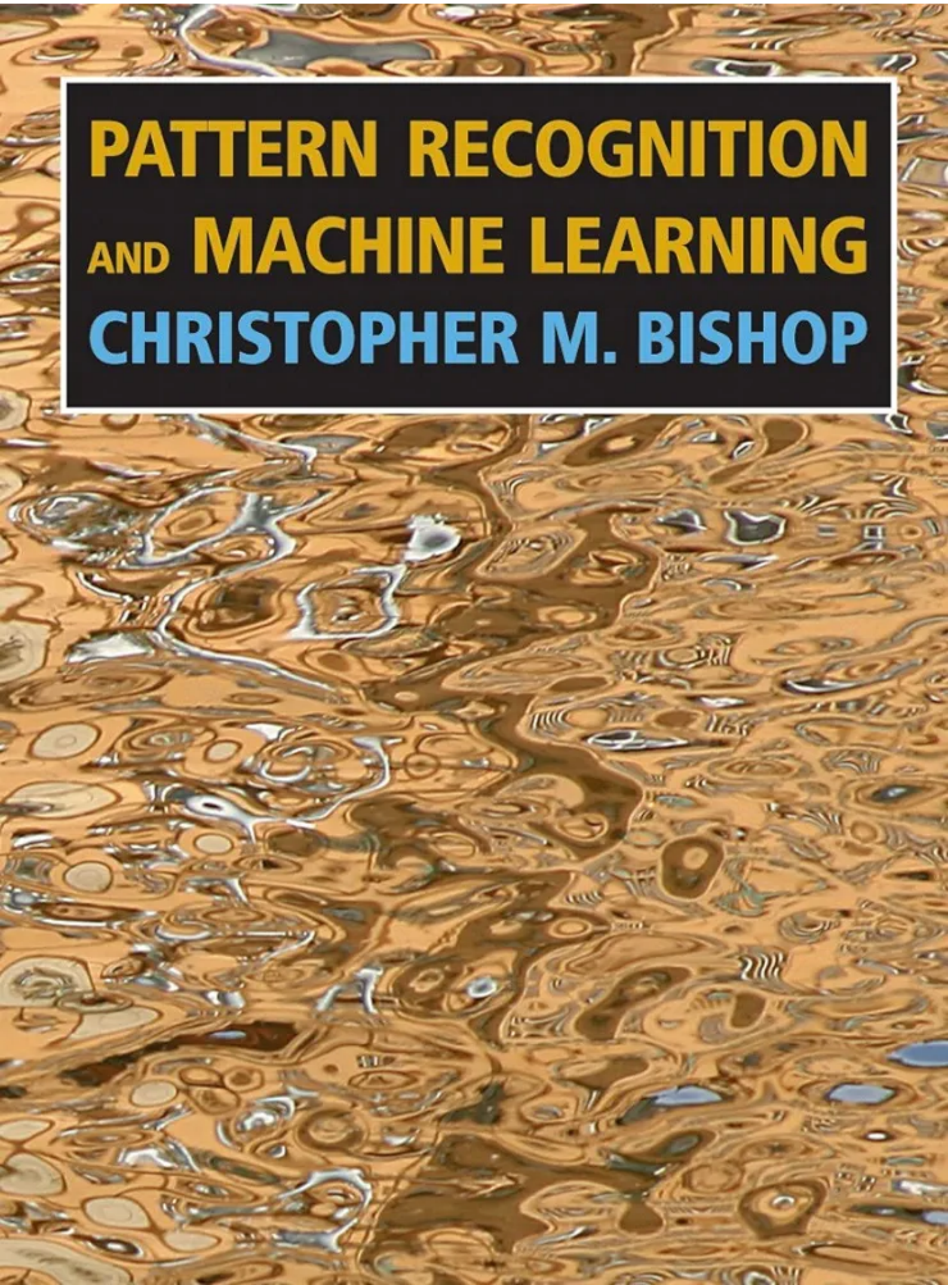

PATTERN RECOGNITION
AND MACHINE LEARNING
CHRISTOPHER M. BISHOP

Christopher M. Bishop

Pattern Recognition and Machine Learning

Springer

Christopher M. Bishop F.R.Eng.
Assistant Director
Microsoft Research Ltd
Cambridge CB3 0FB, U.K.
cmbishop@microsoft.com
http://research.microsoft.com/~cmbishop

Series Editors
Michael Jordan
Department of Computer Science and Department of Statistics
University of California, Berkeley
Berkeley, CA 94720
USA

Professor Jon Kleinberg
Department of Computer Science
Cornell University
Ithaca, NY 14853
USA

Bernhard Schölkopf
Max Planck Institute for Biological Cybernetics
Spemannstrasse 38
72076 Tübingen
Germany

Library of Congress Control Number: 2006922522

ISBN-10: 0-387-31073-8
ISBN-13: 978-0387-31073-2

Printed on acid-free paper.

© 2006 Springer Science+Business Media, LLC
All rights reserved. This work may not be translated or copied in whole or in part without the written permission of the publisher (Springer Science+Business Media, LLC, 233 Spring Street, New York, NY 10013, USA), except for brief excerpts in connection with reviews or scholarly analysis. Use in connection with any form of information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed is forbidden.
The use in this publication of trade names, trademarks, service marks, and similar terms, even if they are not identified as such, is not to be taken as an expression of opinion as to whether or not they are subject to proprietary rights.

Printed in Singapore. (KYO)

9 8 7 6 5 4 3 2 1

springer.com

This book is dedicated to my family:

Jenna, Mark, and Hugh

Total eclipse of the sun, Antalya, Turkey, 29 March 2006.

Preface

Pattern recognition has its origins in engineering, whereas machine learning grew out of computer science. However, these activities can be viewed as two facets of the same field, and together they have undergone substantial development over the past ten years. In particular, Bayesian methods have grown from a specialist niche to become mainstream, while graphical models have emerged as a general framework for describing and applying probabilistic models. Also, the practical applicability of Bayesian methods has been greatly enhanced through the development of a range of approximate inference algorithms such as variational Bayes and expectation propagation. Similarly, new models based on kernels have had significant impact on both algorithms and applications.

This new textbook reflects these recent developments while providing a comprehensive introduction to the fields of pattern recognition and machine learning. It is aimed at advanced undergraduates or first year PhD students, as well as researchers and practitioners, and assumes no previous knowledge of pattern recognition or machine learning concepts. Knowledge of multivariate calculus and basic linear algebra is required, and some familiarity with probabilities would be helpful though not essential as the book includes a self-contained introduction to basic probability theory.

Because this book has broad scope, it is impossible to provide a complete list of references, and in particular no attempt has been made to provide accurate historical attribution of ideas. Instead, the aim has been to give references that offer greater detail than is possible here and that hopefully provide entry points into what, in some cases, is a very extensive literature. For this reason, the references are often to more recent textbooks and review articles rather than to original sources.

The book is supported by a great deal of additional material, including lecture slides as well as the complete set of figures used in the book, and the reader is encouraged to visit the book web site for the latest information:

http://research.microsoft.com/∼cmbishop/PRML

Exercises

The exercises that appear at the end of every chapter form an important component of the book. Each exercise has been carefully chosen to reinforce concepts explained in the text or to develop and generalize them in significant ways, and each is graded according to difficulty ranging from ($\star$), which denotes a simple exercise taking a few minutes to complete, through to ($\star\star\star$), which denotes a significantly more complex exercise.

It has been difficult to know to what extent these solutions should be made widely available. Those engaged in self study will find worked solutions very beneficial, whereas many course tutors request that solutions be available only via the publisher so that the exercises may be used in class. In order to try to meet these conflicting requirements, those exercises that help amplify key points in the text, or that fill in important details, have solutions that are available as a PDF file from the book web site. Such exercises are denoted by **www**. Solutions for the remaining exercises are available to course tutors by contacting the publisher (contact details are given on the book web site). Readers are strongly encouraged to work through the exercises unaided, and to turn to the solutions only as required.

Although this book focuses on concepts and principles, in a taught course the students should ideally have the opportunity to experiment with some of the key algorithms using appropriate data sets. A companion volume (Bishop and Nabney, 2008) will deal with practical aspects of pattern recognition and machine learning, and will be accompanied by Matlab software implementing most of the algorithms discussed in this book.

Acknowledgements

First of all I would like to express my sincere thanks to Markus Svensén who has provided immense help with preparation of figures and with the typesetting of the book in LaTeX. His assistance has been invaluable.

I am very grateful to Microsoft Research for providing a highly stimulating research environment and for giving me the freedom to write this book (the views and opinions expressed in this book, however, are my own and are therefore not necessarily the same as those of Microsoft or its affiliates).

Springer has provided excellent support throughout the final stages of preparation of this book, and I would like to thank my commissioning editor John Kimmel for his support and professionalism, as well as Joseph Piliero for his help in designing the cover and the text format and MaryAnn Brickner for her numerous contributions during the production phase. The inspiration for the cover design came from a discussion with Antonio Criminisi.

I also wish to thank Oxford University Press for permission to reproduce excerpts from an earlier textbook, *Neural Networks for Pattern Recognition* (Bishop, 1995a). The images of the Mark 1 perceptron and of Frank Rosenblatt are reproduced with the permission of Arvin Calspan Advanced Technology Center. I would also like to thank Asela Gunawardana for plotting the spectrogram in Figure 13.1, and Bernhard Schölkopf for permission to use his kernel PCA code to plot Figure 12.17.

Many people have helped by proofreading draft material and providing comments and suggestions, including Shivani Agarwal, Cédric Archambeau, Arik Azran, Andrew Blake, Hakan Cevikalp, Michael Fourman, Brendan Frey, Zoubin Ghahramani, Thore Graepel, Katherine Heller, Ralf Herbrich, Geoffrey Hinton, Adam Johansen, Matthew Johnson, Michael Jordan, Eva Kalyvianaki, Anitha Kannan, Julia Lasserre, David Liu, Tom Minka, Ian Nabney, Tonatiuh Pena, Yuan Qi, Sam Roweis, Balaji Sanjiya, Toby Sharp, Ana Costa e Silva, David Spiegelhalter, Jay Stokes, Tara Symeonides, Martin Szummer, Marshall Tappen, Ilkay Ulusoy, Chris Williams, John Winn, and Andrew Zisserman.

Finally, I would like to thank my wife Jenna who has been hugely supportive throughout the several years it has taken to write this book.

Chris Bishop
Cambridge
February 2006

Mathematical notation

I have tried to keep the mathematical content of the book to the minimum necessary to achieve a proper understanding of the field. However, this minimum level is nonzero, and it should be emphasized that a good grasp of calculus, linear algebra, and probability theory is essential for a clear understanding of modern pattern recognition and machine learning techniques. Nevertheless, the emphasis in this book is on conveying the underlying concepts rather than on mathematical rigour.

I have tried to use a consistent notation throughout the book, although at times this means departing from some of the conventions used in the corresponding research literature. Vectors are denoted by lower case bold Roman letters such as $\mathbf{x}$, and all vectors are assumed to be column vectors. A superscript T denotes the transpose of a matrix or vector, so that $\mathbf{x}^{\mathrm{T}}$ will be a row vector. Uppercase bold roman letters, such as $\mathbf{M}$, denote matrices. The notation $(w_1, \ldots, w_M)$ denotes a row vector with M elements, while the corresponding column vector is written as $\mathbf{w} = (w_1, \ldots, w_M)^{\mathrm{T}}$.

The notation $[a, b]$ is used to denote the *closed* interval from a to b, that is the interval including the values a and b themselves, while (a, b) denotes the corresponding *open* interval, that is the interval excluding a and b. Similarly, $[a, b)$ denotes an interval that includes a but excludes b. For the most part, however, there will be little need to dwell on such refinements as whether the end points of an interval are included or not.

The $M \times M$ identity matrix (also known as the unit matrix) is denoted $\mathbf{I}_M$, which will be abbreviated to $\mathbf{I}$ where there is no ambiguity about it dimensionality. It has elements I_{ij} that equal 1 if $i = j$ and 0 if $i \neq j$.

A functional is denoted $f[y]$ where $y(x)$ is some function. The concept of a functional is discussed in Appendix D.

The notation $g(x) = O(f(x))$ denotes that $|f(x)/g(x)|$ is bounded as $x \to \infty$. For instance if $g(x) = 3x^2 + 2$, then $g(x) = O(x^2)$.

The expectation of a function $f(x, y)$ with respect to a random variable x is denoted by $\mathbb{E}_x[f(x, y)]$. In situations where there is no ambiguity as to which variable is being averaged over, this will be simplified by omitting the suffix, for instance

$\mathbb{E}[x]$. If the distribution of x is conditioned on another variable z, then the corresponding conditional expectation will be written $\mathbb{E}_x[f(x)|z]$. Similarly, the variance is denoted $\mathrm{var}[f(x)]$, and for vector variables the covariance is written $\mathrm{cov}[\mathbf{x}, \mathbf{y}]$. We shall also use $\mathrm{cov}[\mathbf{x}]$ as a shorthand notation for $\mathrm{cov}[\mathbf{x}, \mathbf{x}]$. The concepts of expectations and covariances are introduced in Section 1.2.2.

If we have N values $\mathbf{x}_1, \ldots, \mathbf{x}_N$ of a D-dimensional vector $\mathbf{x} = (x_1, \ldots, x_D)^{\mathrm{T}}$, we can combine the observations into a data matrix $\mathbf{X}$ in which the n^{th} row of $\mathbf{X}$ corresponds to the row vector $\mathbf{x}_n^{\mathrm{T}}$. Thus the n, i element of $\mathbf{X}$ corresponds to the i^{th} element of the n^{th} observation $\mathbf{x}_n$. For the case of one-dimensional variables we shall denote such a matrix by $\mathbf{x}$, which is a column vector whose n^{th} element is x_n. Note that $\mathbf{x}$ (which has dimensionality N) uses a different typeface to distinguish it from $\mathbf{x}$ (which has dimensionality D).

Contents

Preface vii

Mathematical notation xi

Contents xiii

1 Introduction 1
- 1.1 Example: Polynomial Curve Fitting . . . 4
- 1.2 Probability Theory . . . 12
  - 1.2.1 Probability densities . . . 17
  - 1.2.2 Expectations and covariances . . . 19
  - 1.2.3 Bayesian probabilities . . . 21
  - 1.2.4 The Gaussian distribution . . . 24
  - 1.2.5 Curve fitting re-visited . . . 28
  - 1.2.6 Bayesian curve fitting . . . 30
- 1.3 Model Selection . . . 32
- 1.4 The Curse of Dimensionality . . . 33
- 1.5 Decision Theory . . . 38
  - 1.5.1 Minimizing the misclassification rate . . . 39
  - 1.5.2 Minimizing the expected loss . . . 41
  - 1.5.3 The reject option . . . 42
  - 1.5.4 Inference and decision . . . 42
  - 1.5.5 Loss functions for regression . . . 46
- 1.6 Information Theory . . . 48
  - 1.6.1 Relative entropy and mutual information . . . 55

Exercises . . . 58

2 Probability Distributions **67**
2.1 Binary Variables . . . 68
2.1.1 The beta distribution . . . 71
2.2 Multinomial Variables . . . 74
2.2.1 The Dirichlet distribution . . . 76
2.3 The Gaussian Distribution . . . 78
2.3.1 Conditional Gaussian distributions . . . 85
2.3.2 Marginal Gaussian distributions . . . 88
2.3.3 Bayes' theorem for Gaussian variables . . . 90
2.3.4 Maximum likelihood for the Gaussian . . . 93
2.3.5 Sequential estimation . . . 94
2.3.6 Bayesian inference for the Gaussian . . . 97
2.3.7 Student's t-distribution . . . 102
2.3.8 Periodic variables . . . 105
2.3.9 Mixtures of Gaussians . . . 110
2.4 The Exponential Family . . . 113
2.4.1 Maximum likelihood and sufficient statistics . . . 116
2.4.2 Conjugate priors . . . 117
2.4.3 Noninformative priors . . . 117
2.5 Nonparametric Methods . . . 120
2.5.1 Kernel density estimators . . . 122
2.5.2 Nearest-neighbour methods . . . 124
Exercises . . . 127

3 Linear Models for Regression **137**
3.1 Linear Basis Function Models . . . 138
3.1.1 Maximum likelihood and least squares . . . 140
3.1.2 Geometry of least squares . . . 143
3.1.3 Sequential learning . . . 143
3.1.4 Regularized least squares . . . 144
3.1.5 Multiple outputs . . . 146
3.2 The Bias-Variance Decomposition . . . 147
3.3 Bayesian Linear Regression . . . 152
3.3.1 Parameter distribution . . . 152
3.3.2 Predictive distribution . . . 156
3.3.3 Equivalent kernel . . . 159
3.4 Bayesian Model Comparison . . . 161
3.5 The Evidence Approximation . . . 165
3.5.1 Evaluation of the evidence function . . . 166
3.5.2 Maximizing the evidence function . . . 168
3.5.3 Effective number of parameters . . . 170
3.6 Limitations of Fixed Basis Functions . . . 172
Exercises . . . 173

4 Linear Models for Classification **179**
4.1 Discriminant Functions . . . 181
4.1.1 Two classes . . . 181
4.1.2 Multiple classes . . . 182
4.1.3 Least squares for classification . . . 184
4.1.4 Fisher's linear discriminant . . . 186
4.1.5 Relation to least squares . . . 189
4.1.6 Fisher's discriminant for multiple classes . . . 191
4.1.7 The perceptron algorithm . . . 192
4.2 Probabilistic Generative Models . . . 196
4.2.1 Continuous inputs . . . 198
4.2.2 Maximum likelihood solution . . . 200
4.2.3 Discrete features . . . 202
4.2.4 Exponential family . . . 202
4.3 Probabilistic Discriminative Models . . . 203
4.3.1 Fixed basis functions . . . 204
4.3.2 Logistic regression . . . 205
4.3.3 Iterative reweighted least squares . . . 207
4.3.4 Multiclass logistic regression . . . 209
4.3.5 Probit regression . . . 210
4.3.6 Canonical link functions . . . 212
4.4 The Laplace Approximation . . . 213
4.4.1 Model comparison and BIC . . . 216
4.5 Bayesian Logistic Regression . . . 217
4.5.1 Laplace approximation . . . 217
4.5.2 Predictive distribution . . . 218
Exercises . . . 220

5 Neural Networks **225**
5.1 Feed-forward Network Functions . . . 227
5.1.1 Weight-space symmetries . . . 231
5.2 Network Training . . . 232
5.2.1 Parameter optimization . . . 236
5.2.2 Local quadratic approximation . . . 237
5.2.3 Use of gradient information . . . 239
5.2.4 Gradient descent optimization . . . 240
5.3 Error Backpropagation . . . 241
5.3.1 Evaluation of error-function derivatives . . . 242
5.3.2 A simple example . . . 245
5.3.3 Efficiency of backpropagation . . . 246
5.3.4 The Jacobian matrix . . . 247
5.4 The Hessian Matrix . . . 249
5.4.1 Diagonal approximation . . . 250
5.4.2 Outer product approximation . . . 251
5.4.3 Inverse Hessian . . . 252

5.4.4 Finite differences . . . 252
5.4.5 Exact evaluation of the Hessian . . . 253
5.4.6 Fast multiplication by the Hessian . . . 254
5.5 Regularization in Neural Networks . . . 256
5.5.1 Consistent Gaussian priors . . . 257
5.5.2 Early stopping . . . 259
5.5.3 Invariances . . . 261
5.5.4 Tangent propagation . . . 263
5.5.5 Training with transformed data . . . 265
5.5.6 Convolutional networks . . . 267
5.5.7 Soft weight sharing . . . 269
5.6 Mixture Density Networks . . . 272
5.7 Bayesian Neural Networks . . . 277
5.7.1 Posterior parameter distribution . . . 278
5.7.2 Hyperparameter optimization . . . 280
5.7.3 Bayesian neural networks for classification . . . 281
Exercises . . . 284

6 Kernel Methods **291**
6.1 Dual Representations . . . 293
6.2 Constructing Kernels . . . 294
6.3 Radial Basis Function Networks . . . 299
6.3.1 Nadaraya-Watson model . . . 301
6.4 Gaussian Processes . . . 303
6.4.1 Linear regression revisited . . . 304
6.4.2 Gaussian processes for regression . . . 306
6.4.3 Learning the hyperparameters . . . 311
6.4.4 Automatic relevance determination . . . 312
6.4.5 Gaussian processes for classification . . . 313
6.4.6 Laplace approximation . . . 315
6.4.7 Connection to neural networks . . . 319
Exercises . . . 320

7 Sparse Kernel Machines **325**
7.1 Maximum Margin Classifiers . . . 326
7.1.1 Overlapping class distributions . . . 331
7.1.2 Relation to logistic regression . . . 336
7.1.3 Multiclass SVMs . . . 338
7.1.4 SVMs for regression . . . 339
7.1.5 Computational learning theory . . . 344
7.2 Relevance Vector Machines . . . 345
7.2.1 RVM for regression . . . 345
7.2.2 Analysis of sparsity . . . 349
7.2.3 RVM for classification . . . 353
Exercises . . . 357

8 Graphical Models **359**
8.1 Bayesian Networks . . . 360
8.1.1 Example: Polynomial regression . . . 362
8.1.2 Generative models . . . 365
8.1.3 Discrete variables . . . 366
8.1.4 Linear-Gaussian models . . . 370
8.2 Conditional Independence . . . 372
8.2.1 Three example graphs . . . 373
8.2.2 D-separation . . . 378
8.3 Markov Random Fields . . . 383
8.3.1 Conditional independence properties . . . 383
8.3.2 Factorization properties . . . 384
8.3.3 Illustration: Image de-noising . . . 387
8.3.4 Relation to directed graphs . . . 390
8.4 Inference in Graphical Models . . . 393
8.4.1 Inference on a chain . . . 394
8.4.2 Trees . . . 398
8.4.3 Factor graphs . . . 399
8.4.4 The sum-product algorithm . . . 402
8.4.5 The max-sum algorithm . . . 411
8.4.6 Exact inference in general graphs . . . 416
8.4.7 Loopy belief propagation . . . 417
8.4.8 Learning the graph structure . . . 418
Exercises . . . 418

9 Mixture Models and EM **423**
9.1 K-means Clustering . . . 424
9.1.1 Image segmentation and compression . . . 428
9.2 Mixtures of Gaussians . . . 430
9.2.1 Maximum likelihood . . . 432
9.2.2 EM for Gaussian mixtures . . . 435
9.3 An Alternative View of EM . . . 439
9.3.1 Gaussian mixtures revisited . . . 441
9.3.2 Relation to K-means . . . 443
9.3.3 Mixtures of Bernoulli distributions . . . 444
9.3.4 EM for Bayesian linear regression . . . 448
9.4 The EM Algorithm in General . . . 450
Exercises . . . 455

10 Approximate Inference **461**
10.1 Variational Inference . . . 462
10.1.1 Factorized distributions . . . 464
10.1.2 Properties of factorized approximations . . . 466
10.1.3 Example: The univariate Gaussian . . . 470
10.1.4 Model comparison . . . 473
10.2 Illustration: Variational Mixture of Gaussians . . . 474

10.2.1 Variational distribution 475
10.2.2 Variational lower bound 481
10.2.3 Predictive density 482
10.2.4 Determining the number of components 483
10.2.5 Induced factorizations 485
10.3 Variational Linear Regression 486
10.3.1 Variational distribution 486
10.3.2 Predictive distribution 488
10.3.3 Lower bound 489
10.4 Exponential Family Distributions 490
10.4.1 Variational message passing 491
10.5 Local Variational Methods 493
10.6 Variational Logistic Regression 498
10.6.1 Variational posterior distribution 498
10.6.2 Optimizing the variational parameters 500
10.6.3 Inference of hyperparameters 502
10.7 Expectation Propagation 505
10.7.1 Example: The clutter problem 511
10.7.2 Expectation propagation on graphs 513
Exercises 517

11 Sampling Methods **523**
11.1 Basic Sampling Algorithms 526
11.1.1 Standard distributions 526
11.1.2 Rejection sampling 528
11.1.3 Adaptive rejection sampling 530
11.1.4 Importance sampling 532
11.1.5 Sampling-importance-resampling 534
11.1.6 Sampling and the EM algorithm 536
11.2 Markov Chain Monte Carlo 537
11.2.1 Markov chains 539
11.2.2 The Metropolis-Hastings algorithm 541
11.3 Gibbs Sampling 542
11.4 Slice Sampling 546
11.5 The Hybrid Monte Carlo Algorithm 548
11.5.1 Dynamical systems 548
11.5.2 Hybrid Monte Carlo 552
11.6 Estimating the Partition Function 554
Exercises 556

12 Continuous Latent Variables **559**
12.1 Principal Component Analysis 561
12.1.1 Maximum variance formulation 561
12.1.2 Minimum-error formulation 563
12.1.3 Applications of PCA 565
12.1.4 PCA for high-dimensional data 569

12.2 Probabilistic PCA 570
12.2.1 Maximum likelihood PCA 574
12.2.2 EM algorithm for PCA 577
12.2.3 Bayesian PCA 580
12.2.4 Factor analysis 583
12.3 Kernel PCA 586
12.4 Nonlinear Latent Variable Models 591
12.4.1 Independent component analysis 591
12.4.2 Autoassociative neural networks 592
12.4.3 Modelling nonlinear manifolds 595
Exercises 599

13 Sequential Data **605**
13.1 Markov Models 607
13.2 Hidden Markov Models 610
13.2.1 Maximum likelihood for the HMM 615
13.2.2 The forward-backward algorithm 618
13.2.3 The sum-product algorithm for the HMM 625
13.2.4 Scaling factors 627
13.2.5 The Viterbi algorithm 629
13.2.6 Extensions of the hidden Markov model 631
13.3 Linear Dynamical Systems 635
13.3.1 Inference in LDS 638
13.3.2 Learning in LDS 642
13.3.3 Extensions of LDS 644
13.3.4 Particle filters 645
Exercises 646

14 Combining Models **653**
14.1 Bayesian Model Averaging 654
14.2 Committees 655
14.3 Boosting 657
14.3.1 Minimizing exponential error 659
14.3.2 Error functions for boosting 661
14.4 Tree-based Models 663
14.5 Conditional Mixture Models 666
14.5.1 Mixtures of linear regression models 667
14.5.2 Mixtures of logistic models 670
14.5.3 Mixtures of experts 672
Exercises 674

Appendix A Data Sets **677**

Appendix B Probability Distributions **685**

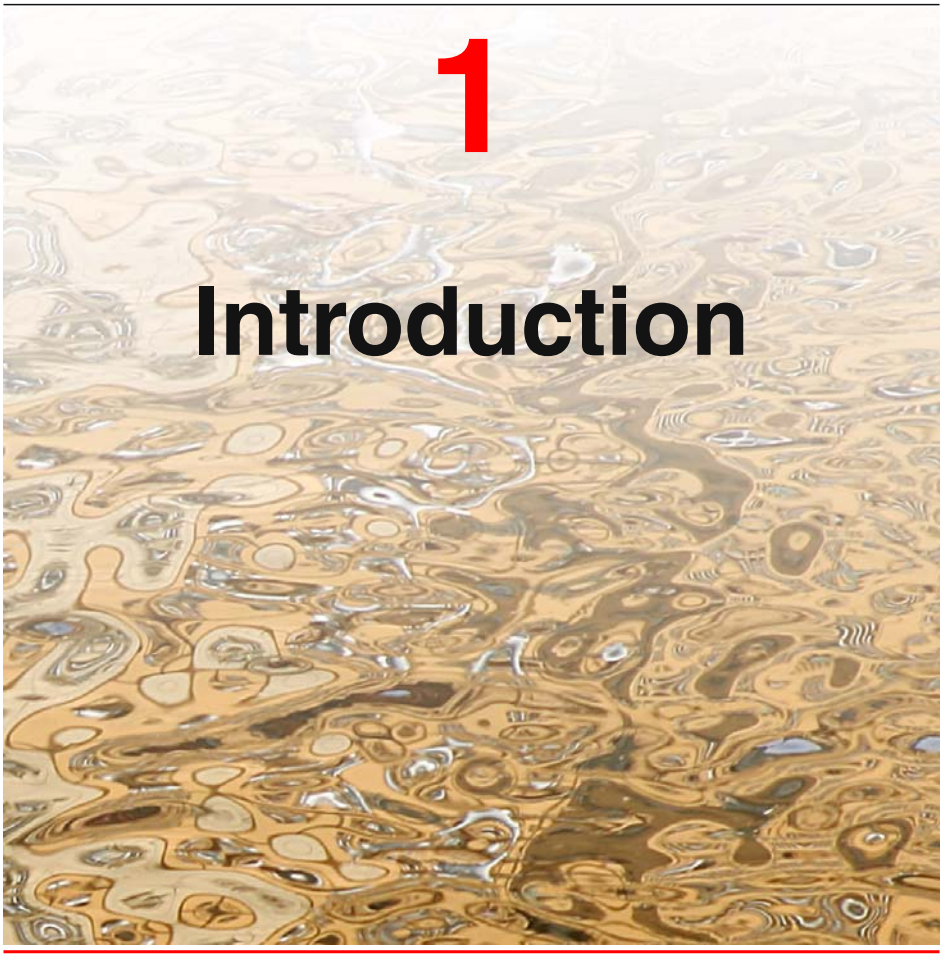
Appendix C Properties of Matrices **695**

Appendix D Calculus of Variations **703**

Appendix E Lagrange Multipliers **707**

References **711**

Index **729**



1 Introduction

The problem of searching for patterns in data is a fundamental one and has a long and successful history. For instance, the extensive astronomical observations of Tycho Brahe in the 16^{th} century allowed Johannes Kepler to discover the empirical laws of planetary motion, which in turn provided a springboard for the development of classical mechanics. Similarly, the discovery of regularities in atomic spectra played a key role in the development and verification of quantum physics in the early twentieth century. The field of pattern recognition is concerned with the automatic discovery of regularities in data through the use of computer algorithms and with the use of these regularities to take actions such as classifying the data into different categories.

Consider the example of recognizing handwritten digits, illustrated in Figure 1.1. Each digit corresponds to a 28×28 pixel image and so can be represented by a vector $\mathbf{x}$ comprising 784 real numbers. The goal is to build a machine that will take such a vector $\mathbf{x}$ as input and that will produce the identity of the digit $0, \ldots, 9$ as the output. This is a nontrivial problem due to the wide variability of handwriting. It could be

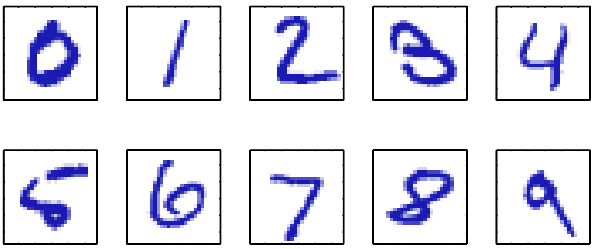


Figure 1.1 Examples of hand-written digits taken from US zip codes.

tackled using handcrafted rules or heuristics for distinguishing the digits based on the shapes of the strokes, but in practice such an approach leads to a proliferation of rules and of exceptions to the rules and so on, and invariably gives poor results.

Far better results can be obtained by adopting a machine learning approach in which a large set of N digits $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ called a *training set* is used to tune the parameters of an adaptive model. The categories of the digits in the training set are known in advance, typically by inspecting them individually and hand-labelling them. We can express the category of a digit using *target vector* $\mathbf{t}$, which represents the identity of the corresponding digit. Suitable techniques for representing categories in terms of vectors will be discussed later. Note that there is one such target vector $\mathbf{t}$ for each digit image $\mathbf{x}$.

The result of running the machine learning algorithm can be expressed as a function $\mathbf{y}(\mathbf{x})$ which takes a new digit image $\mathbf{x}$ as input and that generates an output vector $\mathbf{y}$, encoded in the same way as the target vectors. The precise form of the function $\mathbf{y}(\mathbf{x})$ is determined during the *training* phase, also known as the *learning* phase, on the basis of the training data. Once the model is trained it can then determine the identity of new digit images, which are said to comprise a *test set*. The ability to categorize correctly new examples that differ from those used for training is known as *generalization*. In practical applications, the variability of the input vectors will be such that the training data can comprise only a tiny fraction of all possible input vectors, and so generalization is a central goal in pattern recognition.

For most practical applications, the original input variables are typically *preprocessed* to transform them into some new space of variables where, it is hoped, the pattern recognition problem will be easier to solve. For instance, in the digit recognition problem, the images of the digits are typically translated and scaled so that each digit is contained within a box of a fixed size. This greatly reduces the variability within each digit class, because the location and scale of all the digits are now the same, which makes it much easier for a subsequent pattern recognition algorithm to distinguish between the different classes. This pre-processing stage is sometimes also called *feature extraction*. Note that new test data must be pre-processed using the same steps as the training data.

Pre-processing might also be performed in order to speed up computation. For example, if the goal is real-time face detection in a high-resolution video stream, the computer must handle huge numbers of pixels per second, and presenting these directly to a complex pattern recognition algorithm may be computationally infeasible. Instead, the aim is to find useful features that are fast to compute, and yet that

also preserve useful discriminatory information enabling faces to be distinguished from non-faces. These features are then used as the inputs to the pattern recognition algorithm. For instance, the average value of the image intensity over a rectangular subregion can be evaluated extremely efficiently (Viola and Jones, 2004), and a set of such features can prove very effective in fast face detection. Because the number of such features is smaller than the number of pixels, this kind of pre-processing represents a form of dimensionality reduction. Care must be taken during pre-processing because often information is discarded, and if this information is important to the solution of the problem then the overall accuracy of the system can suffer.

Applications in which the training data comprises examples of the input vectors along with their corresponding target vectors are known as *supervised learning* problems. Cases such as the digit recognition example, in which the aim is to assign each input vector to one of a finite number of discrete categories, are called *classification* problems. If the desired output consists of one or more continuous variables, then the task is called *regression*. An example of a regression problem would be the prediction of the yield in a chemical manufacturing process in which the inputs consist of the concentrations of reactants, the temperature, and the pressure.

In other pattern recognition problems, the training data consists of a set of input vectors $\mathbf{x}$ without any corresponding target values. The goal in such *unsupervised learning* problems may be to discover groups of similar examples within the data, where it is called *clustering*, or to determine the distribution of data within the input space, known as *density estimation*, or to project the data from a high-dimensional space down to two or three dimensions for the purpose of *visualization*.

Finally, the technique of *reinforcement learning* (Sutton and Barto, 1998) is concerned with the problem of finding suitable actions to take in a given situation in order to maximize a reward. Here the learning algorithm is not given examples of optimal outputs, in contrast to supervised learning, but must instead discover them by a process of trial and error. Typically there is a sequence of states and actions in which the learning algorithm is interacting with its environment. In many cases, the current action not only affects the immediate reward but also has an impact on the reward at all subsequent time steps. For example, by using appropriate reinforcement learning techniques a neural network can learn to play the game of backgammon to a high standard (Tesauro, 1994). Here the network must learn to take a board position as input, along with the result of a dice throw, and produce a strong move as the output. This is done by having the network play against a copy of itself for perhaps a million games. A major challenge is that a game of backgammon can involve dozens of moves, and yet it is only at the end of the game that the reward, in the form of victory, is achieved. The reward must then be attributed appropriately to all of the moves that led to it, even though some moves will have been good ones and others less so. This is an example of a *credit assignment* problem. A general feature of reinforcement learning is the trade-off between *exploration*, in which the system tries out new kinds of actions to see how effective they are, and *exploitation*, in which the system makes use of actions that are known to yield a high reward. Too strong a focus on either exploration or exploitation will yield poor results. Reinforcement learning continues to be an active area of machine learning research. However, a

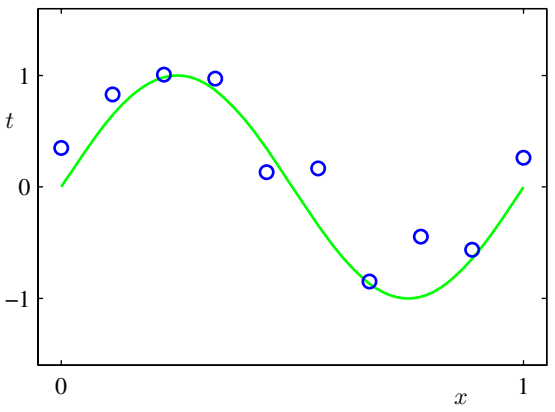


Figure 1.2 Plot of a training data set of $N = 10$ points, shown as blue circles, each comprising an observation of the input variable x along with the corresponding target variable t. The green curve shows the function $\sin(2\pi x)$ used to generate the data. Our goal is to predict the value of t for some new value of x, without knowledge of the green curve.

detailed treatment lies beyond the scope of this book.

Although each of these tasks needs its own tools and techniques, many of the key ideas that underpin them are common to all such problems. One of the main goals of this chapter is to introduce, in a relatively informal way, several of the most important of these concepts and to illustrate them using simple examples. Later in the book we shall see these same ideas re-emerge in the context of more sophisticated models that are applicable to real-world pattern recognition applications. This chapter also provides a self-contained introduction to three important tools that will be used throughout the book, namely probability theory, decision theory, and information theory. Although these might sound like daunting topics, they are in fact straightforward, and a clear understanding of them is essential if machine learning techniques are to be used to best effect in practical applications.

1.1. Example: Polynomial Curve Fitting

We begin by introducing a simple regression problem, which we shall use as a running example throughout this chapter to motivate a number of key concepts. Suppose we observe a real-valued input variable x and we wish to use this observation to predict the value of a real-valued target variable t. For the present purposes, it is instructive to consider an artificial example using synthetically generated data because we then know the precise process that generated the data for comparison against any learned model. The data for this example is generated from the function $\sin(2\pi x)$ with random noise included in the target values, as described in detail in Appendix A.

Now suppose that we are given a training set comprising N observations of x, written $\mathbf{x} \equiv (x_1, \ldots, x_N)^{\mathrm{T}}$, together with corresponding observations of the values of t, denoted $\mathbf{t} \equiv (t_1, \ldots, t_N)^{\mathrm{T}}$. Figure 1.2 shows a plot of a training set comprising $N = 10$ data points. The input data set $\mathbf{x}$ in Figure 1.2 was generated by choosing values of x_n, for $n = 1, \ldots, N$, spaced uniformly in range $[0, 1]$, and the target data set $\mathbf{t}$ was obtained by first computing the corresponding values of the function

$\sin(2\pi x)$ and then adding a small level of random noise having a Gaussian distribution (the Gaussian distribution is discussed in Section 1.2.4) to each such point in order to obtain the corresponding value t_n. By generating data in this way, we are capturing a property of many real data sets, namely that they possess an underlying regularity, which we wish to learn, but that individual observations are corrupted by random noise. This noise might arise from intrinsically stochastic (i.e. random) processes such as radioactive decay but more typically is due to there being sources of variability that are themselves unobserved.

Our goal is to exploit this training set in order to make predictions of the value $\widehat{t}$ of the target variable for some new value $\widehat{x}$ of the input variable. As we shall see later, this involves implicitly trying to discover the underlying function $\sin(2\pi x)$. This is intrinsically a difficult problem as we have to generalize from a finite data set. Furthermore the observed data are corrupted with noise, and so for a given $\widehat{x}$ there is uncertainty as to the appropriate value for $\widehat{t}$. Probability theory, discussed in Section 1.2, provides a framework for expressing such uncertainty in a precise and quantitative manner, and decision theory, discussed in Section 1.5, allows us to exploit this probabilistic representation in order to make predictions that are optimal according to appropriate criteria.

For the moment, however, we shall proceed rather informally and consider a simple approach based on curve fitting. In particular, we shall fit the data using a polynomial function of the form

$$y(x, \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + \ldots + w_M x^M = \sum_{j=0}^{M} w_j x^j \tag{1.1}$$

where M is the *order* of the polynomial, and x^j denotes x raised to the power of j. The polynomial coefficients $w_0, \ldots, w_M$ are collectively denoted by the vector $\mathbf{w}$. Note that, although the polynomial function $y(x, \mathbf{w})$ is a nonlinear function of x, it is a linear function of the coefficients $\mathbf{w}$. Functions, such as the polynomial, which are linear in the unknown parameters have important properties and are called *linear models* and will be discussed extensively in Chapters 3 and 4.

The values of the coefficients will be determined by fitting the polynomial to the training data. This can be done by minimizing an *error function* that measures the misfit between the function $y(x, \mathbf{w})$, for any given value of $\mathbf{w}$, and the training set data points. One simple choice of error function, which is widely used, is given by the sum of the squares of the errors between the predictions $y(x_n, \mathbf{w})$ for each data point x_n and the corresponding target values t_n, so that we minimize

$$E(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^{N} \{y(x_n, \mathbf{w}) - t_n\}^2 \tag{1.2}$$

where the factor of $1/2$ is included for later convenience. We shall discuss the motivation for this choice of error function later in this chapter. For the moment we simply note that it is a nonnegative quantity that would be zero if, and only if, the

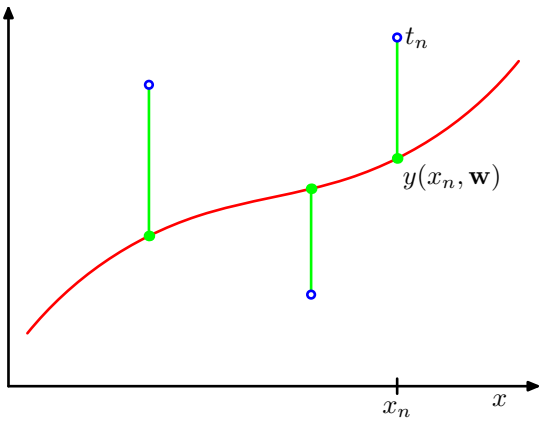


Figure 1.3 The error function (1.2) corresponds to (one half of) the sum of the squares of the displacements (shown by the vertical green bars) of each data point from the function $y(x, \mathbf{w})$.

function $y(x, \mathbf{w})$ were to pass exactly through each training data point. The geometrical interpretation of the sum-of-squares error function is illustrated in Figure 1.3.

We can solve the curve fitting problem by choosing the value of $\mathbf{w}$ for which $E(\mathbf{w})$ is as small as possible. Because the error function is a quadratic function of the coefficients $\mathbf{w}$, its derivatives with respect to the coefficients will be linear in the elements of $\mathbf{w}$, and so the minimization of the error function has a unique solution, denoted by $\mathbf{w}^\star$, which can be found in closed form. The resulting polynomial is given by the function $y(x, \mathbf{w}^\star)$.

Exercise 1.1

There remains the problem of choosing the order M of the polynomial, and as we shall see this will turn out to be an example of an important concept called *model comparison* or *model selection*. In Figure 1.4, we show four examples of the results of fitting polynomials having orders $M = 0, 1, 3$, and 9 to the data set shown in Figure 1.2.

We notice that the constant ($M = 0$) and first order ($M = 1$) polynomials give rather poor fits to the data and consequently rather poor representations of the function $\sin(2\pi x)$. The third order ($M = 3$) polynomial seems to give the best fit to the function $\sin(2\pi x)$ of the examples shown in Figure 1.4. When we go to a much higher order polynomial ($M = 9$), we obtain an excellent fit to the training data. In fact, the polynomial passes exactly through each data point and $E(\mathbf{w}^\star) = 0$. However, the fitted curve oscillates wildly and gives a very poor representation of the function $\sin(2\pi x)$. This latter behaviour is known as *over-fitting*.

As we have noted earlier, the goal is to achieve good generalization by making accurate predictions for new data. We can obtain some quantitative insight into the dependence of the generalization performance on M by considering a separate test set comprising 100 data points generated using exactly the same procedure used to generate the training set points but with new choices for the random noise values included in the target values. For each choice of M, we can then evaluate the residual value of $E(\mathbf{w}^\star)$ given by (1.2) for the training data, and we can also evaluate $E(\mathbf{w}^\star)$ for the test data set. It is sometimes more convenient to use the root-mean-square

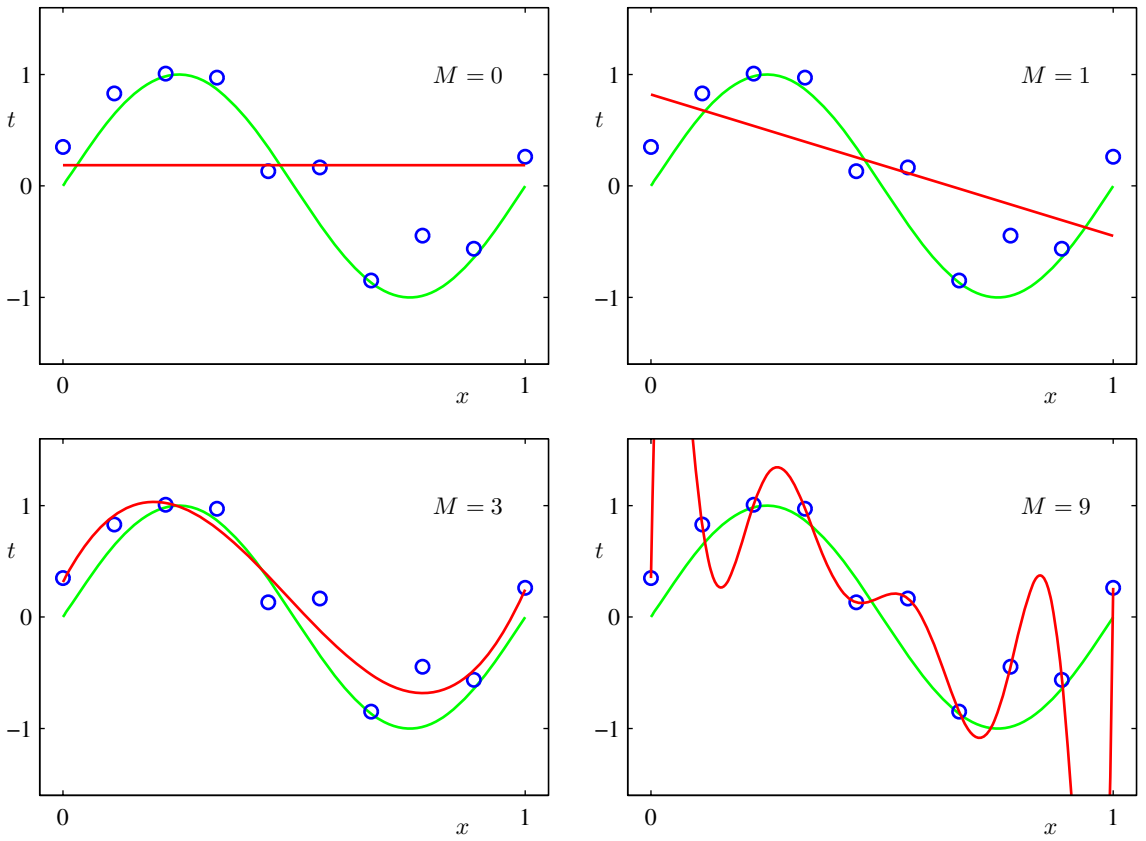


Figure 1.4 Plots of polynomials having various orders M, shown as red curves, fitted to the data set shown in Figure 1.2.

(RMS) error defined by

$$E_{\mathrm{RMS}} = \sqrt{2E(\mathbf{w}^\star)/N} \tag{1.3}$$

in which the division by N allows us to compare different sizes of data sets on an equal footing, and the square root ensures that E_{RMS} is measured on the same scale (and in the same units) as the target variable t. Graphs of the training and test set RMS errors are shown, for various values of M, in Figure 1.5. The test set error is a measure of how well we are doing in predicting the values of t for new data observations of x. We note from Figure 1.5 that small values of M give relatively large values of the test set error, and this can be attributed to the fact that the corresponding polynomials are rather inflexible and are incapable of capturing the oscillations in the function $\sin(2\pi x)$. Values of M in the range $3 \leqslant M \leqslant 8$ give small values for the test set error, and these also give reasonable representations of the generating function $\sin(2\pi x)$, as can be seen, for the case of $M = 3$, from Figure 1.4.

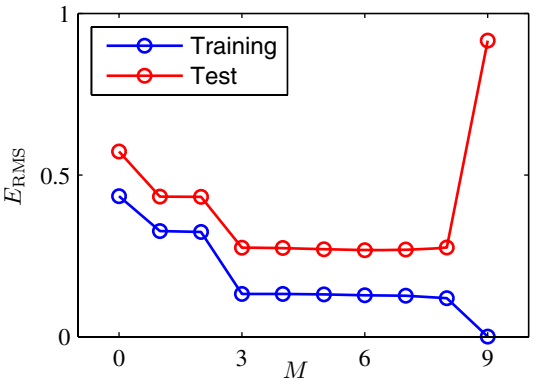


Figure 1.5 Graphs of the root-mean-square error, defined by (1.3), evaluated on the training set and on an independent test set for various values of M.

For $M = 9$, the training set error goes to zero, as we might expect because this polynomial contains 10 degrees of freedom corresponding to the 10 coefficients $w_0, \ldots, w_9$, and so can be tuned exactly to the 10 data points in the training set. However, the test set error has become very large and, as we saw in Figure 1.4, the corresponding function $y(x, \mathbf{w}^\star)$ exhibits wild oscillations.

This may seem paradoxical because a polynomial of given order contains all lower order polynomials as special cases. The $M = 9$ polynomial is therefore capable of generating results at least as good as the $M = 3$ polynomial. Furthermore, we might suppose that the best predictor of new data would be the function $\sin(2\pi x)$ from which the data was generated (and we shall see later that this is indeed the case). We know that a power series expansion of the function $\sin(2\pi x)$ contains terms of all orders, so we might expect that results should improve monotonically as we increase M.

We can gain some insight into the problem by examining the values of the coefficients $\mathbf{w}^\star$ obtained from polynomials of various order, as shown in Table 1.1. We see that, as M increases, the magnitude of the coefficients typically gets larger. In particular for the $M = 9$ polynomial, the coefficients have become finely tuned to the data by developing large positive and negative values so that the correspond-

Table 1.1 Table of the coefficients $\mathbf{w}^\star$ for polynomials of various order. Observe how the typical magnitude of the coefficients increases dramatically as the order of the polynomial increases.

	$M = 0$	$M = 1$	$M = 6$	$M = 9$
$w_0^\star$	0.19	0.82	0.31	0.35
$w_1^\star$		-1.27	7.99	232.37
$w_2^\star$			-25.43	-5321.83
$w_3^\star$			17.37	48568.31
$w_4^\star$				-231639.30
$w_5^\star$				640042.26
$w_6^\star$				-1061800.52
$w_7^\star$				1042400.18
$w_8^\star$				-557682.99
$w_9^\star$				125201.43

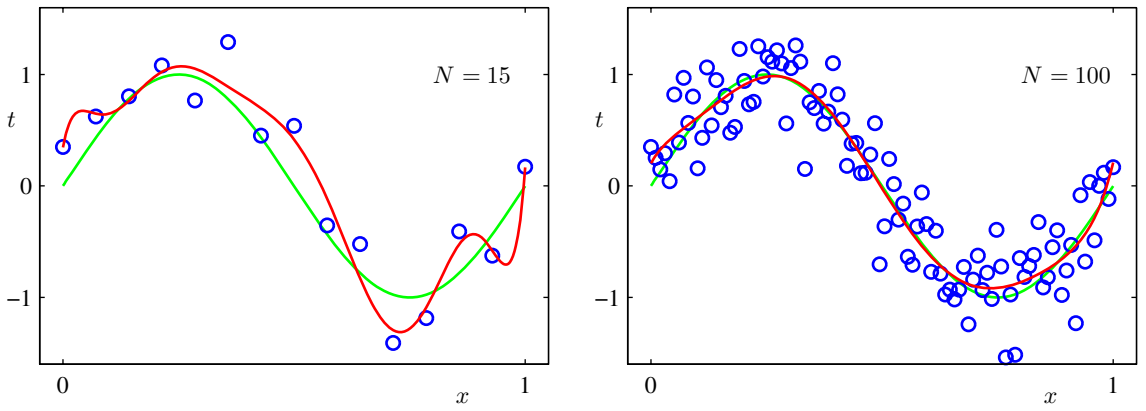


Figure 1.6 Plots of the solutions obtained by minimizing the sum-of-squares error function using the $M = 9$ polynomial for $N = 15$ data points (left plot) and $N = 100$ data points (right plot). We see that increasing the size of the data set reduces the over-fitting problem.

ing polynomial function matches each of the data points exactly, but between data points (particularly near the ends of the range) the function exhibits the large oscillations observed in Figure 1.4. Intuitively, what is happening is that the more flexible polynomials with larger values of M are becoming increasingly tuned to the random noise on the target values.

It is also interesting to examine the behaviour of a given model as the size of the data set is varied, as shown in Figure 1.6. We see that, for a given model complexity, the over-fitting problem become less severe as the size of the data set increases. Another way to say this is that the larger the data set, the more complex (in other words more flexible) the model that we can afford to fit to the data. One rough heuristic that is sometimes advocated is that the number of data points should be no less than some multiple (say 5 or 10) of the number of adaptive parameters in the model. However, as we shall see in Chapter 3, the number of parameters is not necessarily the most appropriate measure of model complexity.

Also, there is something rather unsatisfying about having to limit the number of parameters in a model according to the size of the available training set. It would seem more reasonable to choose the complexity of the model according to the complexity of the problem being solved. We shall see that the least squares approach to finding the model parameters represents a specific case of *maximum likelihood* (discussed in Section 1.2.5), and that the over-fitting problem can be understood as a general property of maximum likelihood. By adopting a *Bayesian* approach, the over-fitting problem can be avoided. We shall see that there is no difficulty from a Bayesian perspective in employing models for which the number of parameters greatly exceeds the number of data points. Indeed, in a Bayesian model the *effective* number of parameters adapts automatically to the size of the data set.

Section 3.4

For the moment, however, it is instructive to continue with the current approach and to consider how in practice we can apply it to data sets of limited size where we

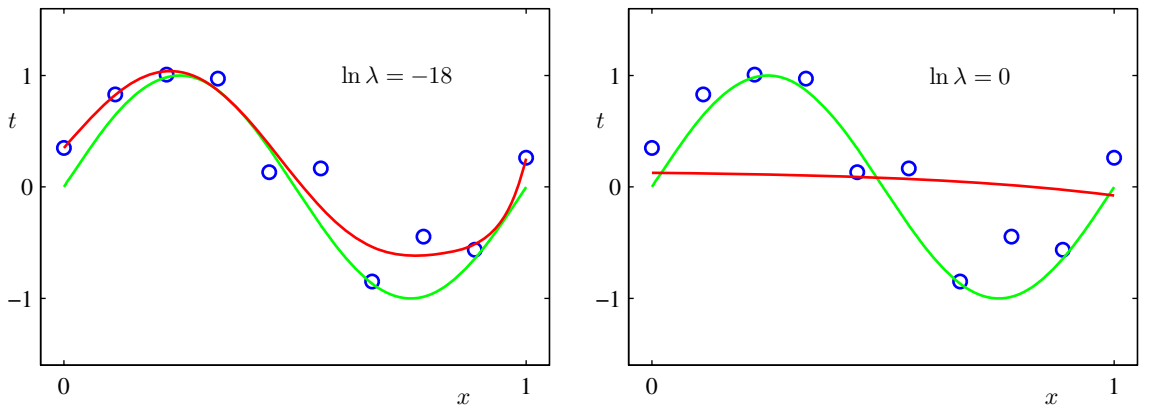


Figure 1.7 Plots of $M = 9$ polynomials fitted to the data set shown in Figure 1.2 using the regularized error function (1.4) for two values of the regularization parameter λ corresponding to $\ln \lambda = -18$ and $\ln \lambda = 0$. The case of no regularizer, i.e., $\lambda = 0$, corresponding to $\ln \lambda = -\infty$, is shown at the bottom right of Figure 1.4.

may wish to use relatively complex and flexible models. One technique that is often used to control the over-fitting phenomenon in such cases is that of *regularization*, which involves adding a penalty term to the error function (1.2) in order to discourage the coefficients from reaching large values. The simplest such penalty term takes the form of a sum of squares of all of the coefficients, leading to a modified error function of the form

$$\widetilde{E}(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n, \mathbf{w}) - t_n\}^2 + \frac{\lambda}{2}\|\mathbf{w}\|^2 \tag{1.4}$$

where $\|\mathbf{w}\|^2 \equiv \mathbf{w}^{\mathrm{T}}\mathbf{w} = w_0^2 + w_1^2 + \ldots + w_M^2$, and the coefficient λ governs the relative importance of the regularization term compared with the sum-of-squares error term. Note that often the coefficient w_0 is omitted from the regularizer because its inclusion causes the results to depend on the choice of origin for the target variable (Hastie *et al.*, 2001), or it may be included but with its own regularization coefficient (we shall discuss this topic in more detail in Section 5.5.1). Again, the error function in (1.4) can be minimized exactly in closed form. Techniques such as this are known in the statistics literature as *shrinkage* methods because they reduce the value of the coefficients. The particular case of a quadratic regularizer is called *ridge regression* (Hoerl and Kennard, 1970). In the context of neural networks, this approach is known as *weight decay*.

Exercise 1.2

Figure 1.7 shows the results of fitting the polynomial of order $M = 9$ to the same data set as before but now using the regularized error function given by (1.4). We see that, for a value of $\ln \lambda = -18$, the over-fitting has been suppressed and we now obtain a much closer representation of the underlying function $\sin(2\pi x)$. If, however, we use too large a value for λ then we again obtain a poor fit, as shown in Figure 1.7 for $\ln \lambda = 0$. The corresponding coefficients from the fitted polynomials are given in Table 1.2, showing that regularization has the desired effect of reducing

Table 1.2 Table of the coefficients $\mathbf{w}^\star$ for $M = 9$ polynomials with various values for the regularization parameter λ. Note that $\ln \lambda = -\infty$ corresponds to a model with no regularization, i.e., to the graph at the bottom right in Figure 1.4. We see that, as the value of λ increases, the typical magnitude of the coefficients gets smaller.

	$\ln \lambda = -\infty$	$\ln \lambda = -18$	$\ln \lambda = 0$
$w_0^\star$	0.35	0.35	0.13
$w_1^\star$	232.37	4.74	-0.05
$w_2^\star$	-5321.83	-0.77	-0.06
$w_3^\star$	48568.31	-31.97	-0.05
$w_4^\star$	-231639.30	-3.89	-0.03
$w_5^\star$	640042.26	55.28	-0.02
$w_6^\star$	-1061800.52	41.32	-0.01
$w_7^\star$	1042400.18	-45.95	-0.00
$w_8^\star$	-557682.99	-91.53	0.00
$w_9^\star$	125201.43	72.68	0.01

the magnitude of the coefficients.

The impact of the regularization term on the generalization error can be seen by plotting the value of the RMS error (1.3) for both training and test sets against $\ln \lambda$, as shown in Figure 1.8. We see that in effect λ now controls the effective complexity of the model and hence determines the degree of over-fitting.

The issue of model complexity is an important one and will be discussed at length in Section 1.3. Here we simply note that, if we were trying to solve a practical application using this approach of minimizing an error function, we would have to find a way to determine a suitable value for the model complexity. The results above suggest a simple way of achieving this, namely by taking the available data and partitioning it into a training set, used to determine the coefficients $\mathbf{w}$, and a separate *validation* set, also called a *hold-out* set, used to optimize the model complexity (either M or λ). In many cases, however, this will prove to be too wasteful of valuable training data, and we have to seek more sophisticated approaches.

Section 1.3

So far our discussion of polynomial curve fitting has appealed largely to intuition. We now seek a more principled approach to solving problems in pattern recognition by turning to a discussion of probability theory. As well as providing the foundation for nearly all of the subsequent developments in this book, it will also

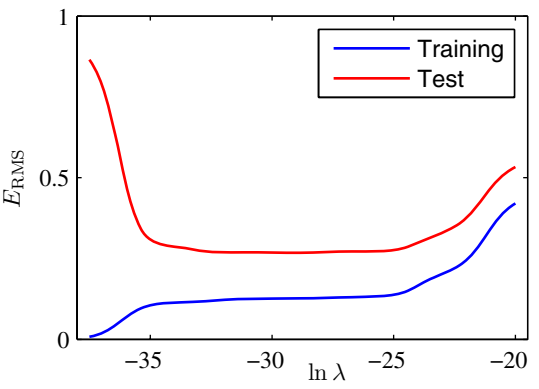


Figure 1.8 Graph of the root-mean-square error (1.3) versus $\ln \lambda$ for the $M = 9$ polynomial.

give us some important insights into the concepts we have introduced in the context of polynomial curve fitting and will allow us to extend these to more complex situations.

1.2. Probability Theory

A key concept in the field of pattern recognition is that of uncertainty. It arises both through noise on measurements, as well as through the finite size of data sets. Probability theory provides a consistent framework for the quantification and manipulation of uncertainty and forms one of the central foundations for pattern recognition. When combined with decision theory, discussed in Section 1.5, it allows us to make optimal predictions given all the information available to us, even though that information may be incomplete or ambiguous.

We will introduce the basic concepts of probability theory by considering a simple example. Imagine we have two boxes, one red and one blue, and in the red box we have 2 apples and 6 oranges, and in the blue box we have 3 apples and 1 orange. This is illustrated in Figure 1.9. Now suppose we randomly pick one of the boxes and from that box we randomly select an item of fruit, and having observed which sort of fruit it is we replace it in the box from which it came. We could imagine repeating this process many times. Let us suppose that in so doing we pick the red box 40% of the time and we pick the blue box 60% of the time, and that when we remove an item of fruit from a box we are equally likely to select any of the pieces of fruit in the box.

In this example, the identity of the box that will be chosen is a random variable, which we shall denote by B. This random variable can take one of two possible values, namely r (corresponding to the red box) or b (corresponding to the blue box). Similarly, the identity of the fruit is also a random variable and will be denoted by F. It can take either of the values a (for apple) or o (for orange).

To begin with, we shall define the probability of an event to be the fraction of times that event occurs out of the total number of trials, in the limit that the total number of trials goes to infinity. Thus the probability of selecting the red box is $4/10$

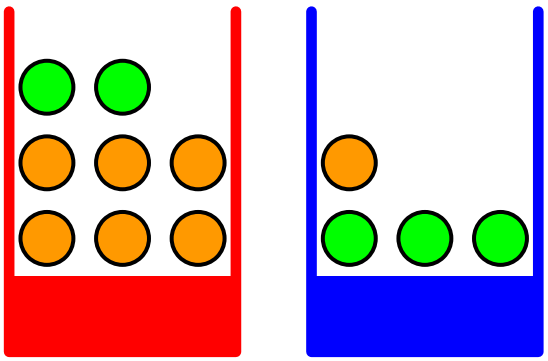


Figure 1.9 We use a simple example of two coloured boxes each containing fruit (apples shown in green and oranges shown in orange) to introduce the basic ideas of probability.

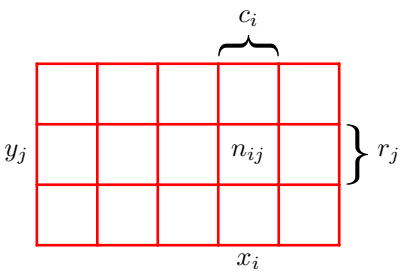


Figure 1.10 We can derive the sum and product rules of probability by considering two random variables, X, which takes the values $\{x_i\}$ where $i = 1, \ldots, M$, and Y, which takes the values $\{y_j\}$ where $j = 1, \ldots, L$. In this illustration we have $M = 5$ and $L = 3$. If we consider a total number N of instances of these variables, then we denote the number of instances where $X = x_i$ and $Y = y_j$ by n_{ij}, which is the number of points in the corresponding cell of the array. The number of points in column i, corresponding to $X = x_i$, is denoted by c_i, and the number of points in row j, corresponding to $Y = y_j$, is denoted by r_j.

and the probability of selecting the blue box is $6/10$. We write these probabilities as $p(B = r) = 4/10$ and $p(B = b) = 6/10$. Note that, by definition, probabilities must lie in the interval $[0, 1]$. Also, if the events are mutually exclusive and if they include all possible outcomes (for instance, in this example the box must be either red or blue), then we see that the probabilities for those events must sum to one.

We can now ask questions such as: "what is the overall probability that the selection procedure will pick an apple?", or "given that we have chosen an orange, what is the probability that the box we chose was the blue one?". We can answer questions such as these, and indeed much more complex questions associated with problems in pattern recognition, once we have equipped ourselves with the two elementary rules of probability, known as the *sum rule* and the *product rule*. Having obtained these rules, we shall then return to our boxes of fruit example.

In order to derive the rules of probability, consider the slightly more general example shown in Figure 1.10 involving two random variables X and Y (which could for instance be the Box and Fruit variables considered above). We shall suppose that X can take any of the values x_i where $i = 1, \ldots, M$, and Y can take the values y_j where $j = 1, \ldots, L$. Consider a total of N trials in which we sample both of the variables X and Y, and let the number of such trials in which $X = x_i$ and $Y = y_j$ be n_{ij}. Also, let the number of trials in which X takes the value x_i (irrespective of the value that Y takes) be denoted by c_i, and similarly let the number of trials in which Y takes the value y_j be denoted by r_j.

The probability that X will take the value x_i and Y will take the value y_j is written $p(X = x_i, Y = y_j)$ and is called the *joint* probability of $X = x_i$ and $Y = y_j$. It is given by the number of points falling in the cell i,j as a fraction of the total number of points, and hence

$$p(X = x_i, Y = y_j) = \frac{n_{ij}}{N}. \tag{1.5}$$

Here we are implicitly considering the limit $N \to \infty$. Similarly, the probability that X takes the value x_i irrespective of the value of Y is written as $p(X = x_i)$ and is given by the fraction of the total number of points that fall in column i, so that

$$p(X = x_i) = \frac{c_i}{N}. \tag{1.6}$$

Because the number of instances in column i in Figure 1.10 is just the sum of the number of instances in each cell of that column, we have $c_i = \sum_j n_{ij}$ and therefore,

from (1.5) and (1.6), we have

$$p(X = x_i) = \sum_{j=1}^{L} p(X = x_i, Y = y_j) \tag{1.7}$$

which is the *sum rule* of probability. Note that $p(X = x_i)$ is sometimes called the *marginal* probability, because it is obtained by marginalizing, or summing out, the other variables (in this case Y).

If we consider only those instances for which $X = x_i$, then the fraction of such instances for which $Y = y_j$ is written $p(Y = y_j|X = x_i)$ and is called the *conditional* probability of $Y = y_j$ given $X = x_i$. It is obtained by finding the fraction of those points in column i that fall in cell i,j and hence is given by

$$p(Y = y_j|X = x_i) = \frac{n_{ij}}{c_i}. \tag{1.8}$$

From (1.5), (1.6), and (1.8), we can then derive the following relationship

$$\begin{aligned} p(X = x_i, Y = y_j) &= \frac{n_{ij}}{N} = \frac{n_{ij}}{c_i} \cdot \frac{c_i}{N} \\ &= p(Y = y_j|X = x_i)p(X = x_i) \end{aligned} \tag{1.9}$$

which is the *product rule* of probability.

So far we have been quite careful to make a distinction between a random variable, such as the box B in the fruit example, and the values that the random variable can take, for example r if the box were the red one. Thus the probability that B takes the value r is denoted $p(B = r)$. Although this helps to avoid ambiguity, it leads to a rather cumbersome notation, and in many cases there will be no need for such pedantry. Instead, we may simply write $p(B)$ to denote a distribution over the random variable B, or $p(r)$ to denote the distribution evaluated for the particular value r, provided that the interpretation is clear from the context.

With this more compact notation, we can write the two fundamental rules of probability theory in the following form.

The Rules of Probability

sum rule
$$p(X) = \sum_{Y} p(X, Y) \tag{1.10}$$

product rule
$$p(X, Y) = p(Y|X)p(X). \tag{1.11}$$

Here $p(X, Y)$ is a joint probability and is verbalized as "the probability of X *and* Y". Similarly, the quantity $p(Y|X)$ is a conditional probability and is verbalized as "the probability of Y *given* X", whereas the quantity $p(X)$ is a marginal probability

and is simply "the probability of X". These two simple rules form the basis for all of the probabilistic machinery that we use throughout this book.

From the product rule, together with the symmetry property $p(X, Y) = p(Y, X)$, we immediately obtain the following relationship between conditional probabilities

$$p(Y|X) = \frac{p(X|Y)p(Y)}{p(X)} \tag{1.12}$$

which is called *Bayes' theorem* and which plays a central role in pattern recognition and machine learning. Using the sum rule, the denominator in Bayes' theorem can be expressed in terms of the quantities appearing in the numerator

$$p(X) = \sum_{Y} p(X|Y)p(Y). \tag{1.13}$$

We can view the denominator in Bayes' theorem as being the normalization constant required to ensure that the sum of the conditional probability on the left-hand side of (1.12) over all values of Y equals one.

In Figure 1.11, we show a simple example involving a joint distribution over two variables to illustrate the concept of marginal and conditional distributions. Here a finite sample of $N = 60$ data points has been drawn from the joint distribution and is shown in the top left. In the top right is a histogram of the fractions of data points having each of the two values of Y. From the definition of probability, these fractions would equal the corresponding probabilities $p(Y)$ in the limit $N \to \infty$. We can view the histogram as a simple way to model a probability distribution given only a finite number of points drawn from that distribution. Modelling distributions from data lies at the heart of statistical pattern recognition and will be explored in great detail in this book. The remaining two plots in Figure 1.11 show the corresponding histogram estimates of $p(X)$ and $p(X|Y = 1)$.

Let us now return to our example involving boxes of fruit. For the moment, we shall once again be explicit about distinguishing between the random variables and their instantiations. We have seen that the probabilities of selecting either the red or the blue boxes are given by

$$p(B = r) = 4/10 \tag{1.14}$$
$$p(B = b) = 6/10 \tag{1.15}$$

respectively. Note that these satisfy $p(B = r) + p(B = b) = 1$.

Now suppose that we pick a box at random, and it turns out to be the blue box. Then the probability of selecting an apple is just the fraction of apples in the blue box which is $3/4$, and so $p(F = a|B = b) = 3/4$. In fact, we can write out all four conditional probabilities for the type of fruit, given the selected box

$$p(F = a|B = r) = 1/4 \tag{1.16}$$
$$p(F = o|B = r) = 3/4 \tag{1.17}$$
$$p(F = a|B = b) = 3/4 \tag{1.18}$$
$$p(F = o|B = b) = 1/4. \tag{1.19}$$

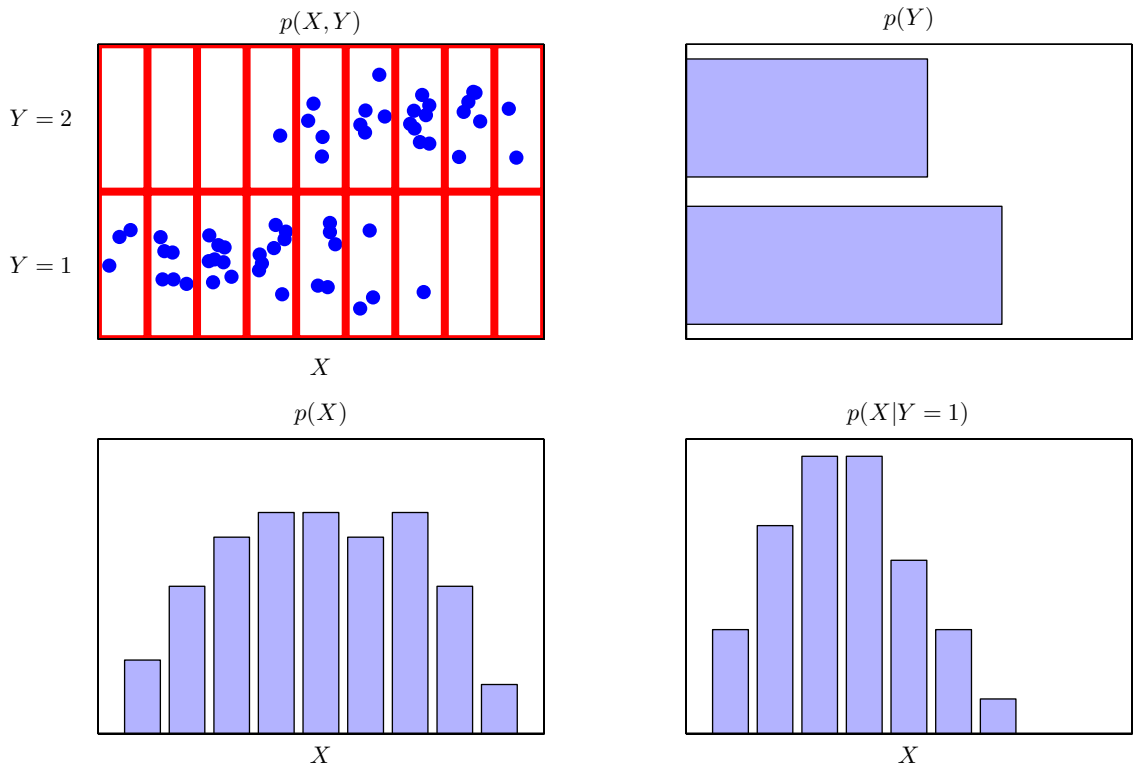


Figure 1.11 An illustration of a distribution over two variables, X, which takes 9 possible values, and Y, which takes two possible values. The top left figure shows a sample of 60 points drawn from a joint probability distribution over these variables. The remaining figures show histogram estimates of the marginal distributions $p(X)$ and $p(Y)$, as well as the conditional distribution $p(X|Y = 1)$ corresponding to the bottom row in the top left figure.

Again, note that these probabilities are normalized so that

$$p(F = a|B = r) + p(F = o|B = r) = 1 \tag{1.20}$$

and similarly

$$p(F = a|B = b) + p(F = o|B = b) = 1. \tag{1.21}$$

We can now use the sum and product rules of probability to evaluate the overall probability of choosing an apple

$$\begin{aligned} p(F = a) &= p(F = a|B = r)p(B = r) + p(F = a|B = b)p(B = b) \\ &= \frac{1}{4} \times \frac{4}{10} + \frac{3}{4} \times \frac{6}{10} = \frac{11}{20} \end{aligned} \tag{1.22}$$

from which it follows, using the sum rule, that $p(F = o) = 1 - 11/20 = 9/20$.

Suppose instead we are told that a piece of fruit has been selected and it is an orange, and we would like to know which box it came from. This requires that we evaluate the probability distribution over boxes conditioned on the identity of the fruit, whereas the probabilities in (1.16)–(1.19) give the probability distribution over the fruit conditioned on the identity of the box. We can solve the problem of reversing the conditional probability by using Bayes' theorem to give

$$p(B = r|F = o) = \frac{p(F = o|B = r)p(B = r)}{p(F = o)} = \frac{3}{4} \times \frac{4}{10} \times \frac{20}{9} = \frac{2}{3}. \tag{1.23}$$

From the sum rule, it then follows that $p(B = b|F = o) = 1 - 2/3 = 1/3$.

We can provide an important interpretation of Bayes' theorem as follows. If we had been asked which box had been chosen before being told the identity of the selected item of fruit, then the most complete information we have available is provided by the probability $p(B)$. We call this the *prior probability* because it is the probability available *before* we observe the identity of the fruit. Once we are told that the fruit is an orange, we can then use Bayes' theorem to compute the probability $p(B|F)$, which we shall call the *posterior probability* because it is the probability obtained *after* we have observed F. Note that in this example, the prior probability of selecting the red box was $4/10$, so that we were more likely to select the blue box than the red one. However, once we have observed that the piece of selected fruit is an orange, we find that the posterior probability of the red box is now $2/3$, so that it is now more likely that the box we selected was in fact the red one. This result accords with our intuition, as the proportion of oranges is much higher in the red box than it is in the blue box, and so the observation that the fruit was an orange provides significant evidence favouring the red box. In fact, the evidence is sufficiently strong that it outweighs the prior and makes it more likely that the red box was chosen rather than the blue one.

Finally, we note that if the joint distribution of two variables factorizes into the product of the marginals, so that $p(X, Y) = p(X)p(Y)$, then X and Y are said to be *independent*. From the product rule, we see that $p(Y|X) = p(Y)$, and so the conditional distribution of Y given X is indeed independent of the value of X. For instance, in our boxes of fruit example, if each box contained the same fraction of apples and oranges, then $p(F|B) = P(F)$, so that the probability of selecting, say, an apple is independent of which box is chosen.

1.2.1 Probability densities

As well as considering probabilities defined over discrete sets of events, we also wish to consider probabilities with respect to continuous variables. We shall limit ourselves to a relatively informal discussion. If the probability of a real-valued variable x falling in the interval $(x, x + \delta x)$ is given by $p(x)\delta x$ for $\delta x \to 0$, then $p(x)$ is called the *probability density* over x. This is illustrated in Figure 1.12. The probability that x will lie in an interval (a, b) is then given by

$$p(x \in (a, b)) = \int_a^b p(x)\,\mathrm{d}x. \tag{1.24}$$

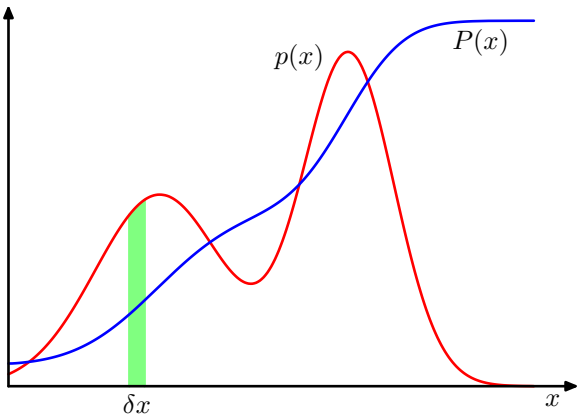


Figure 1.12 The concept of probability for discrete variables can be extended to that of a probability density $p(x)$ over a continuous variable x and is such that the probability of x lying in the interval $(x, x+\delta x)$ is given by $p(x)\delta x$ for $\delta x \to 0$. The probability density can be expressed as the derivative of a cumulative distribution function $P(x)$.

Because probabilities are nonnegative, and because the value of x must lie somewhere on the real axis, the probability density $p(x)$ must satisfy the two conditions

$$p(x) \geqslant 0 \tag{1.25}$$

$$\int_{-\infty}^{\infty} p(x)\,\mathrm{d}x = 1. \tag{1.26}$$

Under a nonlinear change of variable, a probability density transforms differently from a simple function, due to the Jacobian factor. For instance, if we consider a change of variables $x = g(y)$, then a function $f(x)$ becomes $\widetilde{f}(y) = f(g(y))$. Now consider a probability density $p_x(x)$ that corresponds to a density $p_y(y)$ with respect to the new variable y, where the suffices denote the fact that $p_x(x)$ and $p_y(y)$ are different densities. Observations falling in the range $(x, x+\delta x)$ will, for small values of δx, be transformed into the range $(y, y+\delta y)$ where $p_x(x)\delta x \simeq p_y(y)\delta y$, and hence

$$\begin{aligned} p_y(y) &= p_x(x)\left|\frac{\mathrm{d}x}{\mathrm{d}y}\right| \\ &= p_x(g(y))\,|g'(y)|\,. \end{aligned} \tag{1.27}$$

One consequence of this property is that the concept of the maximum of a probability density is dependent on the choice of variable.

Exercise 1.4

The probability that x lies in the interval $(-\infty, z)$ is given by the *cumulative distribution function* defined by

$$P(z) = \int_{-\infty}^{z} p(x)\,\mathrm{d}x \tag{1.28}$$

which satisfies $P'(x) = p(x)$, as shown in Figure 1.12.

If we have several continuous variables $x_1, \ldots, x_D$, denoted collectively by the vector $\mathbf{x}$, then we can define a joint probability density $p(\mathbf{x}) = p(x_1, \ldots, x_D)$ such

that the probability of $\mathbf{x}$ falling in an infinitesimal volume $\delta\mathbf{x}$ containing the point $\mathbf{x}$ is given by $p(\mathbf{x})\delta\mathbf{x}$. This multivariate probability density must satisfy

$$p(\mathbf{x}) \geqslant 0 \tag{1.29}$$

$$\int p(\mathbf{x})\,\mathrm{d}\mathbf{x} = 1 \tag{1.30}$$

in which the integral is taken over the whole of $\mathbf{x}$ space. We can also consider joint probability distributions over a combination of discrete and continuous variables.

Note that if x is a discrete variable, then $p(x)$ is sometimes called a *probability mass function* because it can be regarded as a set of 'probability masses' concentrated at the allowed values of x.

The sum and product rules of probability, as well as Bayes' theorem, apply equally to the case of probability densities, or to combinations of discrete and continuous variables. For instance, if x and y are two real variables, then the sum and product rules take the form

$$p(x) = \int p(x, y)\,\mathrm{d}y \tag{1.31}$$

$$p(x, y) = p(y|x)p(x). \tag{1.32}$$

A formal justification of the sum and product rules for continuous variables (Feller, 1966) requires a branch of mathematics called *measure theory* and lies outside the scope of this book. Its validity can be seen informally, however, by dividing each real variable into intervals of width Δ and considering the discrete probability distribution over these intervals. Taking the limit $\Delta \to 0$ then turns sums into integrals and gives the desired result.

1.2.2 Expectations and covariances

One of the most important operations involving probabilities is that of finding weighted averages of functions. The average value of some function $f(x)$ under a probability distribution $p(x)$ is called the *expectation* of $f(x)$ and will be denoted by $\mathbb{E}[f]$. For a discrete distribution, it is given by

$$\mathbb{E}[f] = \sum_x p(x)f(x) \tag{1.33}$$

so that the average is weighted by the relative probabilities of the different values of x. In the case of continuous variables, expectations are expressed in terms of an integration with respect to the corresponding probability density

$$\mathbb{E}[f] = \int p(x)f(x)\,\mathrm{d}x. \tag{1.34}$$

In either case, if we are given a finite number N of points drawn from the probability distribution or probability density, then the expectation can be approximated as a

finite sum over these points

$$\mathbb{E}[f] \simeq \frac{1}{N}\sum_{n=1}^{N} f(x_n). \tag{1.35}$$

We shall make extensive use of this result when we discuss sampling methods in Chapter 11. The approximation in (1.35) becomes exact in the limit $N \to \infty$.

Sometimes we will be considering expectations of functions of several variables, in which case we can use a subscript to indicate which variable is being averaged over, so that for instance

$$\mathbb{E}_x[f(x, y)] \tag{1.36}$$

denotes the average of the function $f(x, y)$ with respect to the distribution of x. Note that $\mathbb{E}_x[f(x, y)]$ will be a function of y.

We can also consider a *conditional expectation* with respect to a conditional distribution, so that

$$\mathbb{E}_x[f|y] = \sum_x p(x|y)f(x) \tag{1.37}$$

with an analogous definition for continuous variables.

The *variance* of $f(x)$ is defined by

$$\mathrm{var}[f] = \mathbb{E}\left[(f(x) - \mathbb{E}[f(x)])^2\right] \tag{1.38}$$

and provides a measure of how much variability there is in $f(x)$ around its mean value $\mathbb{E}[f(x)]$. Expanding out the square, we see that the variance can also be written in terms of the expectations of $f(x)$ and $f(x)^2$

Exercise 1.5

$$\mathrm{var}[f] = \mathbb{E}[f(x)^2] - \mathbb{E}[f(x)]^2. \tag{1.39}$$

In particular, we can consider the variance of the variable x itself, which is given by

$$\mathrm{var}[x] = \mathbb{E}[x^2] - \mathbb{E}[x]^2. \tag{1.40}$$

For two random variables x and y, the *covariance* is defined by

$$\begin{aligned}\mathrm{cov}[x, y] &= \mathbb{E}_{x,y}\left[\{x - \mathbb{E}[x]\}\{y - \mathbb{E}[y]\}\right] \\ &= \mathbb{E}_{x,y}[xy] - \mathbb{E}[x]\mathbb{E}[y]\end{aligned} \tag{1.41}$$

which expresses the extent to which x and y vary together. If x and y are independent, then their covariance vanishes.

Exercise 1.6

In the case of two vectors of random variables $\mathbf{x}$ and $\mathbf{y}$, the covariance is a matrix

$$\begin{aligned}\mathrm{cov}[\mathbf{x}, \mathbf{y}] &= \mathbb{E}_{\mathbf{x},\mathbf{y}}\left[\{\mathbf{x} - \mathbb{E}[\mathbf{x}]\}\{\mathbf{y}^{\mathrm{T}} - \mathbb{E}[\mathbf{y}^{\mathrm{T}}]\}\right] \\ &= \mathbb{E}_{\mathbf{x},\mathbf{y}}[\mathbf{x}\mathbf{y}^{\mathrm{T}}] - \mathbb{E}[\mathbf{x}]\mathbb{E}[\mathbf{y}^{\mathrm{T}}].\end{aligned} \tag{1.42}$$

If we consider the covariance of the components of a vector $\mathbf{x}$ with each other, then we use a slightly simpler notation $\mathrm{cov}[\mathbf{x}] \equiv \mathrm{cov}[\mathbf{x}, \mathbf{x}]$.

1.2.3 Bayesian probabilities

So far in this chapter, we have viewed probabilities in terms of the frequencies of random, repeatable events. We shall refer to this as the *classical* or *frequentist* interpretation of probability. Now we turn to the more general *Bayesian* view, in which probabilities provide a quantification of uncertainty.

Consider an uncertain event, for example whether the moon was once in its own orbit around the sun, or whether the Arctic ice cap will have disappeared by the end of the century. These are not events that can be repeated numerous times in order to define a notion of probability as we did earlier in the context of boxes of fruit. Nevertheless, we will generally have some idea, for example, of how quickly we think the polar ice is melting. If we now obtain fresh evidence, for instance from a new Earth observation satellite gathering novel forms of diagnostic information, we may revise our opinion on the rate of ice loss. Our assessment of such matters will affect the actions we take, for instance the extent to which we endeavour to reduce the emission of greenhouse gasses. In such circumstances, we would like to be able to quantify our expression of uncertainty and make precise revisions of uncertainty in the light of new evidence, as well as subsequently to be able to take optimal actions or decisions as a consequence. This can all be achieved through the elegant, and very general, Bayesian interpretation of probability.

The use of probability to represent uncertainty, however, is not an ad-hoc choice, but is inevitable if we are to respect common sense while making rational coherent inferences. For instance, Cox (1946) showed that if numerical values are used to represent degrees of belief, then a simple set of axioms encoding common sense properties of such beliefs leads uniquely to a set of rules for manipulating degrees of belief that are equivalent to the sum and product rules of probability. This provided the first rigorous proof that probability theory could be regarded as an extension of Boolean logic to situations involving uncertainty (Jaynes, 2003). Numerous other authors have proposed different sets of properties or axioms that such measures of uncertainty should satisfy (Ramsey, 1931; Good, 1950; Savage, 1961; deFinetti, 1970; Lindley, 1982). In each case, the resulting numerical quantities behave precisely according to the rules of probability. It is therefore natural to refer to these quantities as (Bayesian) probabilities.

In the field of pattern recognition, too, it is helpful to have a more general no-

Thomas Bayes
1701–1761

Thomas Bayes was born in Tunbridge Wells and was a clergyman as well as an amateur scientist and a mathematician. He studied logic and theology at Edinburgh University and was elected Fellow of the Royal Society in 1742. During the 18$^{\text{th}}$ century, issues regarding probability arose in connection with gambling and with the new concept of insurance. One particularly important problem concerned so-called inverse probability. A solution was proposed by Thomas Bayes in his paper 'Essay towards solving a problem in the doctrine of chances', which was published in 1764, some three years after his death, in the *Philosophical Transactions of the Royal Society*. In fact, Bayes only formulated his theory for the case of a uniform prior, and it was Pierre-Simon Laplace who independently rediscovered the theory in general form and who demonstrated its broad applicability.

tion of probability. Consider the example of polynomial curve fitting discussed in Section 1.1. It seems reasonable to apply the frequentist notion of probability to the random values of the observed variables t_n. However, we would like to address and quantify the uncertainty that surrounds the appropriate choice for the model parameters $\mathbf{w}$. We shall see that, from a Bayesian perspective, we can use the machinery of probability theory to describe the uncertainty in model parameters such as $\mathbf{w}$, or indeed in the choice of model itself.

Bayes' theorem now acquires a new significance. Recall that in the boxes of fruit example, the observation of the identity of the fruit provided relevant information that altered the probability that the chosen box was the red one. In that example, Bayes' theorem was used to convert a prior probability into a posterior probability by incorporating the evidence provided by the observed data. As we shall see in detail later, we can adopt a similar approach when making inferences about quantities such as the parameters $\mathbf{w}$ in the polynomial curve fitting example. We capture our assumptions about $\mathbf{w}$, before observing the data, in the form of a prior probability distribution $p(\mathbf{w})$. The effect of the observed data $\mathcal{D} = \{t_1, \ldots, t_N\}$ is expressed through the conditional probability $p(\mathcal{D}|\mathbf{w})$, and we shall see later, in Section 1.2.5, how this can be represented explicitly. Bayes' theorem, which takes the form

$$p(\mathbf{w}|\mathcal{D}) = \frac{p(\mathcal{D}|\mathbf{w})p(\mathbf{w})}{p(\mathcal{D})} \tag{1.43}$$

then allows us to evaluate the uncertainty in $\mathbf{w}$ *after* we have observed $\mathcal{D}$ in the form of the posterior probability $p(\mathbf{w}|\mathcal{D})$.

The quantity $p(\mathcal{D}|\mathbf{w})$ on the right-hand side of Bayes' theorem is evaluated for the observed data set $\mathcal{D}$ and can be viewed as a function of the parameter vector $\mathbf{w}$, in which case it is called the *likelihood function*. It expresses how probable the observed data set is for different settings of the parameter vector $\mathbf{w}$. Note that the likelihood is not a probability distribution over $\mathbf{w}$, and its integral with respect to $\mathbf{w}$ does not (necessarily) equal one.

Given this definition of likelihood, we can state Bayes' theorem in words

$$\text{posterior} \propto \text{likelihood} \times \text{prior} \tag{1.44}$$

where all of these quantities are viewed as functions of $\mathbf{w}$. The denominator in (1.43) is the normalization constant, which ensures that the posterior distribution on the left-hand side is a valid probability density and integrates to one. Indeed, integrating both sides of (1.43) with respect to $\mathbf{w}$, we can express the denominator in Bayes' theorem in terms of the prior distribution and the likelihood function

$$p(\mathcal{D}) = \int p(\mathcal{D}|\mathbf{w})p(\mathbf{w})\,\mathrm{d}\mathbf{w}. \tag{1.45}$$

In both the Bayesian and frequentist paradigms, the likelihood function $p(\mathcal{D}|\mathbf{w})$ plays a central role. However, the manner in which it is used is fundamentally different in the two approaches. In a frequentist setting, $\mathbf{w}$ is considered to be a fixed parameter, whose value is determined by some form of 'estimator', and error bars

on this estimate are obtained by considering the distribution of possible data sets $\mathcal{D}$. By contrast, from the Bayesian viewpoint there is only a single data set $\mathcal{D}$ (namely the one that is actually observed), and the uncertainty in the parameters is expressed through a probability distribution over $\mathbf{w}$.

A widely used frequentist estimator is *maximum likelihood*, in which $\mathbf{w}$ is set to the value that maximizes the likelihood function $p(\mathcal{D}|\mathbf{w})$. This corresponds to choosing the value of $\mathbf{w}$ for which the probability of the observed data set is maximized. In the machine learning literature, the negative log of the likelihood function is called an *error function*. Because the negative logarithm is a monotonically decreasing function, maximizing the likelihood is equivalent to minimizing the error.

One approach to determining frequentist error bars is the *bootstrap* (Efron, 1979; Hastie *et al.*, 2001), in which multiple data sets are created as follows. Suppose our original data set consists of N data points $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$. We can create a new data set $\mathbf{X}_{\mathrm{B}}$ by drawing N points at random from $\mathbf{X}$, with replacement, so that some points in $\mathbf{X}$ may be replicated in $\mathbf{X}_{\mathrm{B}}$, whereas other points in $\mathbf{X}$ may be absent from $\mathbf{X}_{\mathrm{B}}$. This process can be repeated L times to generate L data sets each of size N and each obtained by sampling from the original data set $\mathbf{X}$. The statistical accuracy of parameter estimates can then be evaluated by looking at the variability of predictions between the different bootstrap data sets.

One advantage of the Bayesian viewpoint is that the inclusion of prior knowledge arises naturally. Suppose, for instance, that a fair-looking coin is tossed three times and lands heads each time. A classical maximum likelihood estimate of the probability of landing heads would give 1, implying that all future tosses will land heads! By contrast, a Bayesian approach with any reasonable prior will lead to a much less extreme conclusion.

Section 2.1

There has been much controversy and debate associated with the relative merits of the frequentist and Bayesian paradigms, which have not been helped by the fact that there is no unique frequentist, or even Bayesian, viewpoint. For instance, one common criticism of the Bayesian approach is that the prior distribution is often selected on the basis of mathematical convenience rather than as a reflection of any prior beliefs. Even the subjective nature of the conclusions through their dependence on the choice of prior is seen by some as a source of difficulty. Reducing the dependence on the prior is one motivation for so-called *noninformative* priors. However, these lead to difficulties when comparing different models, and indeed Bayesian methods based on poor choices of prior can give poor results with high confidence. Frequentist evaluation methods offer some protection from such problems, and techniques such as cross-validation remain useful in areas such as model comparison.

Section 2.4.3

Section 1.3

This book places a strong emphasis on the Bayesian viewpoint, reflecting the huge growth in the practical importance of Bayesian methods in the past few years, while also discussing useful frequentist concepts as required.

Although the Bayesian framework has its origins in the 18^{th} century, the practical application of Bayesian methods was for a long time severely limited by the difficulties in carrying through the full Bayesian procedure, particularly the need to marginalize (sum or integrate) over the whole of parameter space, which, as we shall

see, is required in order to make predictions or to compare different models. The development of sampling methods, such as Markov chain Monte Carlo (discussed in Chapter 11) along with dramatic improvements in the speed and memory capacity of computers, opened the door to the practical use of Bayesian techniques in an impressive range of problem domains. Monte Carlo methods are very flexible and can be applied to a wide range of models. However, they are computationally intensive and have mainly been used for small-scale problems.

More recently, highly efficient deterministic approximation schemes such as variational Bayes and expectation propagation (discussed in Chapter 10) have been developed. These offer a complementary alternative to sampling methods and have allowed Bayesian techniques to be used in large-scale applications (Blei *et al.*, 2003).

1.2.4 The Gaussian distribution

We shall devote the whole of Chapter 2 to a study of various probability distributions and their key properties. It is convenient, however, to introduce here one of the most important probability distributions for continuous variables, called the *normal* or *Gaussian* distribution. We shall make extensive use of this distribution in the remainder of this chapter and indeed throughout much of the book.

For the case of a single real-valued variable x, the Gaussian distribution is defined by

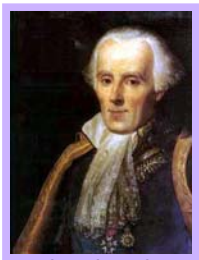
$$\mathcal{N}\left(x|\mu,\sigma^2\right) = \frac{1}{(2\pi\sigma^2)^{1/2}}\exp\left\{-\frac{1}{2\sigma^2}(x-\mu)^2\right\} \tag{1.46}$$

which is governed by two parameters: μ, called the *mean*, and σ^2, called the *variance*. The square root of the variance, given by σ, is called the *standard deviation*, and the reciprocal of the variance, written as $\beta = 1/\sigma^2$, is called the *precision*. We shall see the motivation for these terms shortly. Figure 1.13 shows a plot of the Gaussian distribution.

From the form of (1.46) we see that the Gaussian distribution satisfies

$$\mathcal{N}(x|\mu,\sigma^2) > 0. \tag{1.47}$$

Exercise 1.7 Also it is straightforward to show that the Gaussian is normalized, so that



Pierre-Simon Laplace
1749–1827

It is said that Laplace was seriously lacking in modesty and at one point declared himself to be the best mathematician in France at the time, a claim that was arguably true. As well as being prolific in mathematics, he also made numerous contributions to astronomy, including the nebular hypothesis by which the earth is thought to have formed from the condensation and cooling of a large rotating disk of gas and dust. In 1812 he published the first edition of *Théorie Analytique des Probabilités*, in which Laplace states that "probability theory is nothing but common sense reduced to calculation". This work included a discussion of the inverse probability calculation (later termed Bayes' theorem by Poincaré), which he used to solve problems in life expectancy, jurisprudence, planetary masses, triangulation, and error estimation.

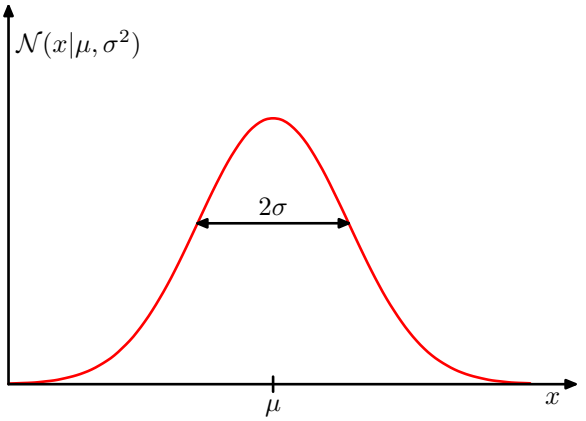


Figure 1.13 Plot of the univariate Gaussian showing the mean μ and the standard deviation σ.

$$\int_{-\infty}^{\infty} \mathcal{N}\left(x|\mu,\sigma^2\right)\,\mathrm{d}x = 1. \tag{1.48}$$

Thus (1.46) satisfies the two requirements for a valid probability density.

We can readily find expectations of functions of x under the Gaussian distribution. In particular, the average value of x is given by

Exercise 1.8

$$\mathbb{E}[x] = \int_{-\infty}^{\infty} \mathcal{N}\left(x|\mu,\sigma^2\right) x\,\mathrm{d}x = \mu. \tag{1.49}$$

Because the parameter μ represents the average value of x under the distribution, it is referred to as the mean. Similarly, for the second order moment

$$\mathbb{E}[x^2] = \int_{-\infty}^{\infty} \mathcal{N}\left(x|\mu,\sigma^2\right) x^2\,\mathrm{d}x = \mu^2 + \sigma^2. \tag{1.50}$$

From (1.49) and (1.50), it follows that the variance of x is given by

$$\mathrm{var}[x] = \mathbb{E}[x^2] - \mathbb{E}[x]^2 = \sigma^2 \tag{1.51}$$

and hence σ^2 is referred to as the variance parameter. The maximum of a distribution is known as its mode. For a Gaussian, the mode coincides with the mean.

Exercise 1.9

We are also interested in the Gaussian distribution defined over a D-dimensional vector $\mathbf{x}$ of continuous variables, which is given by

$$\mathcal{N}(\mathbf{x}|\boldsymbol{\mu},\boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{D/2}}\frac{1}{|\boldsymbol{\Sigma}|^{1/2}}\exp\left\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right\} \tag{1.52}$$

where the D-dimensional vector $\boldsymbol{\mu}$ is called the mean, the $D \times D$ matrix $\boldsymbol{\Sigma}$ is called the covariance, and $|\boldsymbol{\Sigma}|$ denotes the determinant of $\boldsymbol{\Sigma}$. We shall make use of the multivariate Gaussian distribution briefly in this chapter, although its properties will be studied in detail in Section 2.3.

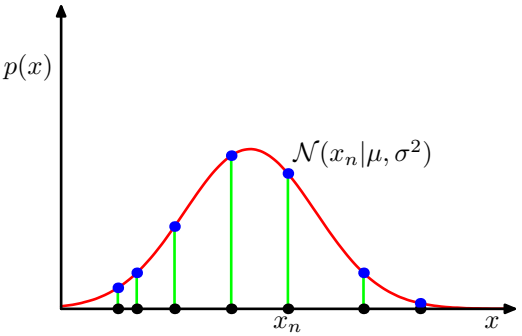


Figure 1.14 Illustration of the likelihood function for a Gaussian distribution, shown by the red curve. Here the black points denote a data set of values $\{x_n\}$, and the likelihood function given by (1.53) corresponds to the product of the blue values. Maximizing the likelihood involves adjusting the mean and variance of the Gaussian so as to maximize this product.

Now suppose that we have a data set of observations $\mathbf{x} = (x_1, \ldots, x_N)^{\mathrm{T}}$, representing N observations of the scalar variable x. Note that we are using the typeface $\mathbf{x}$ to distinguish this from a single observation of the vector-valued variable $(x_1, \ldots, x_D)^{\mathrm{T}}$, which we denote by $\mathbf{x}$. We shall suppose that the observations are drawn independently from a Gaussian distribution whose mean μ and variance σ^2 are unknown, and we would like to determine these parameters from the data set. Data points that are drawn independently from the same distribution are said to be *independent and identically distributed*, which is often abbreviated to i.i.d. We have seen that the joint probability of two independent events is given by the product of the marginal probabilities for each event separately. Because our data set $\mathbf{x}$ is i.i.d., we can therefore write the probability of the data set, given μ and σ^2, in the form

$$p(\mathbf{x}|\mu, \sigma^2) = \prod_{n=1}^{N} \mathcal{N}\left(x_n|\mu, \sigma^2\right). \tag{1.53}$$

When viewed as a function of μ and σ^2, this is the likelihood function for the Gaussian and is interpreted diagrammatically in Figure 1.14.

One common criterion for determining the parameters in a probability distribution using an observed data set is to find the parameter values that maximize the likelihood function. This might seem like a strange criterion because, from our foregoing discussion of probability theory, it would seem more natural to maximize the probability of the parameters given the data, not the probability of the data given the parameters. In fact, these two criteria are related, as we shall discuss in the context of curve fitting.

Section 1.2.5

For the moment, however, we shall determine values for the unknown parameters μ and σ^2 in the Gaussian by maximizing the likelihood function (1.53). In practice, it is more convenient to maximize the log of the likelihood function. Because the logarithm is a monotonically increasing function of its argument, maximization of the log of a function is equivalent to maximization of the function itself. Taking the log not only simplifies the subsequent mathematical analysis, but it also helps numerically because the product of a large number of small probabilities can easily underflow the numerical precision of the computer, and this is resolved by computing instead the sum of the log probabilities. From (1.46) and (1.53), the log likelihood

function can be written in the form

$$\ln p\left(\mathbf{x}|\mu,\sigma^2\right) = -\frac{1}{2\sigma^2}\sum_{n=1}^{N}(x_n-\mu)^2 - \frac{N}{2}\ln\sigma^2 - \frac{N}{2}\ln(2\pi). \tag{1.54}$$

Exercise 1.11

Maximizing (1.54) with respect to μ, we obtain the maximum likelihood solution given by

$$\mu_{\mathrm{ML}} = \frac{1}{N}\sum_{n=1}^{N}x_n \tag{1.55}$$

which is the *sample mean*, i.e., the mean of the observed values $\{x_n\}$. Similarly, maximizing (1.54) with respect to σ^2, we obtain the maximum likelihood solution for the variance in the form

$$\sigma^2_{\mathrm{ML}} = \frac{1}{N}\sum_{n=1}^{N}(x_n-\mu_{\mathrm{ML}})^2 \tag{1.56}$$

which is the *sample variance* measured with respect to the sample mean μ_{ML}. Note that we are performing a joint maximization of (1.54) with respect to μ and σ^2, but in the case of the Gaussian distribution the solution for μ decouples from that for σ^2 so that we can first evaluate (1.55) and then subsequently use this result to evaluate (1.56).

Later in this chapter, and also in subsequent chapters, we shall highlight the significant limitations of the maximum likelihood approach. Here we give an indication of the problem in the context of our solutions for the maximum likelihood parameter settings for the univariate Gaussian distribution. In particular, we shall show that the maximum likelihood approach systematically underestimates the variance of the distribution. This is an example of a phenomenon called *bias* and is related to the problem of over-fitting encountered in the context of polynomial curve fitting.

Section 1.1

We first note that the maximum likelihood solutions μ_{ML} and σ^2_{ML} are functions of the data set values $x_1,\ldots,x_N$. Consider the expectations of these quantities with respect to the data set values, which themselves come from a Gaussian distribution with parameters μ and σ^2. It is straightforward to show that

Exercise 1.12

$$\mathbb{E}[\mu_{\mathrm{ML}}] = \mu \tag{1.57}$$

$$\mathbb{E}[\sigma^2_{\mathrm{ML}}] = \left(\frac{N-1}{N}\right)\sigma^2 \tag{1.58}$$

so that on average the maximum likelihood estimate will obtain the correct mean but will underestimate the true variance by a factor $(N-1)/N$. The intuition behind this result is given by Figure 1.15.

From (1.58) it follows that the following estimate for the variance parameter is unbiased

$$\widetilde{\sigma}^2 = \frac{N}{N-1}\sigma^2_{\mathrm{ML}} = \frac{1}{N-1}\sum_{n=1}^{N}(x_n-\mu_{\mathrm{ML}})^2. \tag{1.59}$$

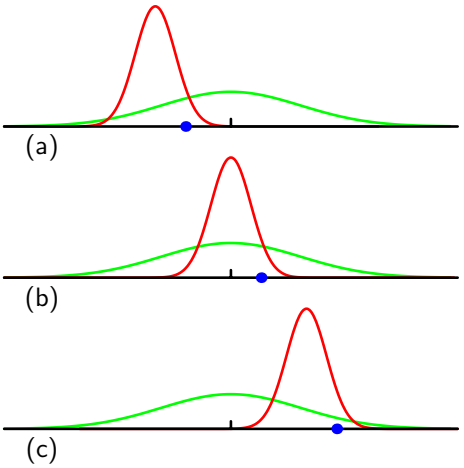


Figure 1.15 Illustration of how bias arises in using maximum likelihood to determine the variance of a Gaussian. The green curve shows the true Gaussian distribution from which data is generated, and the three red curves show the Gaussian distributions obtained by fitting to three data sets, each consisting of two data points shown in blue, using the maximum likelihood results (1.55) and (1.56). Averaged across the three data sets, the mean is correct, but the variance is systematically under-estimated because it is measured relative to the sample mean and not relative to the true mean.

In Section 10.1.3, we shall see how this result arises automatically when we adopt a Bayesian approach.

Note that the bias of the maximum likelihood solution becomes less significant as the number N of data points increases, and in the limit $N \to \infty$ the maximum likelihood solution for the variance equals the true variance of the distribution that generated the data. In practice, for anything other than small N, this bias will not prove to be a serious problem. However, throughout this book we shall be interested in more complex models with many parameters, for which the bias problems associated with maximum likelihood will be much more severe. In fact, as we shall see, the issue of bias in maximum likelihood lies at the root of the over-fitting problem that we encountered earlier in the context of polynomial curve fitting.

1.2.5 Curve fitting re-visited

Section 1.1

We have seen how the problem of polynomial curve fitting can be expressed in terms of error minimization. Here we return to the curve fitting example and view it from a probabilistic perspective, thereby gaining some insights into error functions and regularization, as well as taking us towards a full Bayesian treatment.

The goal in the curve fitting problem is to be able to make predictions for the target variable t given some new value of the input variable x on the basis of a set of training data comprising N input values $\mathbf{x} = (x_1, \ldots, x_N)^{\mathrm{T}}$ and their corresponding target values $\mathbf{t} = (t_1, \ldots, t_N)^{\mathrm{T}}$. We can express our uncertainty over the value of the target variable using a probability distribution. For this purpose, we shall assume that, given the value of x, the corresponding value of t has a Gaussian distribution with a mean equal to the value $y(x, \mathbf{w})$ of the polynomial curve given by (1.1). Thus we have

$$p(t|x, \mathbf{w}, \beta) = \mathcal{N}\left(t|y(x, \mathbf{w}), \beta^{-1}\right) \tag{1.60}$$

where, for consistency with the notation in later chapters, we have defined a precision parameter β corresponding to the inverse variance of the distribution. This is illustrated schematically in Figure 1.16.

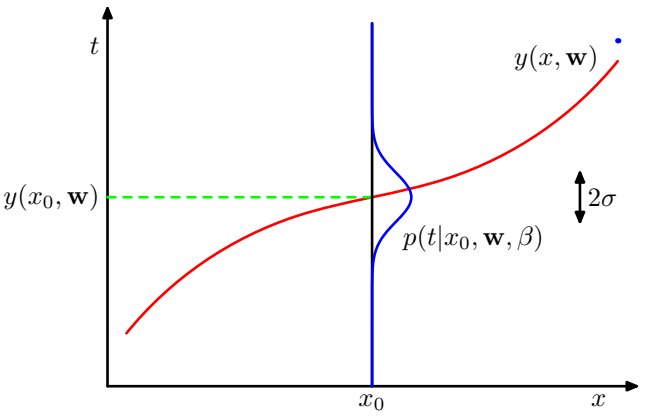


Figure 1.16 Schematic illustration of a Gaussian conditional distribution for t given x given by (1.60), in which the mean is given by the polynomial function $y(x, \mathbf{w})$, and the precision is given by the parameter β, which is related to the variance by $\beta^{-1} = \sigma^2$.

We now use the training data $\{\mathbf{x}, \mathbf{t}\}$ to determine the values of the unknown parameters $\mathbf{w}$ and β by maximum likelihood. If the data are assumed to be drawn independently from the distribution (1.60), then the likelihood function is given by

$$p(\mathbf{t}|\mathbf{x}, \mathbf{w}, \beta) = \prod_{n=1}^{N} \mathcal{N}\left(t_n | y(x_n, \mathbf{w}), \beta^{-1}\right). \tag{1.61}$$

As we did in the case of the simple Gaussian distribution earlier, it is convenient to maximize the logarithm of the likelihood function. Substituting for the form of the Gaussian distribution, given by (1.46), we obtain the log likelihood function in the form

$$\ln p(\mathbf{t}|\mathbf{x}, \mathbf{w}, \beta) = -\frac{\beta}{2} \sum_{n=1}^{N} \{y(x_n, \mathbf{w}) - t_n\}^2 + \frac{N}{2} \ln \beta - \frac{N}{2} \ln(2\pi). \tag{1.62}$$

Consider first the determination of the maximum likelihood solution for the polynomial coefficients, which will be denoted by $\mathbf{w}_{\mathrm{ML}}$. These are determined by maximizing (1.62) with respect to $\mathbf{w}$. For this purpose, we can omit the last two terms on the right-hand side of (1.62) because they do not depend on $\mathbf{w}$. Also, we note that scaling the log likelihood by a positive constant coefficient does not alter the location of the maximum with respect to $\mathbf{w}$, and so we can replace the coefficient $\beta/2$ with $1/2$. Finally, instead of maximizing the log likelihood, we can equivalently minimize the negative log likelihood. We therefore see that maximizing likelihood is equivalent, so far as determining $\mathbf{w}$ is concerned, to minimizing the *sum-of-squares error function* defined by (1.2). Thus the sum-of-squares error function has arisen as a consequence of maximizing likelihood under the assumption of a Gaussian noise distribution.

We can also use maximum likelihood to determine the precision parameter β of the Gaussian conditional distribution. Maximizing (1.62) with respect to β gives

$$\frac{1}{\beta_{\mathrm{ML}}} = \frac{1}{N} \sum_{n=1}^{N} \{y(x_n, \mathbf{w}_{\mathrm{ML}}) - t_n\}^2. \tag{1.63}$$

Again we can first determine the parameter vector $\mathbf{w}_{\mathrm{ML}}$ governing the mean and subsequently use this to find the precision β_{ML} as was the case for the simple Gaussian distribution.

Section 1.2.4

Having determined the parameters $\mathbf{w}$ and β, we can now make predictions for new values of x. Because we now have a probabilistic model, these are expressed in terms of the *predictive distribution* that gives the probability distribution over t, rather than simply a point estimate, and is obtained by substituting the maximum likelihood parameters into (1.60) to give

$$p(t|x, \mathbf{w}_{\mathrm{ML}}, \beta_{\mathrm{ML}}) = \mathcal{N}\left(t|y(x, \mathbf{w}_{\mathrm{ML}}), \beta_{\mathrm{ML}}^{-1}\right). \tag{1.64}$$

Now let us take a step towards a more Bayesian approach and introduce a prior distribution over the polynomial coefficients $\mathbf{w}$. For simplicity, let us consider a Gaussian distribution of the form

$$p(\mathbf{w}|\alpha) = \mathcal{N}(\mathbf{w}|\mathbf{0}, \alpha^{-1}\mathbf{I}) = \left(\frac{\alpha}{2\pi}\right)^{(M+1)/2} \exp\left\{-\frac{\alpha}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w}\right\} \tag{1.65}$$

where α is the precision of the distribution, and $M+1$ is the total number of elements in the vector $\mathbf{w}$ for an M^{th} order polynomial. Variables such as α, which control the distribution of model parameters, are called *hyperparameters*. Using Bayes' theorem, the posterior distribution for $\mathbf{w}$ is proportional to the product of the prior distribution and the likelihood function

$$p(\mathbf{w}|\mathbf{x}, \mathbf{t}, \alpha, \beta) \propto p(\mathbf{t}|\mathbf{x}, \mathbf{w}, \beta)p(\mathbf{w}|\alpha). \tag{1.66}$$

We can now determine $\mathbf{w}$ by finding the most probable value of $\mathbf{w}$ given the data, in other words by maximizing the posterior distribution. This technique is called *maximum posterior*, or simply *MAP*. Taking the negative logarithm of (1.66) and combining with (1.62) and (1.65), we find that the maximum of the posterior is given by the minimum of

$$\frac{\beta}{2}\sum_{n=1}^{N}\{y(x_n, \mathbf{w}) - t_n\}^2 + \frac{\alpha}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w}. \tag{1.67}$$

Thus we see that maximizing the posterior distribution is equivalent to minimizing the regularized sum-of-squares error function encountered earlier in the form (1.4), with a regularization parameter given by $\lambda = \alpha/\beta$.

1.2.6 Bayesian curve fitting

Although we have included a prior distribution $p(\mathbf{w}|\alpha)$, we are so far still making a point estimate of $\mathbf{w}$ and so this does not yet amount to a Bayesian treatment. In a fully Bayesian approach, we should consistently apply the sum and product rules of probability, which requires, as we shall see shortly, that we integrate over all values of $\mathbf{w}$. Such marginalizations lie at the heart of Bayesian methods for pattern recognition.

In the curve fitting problem, we are given the training data $\mathbf{x}$ and $\mathbf{t}$, along with a new test point x, and our goal is to predict the value of t. We therefore wish to evaluate the predictive distribution $p(t|x, \mathbf{x}, \mathbf{t})$. Here we shall assume that the parameters α and β are fixed and known in advance (in later chapters we shall discuss how such parameters can be inferred from data in a Bayesian setting).

A Bayesian treatment simply corresponds to a consistent application of the sum and product rules of probability, which allow the predictive distribution to be written in the form

$$p(t|x, \mathbf{x}, \mathbf{t}) = \int p(t|x, \mathbf{w}) p(\mathbf{w}|\mathbf{x}, \mathbf{t}) \, \mathrm{d}\mathbf{w}. \tag{1.68}$$

Here $p(t|x, \mathbf{w})$ is given by (1.60), and we have omitted the dependence on α and β to simplify the notation. Here $p(\mathbf{w}|\mathbf{x}, \mathbf{t})$ is the posterior distribution over parameters, and can be found by normalizing the right-hand side of (1.66). We shall see in Section 3.3 that, for problems such as the curve-fitting example, this posterior distribution is a Gaussian and can be evaluated analytically. Similarly, the integration in (1.68) can also be performed analytically with the result that the predictive distribution is given by a Gaussian of the form

$$p(t|x, \mathbf{x}, \mathbf{t}) = \mathcal{N}\left(t|m(x), s^2(x)\right) \tag{1.69}$$

where the mean and variance are given by

$$m(x) = \beta \boldsymbol{\phi}(x)^{\mathrm{T}} \mathbf{S} \sum_{n=1}^{N} \boldsymbol{\phi}(x_n) t_n \tag{1.70}$$

$$s^2(x) = \beta^{-1} + \boldsymbol{\phi}(x)^{\mathrm{T}} \mathbf{S} \boldsymbol{\phi}(x). \tag{1.71}$$

Here the matrix $\mathbf{S}$ is given by

$$\mathbf{S}^{-1} = \alpha \mathbf{I} + \beta \sum_{n=1}^{N} \boldsymbol{\phi}(x_n) \boldsymbol{\phi}(x)^{\mathrm{T}} \tag{1.72}$$

where $\mathbf{I}$ is the unit matrix, and we have defined the vector $\boldsymbol{\phi}(x)$ with elements $\phi_i(x) = x^i$ for $i = 0, \ldots, M$.

We see that the variance, as well as the mean, of the predictive distribution in (1.69) is dependent on x. The first term in (1.71) represents the uncertainty in the predicted value of t due to the noise on the target variables and was expressed already in the maximum likelihood predictive distribution (1.64) through β_{ML}^{-1}. However, the second term arises from the uncertainty in the parameters $\mathbf{w}$ and is a consequence of the Bayesian treatment. The predictive distribution for the synthetic sinusoidal regression problem is illustrated in Figure 1.17.

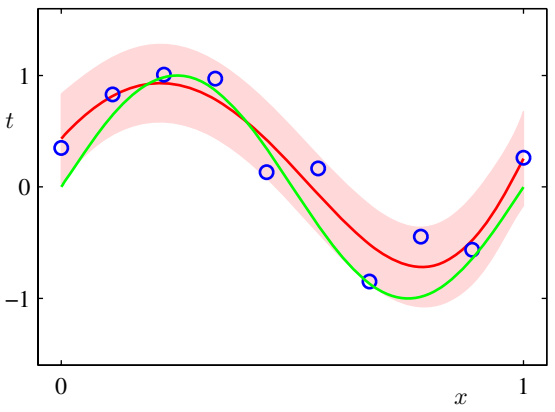


Figure 1.17 The predictive distribution resulting from a Bayesian treatment of polynomial curve fitting using an $M = 9$ polynomial, with the fixed parameters $\alpha = 5 \times 10^{-3}$ and $\beta = 11.1$ (corresponding to the known noise variance), in which the red curve denotes the mean of the predictive distribution and the red region corresponds to ± 1 standard deviation around the mean.

1.3. Model Selection

In our example of polynomial curve fitting using least squares, we saw that there was an optimal order of polynomial that gave the best generalization. The order of the polynomial controls the number of free parameters in the model and thereby governs the model complexity. With regularized least squares, the regularization coefficient λ also controls the effective complexity of the model, whereas for more complex models, such as mixture distributions or neural networks there may be multiple parameters governing complexity. In a practical application, we need to determine the values of such parameters, and the principal objective in doing so is usually to achieve the best predictive performance on new data. Furthermore, as well as finding the appropriate values for complexity parameters within a given model, we may wish to consider a range of different types of model in order to find the best one for our particular application.

We have already seen that, in the maximum likelihood approach, the performance on the training set is not a good indicator of predictive performance on unseen data due to the problem of over-fitting. If data is plentiful, then one approach is simply to use some of the available data to train a range of models, or a given model with a range of values for its complexity parameters, and then to compare them on independent data, sometimes called a *validation set*, and select the one having the best predictive performance. If the model design is iterated many times using a limited size data set, then some over-fitting to the validation data can occur and so it may be necessary to keep aside a third *test set* on which the performance of the selected model is finally evaluated.

In many applications, however, the supply of data for training and testing will be limited, and in order to build good models, we wish to use as much of the available data as possible for training. However, if the validation set is small, it will give a relatively noisy estimate of predictive performance. One solution to this dilemma is to use *cross-validation*, which is illustrated in Figure 1.18. This allows a proportion $(S-1)/S$ of the available data to be used for training while making use of all of the

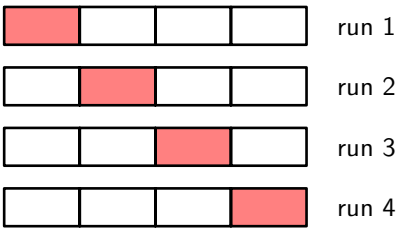


Figure 1.18 The technique of S-fold cross-validation, illustrated here for the case of $S = 4$, involves taking the available data and partitioning it into S groups (in the simplest case these are of equal size). Then $S - 1$ of the groups are used to train a set of models that are then evaluated on the remaining group. This procedure is then repeated for all S possible choices for the held-out group, indicated here by the red blocks, and the performance scores from the S runs are then averaged.

data to assess performance. When data is particularly scarce, it may be appropriate to consider the case $S = N$, where N is the total number of data points, which gives the *leave-one-out* technique.

One major drawback of cross-validation is that the number of training runs that must be performed is increased by a factor of S, and this can prove problematic for models in which the training is itself computationally expensive. A further problem with techniques such as cross-validation that use separate data to assess performance is that we might have multiple complexity parameters for a single model (for instance, there might be several regularization parameters). Exploring combinations of settings for such parameters could, in the worst case, require a number of training runs that is exponential in the number of parameters. Clearly, we need a better approach. Ideally, this should rely only on the training data and should allow multiple hyperparameters and model types to be compared in a single training run. We therefore need to find a measure of performance which depends only on the training data and which does not suffer from bias due to over-fitting.

Historically various 'information criteria' have been proposed that attempt to correct for the bias of maximum likelihood by the addition of a penalty term to compensate for the over-fitting of more complex models. For example, the *Akaike information criterion*, or AIC (Akaike, 1974), chooses the model for which the quantity

$$\ln p(\mathcal{D}|\mathbf{w}_{\mathrm{ML}}) - M \tag{1.73}$$

is largest. Here $p(\mathcal{D}|\mathbf{w}_{\mathrm{ML}})$ is the best-fit log likelihood, and M is the number of adjustable parameters in the model. A variant of this quantity, called the *Bayesian information criterion*, or *BIC*, will be discussed in Section 4.4.1. Such criteria do not take account of the uncertainty in the model parameters, however, and in practice they tend to favour overly simple models. We therefore turn in Section 3.4 to a fully Bayesian approach where we shall see how complexity penalties arise in a natural and principled way.

1.4. The Curse of Dimensionality

In the polynomial curve fitting example we had just one input variable x. For practical applications of pattern recognition, however, we will have to deal with spaces

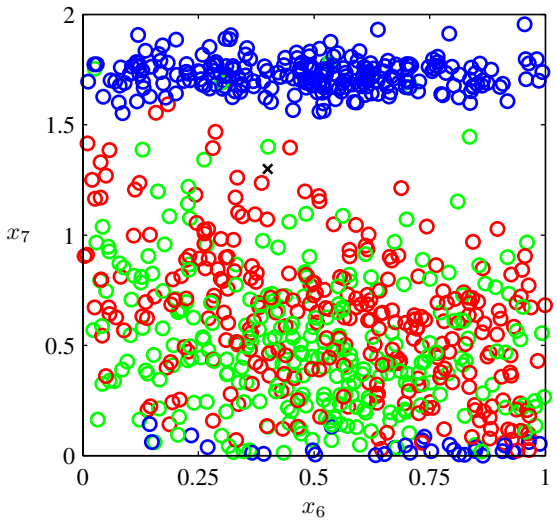


Figure 1.19 Scatter plot of the oil flow data for input variables x_6 and x_7, in which red denotes the 'homogenous' class, green denotes the 'annular' class, and blue denotes the 'laminar' class. Our goal is to classify the new test point denoted by '$\times$'.

of high dimensionality comprising many input variables. As we now discuss, this poses some serious challenges and is an important factor influencing the design of pattern recognition techniques.

In order to illustrate the problem we consider a synthetically generated data set representing measurements taken from a pipeline containing a mixture of oil, water, and gas (Bishop and James, 1993). These three materials can be present in one of three different geometrical configurations known as 'homogenous', 'annular', and 'laminar', and the fractions of the three materials can also vary. Each data point comprises a 12-dimensional input vector consisting of measurements taken with gamma ray densitometers that measure the attenuation of gamma rays passing along narrow beams through the pipe. This data set is described in detail in Appendix A. Figure 1.19 shows 100 points from this data set on a plot showing two of the measurements x_6 and x_7 (the remaining ten input values are ignored for the purposes of this illustration). Each data point is labelled according to which of the three geometrical classes it belongs to, and our goal is to use this data as a training set in order to be able to classify a new observation (x_6, x_7), such as the one denoted by the cross in Figure 1.19. We observe that the cross is surrounded by numerous red points, and so we might suppose that it belongs to the red class. However, there are also plenty of green points nearby, so we might think that it could instead belong to the green class. It seems unlikely that it belongs to the blue class. The intuition here is that the identity of the cross should be determined more strongly by nearby points from the training set and less strongly by more distant points. In fact, this intuition turns out to be reasonable and will be discussed more fully in later chapters.

How can we turn this intuition into a learning algorithm? One very simple approach would be to divide the input space into regular cells, as indicated in Figure 1.20. When we are given a test point and we wish to predict its class, we first decide which cell it belongs to, and we then find all of the training data points that

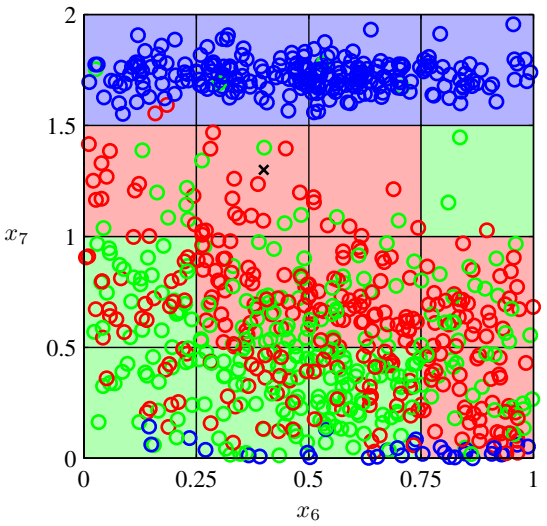


Figure 1.20 Illustration of a simple approach to the solution of a classification problem in which the input space is divided into cells and any new test point is assigned to the class that has a majority number of representatives in the same cell as the test point. As we shall see shortly, this simplistic approach has some severe shortcomings.

fall in the same cell. The identity of the test point is predicted as being the same as the class having the largest number of training points in the same cell as the test point (with ties being broken at random).

There are numerous problems with this naive approach, but one of the most severe becomes apparent when we consider its extension to problems having larger numbers of input variables, corresponding to input spaces of higher dimensionality. The origin of the problem is illustrated in Figure 1.21, which shows that, if we divide a region of a space into regular cells, then the number of such cells grows exponentially with the dimensionality of the space. The problem with an exponentially large number of cells is that we would need an exponentially large quantity of training data in order to ensure that the cells are not empty. Clearly, we have no hope of applying such a technique in a space of more than a few variables, and so we need to find a more sophisticated approach.

We can gain further insight into the problems of high-dimensional spaces by returning to the example of polynomial curve fitting and considering how we would

Section 1.1

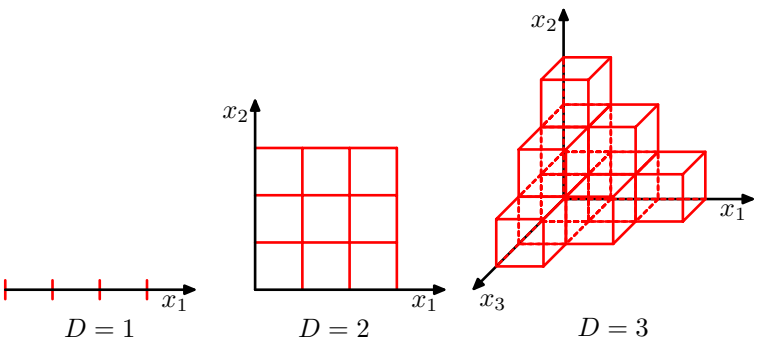


Figure 1.21 Illustration of the curse of dimensionality, showing how the number of regions of a regular grid grows exponentially with the dimensionality D of the space. For clarity, only a subset of the cubical regions are shown for $D = 3$.

extend this approach to deal with input spaces having several variables. If we have D input variables, then a general polynomial with coefficients up to order 3 would take the form

$$y(\mathbf{x}, \mathbf{w}) = w_0 + \sum_{i=1}^{D} w_i x_i + \sum_{i=1}^{D}\sum_{j=1}^{D} w_{ij} x_i x_j + \sum_{i=1}^{D}\sum_{j=1}^{D}\sum_{k=1}^{D} w_{ijk} x_i x_j x_k. \quad (1.74)$$

As D increases, so the number of independent coefficients (not all of the coefficients are independent due to interchange symmetries amongst the x variables) grows proportionally to D^3. In practice, to capture complex dependencies in the data, we may need to use a higher-order polynomial. For a polynomial of order M, the growth in
Exercise 1.16 the number of coefficients is like D^M. Although this is now a power law growth, rather than an exponential growth, it still points to the method becoming rapidly unwieldy and of limited practical utility.

Our geometrical intuitions, formed through a life spent in a space of three dimensions, can fail badly when we consider spaces of higher dimensionality. As a simple example, consider a sphere of radius $r = 1$ in a space of D dimensions, and ask what is the fraction of the volume of the sphere that lies between radius $r = 1-\epsilon$ and $r = 1$. We can evaluate this fraction by noting that the volume of a sphere of radius r in D dimensions must scale as r^D, and so we write

$$V_D(r) = K_D r^D \quad (1.75)$$

Exercise 1.18 where the constant K_D depends only on D. Thus the required fraction is given by

$$\frac{V_D(1) - V_D(1-\epsilon)}{V_D(1)} = 1 - (1-\epsilon)^D \quad (1.76)$$

which is plotted as a function of ϵ for various values of D in Figure 1.22. We see that, for large D, this fraction tends to 1 even for small values of ϵ. Thus, in spaces of high dimensionality, most of the volume of a sphere is concentrated in a thin shell near the surface!

As a further example, of direct relevance to pattern recognition, consider the behaviour of a Gaussian distribution in a high-dimensional space. If we transform from Cartesian to polar coordinates, and then integrate out the directional variables,
Exercise 1.20 we obtain an expression for the density $p(r)$ as a function of radius r from the origin. Thus $p(r)\delta r$ is the probability mass inside a thin shell of thickness δr located at radius r. This distribution is plotted, for various values of D, in Figure 1.23, and we see that for large D the probability mass of the Gaussian is concentrated in a thin shell.

The severe difficulty that can arise in spaces of many dimensions is sometimes called the *curse of dimensionality* (Bellman, 1961). In this book, we shall make extensive use of illustrative examples involving input spaces of one or two dimensions, because this makes it particularly easy to illustrate the techniques graphically. The reader should be warned, however, that not all intuitions developed in spaces of low dimensionality will generalize to spaces of many dimensions.

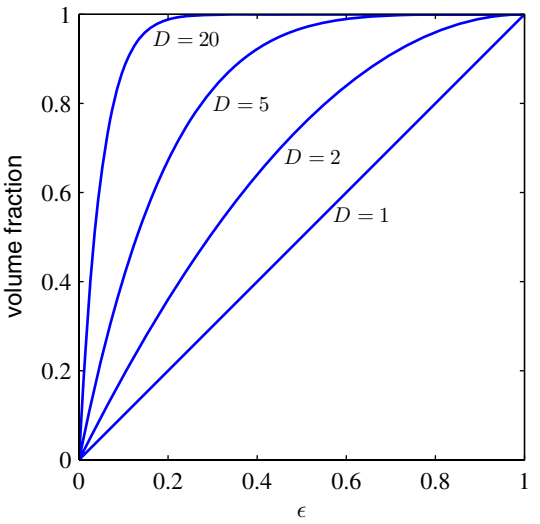


Figure 1.22 Plot of the fraction of the volume of a sphere lying in the range $r = 1-\epsilon$ to $r = 1$ for various values of the dimensionality D.

Although the curse of dimensionality certainly raises important issues for pattern recognition applications, it does not prevent us from finding effective techniques applicable to high-dimensional spaces. The reasons for this are twofold. First, real data will often be confined to a region of the space having lower effective dimensionality, and in particular the directions over which important variations in the target variables occur may be so confined. Second, real data will typically exhibit some smoothness properties (at least locally) so that for the most part small changes in the input variables will produce small changes in the target variables, and so we can exploit local interpolation-like techniques to allow us to make predictions of the target variables for new values of the input variables. Successful pattern recognition techniques exploit one or both of these properties. Consider, for example, an application in manufacturing in which images are captured of identical planar objects on a conveyor belt, in which the goal is to determine their orientation. Each image is a point

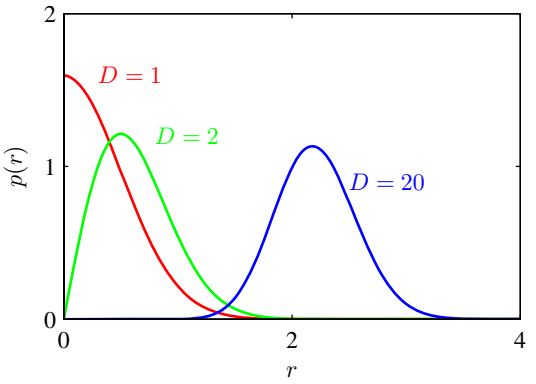


Figure 1.23 Plot of the probability density with respect to radius r of a Gaussian distribution for various values of the dimensionality D. In a high-dimensional space, most of the probability mass of a Gaussian is located within a thin shell at a specific radius.

in a high-dimensional space whose dimensionality is determined by the number of pixels. Because the objects can occur at different positions within the image and in different orientations, there are three degrees of freedom of variability between images, and a set of images will live on a three dimensional *manifold* embedded within the high-dimensional space. Due to the complex relationships between the object position or orientation and the pixel intensities, this manifold will be highly nonlinear. If the goal is to learn a model that can take an input image and output the orientation of the object irrespective of its position, then there is only one degree of freedom of variability within the manifold that is significant.

1.5. Decision Theory

We have seen in Section 1.2 how probability theory provides us with a consistent mathematical framework for quantifying and manipulating uncertainty. Here we turn to a discussion of decision theory that, when combined with probability theory, allows us to make optimal decisions in situations involving uncertainty such as those encountered in pattern recognition.

Suppose we have an input vector $\mathbf{x}$ together with a corresponding vector $\mathbf{t}$ of target variables, and our goal is to predict $\mathbf{t}$ given a new value for $\mathbf{x}$. For regression problems, $\mathbf{t}$ will comprise continuous variables, whereas for classification problems $\mathbf{t}$ will represent class labels. The joint probability distribution $p(\mathbf{x}, \mathbf{t})$ provides a complete summary of the uncertainty associated with these variables. Determination of $p(\mathbf{x}, \mathbf{t})$ from a set of training data is an example of *inference* and is typically a very difficult problem whose solution forms the subject of much of this book. In a practical application, however, we must often make a specific prediction for the value of $\mathbf{t}$, or more generally take a specific action based on our understanding of the values $\mathbf{t}$ is likely to take, and this aspect is the subject of decision theory.

Consider, for example, a medical diagnosis problem in which we have taken an X-ray image of a patient, and we wish to determine whether the patient has cancer or not. In this case, the input vector $\mathbf{x}$ is the set of pixel intensities in the image, and output variable t will represent the presence of cancer, which we denote by the class $\mathcal{C}_1$, or the absence of cancer, which we denote by the class $\mathcal{C}_2$. We might, for instance, choose t to be a binary variable such that $t = 0$ corresponds to class $\mathcal{C}_1$ and $t = 1$ corresponds to class $\mathcal{C}_2$. We shall see later that this choice of label values is particularly convenient for probabilistic models. The general inference problem then involves determining the joint distribution $p(\mathbf{x}, \mathcal{C}_k)$, or equivalently $p(\mathbf{x}, t)$, which gives us the most complete probabilistic description of the situation. Although this can be a very useful and informative quantity, in the end we must decide either to give treatment to the patient or not, and we would like this choice to be optimal in some appropriate sense (Duda and Hart, 1973). This is the *decision* step, and it is the subject of decision theory to tell us how to make optimal decisions given the appropriate probabilities. We shall see that the decision stage is generally very simple, even trivial, once we have solved the inference problem.

Here we give an introduction to the key ideas of decision theory as required for

the rest of the book. Further background, as well as more detailed accounts, can be found in Berger (1985) and Bather (2000).

Before giving a more detailed analysis, let us first consider informally how we might expect probabilities to play a role in making decisions. When we obtain the X-ray image $\mathbf{x}$ for a new patient, our goal is to decide which of the two classes to assign to the image. We are interested in the probabilities of the two classes given the image, which are given by $p(\mathcal{C}_k|\mathbf{x})$. Using Bayes' theorem, these probabilities can be expressed in the form

$$p(\mathcal{C}_k|\mathbf{x}) = \frac{p(\mathbf{x}|\mathcal{C}_k)p(\mathcal{C}_k)}{p(\mathbf{x})}. \tag{1.77}$$

Note that any of the quantities appearing in Bayes' theorem can be obtained from the joint distribution $p(\mathbf{x}, \mathcal{C}_k)$ by either marginalizing or conditioning with respect to the appropriate variables. We can now interpret $p(\mathcal{C}_k)$ as the prior probability for the class $\mathcal{C}_k$, and $p(\mathcal{C}_k|\mathbf{x})$ as the corresponding posterior probability. Thus $p(\mathcal{C}_1)$ represents the probability that a person has cancer, before we take the X-ray measurement. Similarly, $p(\mathcal{C}_1|\mathbf{x})$ is the corresponding probability, revised using Bayes' theorem in light of the information contained in the X-ray. If our aim is to minimize the chance of assigning $\mathbf{x}$ to the wrong class, then intuitively we would choose the class having the higher posterior probability. We now show that this intuition is correct, and we also discuss more general criteria for making decisions.

1.5.1 Minimizing the misclassification rate

Suppose that our goal is simply to make as few misclassifications as possible. We need a rule that assigns each value of $\mathbf{x}$ to one of the available classes. Such a rule will divide the input space into regions $\mathcal{R}_k$ called *decision regions*, one for each class, such that all points in $\mathcal{R}_k$ are assigned to class $\mathcal{C}_k$. The boundaries between decision regions are called *decision boundaries* or *decision surfaces*. Note that each decision region need not be contiguous but could comprise some number of disjoint regions. We shall encounter examples of decision boundaries and decision regions in later chapters. In order to find the optimal decision rule, consider first of all the case of two classes, as in the cancer problem for instance. A mistake occurs when an input vector belonging to class $\mathcal{C}_1$ is assigned to class $\mathcal{C}_2$ or vice versa. The probability of this occurring is given by

$$\begin{aligned} p(\text{mistake}) &= p(\mathbf{x} \in \mathcal{R}_1, \mathcal{C}_2) + p(\mathbf{x} \in \mathcal{R}_2, \mathcal{C}_1) \\ &= \int_{\mathcal{R}_1} p(\mathbf{x}, \mathcal{C}_2)\,\mathrm{d}\mathbf{x} + \int_{\mathcal{R}_2} p(\mathbf{x}, \mathcal{C}_1)\,\mathrm{d}\mathbf{x}. \end{aligned} \tag{1.78}$$

We are free to choose the decision rule that assigns each point $\mathbf{x}$ to one of the two classes. Clearly to minimize $p(\text{mistake})$ we should arrange that each $\mathbf{x}$ is assigned to whichever class has the smaller value of the integrand in (1.78). Thus, if $p(\mathbf{x}, \mathcal{C}_1) > p(\mathbf{x}, \mathcal{C}_2)$ for a given value of $\mathbf{x}$, then we should assign that $\mathbf{x}$ to class $\mathcal{C}_1$. From the product rule of probability we have $p(\mathbf{x}, \mathcal{C}_k) = p(\mathcal{C}_k|\mathbf{x})p(\mathbf{x})$. Because the factor $p(\mathbf{x})$ is common to both terms, we can restate this result as saying that the minimum

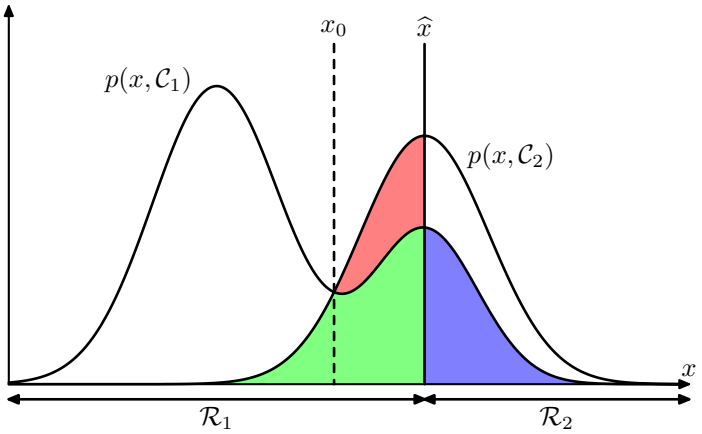


Figure 1.24 Schematic illustration of the joint probabilities $p(x, \mathcal{C}_k)$ for each of two classes plotted against x, together with the decision boundary $x = \widehat{x}$. Values of $x \geqslant \widehat{x}$ are classified as class $\mathcal{C}_2$ and hence belong to decision region $\mathcal{R}_2$, whereas points $x < \widehat{x}$ are classified as $\mathcal{C}_1$ and belong to $\mathcal{R}_1$. Errors arise from the blue, green, and red regions, so that for $x < \widehat{x}$ the errors are due to points from class $\mathcal{C}_2$ being misclassified as $\mathcal{C}_1$ (represented by the sum of the red and green regions), and conversely for points in the region $x \geqslant \widehat{x}$ the errors are due to points from class $\mathcal{C}_1$ being misclassified as $\mathcal{C}_2$ (represented by the blue region). As we vary the location $\widehat{x}$ of the decision boundary, the combined areas of the blue and green regions remains constant, whereas the size of the red region varies. The optimal choice for $\widehat{x}$ is where the curves for $p(x, \mathcal{C}_1)$ and $p(x, \mathcal{C}_2)$ cross, corresponding to $\widehat{x} = x_0$, because in this case the red region disappears. This is equivalent to the minimum misclassification rate decision rule, which assigns each value of x to the class having the higher posterior probability $p(\mathcal{C}_k|x)$.

probability of making a mistake is obtained if each value of $\mathbf{x}$ is assigned to the class for which the posterior probability $p(\mathcal{C}_k|\mathbf{x})$ is largest. This result is illustrated for two classes, and a single input variable x, in Figure 1.24.

For the more general case of K classes, it is slightly easier to maximize the probability of being correct, which is given by

$$\begin{aligned} p(\text{correct}) &= \sum_{k=1}^{K} p(\mathbf{x} \in \mathcal{R}_k, \mathcal{C}_k) \\ &= \sum_{k=1}^{K} \int_{\mathcal{R}_k} p(\mathbf{x}, \mathcal{C}_k)\, \mathrm{d}\mathbf{x} \end{aligned} \tag{1.79}$$

which is maximized when the regions $\mathcal{R}_k$ are chosen such that each $\mathbf{x}$ is assigned to the class for which $p(\mathbf{x}, \mathcal{C}_k)$ is largest. Again, using the product rule $p(\mathbf{x}, \mathcal{C}_k) = p(\mathcal{C}_k|\mathbf{x})p(\mathbf{x})$, and noting that the factor of $p(\mathbf{x})$ is common to all terms, we see that each $\mathbf{x}$ should be assigned to the class having the largest posterior probability $p(\mathcal{C}_k|\mathbf{x})$.

$$\begin{array}{c} \\ \text{cancer} \\ \text{normal} \end{array} \begin{array}{c} \begin{array}{cc} \text{cancer} & \text{normal} \end{array} \\ \begin{pmatrix} 0 & 1000 \\ 1 & 0 \end{pmatrix} \end{array}$$

Figure 1.25 An example of a loss matrix with elements L_{kj} for the cancer treatment problem. The rows correspond to the true class, whereas the columns correspond to the assignment of class made by our decision criterion.

1.5.2 Minimizing the expected loss

For many applications, our objective will be more complex than simply minimizing the number of misclassifications. Let us consider again the medical diagnosis problem. We note that, if a patient who does not have cancer is incorrectly diagnosed as having cancer, the consequences may be some patient distress plus the need for further investigations. Conversely, if a patient with cancer is diagnosed as healthy, the result may be premature death due to lack of treatment. Thus the consequences of these two types of mistake can be dramatically different. It would clearly be better to make fewer mistakes of the second kind, even if this was at the expense of making more mistakes of the first kind.

We can formalize such issues through the introduction of a *loss function*, also called a *cost function*, which is a single, overall measure of loss incurred in taking any of the available decisions or actions. Our goal is then to minimize the total loss incurred. Note that some authors consider instead a *utility function*, whose value they aim to maximize. These are equivalent concepts if we take the utility to be simply the negative of the loss, and throughout this text we shall use the loss function convention. Suppose that, for a new value of $\mathbf{x}$, the true class is $\mathcal{C}_k$ and that we assign $\mathbf{x}$ to class $\mathcal{C}_j$ (where j may or may not be equal to k). In so doing, we incur some level of loss that we denote by L_{kj}, which we can view as the k, j element of a *loss matrix*. For instance, in our cancer example, we might have a loss matrix of the form shown in Figure 1.25. This particular loss matrix says that there is no loss incurred if the correct decision is made, there is a loss of 1 if a healthy patient is diagnosed as having cancer, whereas there is a loss of 1000 if a patient having cancer is diagnosed as healthy.

The optimal solution is the one which minimizes the loss function. However, the loss function depends on the true class, which is unknown. For a given input vector $\mathbf{x}$, our uncertainty in the true class is expressed through the joint probability distribution $p(\mathbf{x}, \mathcal{C}_k)$ and so we seek instead to minimize the average loss, where the average is computed with respect to this distribution, which is given by

$$\mathbb{E}[L] = \sum_k \sum_j \int_{\mathcal{R}_j} L_{kj} p(\mathbf{x}, \mathcal{C}_k) \, \mathrm{d}\mathbf{x}. \tag{1.80}$$

Each $\mathbf{x}$ can be assigned independently to one of the decision regions $\mathcal{R}_j$. Our goal is to choose the regions $\mathcal{R}_j$ in order to minimize the expected loss (1.80), which implies that for each $\mathbf{x}$ we should minimize $\sum_k L_{kj} p(\mathbf{x}, \mathcal{C}_k)$. As before, we can use the product rule $p(\mathbf{x}, \mathcal{C}_k) = p(\mathcal{C}_k|\mathbf{x})p(\mathbf{x})$ to eliminate the common factor of $p(\mathbf{x})$. Thus the decision rule that minimizes the expected loss is the one that assigns each

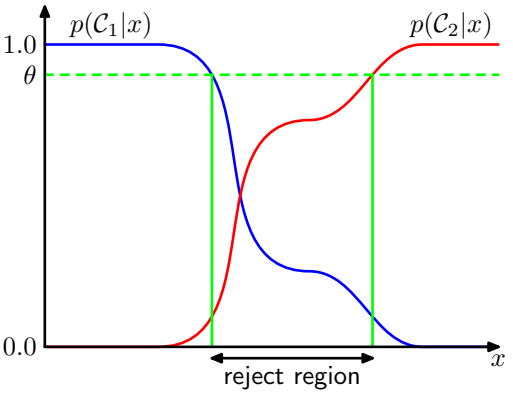


Figure 1.26 Illustration of the reject option. Inputs x such that the larger of the two posterior probabilities is less than or equal to some threshold θ will be rejected.

new $\mathbf{x}$ to the class j for which the quantity

$$\sum_k L_{kj} p(\mathcal{C}_k|\mathbf{x}) \tag{1.81}$$

is a minimum. This is clearly trivial to do, once we know the posterior class probabilities $p(\mathcal{C}_k|\mathbf{x})$.

1.5.3 The reject option

We have seen that classification errors arise from the regions of input space where the largest of the posterior probabilities $p(\mathcal{C}_k|\mathbf{x})$ is significantly less than unity, or equivalently where the joint distributions $p(\mathbf{x}, \mathcal{C}_k)$ have comparable values. These are the regions where we are relatively uncertain about class membership. In some applications, it will be appropriate to avoid making decisions on the difficult cases in anticipation of a lower error rate on those examples for which a classification decision is made. This is known as the *reject option*. For example, in our hypothetical medical illustration, it may be appropriate to use an automatic system to classify those X-ray images for which there is little doubt as to the correct class, while leaving a human expert to classify the more ambiguous cases. We can achieve this by introducing a threshold θ and rejecting those inputs $\mathbf{x}$ for which the largest of the posterior probabilities $p(\mathcal{C}_k|\mathbf{x})$ is less than or equal to θ. This is illustrated for the case of two classes, and a single continuous input variable x, in Figure 1.26. Note that setting $\theta = 1$ will ensure that all examples are rejected, whereas if there are K classes then setting $\theta < 1/K$ will ensure that no examples are rejected. Thus the fraction of examples that get rejected is controlled by the value of θ.

We can easily extend the reject criterion to minimize the expected loss, when a loss matrix is given, taking account of the loss incurred when a reject decision is made.

Exercise 1.24

1.5.4 Inference and decision

We have broken the classification problem down into two separate stages, the *inference stage* in which we use training data to learn a model for $p(\mathcal{C}_k|\mathbf{x})$, and the

subsequent *decision* stage in which we use these posterior probabilities to make optimal class assignments. An alternative possibility would be to solve both problems together and simply learn a function that maps inputs $\mathbf{x}$ directly into decisions. Such a function is called a *discriminant function*.

In fact, we can identify three distinct approaches to solving decision problems, all of which have been used in practical applications. These are given, in decreasing order of complexity, by:

(a) First solve the inference problem of determining the class-conditional densities $p(\mathbf{x}|\mathcal{C}_k)$ for each class $\mathcal{C}_k$ individually. Also separately infer the prior class probabilities $p(\mathcal{C}_k)$. Then use Bayes' theorem in the form

$$p(\mathcal{C}_k|\mathbf{x}) = \frac{p(\mathbf{x}|\mathcal{C}_k)p(\mathcal{C}_k)}{p(\mathbf{x})} \tag{1.82}$$

to find the posterior class probabilities $p(\mathcal{C}_k|\mathbf{x})$. As usual, the denominator in Bayes' theorem can be found in terms of the quantities appearing in the numerator, because

$$p(\mathbf{x}) = \sum_k p(\mathbf{x}|\mathcal{C}_k)p(\mathcal{C}_k). \tag{1.83}$$

Equivalently, we can model the joint distribution $p(\mathbf{x}, \mathcal{C}_k)$ directly and then normalize to obtain the posterior probabilities. Having found the posterior probabilities, we use decision theory to determine class membership for each new input $\mathbf{x}$. Approaches that explicitly or implicitly model the distribution of inputs as well as outputs are known as *generative models*, because by sampling from them it is possible to generate synthetic data points in the input space.

(b) First solve the inference problem of determining the posterior class probabilities $p(\mathcal{C}_k|\mathbf{x})$, and then subsequently use decision theory to assign each new $\mathbf{x}$ to one of the classes. Approaches that model the posterior probabilities directly are called *discriminative models*.

(c) Find a function $f(\mathbf{x})$, called a discriminant function, which maps each input $\mathbf{x}$ directly onto a class label. For instance, in the case of two-class problems, $f(\cdot)$ might be binary valued and such that $f = 0$ represents class $\mathcal{C}_1$ and $f = 1$ represents class $\mathcal{C}_2$. In this case, probabilities play no role.

Let us consider the relative merits of these three alternatives. Approach (a) is the most demanding because it involves finding the joint distribution over both $\mathbf{x}$ and $\mathcal{C}_k$. For many applications, $\mathbf{x}$ will have high dimensionality, and consequently we may need a large training set in order to be able to determine the class-conditional densities to reasonable accuracy. Note that the class priors $p(\mathcal{C}_k)$ can often be estimated simply from the fractions of the training set data points in each of the classes. One advantage of approach (a), however, is that it also allows the marginal density of data $p(\mathbf{x})$ to be determined from (1.83). This can be useful for detecting new data points that have low probability under the model and for which the predictions may

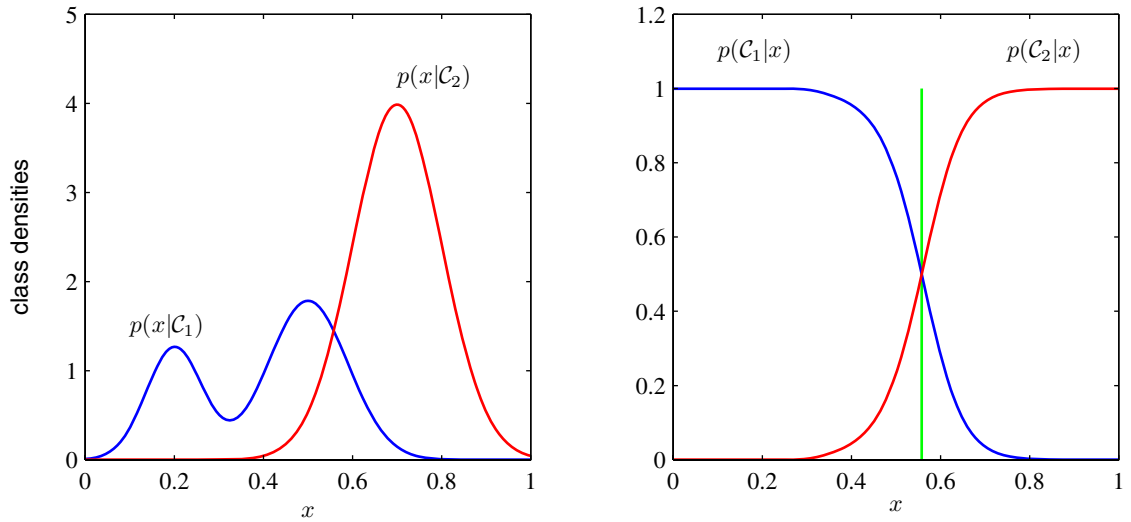


Figure 1.27 Example of the class-conditional densities for two classes having a single input variable x (left plot) together with the corresponding posterior probabilities (right plot). Note that the left-hand mode of the class-conditional density $p(\mathbf{x}|\mathcal{C}_1)$, shown in blue on the left plot, has no effect on the posterior probabilities. The vertical green line in the right plot shows the decision boundary in x that gives the minimum misclassification rate.

be of low accuracy, which is known as *outlier detection* or *novelty detection* (Bishop, 1994; Tarassenko, 1995).

However, if we only wish to make classification decisions, then it can be wasteful of computational resources, and excessively demanding of data, to find the joint distribution $p(\mathbf{x}, \mathcal{C}_k)$ when in fact we only really need the posterior probabilities $p(\mathcal{C}_k|\mathbf{x})$, which can be obtained directly through approach (b). Indeed, the class-conditional densities may contain a lot of structure that has little effect on the posterior probabilities, as illustrated in Figure 1.27. There has been much interest in exploring the relative merits of generative and discriminative approaches to machine learning, and in finding ways to combine them (Jebara, 2004; Lasserre *et al.*, 2006).

An even simpler approach is (c) in which we use the training data to find a discriminant function $f(\mathbf{x})$ that maps each $\mathbf{x}$ directly onto a class label, thereby combining the inference and decision stages into a single learning problem. In the example of Figure 1.27, this would correspond to finding the value of x shown by the vertical green line, because this is the decision boundary giving the minimum probability of misclassification.

With option (c), however, we no longer have access to the posterior probabilities $p(\mathcal{C}_k|\mathbf{x})$. There are many powerful reasons for wanting to compute the posterior probabilities, even if we subsequently use them to make decisions. These include:

Minimizing risk. Consider a problem in which the elements of the loss matrix are subjected to revision from time to time (such as might occur in a financial

application). If we know the posterior probabilities, we can trivially revise the minimum risk decision criterion by modifying (1.81) appropriately. If we have only a discriminant function, then any change to the loss matrix would require that we return to the training data and solve the classification problem afresh.

Reject option. Posterior probabilities allow us to determine a rejection criterion that will minimize the misclassification rate, or more generally the expected loss, for a given fraction of rejected data points.

Compensating for class priors. Consider our medical X-ray problem again, and suppose that we have collected a large number of X-ray images from the general population for use as training data in order to build an automated screening system. Because cancer is rare amongst the general population, we might find that, say, only 1 in every 1,000 examples corresponds to the presence of cancer. If we used such a data set to train an adaptive model, we could run into severe difficulties due to the small proportion of the cancer class. For instance, a classifier that assigned every point to the normal class would already achieve 99.9% accuracy and it would be difficult to avoid this trivial solution. Also, even a large data set will contain very few examples of X-ray images corresponding to cancer, and so the learning algorithm will not be exposed to a broad range of examples of such images and hence is not likely to generalize well. A balanced data set in which we have selected equal numbers of examples from each of the classes would allow us to find a more accurate model. However, we then have to compensate for the effects of our modifications to the training data. Suppose we have used such a modified data set and found models for the posterior probabilities. From Bayes' theorem (1.82), we see that the posterior probabilities are proportional to the prior probabilities, which we can interpret as the fractions of points in each class. We can therefore simply take the posterior probabilities obtained from our artificially balanced data set and first divide by the class fractions in that data set and then multiply by the class fractions in the population to which we wish to apply the model. Finally, we need to normalize to ensure that the new posterior probabilities sum to one. Note that this procedure cannot be applied if we have learned a discriminant function directly instead of determining posterior probabilities.

Combining models. For complex applications, we may wish to break the problem into a number of smaller subproblems each of which can be tackled by a separate module. For example, in our hypothetical medical diagnosis problem, we may have information available from, say, blood tests as well as X-ray images. Rather than combine all of this heterogeneous information into one huge input space, it may be more effective to build one system to interpret the X-ray images and a different one to interpret the blood data. As long as each of the two models gives posterior probabilities for the classes, we can combine the outputs systematically using the rules of probability. One simple way to do this is to assume that, for each class separately, the distributions of inputs for the X-ray images, denoted by $\mathbf{x}_{\mathrm{I}}$, and the blood data, denoted by $\mathbf{x}_{\mathrm{B}}$, are

independent, so that

$$p(\mathbf{x}_{\mathrm{I}}, \mathbf{x}_{\mathrm{B}}|\mathcal{C}_k) = p(\mathbf{x}_{\mathrm{I}}|\mathcal{C}_k)p(\mathbf{x}_{\mathrm{B}}|\mathcal{C}_k). \tag{1.84}$$

Section 8.2

This is an example of *conditional independence* property, because the independence holds when the distribution is conditioned on the class $\mathcal{C}_k$. The posterior probability, given both the X-ray and blood data, is then given by

$$\begin{aligned} p(\mathcal{C}_k|\mathbf{x}_{\mathrm{I}}, \mathbf{x}_{\mathrm{B}}) &\propto p(\mathbf{x}_{\mathrm{I}}, \mathbf{x}_{\mathrm{B}}|\mathcal{C}_k)p(\mathcal{C}_k) \\ &\propto p(\mathbf{x}_{\mathrm{I}}|\mathcal{C}_k)p(\mathbf{x}_{\mathrm{B}}|\mathcal{C}_k)p(\mathcal{C}_k) \\ &\propto \frac{p(\mathcal{C}_k|\mathbf{x}_{\mathrm{I}})p(\mathcal{C}_k|\mathbf{x}_{\mathrm{B}})}{p(\mathcal{C}_k)} \end{aligned} \tag{1.85}$$

Thus we need the class prior probabilities $p(\mathcal{C}_k)$, which we can easily estimate from the fractions of data points in each class, and then we need to normalize the resulting posterior probabilities so they sum to one. The particular conditional independence assumption (1.84) is an example of the *naive Bayes model*. Note that the joint marginal distribution $p(\mathbf{x}_{\mathrm{I}}, \mathbf{x}_{\mathrm{B}})$ will typically not factorize under this model. We shall see in later chapters how to construct models for combining data that do not require the conditional independence assumption (1.84).

Section 8.2.2

1.5.5 Loss functions for regression

So far, we have discussed decision theory in the context of classification problems. We now turn to the case of regression problems, such as the curve fitting example discussed earlier. The decision stage consists of choosing a specific estimate $y(\mathbf{x})$ of the value of t for each input $\mathbf{x}$. Suppose that in doing so, we incur a loss $L(t, y(\mathbf{x}))$. The average, or expected, loss is then given by

Section 1.1

$$\mathbb{E}[L] = \iint L(t, y(\mathbf{x}))p(\mathbf{x}, t)\,\mathrm{d}\mathbf{x}\,\mathrm{d}t. \tag{1.86}$$

A common choice of loss function in regression problems is the squared loss given by $L(t, y(\mathbf{x})) = \{y(\mathbf{x}) - t\}^2$. In this case, the expected loss can be written

$$\mathbb{E}[L] = \iint \{y(\mathbf{x}) - t\}^2 p(\mathbf{x}, t)\,\mathrm{d}\mathbf{x}\,\mathrm{d}t. \tag{1.87}$$

Our goal is to choose $y(\mathbf{x})$ so as to minimize $\mathbb{E}[L]$. If we assume a completely flexible function $y(\mathbf{x})$, we can do this formally using the calculus of variations to give

Appendix D

$$\frac{\delta \mathbb{E}[L]}{\delta y(\mathbf{x})} = 2\int \{y(\mathbf{x}) - t\}p(\mathbf{x}, t)\,\mathrm{d}t = 0. \tag{1.88}$$

Solving for $y(\mathbf{x})$, and using the sum and product rules of probability, we obtain

$$y(\mathbf{x}) = \frac{\int t p(\mathbf{x}, t)\,\mathrm{d}t}{p(\mathbf{x})} = \int t p(t|\mathbf{x})\,\mathrm{d}t = \mathbb{E}_t[t|\mathbf{x}] \tag{1.89}$$

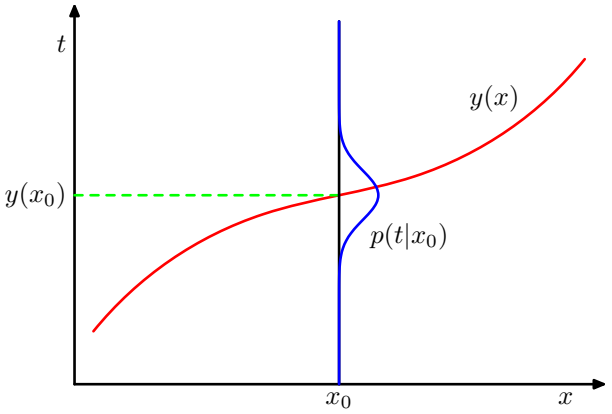


Figure 1.28 The regression function $y(x)$, which minimizes the expected squared loss, is given by the mean of the conditional distribution $p(t|x)$.

which is the conditional average of t conditioned on $\mathbf{x}$ and is known as the *regression function*. This result is illustrated in Figure 1.28. It can readily be extended to multiple target variables represented by the vector $\mathbf{t}$, in which case the optimal solution is the conditional average $\mathbf{y}(\mathbf{x}) = \mathbb{E}_t[\mathbf{t}|\mathbf{x}]$.

Exercise 1.25

We can also derive this result in a slightly different way, which will also shed light on the nature of the regression problem. Armed with the knowledge that the optimal solution is the conditional expectation, we can expand the square term as follows

$$\begin{aligned}\{y(\mathbf{x}) - t\}^2 &= \{y(\mathbf{x}) - \mathbb{E}[t|\mathbf{x}] + \mathbb{E}[t|\mathbf{x}] - t\}^2 \\ &= \{y(\mathbf{x}) - \mathbb{E}[t|\mathbf{x}]\}^2 + 2\{y(\mathbf{x}) - \mathbb{E}[t|\mathbf{x}]\}\{\mathbb{E}[t|\mathbf{x}] - t\} + \{\mathbb{E}[t|\mathbf{x}] - t\}^2\end{aligned}$$

where, to keep the notation uncluttered, we use $\mathbb{E}[t|\mathbf{x}]$ to denote $\mathbb{E}_t[t|\mathbf{x}]$. Substituting into the loss function and performing the integral over t, we see that the cross-term vanishes and we obtain an expression for the loss function in the form

$$\mathbb{E}[L] = \int \{y(\mathbf{x}) - \mathbb{E}[t|\mathbf{x}]\}^2 p(\mathbf{x})\,\mathrm{d}\mathbf{x} + \int \{\mathbb{E}[t|\mathbf{x}] - t\}^2 p(\mathbf{x})\,\mathrm{d}\mathbf{x}. \tag{1.90}$$

The function $y(\mathbf{x})$ we seek to determine enters only in the first term, which will be minimized when $y(\mathbf{x})$ is equal to $\mathbb{E}[t|\mathbf{x}]$, in which case this term will vanish. This is simply the result that we derived previously and that shows that the optimal least squares predictor is given by the conditional mean. The second term is the variance of the distribution of t, averaged over $\mathbf{x}$. It represents the intrinsic variability of the target data and can be regarded as noise. Because it is independent of $y(\mathbf{x})$, it represents the irreducible minimum value of the loss function.

As with the classification problem, we can either determine the appropriate probabilities and then use these to make optimal decisions, or we can build models that make decisions directly. Indeed, we can identify three distinct approaches to solving regression problems given, in order of decreasing complexity, by:

(a) First solve the inference problem of determining the joint density $p(\mathbf{x}, t)$. Then normalize to find the conditional density $p(t|\mathbf{x})$, and finally marginalize to find the conditional mean given by (1.89).

(b) First solve the inference problem of determining the conditional density $p(t|\mathbf{x})$, and then subsequently marginalize to find the conditional mean given by (1.89).

(c) Find a regression function $y(\mathbf{x})$ directly from the training data.

The relative merits of these three approaches follow the same lines as for classification problems above.

The squared loss is not the only possible choice of loss function for regression. Indeed, there are situations in which squared loss can lead to very poor results and where we need to develop more sophisticated approaches. An important example concerns situations in which the conditional distribution $p(t|\mathbf{x})$ is multimodal, as often arises in the solution of inverse problems. Here we consider briefly one simple generalization of the squared loss, called the *Minkowski* loss, whose expectation is given by

Section 5.6

$$\mathbb{E}[L_q] = \iint |y(\mathbf{x}) - t|^q p(\mathbf{x}, t)\, \mathrm{d}\mathbf{x}\, \mathrm{d}t \tag{1.91}$$

which reduces to the expected squared loss for $q = 2$. The function $|y - t|^q$ is plotted against $y - t$ for various values of q in Figure 1.29. The minimum of $\mathbb{E}[L_q]$ is given by the conditional mean for $q = 2$, the conditional median for $q = 1$, and the conditional mode for $q \to 0$.

Exercise 1.27

1.6. Information Theory

In this chapter, we have discussed a variety of concepts from probability theory and decision theory that will form the foundations for much of the subsequent discussion in this book. We close this chapter by introducing some additional concepts from the field of information theory, which will also prove useful in our development of pattern recognition and machine learning techniques. Again, we shall focus only on the key concepts, and we refer the reader elsewhere for more detailed discussions (Viterbi and Omura, 1979; Cover and Thomas, 1991; MacKay, 2003) .

We begin by considering a discrete random variable x and we ask how much information is received when we observe a specific value for this variable. The amount of information can be viewed as the 'degree of surprise' on learning the value of x. If we are told that a highly improbable event has just occurred, we will have received more information than if we were told that some very likely event has just occurred, and if we knew that the event was certain to happen we would receive no information. Our measure of information content will therefore depend on the probability distribution $p(x)$, and we therefore look for a quantity $h(x)$ that is a monotonic function of the probability $p(x)$ and that expresses the information content. The form of $h(\cdot)$ can be found by noting that if we have two events x and y that are unrelated, then the information gain from observing both of them should be the sum of the information gained from each of them separately, so that $h(x, y) = h(x) + h(y)$. Two unrelated events will be statistically independent and so $p(x, y) = p(x)p(y)$. From these two relationships, it is easily shown that $h(x)$ must be given by the logarithm of $p(x)$ and so we have

Exercise 1.28

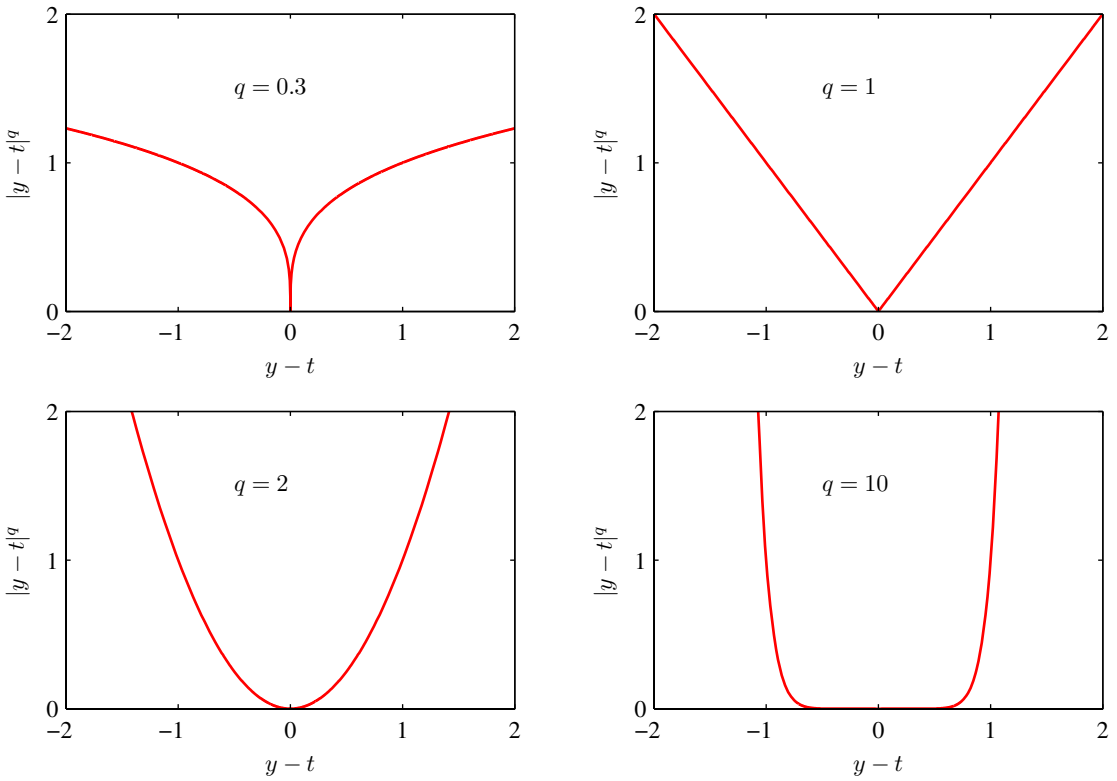


Figure 1.29 Plots of the quantity $L_q = |y - t|^q$ for various values of q.

$$h(x) = -\log_2 p(x) \tag{1.92}$$

where the negative sign ensures that information is positive or zero. Note that low probability events x correspond to high information content. The choice of basis for the logarithm is arbitrary, and for the moment we shall adopt the convention prevalent in information theory of using logarithms to the base of 2. In this case, as we shall see shortly, the units of $h(x)$ are bits ('binary digits').

Now suppose that a sender wishes to transmit the value of a random variable to a receiver. The average amount of information that they transmit in the process is obtained by taking the expectation of (1.92) with respect to the distribution $p(x)$ and is given by

$$\mathrm{H}[x] = -\sum_x p(x) \log_2 p(x). \tag{1.93}$$

This important quantity is called the *entropy* of the random variable x. Note that $\lim_{p \to 0} p \ln p = 0$ and so we shall take $p(x) \ln p(x) = 0$ whenever we encounter a value for x such that $p(x) = 0$.

So far we have given a rather heuristic motivation for the definition of informa-

tion (1.92) and the corresponding entropy (1.93). We now show that these definitions indeed possess useful properties. Consider a random variable x having 8 possible states, each of which is equally likely. In order to communicate the value of x to a receiver, we would need to transmit a message of length 3 bits. Notice that the entropy of this variable is given by

$$\mathrm{H}[x] = -8 \times \frac{1}{8} \log_2 \frac{1}{8} = 3 \text{ bits}.$$

Now consider an example (Cover and Thomas, 1991) of a variable having 8 possible states $\{a, b, c, d, e, f, g, h\}$ for which the respective probabilities are given by $(\frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \frac{1}{64}, \frac{1}{64}, \frac{1}{64}, \frac{1}{64})$. The entropy in this case is given by

$$\mathrm{H}[x] = -\frac{1}{2} \log_2 \frac{1}{2} - \frac{1}{4} \log_2 \frac{1}{4} - \frac{1}{8} \log_2 \frac{1}{8} - \frac{1}{16} \log_2 \frac{1}{16} - \frac{4}{64} \log_2 \frac{1}{64} = 2 \text{ bits}.$$

We see that the nonuniform distribution has a smaller entropy than the uniform one, and we shall gain some insight into this shortly when we discuss the interpretation of entropy in terms of disorder. For the moment, let us consider how we would transmit the identity of the variable's state to a receiver. We could do this, as before, using a 3-bit number. However, we can take advantage of the nonuniform distribution by using shorter codes for the more probable events, at the expense of longer codes for the less probable events, in the hope of getting a shorter average code length. This can be done by representing the states $\{a, b, c, d, e, f, g, h\}$ using, for instance, the following set of code strings: 0, 10, 110, 1110, 111100, 111101, 111110, 111111. The average length of the code that has to be transmitted is then

$$\text{average code length} = \frac{1}{2} \times 1 + \frac{1}{4} \times 2 + \frac{1}{8} \times 3 + \frac{1}{16} \times 4 + 4 \times \frac{1}{64} \times 6 = 2 \text{ bits}$$

which again is the same as the entropy of the random variable. Note that shorter code strings cannot be used because it must be possible to disambiguate a concatenation of such strings into its component parts. For instance, 11001110 decodes uniquely into the state sequence c, a, d.

This relation between entropy and shortest coding length is a general one. The *noiseless coding theorem* (Shannon, 1948) states that the entropy is a lower bound on the number of bits needed to transmit the state of a random variable.

From now on, we shall switch to the use of natural logarithms in defining entropy, as this will provide a more convenient link with ideas elsewhere in this book. In this case, the entropy is measured in units of 'nats' instead of bits, which differ simply by a factor of $\ln 2$.

We have introduced the concept of entropy in terms of the average amount of information needed to specify the state of a random variable. In fact, the concept of entropy has much earlier origins in physics where it was introduced in the context of equilibrium thermodynamics and later given a deeper interpretation as a measure of disorder through developments in statistical mechanics. We can understand this alternative view of entropy by considering a set of N identical objects that are to be divided amongst a set of bins, such that there are n_i objects in the i^{th} bin. Consider

the number of different ways of allocating the objects to the bins. There are N ways to choose the first object, $(N-1)$ ways to choose the second object, and so on, leading to a total of $N!$ ways to allocate all N objects to the bins, where $N!$ (pronounced 'factorial N') denotes the product $N \times (N-1) \times \cdots \times 2 \times 1$. However, we don't wish to distinguish between rearrangements of objects within each bin. In the i^{th} bin there are $n_i!$ ways of reordering the objects, and so the total number of ways of allocating the N objects to the bins is given by

$$W = \frac{N!}{\prod_i n_i!} \tag{1.94}$$

which is called the *multiplicity*. The entropy is then defined as the logarithm of the multiplicity scaled by an appropriate constant

$$\mathrm{H} = \frac{1}{N} \ln W = \frac{1}{N} \ln N! - \frac{1}{N} \sum_i \ln n_i!. \tag{1.95}$$

We now consider the limit $N \to \infty$, in which the fractions n_i/N are held fixed, and apply Stirling's approximation

$$\ln N! \simeq N \ln N - N \tag{1.96}$$

which gives

$$\mathrm{H} = -\lim_{N\to\infty} \sum_i \left(\frac{n_i}{N}\right) \ln\left(\frac{n_i}{N}\right) = -\sum_i p_i \ln p_i \tag{1.97}$$

where we have used $\sum_i n_i = N$. Here $p_i = \lim_{N\to\infty}(n_i/N)$ is the probability of an object being assigned to the i^{th} bin. In physics terminology, the specific arrangements of objects in the bins is called a *microstate*, and the overall distribution of occupation numbers, expressed through the ratios n_i/N, is called a *macrostate*. The multiplicity W is also known as the *weight* of the macrostate.

We can interpret the bins as the states x_i of a discrete random variable X, where $p(X = x_i) = p_i$. The entropy of the random variable X is then

$$\mathrm{H}[p] = -\sum_i p(x_i) \ln p(x_i). \tag{1.98}$$

Distributions $p(x_i)$ that are sharply peaked around a few values will have a relatively low entropy, whereas those that are spread more evenly across many values will have higher entropy, as illustrated in Figure 1.30. Because $0 \leqslant p_i \leqslant 1$, the entropy is nonnegative, and it will equal its minimum value of 0 when one of the $p_i = 1$ and all other $p_{j\neq i} = 0$. The maximum entropy configuration can be found by maximizing H using a Lagrange multiplier to enforce the normalization constraint on the probabilities. Thus we maximize

Appendix E

$$\widetilde{\mathrm{H}} = -\sum_i p(x_i) \ln p(x_i) + \lambda \left(\sum_i p(x_i) - 1 \right) \tag{1.99}$$

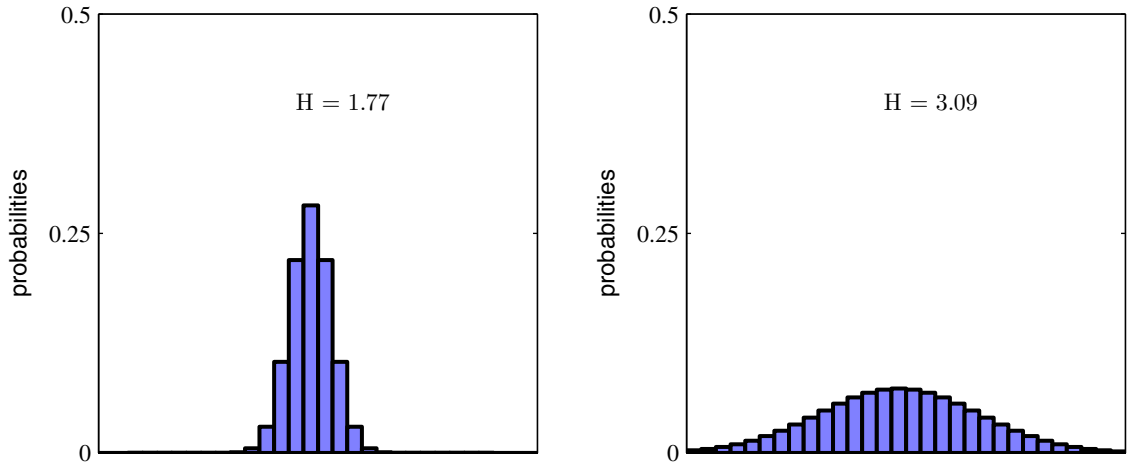


Figure 1.30 Histograms of two probability distributions over 30 bins illustrating the higher value of the entropy H for the broader distribution. The largest entropy would arise from a uniform distribution that would give $\mathrm{H} = -\ln(1/30) = 3.40$.

Exercise 1.29

from which we find that all of the $p(x_i)$ are equal and are given by $p(x_i) = 1/M$ where M is the total number of states x_i. The corresponding value of the entropy is then $\mathrm{H} = \ln M$. This result can also be derived from Jensen's inequality (to be discussed shortly). To verify that the stationary point is indeed a maximum, we can evaluate the second derivative of the entropy, which gives

$$\frac{\partial \widetilde{\mathrm{H}}}{\partial p(x_i)\partial p(x_j)} = -I_{ij}\frac{1}{p_i} \tag{1.100}$$

where I_{ij} are the elements of the identity matrix.

We can extend the definition of entropy to include distributions $p(x)$ over continuous variables x as follows. First divide x into bins of width Δ. Then, assuming $p(x)$ is continuous, the *mean value theorem* (Weisstein, 1999) tells us that, for each such bin, there must exist a value x_i such that

$$\int_{i\Delta}^{(i+1)\Delta} p(x)\,\mathrm{d}x = p(x_i)\Delta. \tag{1.101}$$

We can now quantize the continuous variable x by assigning any value x to the value x_i whenever x falls in the i^{th} bin. The probability of observing the value x_i is then $p(x_i)\Delta$. This gives a discrete distribution for which the entropy takes the form

$$\mathrm{H}_\Delta = -\sum_i p(x_i)\Delta \ln\left(p(x_i)\Delta\right) = -\sum_i p(x_i)\Delta \ln p(x_i) - \ln\Delta \tag{1.102}$$

where we have used $\sum_i p(x_i)\Delta = 1$, which follows from (1.101). We now omit the second term $-\ln\Delta$ on the right-hand side of (1.102) and then consider the limit

$\Delta \to 0$. The first term on the right-hand side of (1.102) will approach the integral of $p(x) \ln p(x)$ in this limit so that

$$\lim_{\Delta \to 0} \left\{ \sum_i p(x_i) \Delta \ln p(x_i) \right\} = - \int p(x) \ln p(x) \, \mathrm{d}x \tag{1.103}$$

where the quantity on the right-hand side is called the *differential entropy*. We see that the discrete and continuous forms of the entropy differ by a quantity $\ln \Delta$, which diverges in the limit $\Delta \to 0$. This reflects the fact that to specify a continuous variable very precisely requires a large number of bits. For a density defined over multiple continuous variables, denoted collectively by the vector $\mathbf{x}$, the differential entropy is given by

$$\mathrm{H}[\mathbf{x}] = - \int p(\mathbf{x}) \ln p(\mathbf{x}) \, \mathrm{d}\mathbf{x}. \tag{1.104}$$

In the case of discrete distributions, we saw that the maximum entropy configuration corresponded to an equal distribution of probabilities across the possible states of the variable. Let us now consider the maximum entropy configuration for a continuous variable. In order for this maximum to be well defined, it will be necessary to constrain the first and second moments of $p(x)$ as well as preserving the normalization constraint. We therefore maximize the differential entropy with the

Ludwig Boltzmann
1844–1906

Ludwig Eduard Boltzmann was an Austrian physicist who created the field of statistical mechanics. Prior to Boltzmann, the concept of entropy was already known from classical thermodynamics where it quantifies the fact that when we take energy from a system, not all of that energy is typically available to do useful work. Boltzmann showed that the thermodynamic entropy S, a macroscopic quantity, could be related to the statistical properties at the microscopic level. This is expressed through the famous equation $S = k \ln W$ in which W represents the number of possible microstates in a macrostate, and $k \simeq 1.38 \times 10^{-23}$ (in units of Joules per Kelvin) is known as Boltzmann's constant. Boltzmann's ideas were disputed by many scientists of they day. One difficulty they saw arose from the second law of thermodynamics, which states that the entropy of a closed system tends to increase with time. By contrast, at the microscopic level the classical Newtonian equations of physics are reversible, and so they found it difficult to see how the latter could explain the former. They didn't fully appreciate Boltzmann's arguments, which were statistical in nature and which concluded not that entropy could never decrease over time but simply that with overwhelming probability it would generally increase. Boltzmann even had a long-running dispute with the editor of the leading German physics journal who refused to let him refer to atoms and molecules as anything other than convenient theoretical constructs. The continued attacks on his work lead to bouts of depression, and eventually he committed suicide. Shortly after Boltzmann's death, new experiments by Perrin on colloidal suspensions verified his theories and confirmed the value of the Boltzmann constant. The equation $S = k \ln W$ is carved on Boltzmann's tombstone.

three constraints

$$\int_{-\infty}^{\infty} p(x)\,\mathrm{d}x = 1 \tag{1.105}$$

$$\int_{-\infty}^{\infty} xp(x)\,\mathrm{d}x = \mu \tag{1.106}$$

$$\int_{-\infty}^{\infty} (x-\mu)^2 p(x)\,\mathrm{d}x = \sigma^2. \tag{1.107}$$

Appendix E The constrained maximization can be performed using Lagrange multipliers so that we maximize the following functional with respect to $p(x)$

$$-\int_{-\infty}^{\infty} p(x)\ln p(x)\,\mathrm{d}x + \lambda_1\left(\int_{-\infty}^{\infty} p(x)\,\mathrm{d}x - 1\right) + \lambda_2\left(\int_{-\infty}^{\infty} xp(x)\,\mathrm{d}x - \mu\right) + \lambda_3\left(\int_{-\infty}^{\infty} (x-\mu)^2 p(x)\,\mathrm{d}x - \sigma^2\right).$$

Appendix D Using the calculus of variations, we set the derivative of this functional to zero giving

$$p(x) = \exp\left\{-1 + \lambda_1 + \lambda_2 x + \lambda_3 (x-\mu)^2\right\}. \tag{1.108}$$

Exercise 1.34 The Lagrange multipliers can be found by back substitution of this result into the three constraint equations, leading finally to the result

$$p(x) = \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left\{-\frac{(x-\mu)^2}{2\sigma^2}\right\} \tag{1.109}$$

and so the distribution that maximizes the differential entropy is the Gaussian. Note that we did not constrain the distribution to be nonnegative when we maximized the entropy. However, because the resulting distribution is indeed nonnegative, we see with hindsight that such a constraint is not necessary.

Exercise 1.35 If we evaluate the differential entropy of the Gaussian, we obtain

$$\mathrm{H}[x] = \frac{1}{2}\left\{1 + \ln(2\pi\sigma^2)\right\}. \tag{1.110}$$

Thus we see again that the entropy increases as the distribution becomes broader, i.e., as σ^2 increases. This result also shows that the differential entropy, unlike the discrete entropy, can be negative, because $\mathrm{H}(x) < 0$ in (1.110) for $\sigma^2 < 1/(2\pi e)$.

Suppose we have a joint distribution $p(\mathbf{x}, \mathbf{y})$ from which we draw pairs of values of $\mathbf{x}$ and $\mathbf{y}$. If a value of $\mathbf{x}$ is already known, then the additional information needed to specify the corresponding value of $\mathbf{y}$ is given by $-\ln p(\mathbf{y}|\mathbf{x})$. Thus the average additional information needed to specify $\mathbf{y}$ can be written as

$$\mathrm{H}[\mathbf{y}|\mathbf{x}] = -\iint p(\mathbf{y}, \mathbf{x}) \ln p(\mathbf{y}|\mathbf{x})\,\mathrm{d}\mathbf{y}\,\mathrm{d}\mathbf{x} \tag{1.111}$$

Exercise 1.37

which is called the *conditional entropy* of $\mathbf{y}$ given $\mathbf{x}$. It is easily seen, using the product rule, that the conditional entropy satisfies the relation

$$\mathrm{H}[\mathbf{x}, \mathbf{y}] = \mathrm{H}[\mathbf{y}|\mathbf{x}] + \mathrm{H}[\mathbf{x}] \tag{1.112}$$

where $\mathrm{H}[\mathbf{x}, \mathbf{y}]$ is the differential entropy of $p(\mathbf{x}, \mathbf{y})$ and $\mathrm{H}[\mathbf{x}]$ is the differential entropy of the marginal distribution $p(\mathbf{x})$. Thus the information needed to describe $\mathbf{x}$ and $\mathbf{y}$ is given by the sum of the information needed to describe $\mathbf{x}$ alone plus the additional information required to specify $\mathbf{y}$ given $\mathbf{x}$.

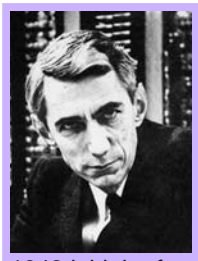
1.6.1 Relative entropy and mutual information

So far in this section, we have introduced a number of concepts from information theory, including the key notion of entropy. We now start to relate these ideas to pattern recognition. Consider some unknown distribution $p(\mathbf{x})$, and suppose that we have modelled this using an approximating distribution $q(\mathbf{x})$. If we use $q(\mathbf{x})$ to construct a coding scheme for the purpose of transmitting values of $\mathbf{x}$ to a receiver, then the average *additional* amount of information (in nats) required to specify the value of $\mathbf{x}$ (assuming we choose an efficient coding scheme) as a result of using $q(\mathbf{x})$ instead of the true distribution $p(\mathbf{x})$ is given by

$$\begin{aligned} \mathrm{KL}(p\|q) &= -\int p(\mathbf{x}) \ln q(\mathbf{x})\, \mathrm{d}\mathbf{x} - \left(-\int p(\mathbf{x}) \ln p(\mathbf{x})\, \mathrm{d}\mathbf{x}\right) \\ &= -\int p(\mathbf{x}) \ln \left\{\frac{q(\mathbf{x})}{p(\mathbf{x})}\right\} \mathrm{d}\mathbf{x}. \end{aligned} \tag{1.113}$$

This is known as the *relative entropy* or *Kullback-Leibler divergence*, or *KL divergence* (Kullback and Leibler, 1951), between the distributions $p(\mathbf{x})$ and $q(\mathbf{x})$. Note that it is not a symmetrical quantity, that is to say $\mathrm{KL}(p\|q) \not\equiv \mathrm{KL}(q\|p)$.

We now show that the Kullback-Leibler divergence satisfies $\mathrm{KL}(p\|q) \geqslant 0$ with equality if, and only if, $p(\mathbf{x}) = q(\mathbf{x})$. To do this we first introduce the concept of *convex* functions. A function $f(x)$ is said to be convex if it has the property that every chord lies on or above the function, as shown in Figure 1.31. Any value of x in the interval from $x = a$ to $x = b$ can be written in the form $\lambda a + (1 - \lambda)b$ where $0 \leqslant \lambda \leqslant 1$. The corresponding point on the chord is given by $\lambda f(a) + (1 - \lambda)f(b)$,



Claude Shannon
1916–2001

After graduating from Michigan and MIT, Shannon joined the AT&T Bell Telephone laboratories in 1941. His paper 'A Mathematical Theory of Communication' published in the *Bell System Technical Journal* in 1948 laid the foundations for modern information theory. This paper introduced the word 'bit', and his concept that information could be sent as a stream of 1s and 0s paved the way for the communications revolution. It is said that von Neumann recommended to Shannon that he use the term entropy, not only because of its similarity to the quantity used in physics, but also because "nobody knows what entropy really is, so in any discussion you will always have an advantage".

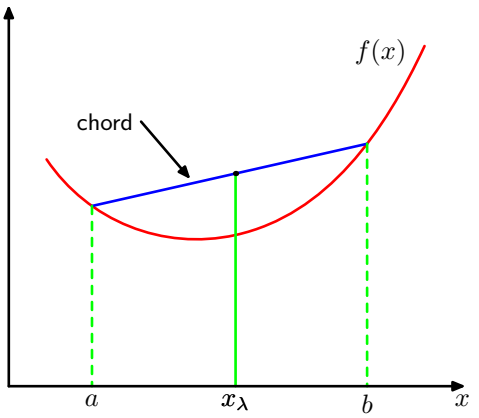


Figure 1.31 A convex function $f(x)$ is one for which every chord (shown in blue) lies on or above the function (shown in red).

and the corresponding value of the function is $f(\lambda a + (1-\lambda)b)$. Convexity then implies

$$f(\lambda a + (1-\lambda)b) \leqslant \lambda f(a) + (1-\lambda)f(b). \tag{1.114}$$

This is equivalent to the requirement that the second derivative of the function be everywhere positive. Examples of convex functions are $x \ln x$ (for $x > 0$) and x^2. A function is called *strictly convex* if the equality is satisfied only for $\lambda = 0$ and $\lambda = 1$. If a function has the opposite property, namely that every chord lies on or below the function, it is called *concave*, with a corresponding definition for *strictly concave*. If a function $f(x)$ is convex, then $-f(x)$ will be concave.

Exercise 1.36

Exercise 1.38

Using the technique of proof by induction, we can show from (1.114) that a convex function $f(x)$ satisfies

$$f\left(\sum_{i=1}^{M} \lambda_i x_i\right) \leqslant \sum_{i=1}^{M} \lambda_i f(x_i) \tag{1.115}$$

where $\lambda_i \geqslant 0$ and $\sum_i \lambda_i = 1$, for any set of points $\{x_i\}$. The result (1.115) is known as *Jensen's inequality*. If we interpret the λ_i as the probability distribution over a discrete variable x taking the values $\{x_i\}$, then (1.115) can be written

$$f(\mathbb{E}[x]) \leqslant \mathbb{E}[f(x)] \tag{1.116}$$

where $\mathbb{E}[\cdot]$ denotes the expectation. For continuous variables, Jensen's inequality takes the form

$$f\left(\int \mathbf{x}p(\mathbf{x})\,\mathrm{d}\mathbf{x}\right) \leqslant \int f(\mathbf{x})p(\mathbf{x})\,\mathrm{d}\mathbf{x}. \tag{1.117}$$

We can apply Jensen's inequality in the form (1.117) to the Kullback-Leibler divergence (1.113) to give

$$\mathrm{KL}(p\|q) = -\int p(\mathbf{x}) \ln \left\{\frac{q(\mathbf{x})}{p(\mathbf{x})}\right\} \mathrm{d}\mathbf{x} \geqslant -\ln \int q(\mathbf{x})\,\mathrm{d}\mathbf{x} = 0 \tag{1.118}$$

where we have used the fact that $-\ln x$ is a convex function, together with the normalization condition $\int q(\mathbf{x})\,\mathrm{d}\mathbf{x} = 1$. In fact, $-\ln x$ is a strictly convex function, so the equality will hold if, and only if, $q(\mathbf{x}) = p(\mathbf{x})$ for all $\mathbf{x}$. Thus we can interpret the Kullback-Leibler divergence as a measure of the dissimilarity of the two distributions $p(\mathbf{x})$ and $q(\mathbf{x})$.

We see that there is an intimate relationship between data compression and density estimation (i.e., the problem of modelling an unknown probability distribution) because the most efficient compression is achieved when we know the true distribution. If we use a distribution that is different from the true one, then we must necessarily have a less efficient coding, and on average the additional information that must be transmitted is (at least) equal to the Kullback-Leibler divergence between the two distributions.

Suppose that data is being generated from an unknown distribution $p(\mathbf{x})$ that we wish to model. We can try to approximate this distribution using some parametric distribution $q(\mathbf{x}|\boldsymbol{\theta})$, governed by a set of adjustable parameters $\boldsymbol{\theta}$, for example a multivariate Gaussian. One way to determine $\boldsymbol{\theta}$ is to minimize the Kullback-Leibler divergence between $p(\mathbf{x})$ and $q(\mathbf{x}|\boldsymbol{\theta})$ with respect to $\boldsymbol{\theta}$. We cannot do this directly because we don't know $p(\mathbf{x})$. Suppose, however, that we have observed a finite set of training points $\mathbf{x}_n$, for $n = 1, \ldots, N$, drawn from $p(\mathbf{x})$. Then the expectation with respect to $p(\mathbf{x})$ can be approximated by a finite sum over these points, using (1.35), so that

$$\mathrm{KL}(p\|q) \simeq \sum_{n=1}^{N} \left\{-\ln q(\mathbf{x}_n|\boldsymbol{\theta}) + \ln p(\mathbf{x}_n)\right\}. \tag{1.119}$$

The second term on the right-hand side of (1.119) is independent of $\boldsymbol{\theta}$, and the first term is the negative log likelihood function for $\boldsymbol{\theta}$ under the distribution $q(\mathbf{x}|\boldsymbol{\theta})$ evaluated using the training set. Thus we see that minimizing this Kullback-Leibler divergence is equivalent to maximizing the likelihood function.

Now consider the joint distribution between two sets of variables $\mathbf{x}$ and $\mathbf{y}$ given by $p(\mathbf{x}, \mathbf{y})$. If the sets of variables are independent, then their joint distribution will factorize into the product of their marginals $p(\mathbf{x}, \mathbf{y}) = p(\mathbf{x})p(\mathbf{y})$. If the variables are not independent, we can gain some idea of whether they are 'close' to being independent by considering the Kullback-Leibler divergence between the joint distribution and the product of the marginals, given by

$$\begin{aligned} \mathrm{I}[\mathbf{x}, \mathbf{y}] &\equiv \mathrm{KL}(p(\mathbf{x}, \mathbf{y})\|p(\mathbf{x})p(\mathbf{y})) \\ &= -\iint p(\mathbf{x}, \mathbf{y}) \ln \left(\frac{p(\mathbf{x})p(\mathbf{y})}{p(\mathbf{x}, \mathbf{y})} \right) \mathrm{d}\mathbf{x}\,\mathrm{d}\mathbf{y} \end{aligned} \tag{1.120}$$

which is called the *mutual information* between the variables $\mathbf{x}$ and $\mathbf{y}$. From the properties of the Kullback-Leibler divergence, we see that $I(\mathbf{x}, \mathbf{y}) \geqslant 0$ with equality if, and only if, $\mathbf{x}$ and $\mathbf{y}$ are independent. Using the sum and product rules of probability, we see that the mutual information is related to the conditional entropy through

Exercise 1.41

$$\mathrm{I}[\mathbf{x}, \mathbf{y}] = \mathrm{H}[\mathbf{x}] - \mathrm{H}[\mathbf{x}|\mathbf{y}] = \mathrm{H}[\mathbf{y}] - \mathrm{H}[\mathbf{y}|\mathbf{x}]. \tag{1.121}$$

Thus we can view the mutual information as the reduction in the uncertainty about $\mathbf{x}$ by virtue of being told the value of $\mathbf{y}$ (or vice versa). From a Bayesian perspective, we can view $p(\mathbf{x})$ as the prior distribution for $\mathbf{x}$ and $p(\mathbf{x}|\mathbf{y})$ as the posterior distribution after we have observed new data $\mathbf{y}$. The mutual information therefore represents the reduction in uncertainty about $\mathbf{x}$ as a consequence of the new observation $\mathbf{y}$.

Exercises

1.1 ($\star$) **www** Consider the sum-of-squares error function given by (1.2) in which the function $y(x, \mathbf{w})$ is given by the polynomial (1.1). Show that the coefficients $\mathbf{w} = \{w_i\}$ that minimize this error function are given by the solution to the following set of linear equations

$$\sum_{j=0}^{M} A_{ij} w_j = T_i \tag{1.122}$$

where

$$A_{ij} = \sum_{n=1}^{N} (x_n)^{i+j}, \qquad T_i = \sum_{n=1}^{N} (x_n)^i t_n. \tag{1.123}$$

Here a suffix i or j denotes the index of a component, whereas $(x)^i$ denotes x raised to the power of i.

1.2 ($\star$) Write down the set of coupled linear equations, analogous to (1.122), satisfied by the coefficients w_i which minimize the regularized sum-of-squares error function given by (1.4).

1.3 ($\star\star$) Suppose that we have three coloured boxes r (red), b (blue), and g (green). Box r contains 3 apples, 4 oranges, and 3 limes, box b contains 1 apple, 1 orange, and 0 limes, and box g contains 3 apples, 3 oranges, and 4 limes. If a box is chosen at random with probabilities $p(r) = 0.2$, $p(b) = 0.2$, $p(g) = 0.6$, and a piece of fruit is removed from the box (with equal probability of selecting any of the items in the box), then what is the probability of selecting an apple? If we observe that the selected fruit is in fact an orange, what is the probability that it came from the green box?

1.4 ($\star\star$) **www** Consider a probability density $p_x(x)$ defined over a continuous variable x, and suppose that we make a nonlinear change of variable using $x = g(y)$, so that the density transforms according to (1.27). By differentiating (1.27), show that the location $\widehat{y}$ of the maximum of the density in y is not in general related to the location $\widehat{x}$ of the maximum of the density over x by the simple functional relation $\widehat{x} = g(\widehat{y})$ as a consequence of the Jacobian factor. This shows that the maximum of a probability density (in contrast to a simple function) is dependent on the choice of variable. Verify that, in the case of a linear transformation, the location of the maximum transforms in the same way as the variable itself.

1.5 ($\star$) Using the definition (1.38) show that $\mathrm{var}[f(x)]$ satisfies (1.39).

1.6 (⋆) Show that if two variables x and y are independent, then their covariance is zero.

1.7 (⋆⋆) **www** In this exercise, we prove the normalization condition (1.48) for the univariate Gaussian. To do this consider, the integral

$$I = \int_{-\infty}^{\infty} \exp\left(-\frac{1}{2\sigma^2}x^2\right) \mathrm{d}x \tag{1.124}$$

which we can evaluate by first writing its square in the form

$$I^2 = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \exp\left(-\frac{1}{2\sigma^2}x^2 - \frac{1}{2\sigma^2}y^2\right) \mathrm{d}x\,\mathrm{d}y. \tag{1.125}$$

Now make the transformation from Cartesian coordinates (x, y) to polar coordinates (r, θ) and then substitute $u = r^2$. Show that, by performing the integrals over θ and u, and then taking the square root of both sides, we obtain

$$I = \left(2\pi\sigma^2\right)^{1/2}. \tag{1.126}$$

Finally, use this result to show that the Gaussian distribution $\mathcal{N}(x|\mu, \sigma^2)$ is normalized.

1.8 (⋆⋆) **www** By using a change of variables, verify that the univariate Gaussian distribution given by (1.46) satisfies (1.49). Next, by differentiating both sides of the normalization condition

$$\int_{-\infty}^{\infty} \mathcal{N}\left(x|\mu, \sigma^2\right) \mathrm{d}x = 1 \tag{1.127}$$

with respect to σ^2, verify that the Gaussian satisfies (1.50). Finally, show that (1.51) holds.

1.9 (⋆) **www** Show that the mode (i.e. the maximum) of the Gaussian distribution (1.46) is given by μ. Similarly, show that the mode of the multivariate Gaussian (1.52) is given by $\boldsymbol{\mu}$.

1.10 (⋆) **www** Suppose that the two variables x and z are statistically independent. Show that the mean and variance of their sum satisfies

$$\mathbb{E}[x + z] = \mathbb{E}[x] + \mathbb{E}[z] \tag{1.128}$$

$$\mathrm{var}[x + z] = \mathrm{var}[x] + \mathrm{var}[z]. \tag{1.129}$$

1.11 (⋆) By setting the derivatives of the log likelihood function (1.54) with respect to μ and σ^2 equal to zero, verify the results (1.55) and (1.56).

1.12 (⋆⋆) **www** Using the results (1.49) and (1.50), show that

$$\mathbb{E}[x_n x_m] = \mu^2 + I_{nm}\sigma^2 \tag{1.130}$$

where x_n and x_m denote data points sampled from a Gaussian distribution with mean μ and variance σ^2, and I_{nm} satisfies $I_{nm} = 1$ if $n = m$ and $I_{nm} = 0$ otherwise. Hence prove the results (1.57) and (1.58).

1.13 (⋆) Suppose that the variance of a Gaussian is estimated using the result (1.56) but with the maximum likelihood estimate μ_{ML} replaced with the true value μ of the mean. Show that this estimator has the property that its expectation is given by the true variance σ^2.

1.14 (⋆⋆) Show that an arbitrary square matrix with elements w_{ij} can be written in the form $w_{ij} = w_{ij}^{\mathrm{S}} + w_{ij}^{\mathrm{A}}$ where w_{ij}^{S} and w_{ij}^{A} are symmetric and anti-symmetric matrices, respectively, satisfying $w_{ij}^{\mathrm{S}} = w_{ji}^{\mathrm{S}}$ and $w_{ij}^{\mathrm{A}} = -w_{ji}^{\mathrm{A}}$ for all i and j. Now consider the second order term in a higher order polynomial in D dimensions, given by

$$\sum_{i=1}^{D}\sum_{j=1}^{D} w_{ij} x_i x_j. \tag{1.131}$$

Show that

$$\sum_{i=1}^{D}\sum_{j=1}^{D} w_{ij} x_i x_j = \sum_{i=1}^{D}\sum_{j=1}^{D} w_{ij}^{\mathrm{S}} x_i x_j \tag{1.132}$$

so that the contribution from the anti-symmetric matrix vanishes. We therefore see that, without loss of generality, the matrix of coefficients w_{ij} can be chosen to be symmetric, and so not all of the D^2 elements of this matrix can be chosen independently. Show that the number of independent parameters in the matrix w_{ij}^{S} is given by $D(D+1)/2$.

1.15 (⋆⋆⋆) **www** In this exercise and the next, we explore how the number of independent parameters in a polynomial grows with the order M of the polynomial and with the dimensionality D of the input space. We start by writing down the M^{th} order term for a polynomial in D dimensions in the form

$$\sum_{i_1=1}^{D}\sum_{i_2=1}^{D}\cdots\sum_{i_M=1}^{D} w_{i_1 i_2 \cdots i_M} x_{i_1} x_{i_2} \cdots x_{i_M}. \tag{1.133}$$

The coefficients $w_{i_1 i_2 \cdots i_M}$ comprise D^M elements, but the number of independent parameters is significantly fewer due to the many interchange symmetries of the factor $x_{i_1} x_{i_2} \cdots x_{i_M}$. Begin by showing that the redundancy in the coefficients can be removed by rewriting this M^{th} order term in the form

$$\sum_{i_1=1}^{D}\sum_{i_2=1}^{i_1}\cdots\sum_{i_M=1}^{i_{M-1}} \widetilde{w}_{i_1 i_2 \cdots i_M} x_{i_1} x_{i_2} \cdots x_{i_M}. \tag{1.134}$$

Note that the precise relationship between the $\widetilde{w}$ coefficients and w coefficients need not be made explicit. Use this result to show that the number of *independent* parameters $n(D, M)$, which appear at order M, satisfies the following recursion relation

$$n(D, M) = \sum_{i=1}^{D} n(i, M-1). \tag{1.135}$$

Next use proof by induction to show that the following result holds

$$\sum_{i=1}^{D} \frac{(i+M-2)!}{(i-1)!\,(M-1)!} = \frac{(D+M-1)!}{(D-1)!\,M!} \tag{1.136}$$

which can be done by first proving the result for $D = 1$ and arbitrary M by making use of the result $0! = 1$, then assuming it is correct for dimension D and verifying that it is correct for dimension $D + 1$. Finally, use the two previous results, together with proof by induction, to show

$$n(D, M) = \frac{(D+M-1)!}{(D-1)!\,M!}. \tag{1.137}$$

To do this, first show that the result is true for $M = 2$, and any value of $D \geqslant 1$, by comparison with the result of Exercise 1.14. Then make use of (1.135), together with (1.136), to show that, if the result holds at order $M - 1$, then it will also hold at order M

1.16 ($\star\star\star$) In Exercise 1.15, we proved the result (1.135) for the number of independent parameters in the M^{th} order term of a D-dimensional polynomial. We now find an expression for the total number $N(D, M)$ of independent parameters in all of the terms up to and including the $M6\text{th}$ order. First show that $N(D, M)$ satisfies

$$N(D, M) = \sum_{m=0}^{M} n(D, m) \tag{1.138}$$

where $n(D, m)$ is the number of independent parameters in the term of order m. Now make use of the result (1.137), together with proof by induction, to show that

$$N(d, M) = \frac{(D+M)!}{D!\,M!}. \tag{1.139}$$

This can be done by first proving that the result holds for $M = 0$ and arbitrary $D \geqslant 1$, then assuming that it holds at order M, and hence showing that it holds at order $M + 1$. Finally, make use of Stirling's approximation in the form

$$n! \simeq n^n e^{-n} \tag{1.140}$$

for large n to show that, for $D \gg M$, the quantity $N(D, M)$ grows like D^M, and for $M \gg D$ it grows like M^D. Consider a cubic ($M = 3$) polynomial in D dimensions, and evaluate numerically the total number of independent parameters for (i) $D = 10$ and (ii) $D = 100$, which correspond to typical small-scale and medium-scale machine learning applications.

1.17 (⋆⋆) **www** The gamma function is defined by

$$\Gamma(x) \equiv \int_0^\infty u^{x-1} e^{-u}\,\mathrm{d}u. \tag{1.141}$$

Using integration by parts, prove the relation $\Gamma(x+1) = x\Gamma(x)$. Show also that $\Gamma(1) = 1$ and hence that $\Gamma(x+1) = x!$ when x is an integer.

1.18 (⋆⋆) **www** We can use the result (1.126) to derive an expression for the surface area S_D, and the volume V_D, of a sphere of unit radius in D dimensions. To do this, consider the following result, which is obtained by transforming from Cartesian to polar coordinates

$$\prod_{i=1}^{D} \int_{-\infty}^{\infty} e^{-x_i^2}\,\mathrm{d}x_i = S_D \int_0^\infty e^{-r^2} r^{D-1}\,\mathrm{d}r. \tag{1.142}$$

Using the definition (1.141) of the Gamma function, together with (1.126), evaluate both sides of this equation, and hence show that

$$S_D = \frac{2\pi^{D/2}}{\Gamma(D/2)}. \tag{1.143}$$

Next, by integrating with respect to radius from 0 to 1, show that the volume of the unit sphere in D dimensions is given by

$$V_D = \frac{S_D}{D}. \tag{1.144}$$

Finally, use the results $\Gamma(1) = 1$ and $\Gamma(3/2) = \sqrt{\pi}/2$ to show that (1.143) and (1.144) reduce to the usual expressions for $D = 2$ and $D = 3$.

1.19 (⋆⋆) Consider a sphere of radius a in D-dimensions together with the concentric hypercube of side $2a$, so that the sphere touches the hypercube at the centres of each of its sides. By using the results of Exercise 1.18, show that the ratio of the volume of the sphere to the volume of the cube is given by

$$\frac{\text{volume of sphere}}{\text{volume of cube}} = \frac{\pi^{D/2}}{D2^{D-1}\Gamma(D/2)}. \tag{1.145}$$

Now make use of Stirling's formula in the form

$$\Gamma(x+1) \simeq (2\pi)^{1/2} e^{-x} x^{x+1/2} \tag{1.146}$$

which is valid for $x \gg 1$, to show that, as $D \to \infty$, the ratio (1.145) goes to zero. Show also that the ratio of the distance from the centre of the hypercube to one of the corners, divided by the perpendicular distance to one of the sides, is $\sqrt{D}$, which therefore goes to ∞ as $D \to \infty$. From these results we see that, in a space of high dimensionality, most of the volume of a cube is concentrated in the large number of corners, which themselves become very long 'spikes'!

1.20 (⋆⋆) **www** In this exercise, we explore the behaviour of the Gaussian distribution in high-dimensional spaces. Consider a Gaussian distribution in D dimensions given by

$$p(\mathbf{x}) = \frac{1}{(2\pi\sigma^2)^{D/2}} \exp\left(-\frac{\|\mathbf{x}\|^2}{2\sigma^2}\right). \tag{1.147}$$

We wish to find the density with respect to radius in polar coordinates in which the direction variables have been integrated out. To do this, show that the integral of the probability density over a thin shell of radius r and thickness ϵ, where $\epsilon \ll 1$, is given by $p(r)\epsilon$ where

$$p(r) = \frac{S_D r^{D-1}}{(2\pi\sigma^2)^{D/2}} \exp\left(-\frac{r^2}{2\sigma^2}\right) \tag{1.148}$$

where S_D is the surface area of a unit sphere in D dimensions. Show that the function $p(r)$ has a single stationary point located, for large D, at $\widehat{r} \simeq \sqrt{D}\sigma$. By considering $p(\widehat{r} + \epsilon)$ where $\epsilon \ll \widehat{r}$, show that for large D,

$$p(\widehat{r} + \epsilon) = p(\widehat{r}) \exp\left(-\frac{3\epsilon^2}{2\sigma^2}\right) \tag{1.149}$$

which shows that $\widehat{r}$ is a maximum of the radial probability density and also that $p(r)$ decays exponentially away from its maximum at $\widehat{r}$ with length scale σ. We have already seen that $\sigma \ll \widehat{r}$ for large D, and so we see that most of the probability mass is concentrated in a thin shell at large radius. Finally, show that the probability density $p(\mathbf{x})$ is larger at the origin than at the radius $\widehat{r}$ by a factor of $\exp(D/2)$. We therefore see that most of the probability mass in a high-dimensional Gaussian distribution is located at a different radius from the region of high probability density. This property of distributions in spaces of high dimensionality will have important consequences when we consider Bayesian inference of model parameters in later chapters.

1.21 (⋆⋆) Consider two nonnegative numbers a and b, and show that, if $a \leqslant b$, then $a \leqslant (ab)^{1/2}$. Use this result to show that, if the decision regions of a two-class classification problem are chosen to minimize the probability of misclassification, this probability will satisfy

$$p(\text{mistake}) \leqslant \int \{p(\mathbf{x}, \mathcal{C}_1)p(\mathbf{x}, \mathcal{C}_2)\}^{1/2} \, \mathrm{d}\mathbf{x}. \tag{1.150}$$

1.22 (⋆) **www** Given a loss matrix with elements L_{kj}, the expected risk is minimized if, for each $\mathbf{x}$, we choose the class that minimizes (1.81). Verify that, when the loss matrix is given by $L_{kj} = 1 - I_{kj}$, where I_{kj} are the elements of the identity matrix, this reduces to the criterion of choosing the class having the largest posterior probability. What is the interpretation of this form of loss matrix?

1.23 (⋆) Derive the criterion for minimizing the expected loss when there is a general loss matrix and general prior probabilities for the classes.

1.24 (⋆⋆) www Consider a classification problem in which the loss incurred when an input vector from class $\mathcal{C}_k$ is classified as belonging to class $\mathcal{C}_j$ is given by the loss matrix L_{kj}, and for which the loss incurred in selecting the reject option is λ. Find the decision criterion that will give the minimum expected loss. Verify that this reduces to the reject criterion discussed in Section 1.5.3 when the loss matrix is given by $L_{kj} = 1 - I_{kj}$. What is the relationship between λ and the rejection threshold θ?

1.25 (⋆) www Consider the generalization of the squared loss function (1.87) for a single target variable t to the case of multiple target variables described by the vector $\mathbf{t}$ given by

$$\mathbb{E}[L(\mathbf{t}, \mathbf{y}(\mathbf{x}))] = \iint \|\mathbf{y}(\mathbf{x}) - \mathbf{t}\|^2 p(\mathbf{x}, \mathbf{t})\, \mathrm{d}\mathbf{x}\, \mathrm{d}\mathbf{t}. \tag{1.151}$$

Using the calculus of variations, show that the function $\mathbf{y}(\mathbf{x})$ for which this expected loss is minimized is given by $\mathbf{y}(\mathbf{x}) = \mathbb{E}_{\mathbf{t}}[\mathbf{t}|\mathbf{x}]$. Show that this result reduces to (1.89) for the case of a single target variable t.

1.26 (⋆) By expansion of the square in (1.151), derive a result analogous to (1.90) and hence show that the function $\mathbf{y}(\mathbf{x})$ that minimizes the expected squared loss for the case of a vector $\mathbf{t}$ of target variables is again given by the conditional expectation of $\mathbf{t}$.

1.27 (⋆⋆) www Consider the expected loss for regression problems under the L_q loss function given by (1.91). Write down the condition that $y(\mathbf{x})$ must satisfy in order to minimize $\mathbb{E}[L_q]$. Show that, for $q = 1$, this solution represents the conditional median, i.e., the function $y(\mathbf{x})$ such that the probability mass for $t < y(\mathbf{x})$ is the same as for $t \geqslant y(\mathbf{x})$. Also show that the minimum expected L_q loss for $q \to 0$ is given by the conditional mode, i.e., by the function $y(\mathbf{x})$ equal to the value of t that maximizes $p(t|\mathbf{x})$ for each $\mathbf{x}$.

1.28 (⋆) In Section 1.6, we introduced the idea of entropy $h(x)$ as the information gained on observing the value of a random variable x having distribution $p(x)$. We saw that, for independent variables x and y for which $p(x, y) = p(x)p(y)$, the entropy functions are additive, so that $h(x, y) = h(x) + h(y)$. In this exercise, we derive the relation between h and p in the form of a function $h(p)$. First show that $h(p^2) = 2h(p)$, and hence by induction that $h(p^n) = nh(p)$ where n is a positive integer. Hence show that $h(p^{n/m}) = (n/m)h(p)$ where m is also a positive integer. This implies that $h(p^x) = xh(p)$ where x is a positive rational number, and hence by continuity when it is a positive real number. Finally, show that this implies $h(p)$ must take the form $h(p) \propto \ln p$.

1.29 (⋆) www Consider an M-state discrete random variable x, and use Jensen's inequality in the form (1.115) to show that the entropy of its distribution $p(x)$ satisfies $\mathrm{H}[x] \leqslant \ln M$.

1.30 (⋆⋆) Evaluate the Kullback-Leibler divergence (1.113) between two Gaussians $p(x) = \mathcal{N}(x|\mu, \sigma^2)$ and $q(x) = \mathcal{N}(x|m, s^2)$.

Table 1.3 The joint distribution $p(x, y)$ for two binary variables x and y used in Exercise 1.39.

		y = 0	y = 1
x	0	1/3	1/3
	1	0	1/3

1.31 ($\star\star$) **www** Consider two variables $\mathbf{x}$ and $\mathbf{y}$ having joint distribution $p(\mathbf{x}, \mathbf{y})$. Show that the differential entropy of this pair of variables satisfies

$$\mathrm{H}[\mathbf{x}, \mathbf{y}] \leqslant \mathrm{H}[\mathbf{x}] + \mathrm{H}[\mathbf{y}] \tag{1.152}$$

with equality if, and only if, $\mathbf{x}$ and $\mathbf{y}$ are statistically independent.

1.32 ($\star$) Consider a vector $\mathbf{x}$ of continuous variables with distribution $p(\mathbf{x})$ and corresponding entropy $\mathrm{H}[\mathbf{x}]$. Suppose that we make a nonsingular linear transformation of $\mathbf{x}$ to obtain a new variable $\mathbf{y} = \mathbf{A}\mathbf{x}$. Show that the corresponding entropy is given by $\mathrm{H}[\mathbf{y}] = \mathrm{H}[\mathbf{x}] + \ln |\mathbf{A}|$ where $|\mathbf{A}|$ denotes the determinant of $\mathbf{A}$.

1.33 ($\star\star$) Suppose that the conditional entropy $\mathrm{H}[y|x]$ between two discrete random variables x and y is zero. Show that, for all values of x such that $p(x) > 0$, the variable y must be a function of x, in other words for each x there is only one value of y such that $p(y|x) \neq 0$.

1.34 ($\star\star$) **www** Use the calculus of variations to show that the stationary point of the functional (1.108) is given by (1.108). Then use the constraints (1.105), (1.106), and (1.107) to eliminate the Lagrange multipliers and hence show that the maximum entropy solution is given by the Gaussian (1.109).

1.35 ($\star$) **www** Use the results (1.106) and (1.107) to show that the entropy of the univariate Gaussian (1.109) is given by (1.110).

1.36 ($\star$) A strictly convex function is defined as one for which every chord lies above the function. Show that this is equivalent to the condition that the second derivative of the function be positive.

1.37 ($\star$) Using the definition (1.111) together with the product rule of probability, prove the result (1.112).

1.38 ($\star\star$) **www** Using proof by induction, show that the inequality (1.114) for convex functions implies the result (1.115).

1.39 ($\star\star\star$) Consider two binary variables x and y having the joint distribution given in Table 1.3.

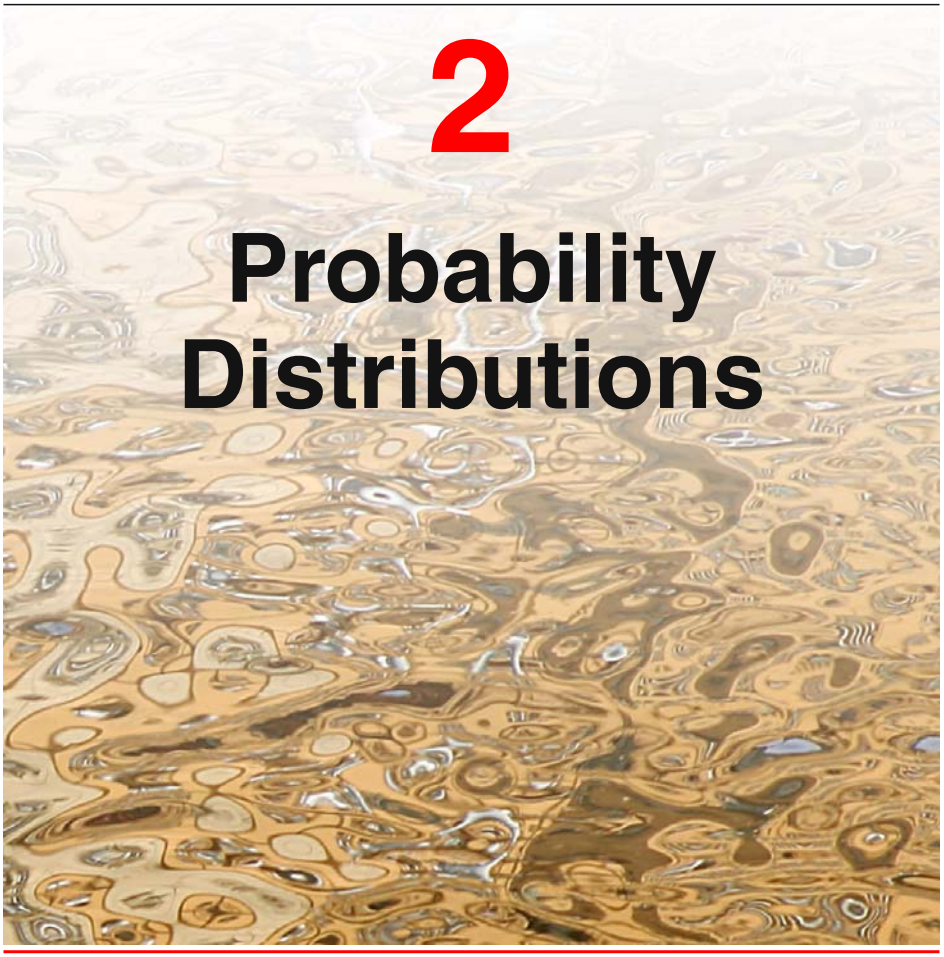
Evaluate the following quantities

(a) $\mathrm{H}[x]$ **(c)** $\mathrm{H}[y|x]$ **(e)** $\mathrm{H}[x, y]$
(b) $\mathrm{H}[y]$ **(d)** $\mathrm{H}[x|y]$ **(f)** $\mathrm{I}[x, y]$.

Draw a diagram to show the relationship between these various quantities.

1.40 ($\star$) By applying Jensen's inequality (1.115) with $f(x) = \ln x$, show that the arithmetic mean of a set of real numbers is never less than their geometrical mean.

1.41 ($\star$) **WWW** Using the sum and product rules of probability, show that the mutual information $I(\mathbf{x}, \mathbf{y})$ satisfies the relation (1.121).



2 Probability Distributions

In Chapter 1, we emphasized the central role played by probability theory in the solution of pattern recognition problems. We turn now to an exploration of some particular examples of probability distributions and their properties. As well as being of great interest in their own right, these distributions can form building blocks for more complex models and will be used extensively throughout the book. The distributions introduced in this chapter will also serve another important purpose, namely to provide us with the opportunity to discuss some key statistical concepts, such as Bayesian inference, in the context of simple models before we encounter them in more complex situations in later chapters.

One role for the distributions discussed in this chapter is to model the probability distribution $p(\mathbf{x})$ of a random variable $\mathbf{x}$, given a finite set $\mathbf{x}_1, \ldots, \mathbf{x}_N$ of observations. This problem is known as *density estimation*. For the purposes of this chapter, we shall assume that the data points are independent and identically distributed. It should be emphasized that the problem of density estimation is fun-

damentally ill-posed, because there are infinitely many probability distributions that could have given rise to the observed finite data set. Indeed, any distribution $p(\mathbf{x})$ that is nonzero at each of the data points $\mathbf{x}_1, \ldots, \mathbf{x}_N$ is a potential candidate. The issue of choosing an appropriate distribution relates to the problem of model selection that has already been encountered in the context of polynomial curve fitting in Chapter 1 and that is a central issue in pattern recognition.

We begin by considering the binomial and multinomial distributions for discrete random variables and the Gaussian distribution for continuous random variables. These are specific examples of *parametric* distributions, so-called because they are governed by a small number of adaptive parameters, such as the mean and variance in the case of a Gaussian for example. To apply such models to the problem of density estimation, we need a procedure for determining suitable values for the parameters, given an observed data set. In a frequentist treatment, we choose specific values for the parameters by optimizing some criterion, such as the likelihood function. By contrast, in a Bayesian treatment we introduce prior distributions over the parameters and then use Bayes' theorem to compute the corresponding posterior distribution given the observed data.

We shall see that an important role is played by *conjugate* priors, that lead to posterior distributions having the same functional form as the prior, and that therefore lead to a greatly simplified Bayesian analysis. For example, the conjugate prior for the parameters of the multinomial distribution is called the *Dirichlet* distribution, while the conjugate prior for the mean of a Gaussian is another Gaussian. All of these distributions are examples of the *exponential family* of distributions, which possess a number of important properties, and which will be discussed in some detail.

One limitation of the parametric approach is that it assumes a specific functional form for the distribution, which may turn out to be inappropriate for a particular application. An alternative approach is given by *nonparametric* density estimation methods in which the form of the distribution typically depends on the size of the data set. Such models still contain parameters, but these control the model complexity rather than the form of the distribution. We end this chapter by considering three nonparametric methods based respectively on histograms, nearest-neighbours, and kernels.

2.1. Binary Variables

We begin by considering a single binary random variable $x \in \{0, 1\}$. For example, x might describe the outcome of flipping a coin, with $x = 1$ representing 'heads', and $x = 0$ representing 'tails'. We can imagine that this is a damaged coin so that the probability of landing heads is not necessarily the same as that of landing tails. The probability of $x = 1$ will be denoted by the parameter μ so that

$$p(x = 1|\mu) = \mu \tag{2.1}$$

where $0 \leqslant \mu \leqslant 1$, from which it follows that $p(x = 0|\mu) = 1 - \mu$. The probability distribution over x can therefore be written in the form

$$\text{Bern}(x|\mu) = \mu^x (1 - \mu)^{1-x} \tag{2.2}$$

Exercise 2.1

which is known as the *Bernoulli* distribution. It is easily verified that this distribution is normalized and that it has mean and variance given by

$$\mathbb{E}[x] = \mu \tag{2.3}$$
$$\text{var}[x] = \mu(1 - \mu). \tag{2.4}$$

Now suppose we have a data set $\mathcal{D} = \{x_1, \ldots, x_N\}$ of observed values of x. We can construct the likelihood function, which is a function of μ, on the assumption that the observations are drawn independently from $p(x|\mu)$, so that

$$p(\mathcal{D}|\mu) = \prod_{n=1}^{N} p(x_n|\mu) = \prod_{n=1}^{N} \mu^{x_n} (1 - \mu)^{1-x_n}. \tag{2.5}$$

In a frequentist setting, we can estimate a value for μ by maximizing the likelihood function, or equivalently by maximizing the logarithm of the likelihood. In the case of the Bernoulli distribution, the log likelihood function is given by

$$\ln p(\mathcal{D}|\mu) = \sum_{n=1}^{N} \ln p(x_n|\mu) = \sum_{n=1}^{N} \{x_n \ln \mu + (1 - x_n) \ln(1 - \mu)\}. \tag{2.6}$$

At this point, it is worth noting that the log likelihood function depends on the N observations x_n only through their sum $\sum_n x_n$. This sum provides an example of a *sufficient statistic* for the data under this distribution, and we shall study the important role of sufficient statistics in some detail. If we set the derivative of $\ln p(\mathcal{D}|\mu)$ with respect to μ equal to zero, we obtain the maximum likelihood estimator

Section 2.4

$$\mu_{\text{ML}} = \frac{1}{N} \sum_{n=1}^{N} x_n \tag{2.7}$$

Jacob Bernoulli
1654–1705

Jacob Bernoulli, also known as Jacques or James Bernoulli, was a Swiss mathematician and was the first of many in the Bernoulli family to pursue a career in science and mathematics. Although compelled to study philosophy and theology against his will by his parents, he travelled extensively after graduating in order to meet with many of the leading scientists of his time, including Boyle and Hooke in England. When he returned to Switzerland, he taught mechanics and became Professor of Mathematics at Basel in 1687. Unfortunately, rivalry between Jacob and his younger brother Johann turned an initially productive collaboration into a bitter and public dispute. Jacob's most significant contributions to mathematics appeared in *The Art of Conjecture* published in 1713, eight years after his death, which deals with topics in probability theory including what has become known as the Bernoulli distribution.

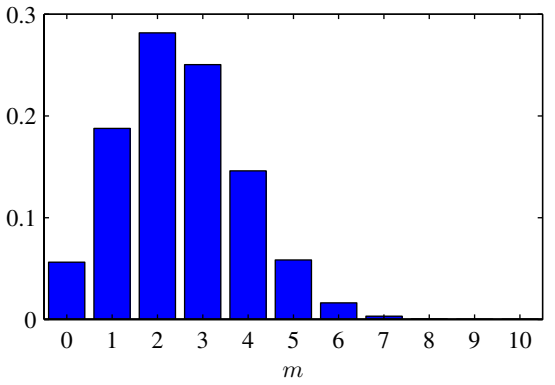


Figure 2.1 Histogram plot of the binomial distribution (2.9) as a function of m for $N = 10$ and $\mu = 0.25$.

which is also known as the *sample mean*. If we denote the number of observations of $x = 1$ (heads) within this data set by m, then we can write (2.7) in the form

$$\mu_{\mathrm{ML}} = \frac{m}{N} \tag{2.8}$$

so that the probability of landing heads is given, in this maximum likelihood framework, by the fraction of observations of heads in the data set.

Now suppose we flip a coin, say, 3 times and happen to observe 3 heads. Then $N = m = 3$ and $\mu_{\mathrm{ML}} = 1$. In this case, the maximum likelihood result would predict that all future observations should give heads. Common sense tells us that this is unreasonable, and in fact this is an extreme example of the over-fitting associated with maximum likelihood. We shall see shortly how to arrive at more sensible conclusions through the introduction of a prior distribution over μ.

We can also work out the distribution of the number m of observations of $x = 1$, given that the data set has size N. This is called the *binomial* distribution, and from (2.5) we see that it is proportional to $\mu^m(1-\mu)^{N-m}$. In order to obtain the normalization coefficient we note that out of N coin flips, we have to add up all of the possible ways of obtaining m heads, so that the binomial distribution can be written

$$\mathrm{Bin}(m|N,\mu) = \binom{N}{m}\mu^m(1-\mu)^{N-m} \tag{2.9}$$

where

$$\binom{N}{m} \equiv \frac{N!}{(N-m)!m!} \tag{2.10}$$

Exercise 2.3

is the number of ways of choosing m objects out of a total of N identical objects. Figure 2.1 shows a plot of the binomial distribution for $N = 10$ and $\mu = 0.25$.

The mean and variance of the binomial distribution can be found by using the result of Exercise 1.10, which shows that for independent events the mean of the sum is the sum of the means, and the variance of the sum is the sum of the variances. Because $m = x_1 + \ldots + x_N$, and for each observation the mean and variance are

given by (2.3) and (2.4), respectively, we have

$$\mathbb{E}[m] \equiv \sum_{m=0}^{N} m\mathrm{Bin}(m|N,\mu) = N\mu \tag{2.11}$$

$$\mathrm{var}[m] \equiv \sum_{m=0}^{N} (m-\mathbb{E}[m])^2 \mathrm{Bin}(m|N,\mu) = N\mu(1-\mu). \tag{2.12}$$

Exercise 2.4 These results can also be proved directly using calculus.

2.1.1 The beta distribution

We have seen in (2.8) that the maximum likelihood setting for the parameter μ in the Bernoulli distribution, and hence in the binomial distribution, is given by the fraction of the observations in the data set having $x = 1$. As we have already noted, this can give severely over-fitted results for small data sets. In order to develop a Bayesian treatment for this problem, we need to introduce a prior distribution $p(\mu)$ over the parameter μ. Here we consider a form of prior distribution that has a simple interpretation as well as some useful analytical properties. To motivate this prior, we note that the likelihood function takes the form of the product of factors of the form $\mu^x(1-\mu)^{1-x}$. If we choose a prior to be proportional to powers of μ and $(1-\mu)$, then the posterior distribution, which is proportional to the product of the prior and the likelihood function, will have the same functional form as the prior. This property is called *conjugacy* and we will see several examples of it later in this chapter. We therefore choose a prior, called the *beta* distribution, given by

$$\mathrm{Beta}(\mu|a,b) = \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)}\mu^{a-1}(1-\mu)^{b-1} \tag{2.13}$$

where $\Gamma(x)$ is the gamma function defined by (1.141), and the coefficient in (2.13) *Exercise 2.5* ensures that the beta distribution is normalized, so that

$$\int_0^1 \mathrm{Beta}(\mu|a,b)\,\mathrm{d}\mu = 1. \tag{2.14}$$

Exercise 2.6 The mean and variance of the beta distribution are given by

$$\mathbb{E}[\mu] = \frac{a}{a+b} \tag{2.15}$$

$$\mathrm{var}[\mu] = \frac{ab}{(a+b)^2(a+b+1)}. \tag{2.16}$$

The parameters a and b are often called *hyperparameters* because they control the distribution of the parameter μ. Figure 2.2 shows plots of the beta distribution for various values of the hyperparameters.

The posterior distribution of μ is now obtained by multiplying the beta prior (2.13) by the binomial likelihood function (2.9) and normalizing. Keeping only the factors that depend on μ, we see that this posterior distribution has the form

$$p(\mu|m,l,a,b) \propto \mu^{m+a-1}(1-\mu)^{l+b-1} \tag{2.17}$$

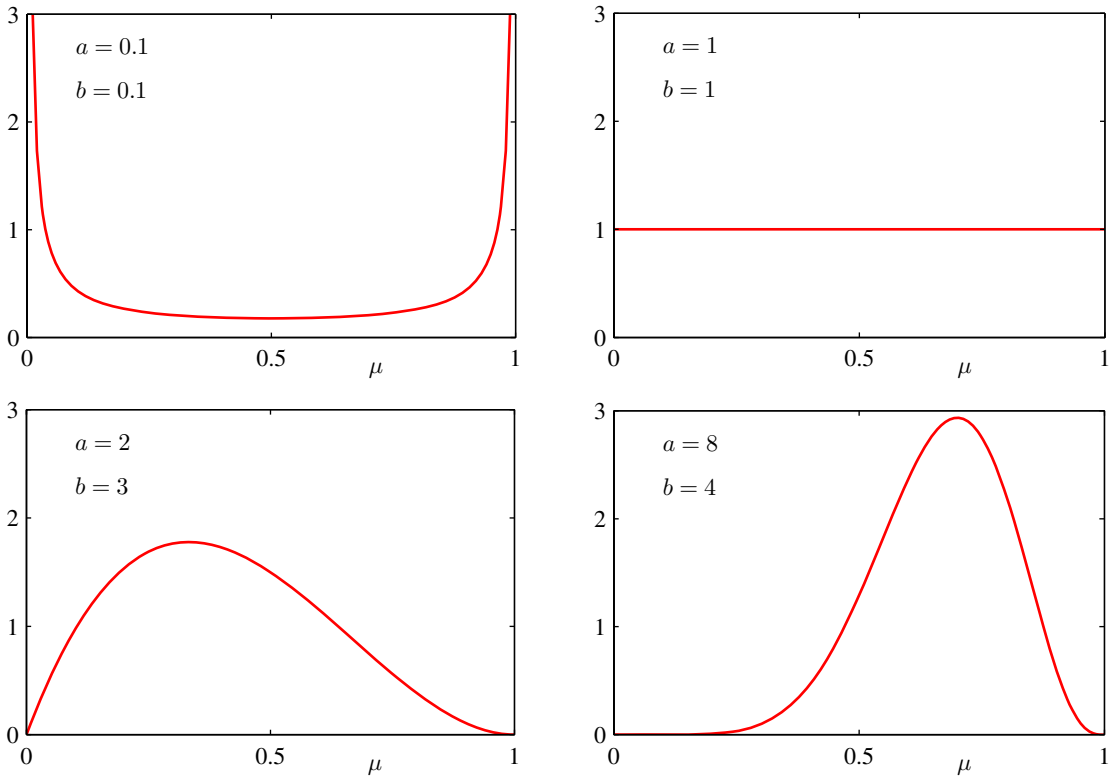


Figure 2.2 Plots of the beta distribution $\mathrm{Beta}(\mu|a,b)$ given by (2.13) as a function of μ for various values of the hyperparameters a and b.

where $l = N - m$, and therefore corresponds to the number of 'tails' in the coin example. We see that (2.17) has the same functional dependence on μ as the prior distribution, reflecting the conjugacy properties of the prior with respect to the likelihood function. Indeed, it is simply another beta distribution, and its normalization coefficient can therefore be obtained by comparison with (2.13) to give

$$p(\mu|m,l,a,b) = \frac{\Gamma(m+a+l+b)}{\Gamma(m+a)\Gamma(l+b)}\mu^{m+a-1}(1-\mu)^{l+b-1}. \tag{2.18}$$

We see that the effect of observing a data set of m observations of $x = 1$ and l observations of $x = 0$ has been to increase the value of a by m, and the value of b by l, in going from the prior distribution to the posterior distribution. This allows us to provide a simple interpretation of the hyperparameters a and b in the prior as an *effective number of observations* of $x = 1$ and $x = 0$, respectively. Note that a and b need not be integers. Furthermore, the posterior distribution can act as the prior if we subsequently observe additional data. To see this, we can imagine taking observations one at a time and after each observation updating the current posterior

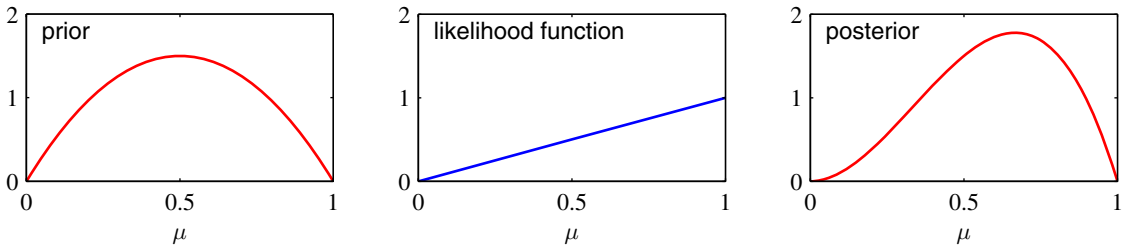


Figure 2.3 Illustration of one step of sequential Bayesian inference. The prior is given by a beta distribution with parameters $a = 2$, $b = 2$, and the likelihood function, given by (2.9) with $N = m = 1$, corresponds to a single observation of $x = 1$, so that the posterior is given by a beta distribution with parameters $a = 3$, $b = 2$.

distribution by multiplying by the likelihood function for the new observation and then normalizing to obtain the new, revised posterior distribution. At each stage, the posterior is a beta distribution with some total number of (prior and actual) observed values for $x = 1$ and $x = 0$ given by the parameters a and b. Incorporation of an additional observation of $x = 1$ simply corresponds to incrementing the value of a by 1, whereas for an observation of $x = 0$ we increment b by 1. Figure 2.3 illustrates one step in this process.

We see that this *sequential* approach to learning arises naturally when we adopt a Bayesian viewpoint. It is independent of the choice of prior and of the likelihood function and depends only on the assumption of i.i.d. data. Sequential methods make use of observations one at a time, or in small batches, and then discard them before the next observations are used. They can be used, for example, in real-time learning scenarios where a steady stream of data is arriving, and predictions must be made before all of the data is seen. Because they do not require the whole data set to be stored or loaded into memory, sequential methods are also useful for large data sets. Maximum likelihood methods can also be cast into a sequential framework.

Section 2.3.5

If our goal is to predict, as best we can, the outcome of the next trial, then we must evaluate the predictive distribution of x, given the observed data set $\mathcal{D}$. From the sum and product rules of probability, this takes the form

$$p(x = 1|\mathcal{D}) = \int_0^1 p(x = 1|\mu)p(\mu|\mathcal{D})\,\mathrm{d}\mu = \int_0^1 \mu p(\mu|\mathcal{D})\,\mathrm{d}\mu = \mathbb{E}[\mu|\mathcal{D}]. \quad (2.19)$$

Using the result (2.18) for the posterior distribution $p(\mu|\mathcal{D})$, together with the result (2.15) for the mean of the beta distribution, we obtain

$$p(x = 1|\mathcal{D}) = \frac{m + a}{m + a + l + b} \quad (2.20)$$

which has a simple interpretation as the total fraction of observations (both real observations and fictitious prior observations) that correspond to $x = 1$. Note that in the limit of an infinitely large data set $m, l \to \infty$ the result (2.20) reduces to the maximum likelihood result (2.8). As we shall see, it is a very general property that the Bayesian and maximum likelihood results will agree in the limit of an infinitely

large data set. For a finite data set, the posterior mean for μ always lies between the prior mean and the maximum likelihood estimate for μ corresponding to the relative frequencies of events given by (2.7).

Exercise 2.7

From Figure 2.2, we see that as the number of observations increases, so the posterior distribution becomes more sharply peaked. This can also be seen from the result (2.16) for the variance of the beta distribution, in which we see that the variance goes to zero for $a \to \infty$ or $b \to \infty$. In fact, we might wonder whether it is a general property of Bayesian learning that, as we observe more and more data, the uncertainty represented by the posterior distribution will steadily decrease.

To address this, we can take a frequentist view of Bayesian learning and show that, on average, such a property does indeed hold. Consider a general Bayesian inference problem for a parameter $\boldsymbol{\theta}$ for which we have observed a data set $\mathcal{D}$, described by the joint distribution $p(\boldsymbol{\theta}, \mathcal{D})$. The following result

Exercise 2.8

$$\mathbb{E}_{\boldsymbol{\theta}}[\boldsymbol{\theta}] = \mathbb{E}_{\mathcal{D}}\left[\mathbb{E}_{\boldsymbol{\theta}}[\boldsymbol{\theta}|\mathcal{D}]\right] \tag{2.21}$$

where

$$\mathbb{E}_{\boldsymbol{\theta}}[\boldsymbol{\theta}] \equiv \int p(\boldsymbol{\theta})\boldsymbol{\theta}\,\mathrm{d}\boldsymbol{\theta} \tag{2.22}$$

$$\mathbb{E}_{\mathcal{D}}[\mathbb{E}_{\boldsymbol{\theta}}[\boldsymbol{\theta}|\mathcal{D}]] \equiv \int \left\{ \int \boldsymbol{\theta} p(\boldsymbol{\theta}|\mathcal{D})\,\mathrm{d}\boldsymbol{\theta} \right\} p(\mathcal{D})\,\mathrm{d}\mathcal{D} \tag{2.23}$$

says that the posterior mean of $\boldsymbol{\theta}$, averaged over the distribution generating the data, is equal to the prior mean of $\boldsymbol{\theta}$. Similarly, we can show that

$$\mathrm{var}_{\boldsymbol{\theta}}[\boldsymbol{\theta}] = \mathbb{E}_{\mathcal{D}}\left[\mathrm{var}_{\boldsymbol{\theta}}[\boldsymbol{\theta}|\mathcal{D}]\right] + \mathrm{var}_{\mathcal{D}}\left[\mathbb{E}_{\boldsymbol{\theta}}[\boldsymbol{\theta}|\mathcal{D}]\right]. \tag{2.24}$$

The term on the left-hand side of (2.24) is the prior variance of $\boldsymbol{\theta}$. On the right-hand side, the first term is the average posterior variance of $\boldsymbol{\theta}$, and the second term measures the variance in the posterior mean of $\boldsymbol{\theta}$. Because this variance is a positive quantity, this result shows that, on average, the posterior variance of $\boldsymbol{\theta}$ is smaller than the prior variance. The reduction in variance is greater if the variance in the posterior mean is greater. Note, however, that this result only holds on average, and that for a particular observed data set it is possible for the posterior variance to be larger than the prior variance.

2.2. Multinomial Variables

Binary variables can be used to describe quantities that can take one of two possible values. Often, however, we encounter discrete variables that can take on one of K possible mutually exclusive states. Although there are various alternative ways to express such variables, we shall see shortly that a particularly convenient representation is the 1-of-K scheme in which the variable is represented by a K-dimensional vector $\mathbf{x}$ in which one of the elements x_k equals 1, and all remaining elements equal

0. So, for instance if we have a variable that can take $K = 6$ states and a particular observation of the variable happens to correspond to the state where $x_3 = 1$, then $\mathbf{x}$ will be represented by

$$\mathbf{x} = (0, 0, 1, 0, 0, 0)^{\mathrm{T}}. \tag{2.25}$$

Note that such vectors satisfy $\sum_{k=1}^{K} x_k = 1$. If we denote the probability of $x_k = 1$ by the parameter μ_k, then the distribution of $\mathbf{x}$ is given

$$p(\mathbf{x}|\boldsymbol{\mu}) = \prod_{k=1}^{K} \mu_k^{x_k} \tag{2.26}$$

where $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_K)^{\mathrm{T}}$, and the parameters μ_k are constrained to satisfy $\mu_k \geqslant 0$ and $\sum_k \mu_k = 1$, because they represent probabilities. The distribution (2.26) can be regarded as a generalization of the Bernoulli distribution to more than two outcomes. It is easily seen that the distribution is normalized

$$\sum_{\mathbf{x}} p(\mathbf{x}|\boldsymbol{\mu}) = \sum_{k=1}^{K} \mu_k = 1 \tag{2.27}$$

and that

$$\mathbb{E}[\mathbf{x}|\boldsymbol{\mu}] = \sum_{\mathbf{x}} p(\mathbf{x}|\boldsymbol{\mu})\mathbf{x} = (\mu_1, \ldots, \mu_M)^{\mathrm{T}} = \boldsymbol{\mu}. \tag{2.28}$$

Now consider a data set $\mathcal{D}$ of N independent observations $\mathbf{x}_1, \ldots, \mathbf{x}_N$. The corresponding likelihood function takes the form

$$p(\mathcal{D}|\boldsymbol{\mu}) = \prod_{n=1}^{N} \prod_{k=1}^{K} \mu_k^{x_{nk}} = \prod_{k=1}^{K} \mu_k^{\left(\sum_n x_{nk}\right)} = \prod_{k=1}^{K} \mu_k^{m_k}. \tag{2.29}$$

We see that the likelihood function depends on the N data points only through the K quantities

$$m_k = \sum_n x_{nk} \tag{2.30}$$

which represent the number of observations of $x_k = 1$. These are called the *sufficient statistics* for this distribution.

Section 2.4

In order to find the maximum likelihood solution for $\boldsymbol{\mu}$, we need to maximize $\ln p(\mathcal{D}|\boldsymbol{\mu})$ with respect to μ_k taking account of the constraint that the μ_k must sum to one. This can be achieved using a Lagrange multiplier λ and maximizing

Appendix E

$$\sum_{k=1}^{K} m_k \ln \mu_k + \lambda \left(\sum_{k=1}^{K} \mu_k - 1 \right). \tag{2.31}$$

Setting the derivative of (2.31) with respect to μ_k to zero, we obtain

$$\mu_k = -m_k/\lambda. \tag{2.32}$$

We can solve for the Lagrange multiplier λ by substituting (2.32) into the constraint $\sum_k \mu_k = 1$ to give $\lambda = -N$. Thus we obtain the maximum likelihood solution in the form

$$\mu_k^{\mathrm{ML}} = \frac{m_k}{N} \tag{2.33}$$

which is the fraction of the N observations for which $x_k = 1$.

We can consider the joint distribution of the quantities $m_1, \ldots, m_K$, conditioned on the parameters $\boldsymbol{\mu}$ and on the total number N of observations. From (2.29) this takes the form

$$\mathrm{Mult}(m_1, m_2, \ldots, m_K|\boldsymbol{\mu}, N) = \binom{N}{m_1 m_2 \ldots m_K} \prod_{k=1}^{K} \mu_k^{m_k} \tag{2.34}$$

which is known as the *multinomial* distribution. The normalization coefficient is the number of ways of partitioning N objects into K groups of size $m_1, \ldots, m_K$ and is given by

$$\binom{N}{m_1 m_2 \ldots m_K} = \frac{N!}{m_1! m_2! \ldots m_K!}. \tag{2.35}$$

Note that the variables m_k are subject to the constraint

$$\sum_{k=1}^{K} m_k = N. \tag{2.36}$$

2.2.1 The Dirichlet distribution

We now introduce a family of prior distributions for the parameters $\{\mu_k\}$ of the multinomial distribution (2.34). By inspection of the form of the multinomial distribution, we see that the conjugate prior is given by

$$p(\boldsymbol{\mu}|\boldsymbol{\alpha}) \propto \prod_{k=1}^{K} \mu_k^{\alpha_k - 1} \tag{2.37}$$

where $0 \leqslant \mu_k \leqslant 1$ and $\sum_k \mu_k = 1$. Here $\alpha_1, \ldots, \alpha_K$ are the parameters of the distribution, and $\boldsymbol{\alpha}$ denotes $(\alpha_1, \ldots, \alpha_K)^{\mathrm{T}}$. Note that, because of the summation constraint, the distribution over the space of the $\{\mu_k\}$ is confined to a *simplex* of dimensionality $K - 1$, as illustrated for $K = 3$ in Figure 2.4.

Exercise 2.9

The normalized form for this distribution is by

$$\mathrm{Dir}(\boldsymbol{\mu}|\boldsymbol{\alpha}) = \frac{\Gamma(\alpha_0)}{\Gamma(\alpha_1) \cdots \Gamma(\alpha_K)} \prod_{k=1}^{K} \mu_k^{\alpha_k - 1} \tag{2.38}$$

which is called the *Dirichlet* distribution. Here $\Gamma(x)$ is the gamma function defined by (1.141) while

$$\alpha_0 = \sum_{k=1}^{K} \alpha_k. \tag{2.39}$$

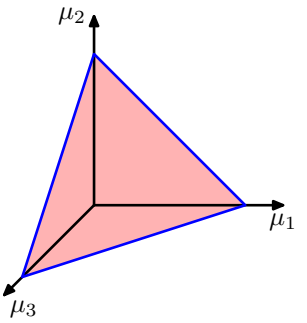


Figure 2.4 The Dirichlet distribution over three variables μ_1, μ_2, μ_3 is confined to a simplex (a bounded linear manifold) of the form shown, as a consequence of the constraints $0 \leqslant \mu_k \leqslant 1$ and $\sum_k \mu_k = 1$.

Plots of the Dirichlet distribution over the simplex, for various settings of the parameters α_k, are shown in Figure 2.5.

Multiplying the prior (2.38) by the likelihood function (2.34), we obtain the posterior distribution for the parameters $\{\mu_k\}$ in the form

$$p(\boldsymbol{\mu}|\mathcal{D}, \boldsymbol{\alpha}) \propto p(\mathcal{D}|\boldsymbol{\mu})p(\boldsymbol{\mu}|\boldsymbol{\alpha}) \propto \prod_{k=1}^{K} \mu_k^{\alpha_k + m_k - 1}. \tag{2.40}$$

We see that the posterior distribution again takes the form of a Dirichlet distribution, confirming that the Dirichlet is indeed a conjugate prior for the multinomial. This allows us to determine the normalization coefficient by comparison with (2.38) so that

$$\begin{aligned} p(\boldsymbol{\mu}|\mathcal{D}, \boldsymbol{\alpha}) &= \mathrm{Dir}(\boldsymbol{\mu}|\boldsymbol{\alpha} + \mathbf{m}) \\ &= \frac{\Gamma(\alpha_0 + N)}{\Gamma(\alpha_1 + m_1) \cdots \Gamma(\alpha_K + m_K)} \prod_{k=1}^{K} \mu_k^{\alpha_k + m_k - 1} \end{aligned} \tag{2.41}$$

where we have denoted $\mathbf{m} = (m_1, \ldots, m_K)^{\mathrm{T}}$. As for the case of the binomial distribution with its beta prior, we can interpret the parameters α_k of the Dirichlet prior as an effective number of observations of $x_k = 1$.

Note that two-state quantities can either be represented as binary variables and

Lejeune Dirichlet
1805–1859

Johann Peter Gustav Lejeune Dirichlet was a modest and reserved mathematician who made contributions in number theory, mechanics, and astronomy, and who gave the first rigorous analysis of Fourier series. His family originated from Richelet in Belgium, and the name Lejeune Dirichlet comes from 'le jeune de Richelet' (the young person from Richelet). Dirichlet's first paper, which was published in 1825, brought him instant fame. It concerned Fermat's last theorem, which claims that there are no positive integer solutions to $x^n + y^n = z^n$ for $n > 2$. Dirichlet gave a partial proof for the case $n = 5$, which was sent to Legendre for review and who in turn completed the proof. Later, Dirichlet gave a complete proof for $n = 14$, although a full proof of Fermat's last theorem for arbitrary n had to wait until the work of Andrew Wiles in the closing years of the 20$^{\mathrm{th}}$ century.

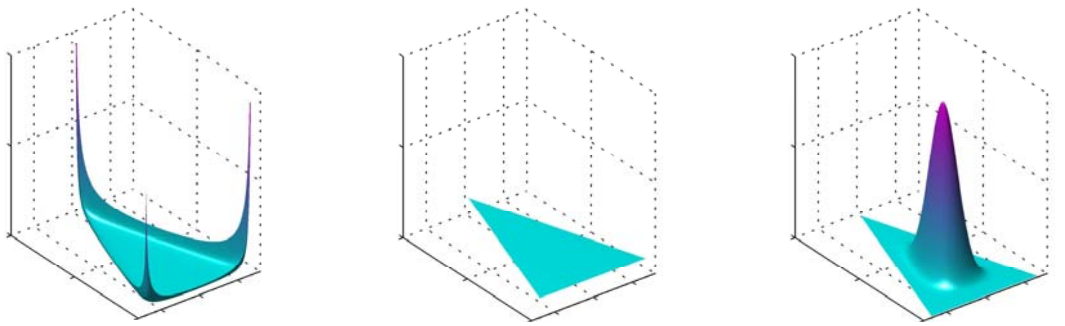


Figure 2.5 Plots of the Dirichlet distribution over three variables, where the two horizontal axes are coordinates in the plane of the simplex and the vertical axis corresponds to the value of the density. Here $\{\alpha_k\} = 0.1$ on the left plot, $\{\alpha_k\} = 1$ in the centre plot, and $\{\alpha_k\} = 10$ in the right plot.

modelled using the binomial distribution (2.9) or as 1-of-2 variables and modelled using the multinomial distribution (2.34) with $K = 2$.

2.3. The Gaussian Distribution

The Gaussian, also known as the normal distribution, is a widely used model for the distribution of continuous variables. In the case of a single variable x, the Gaussian distribution can be written in the form

$$\mathcal{N}(x|\mu, \sigma^2) = \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left\{-\frac{1}{2\sigma^2}(x-\mu)^2\right\} \tag{2.42}$$

where μ is the mean and σ^2 is the variance. For a D-dimensional vector $\mathbf{x}$, the multivariate Gaussian distribution takes the form

$$\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{D/2}} \frac{1}{|\boldsymbol{\Sigma}|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right\} \tag{2.43}$$

where $\boldsymbol{\mu}$ is a D-dimensional mean vector, $\boldsymbol{\Sigma}$ is a $D \times D$ covariance matrix, and $|\boldsymbol{\Sigma}|$ denotes the determinant of $\boldsymbol{\Sigma}$.

The Gaussian distribution arises in many different contexts and can be motivated from a variety of different perspectives. For example, we have already seen that for a single real variable, the distribution that maximizes the entropy is the Gaussian. This property applies also to the multivariate Gaussian.

Section 1.6

Exercise 2.14

Another situation in which the Gaussian distribution arises is when we consider the sum of multiple random variables. The *central limit theorem* (due to Laplace) tells us that, subject to certain mild conditions, the sum of a set of random variables, which is of course itself a random variable, has a distribution that becomes increasingly Gaussian as the number of terms in the sum increases (Walker, 1969). We can

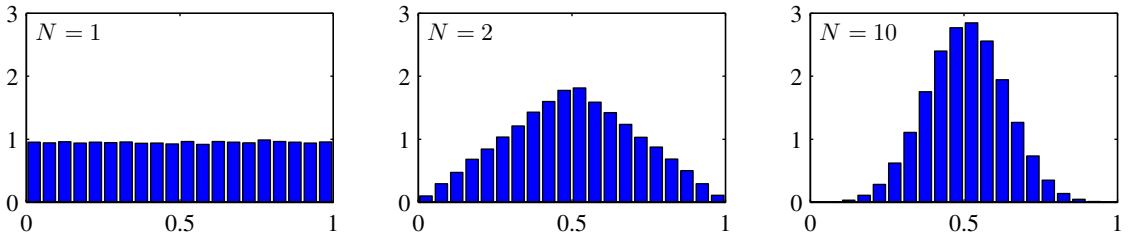


Figure 2.6 Histogram plots of the mean of N uniformly distributed numbers for various values of N. We observe that as N increases, the distribution tends towards a Gaussian.

illustrate this by considering N variables $x_1, \ldots, x_N$ each of which has a uniform distribution over the interval $[0, 1]$ and then considering the distribution of the mean $(x_1 + \cdots + x_N)/N$. For large N, this distribution tends to a Gaussian, as illustrated in Figure 2.6. In practice, the convergence to a Gaussian as N increases can be very rapid. One consequence of this result is that the binomial distribution (2.9), which is a distribution over m defined by the sum of N observations of the random binary variable x, will tend to a Gaussian as $N \to \infty$ (see Figure 2.1 for the case of $N = 10$).

The Gaussian distribution has many important analytical properties, and we shall consider several of these in detail. As a result, this section will be rather more technically involved than some of the earlier sections, and will require familiarity with various matrix identities. However, we strongly encourage the reader to become proficient in manipulating Gaussian distributions using the techniques presented here as this will prove invaluable in understanding the more complex models presented in later chapters.

Appendix C

We begin by considering the geometrical form of the Gaussian distribution. The

Carl Friedrich Gauss
1777–1855

It is said that when Gauss went to elementary school at age 7, his teacher Büttner, trying to keep the class occupied, asked the pupils to sum the integers from 1 to 100. To the teacher's amazement, Gauss arrived at the answer in a matter of moments by noting that the sum can be represented as 50 pairs ($1 + 100$, $2 + 99$, etc.) each of which added to 101, giving the answer 5,050. It is now believed that the problem which was actually set was of the same form but somewhat harder in that the sequence had a larger starting value and a larger increment. Gauss was a German mathematician and scientist with a reputation for being a hard-working perfectionist. One of his many contributions was to show that least squares can be derived under the assumption of normally distributed errors. He also created an early formulation of non-Euclidean geometry (a self-consistent geometrical theory that violates the axioms of Euclid) but was reluctant to discuss it openly for fear that his reputation might suffer if it were seen that he believed in such a geometry. At one point, Gauss was asked to conduct a geodetic survey of the state of Hanover, which led to his formulation of the normal distribution, now also known as the Gaussian. After his death, a study of his diaries revealed that he had discovered several important mathematical results years or even decades before they were published by others.

functional dependence of the Gaussian on $\mathbf{x}$ is through the quadratic form

$$\Delta^2 = (\mathbf{x} - \boldsymbol{\mu})^{\mathrm{T}} \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}) \tag{2.44}$$

which appears in the exponent. The quantity Δ is called the *Mahalanobis distance* from $\boldsymbol{\mu}$ to $\mathbf{x}$ and reduces to the Euclidean distance when $\boldsymbol{\Sigma}$ is the identity matrix. The Gaussian distribution will be constant on surfaces in $\mathbf{x}$-space for which this quadratic form is constant.

First of all, we note that the matrix $\boldsymbol{\Sigma}$ can be taken to be symmetric, without loss of generality, because any antisymmetric component would disappear from the exponent. Now consider the eigenvector equation for the covariance matrix

Exercise 2.17

$$\boldsymbol{\Sigma} \mathbf{u}_i = \lambda_i \mathbf{u}_i \tag{2.45}$$

where $i = 1, \ldots, D$. Because $\boldsymbol{\Sigma}$ is a real, symmetric matrix its eigenvalues will be real, and its eigenvectors can be chosen to form an orthonormal set, so that

Exercise 2.18

$$\mathbf{u}_i^{\mathrm{T}} \mathbf{u}_j = I_{ij} \tag{2.46}$$

where I_{ij} is the i, j element of the identity matrix and satisfies

$$I_{ij} = \begin{cases} 1, & \text{if } i = j \\ 0, & \text{otherwise.} \end{cases} \tag{2.47}$$

The covariance matrix $\boldsymbol{\Sigma}$ can be expressed as an expansion in terms of its eigenvectors in the form

Exercise 2.19

$$\boldsymbol{\Sigma} = \sum_{i=1}^{D} \lambda_i \mathbf{u}_i \mathbf{u}_i^{\mathrm{T}} \tag{2.48}$$

and similarly the inverse covariance matrix $\boldsymbol{\Sigma}^{-1}$ can be expressed as

$$\boldsymbol{\Sigma}^{-1} = \sum_{i=1}^{D} \frac{1}{\lambda_i} \mathbf{u}_i \mathbf{u}_i^{\mathrm{T}}. \tag{2.49}$$

Substituting (2.49) into (2.44), the quadratic form becomes

$$\Delta^2 = \sum_{i=1}^{D} \frac{y_i^2}{\lambda_i} \tag{2.50}$$

where we have defined

$$y_i = \mathbf{u}_i^{\mathrm{T}} (\mathbf{x} - \boldsymbol{\mu}). \tag{2.51}$$

We can interpret $\{y_i\}$ as a new coordinate system defined by the orthonormal vectors $\mathbf{u}_i$ that are shifted and rotated with respect to the original x_i coordinates. Forming the vector $\mathbf{y} = (y_1, \ldots, y_D)^{\mathrm{T}}$, we have

$$\mathbf{y} = \mathbf{U}(\mathbf{x} - \boldsymbol{\mu}) \tag{2.52}$$

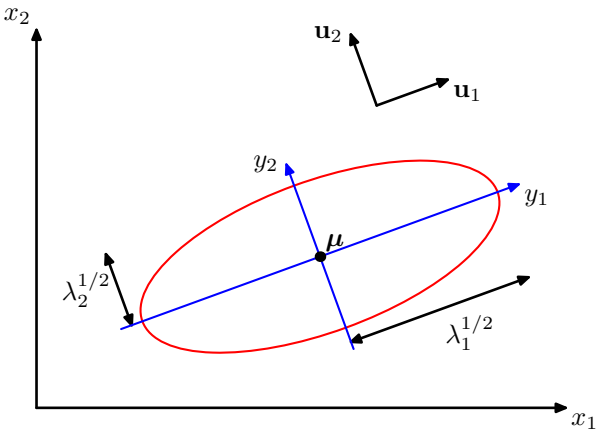


Figure 2.7 The red curve shows the elliptical surface of constant probability density for a Gaussian in a two-dimensional space $\mathbf{x} = (x_1, x_2)$ on which the density is $\exp(-1/2)$ of its value at $\mathbf{x} = \boldsymbol{\mu}$. The major axes of the ellipse are defined by the eigenvectors $\mathbf{u}_i$ of the covariance matrix, with corresponding eigenvalues λ_i.

Appendix C

where $\mathbf{U}$ is a matrix whose rows are given by $\mathbf{u}_i^{\mathrm{T}}$. From (2.46) it follows that $\mathbf{U}$ is an *orthogonal* matrix, i.e., it satisfies $\mathbf{U}\mathbf{U}^{\mathrm{T}} = \mathbf{I}$, and hence also $\mathbf{U}^{\mathrm{T}}\mathbf{U} = \mathbf{I}$, where $\mathbf{I}$ is the identity matrix.

The quadratic form, and hence the Gaussian density, will be constant on surfaces for which (2.51) is constant. If all of the eigenvalues λ_i are positive, then these surfaces represent ellipsoids, with their centres at $\boldsymbol{\mu}$ and their axes oriented along $\mathbf{u}_i$, and with scaling factors in the directions of the axes given by $\lambda_i^{1/2}$, as illustrated in Figure 2.7.

For the Gaussian distribution to be well defined, it is necessary for all of the eigenvalues λ_i of the covariance matrix to be strictly positive, otherwise the distribution cannot be properly normalized. A matrix whose eigenvalues are strictly positive is said to be *positive definite*. In Chapter 12, we will encounter Gaussian distributions for which one or more of the eigenvalues are zero, in which case the distribution is singular and is confined to a subspace of lower dimensionality. If all of the eigenvalues are nonnegative, then the covariance matrix is said to be *positive semidefinite*.

Now consider the form of the Gaussian distribution in the new coordinate system defined by the y_i. In going from the $\mathbf{x}$ to the $\mathbf{y}$ coordinate system, we have a Jacobian matrix $\mathbf{J}$ with elements given by

$$J_{ij} = \frac{\partial x_i}{\partial y_j} = U_{ji} \tag{2.53}$$

where U_{ji} are the elements of the matrix $\mathbf{U}^{\mathrm{T}}$. Using the orthonormality property of the matrix $\mathbf{U}$, we see that the square of the determinant of the Jacobian matrix is

$$|\mathbf{J}|^2 = \left|\mathbf{U}^{\mathrm{T}}\right|^2 = \left|\mathbf{U}^{\mathrm{T}}\right||\mathbf{U}| = \left|\mathbf{U}^{\mathrm{T}}\mathbf{U}\right| = |\mathbf{I}| = 1 \tag{2.54}$$

and hence $|\mathbf{J}| = 1$. Also, the determinant $|\boldsymbol{\Sigma}|$ of the covariance matrix can be written

as the product of its eigenvalues, and hence

$$|\boldsymbol{\Sigma}|^{1/2} = \prod_{j=1}^{D} \lambda_j^{1/2}. \tag{2.55}$$

Thus in the y_j coordinate system, the Gaussian distribution takes the form

$$p(\mathbf{y}) = p(\mathbf{x})|\mathbf{J}| = \prod_{j=1}^{D} \frac{1}{(2\pi\lambda_j)^{1/2}} \exp\left\{-\frac{y_j^2}{2\lambda_j}\right\} \tag{2.56}$$

which is the product of D independent univariate Gaussian distributions. The eigenvectors therefore define a new set of shifted and rotated coordinates with respect to which the joint probability distribution factorizes into a product of independent distributions. The integral of the distribution in the $\mathbf{y}$ coordinate system is then

$$\int p(\mathbf{y})\,\mathrm{d}\mathbf{y} = \prod_{j=1}^{D} \int_{-\infty}^{\infty} \frac{1}{(2\pi\lambda_j)^{1/2}} \exp\left\{-\frac{y_j^2}{2\lambda_j}\right\} \mathrm{d}y_j = 1 \tag{2.57}$$

where we have used the result (1.48) for the normalization of the univariate Gaussian. This confirms that the multivariate Gaussian (2.43) is indeed normalized.

We now look at the moments of the Gaussian distribution and thereby provide an interpretation of the parameters $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$. The expectation of $\mathbf{x}$ under the Gaussian distribution is given by

$$\begin{aligned}\mathbb{E}[\mathbf{x}] &= \frac{1}{(2\pi)^{D/2}} \frac{1}{|\boldsymbol{\Sigma}|^{1/2}} \int \exp\left\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right\} \mathbf{x}\,\mathrm{d}\mathbf{x} \\ &= \frac{1}{(2\pi)^{D/2}} \frac{1}{|\boldsymbol{\Sigma}|^{1/2}} \int \exp\left\{-\frac{1}{2}\mathbf{z}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\mathbf{z}\right\} (\mathbf{z}+\boldsymbol{\mu})\,\mathrm{d}\mathbf{z}\end{aligned} \tag{2.58}$$

where we have changed variables using $\mathbf{z} = \mathbf{x} - \boldsymbol{\mu}$. We now note that the exponent is an even function of the components of $\mathbf{z}$ and, because the integrals over these are taken over the range $(-\infty, \infty)$, the term in $\mathbf{z}$ in the factor $(\mathbf{z} + \boldsymbol{\mu})$ will vanish by symmetry. Thus

$$\mathbb{E}[\mathbf{x}] = \boldsymbol{\mu} \tag{2.59}$$

and so we refer to $\boldsymbol{\mu}$ as the mean of the Gaussian distribution.

We now consider second order moments of the Gaussian. In the univariate case, we considered the second order moment given by $\mathbb{E}[x^2]$. For the multivariate Gaussian, there are D^2 second order moments given by $\mathbb{E}[x_i x_j]$, which we can group together to form the matrix $\mathbb{E}[\mathbf{x}\mathbf{x}^{\mathrm{T}}]$. This matrix can be written as

$$\begin{aligned}\mathbb{E}[\mathbf{x}\mathbf{x}^{\mathrm{T}}] &= \frac{1}{(2\pi)^{D/2}} \frac{1}{|\boldsymbol{\Sigma}|^{1/2}} \int \exp\left\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right\} \mathbf{x}\mathbf{x}^{\mathrm{T}}\,\mathrm{d}\mathbf{x} \\ &= \frac{1}{(2\pi)^{D/2}} \frac{1}{|\boldsymbol{\Sigma}|^{1/2}} \int \exp\left\{-\frac{1}{2}\mathbf{z}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\mathbf{z}\right\} (\mathbf{z}+\boldsymbol{\mu})(\mathbf{z}+\boldsymbol{\mu})^{\mathrm{T}}\,\mathrm{d}\mathbf{z}\end{aligned}$$

where again we have changed variables using $\mathbf{z} = \mathbf{x} - \boldsymbol{\mu}$. Note that the cross-terms involving $\boldsymbol{\mu}\mathbf{z}^{\mathrm{T}}$ and $\boldsymbol{\mu}^{\mathrm{T}}\mathbf{z}$ will again vanish by symmetry. The term $\boldsymbol{\mu}\boldsymbol{\mu}^{\mathrm{T}}$ is constant and can be taken outside the integral, which itself is unity because the Gaussian distribution is normalized. Consider the term involving $\mathbf{z}\mathbf{z}^{\mathrm{T}}$. Again, we can make use of the eigenvector expansion of the covariance matrix given by (2.45), together with the completeness of the set of eigenvectors, to write

$$\mathbf{z} = \sum_{j=1}^{D} y_j \mathbf{u}_j \tag{2.60}$$

where $y_j = \mathbf{u}_j^{\mathrm{T}}\mathbf{z}$, which gives

$$\begin{aligned} &\frac{1}{(2\pi)^{D/2}} \frac{1}{|\boldsymbol{\Sigma}|^{1/2}} \int \exp\left\{-\frac{1}{2}\mathbf{z}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\mathbf{z}\right\} \mathbf{z}\mathbf{z}^{\mathrm{T}}\,\mathrm{d}\mathbf{z} \\ &= \frac{1}{(2\pi)^{D/2}} \frac{1}{|\boldsymbol{\Sigma}|^{1/2}} \sum_{i=1}^{D}\sum_{j=1}^{D} \mathbf{u}_i\mathbf{u}_j^{\mathrm{T}} \int \exp\left\{-\sum_{k=1}^{D}\frac{y_k^2}{2\lambda_k}\right\} y_i y_j\,\mathrm{d}\mathbf{y} \\ &= \sum_{i=1}^{D} \mathbf{u}_i\mathbf{u}_i^{\mathrm{T}}\lambda_i = \boldsymbol{\Sigma} \end{aligned} \tag{2.61}$$

where we have made use of the eigenvector equation (2.45), together with the fact that the integral on the right-hand side of the middle line vanishes by symmetry unless $i = j$, and in the final line we have made use of the results (1.50) and (2.55), together with (2.48). Thus we have

$$\mathbb{E}[\mathbf{x}\mathbf{x}^{\mathrm{T}}] = \boldsymbol{\mu}\boldsymbol{\mu}^{\mathrm{T}} + \boldsymbol{\Sigma}. \tag{2.62}$$

For single random variables, we subtracted the mean before taking second moments in order to define a variance. Similarly, in the multivariate case it is again convenient to subtract off the mean, giving rise to the *covariance* of a random vector $\mathbf{x}$ defined by

$$\mathrm{cov}[\mathbf{x}] = \mathbb{E}\left[(\mathbf{x} - \mathbb{E}[\mathbf{x}])(\mathbf{x} - \mathbb{E}[\mathbf{x}])^{\mathrm{T}}\right]. \tag{2.63}$$

For the specific case of a Gaussian distribution, we can make use of $\mathbb{E}[\mathbf{x}] = \boldsymbol{\mu}$, together with the result (2.62), to give

$$\mathrm{cov}[\mathbf{x}] = \boldsymbol{\Sigma}. \tag{2.64}$$

Because the parameter matrix $\boldsymbol{\Sigma}$ governs the covariance of $\mathbf{x}$ under the Gaussian distribution, it is called the covariance matrix.

Although the Gaussian distribution (2.43) is widely used as a density model, it suffers from some significant limitations. Consider the number of free parameters in the distribution. A general symmetric covariance matrix $\boldsymbol{\Sigma}$ will have $D(D+1)/2$ independent parameters, and there are another D independent parameters in $\boldsymbol{\mu}$, giving $D(D+3)/2$ parameters in total. For large D, the total number of parameters

Exercise 2.21

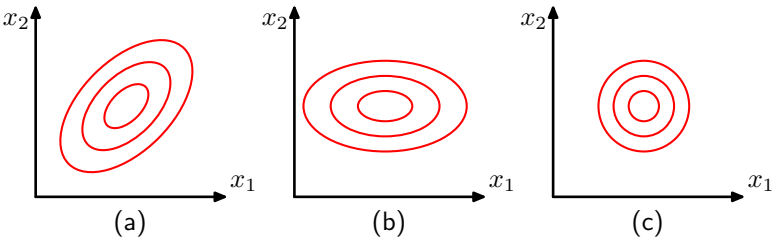


Figure 2.8 Contours of constant probability density for a Gaussian distribution in two dimensions in which the covariance matrix is (a) of general form, (b) diagonal, in which the elliptical contours are aligned with the coordinate axes, and (c) proportional to the identity matrix, in which the contours are concentric circles.

therefore grows quadratically with D, and the computational task of manipulating and inverting large matrices can become prohibitive. One way to address this problem is to use restricted forms of the covariance matrix. If we consider covariance matrices that are *diagonal*, so that $\boldsymbol{\Sigma} = \mathrm{diag}(\sigma_i^2)$, we then have a total of $2D$ independent parameters in the density model. The corresponding contours of constant density are given by axis-aligned ellipsoids. We could further restrict the covariance matrix to be proportional to the identity matrix, $\boldsymbol{\Sigma} = \sigma^2\mathbf{I}$, known as an *isotropic* covariance, giving $D + 1$ independent parameters in the model and spherical surfaces of constant density. The three possibilities of general, diagonal, and isotropic covariance matrices are illustrated in Figure 2.8. Unfortunately, whereas such approaches limit the number of degrees of freedom in the distribution and make inversion of the covariance matrix a much faster operation, they also greatly restrict the form of the probability density and limit its ability to capture interesting correlations in the data.

A further limitation of the Gaussian distribution is that it is intrinsically unimodal (i.e., has a single maximum) and so is unable to provide a good approximation to multimodal distributions. Thus the Gaussian distribution can be both too flexible, in the sense of having too many parameters, while also being too limited in the range of distributions that it can adequately represent. We will see later that the introduction of *latent* variables, also called *hidden* variables or *unobserved* variables, allows both of these problems to be addressed. In particular, a rich family of multimodal distributions is obtained by introducing discrete latent variables leading to mixtures of Gaussians, as discussed in Section 2.3.9. Similarly, the introduction of continuous latent variables, as described in Chapter 12, leads to models in which the number of free parameters can be controlled independently of the dimensionality D of the data space while still allowing the model to capture the dominant correlations in the data set. Indeed, these two approaches can be combined and further extended to derive a very rich set of hierarchical models that can be adapted to a broad range of practical applications. For instance, the Gaussian version of the *Markov random field*, which is widely used as a probabilistic model of images, is a Gaussian distribution over the joint space of pixel intensities but rendered tractable through the imposition of considerable structure reflecting the spatial organization of the pixels. Similarly, the *linear dynamical system*, used to model time series data for applications such as tracking, is also a joint Gaussian distribution over a potentially large number of observed and latent variables and again is tractable due to the structure imposed on the distribution. A powerful framework for expressing the form and properties of

Section 8.3

Section 13.3

such complex distributions is that of probabilistic graphical models, which will form the subject of Chapter 8.

2.3.1 Conditional Gaussian distributions

An important property of the multivariate Gaussian distribution is that if two sets of variables are jointly Gaussian, then the conditional distribution of one set conditioned on the other is again Gaussian. Similarly, the marginal distribution of either set is also Gaussian.

Consider first the case of conditional distributions. Suppose $\mathbf{x}$ is a D-dimensional vector with Gaussian distribution $\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ and that we partition $\mathbf{x}$ into two disjoint subsets $\mathbf{x}_a$ and $\mathbf{x}_b$. Without loss of generality, we can take $\mathbf{x}_a$ to form the first M components of $\mathbf{x}$, with $\mathbf{x}_b$ comprising the remaining $D - M$ components, so that

$$\mathbf{x} = \begin{pmatrix} \mathbf{x}_a \\ \mathbf{x}_b \end{pmatrix}. \tag{2.65}$$

We also define corresponding partitions of the mean vector $\boldsymbol{\mu}$ given by

$$\boldsymbol{\mu} = \begin{pmatrix} \boldsymbol{\mu}_a \\ \boldsymbol{\mu}_b \end{pmatrix} \tag{2.66}$$

and of the covariance matrix $\boldsymbol{\Sigma}$ given by

$$\boldsymbol{\Sigma} = \begin{pmatrix} \boldsymbol{\Sigma}_{aa} & \boldsymbol{\Sigma}_{ab} \\ \boldsymbol{\Sigma}_{ba} & \boldsymbol{\Sigma}_{bb} \end{pmatrix}. \tag{2.67}$$

Note that the symmetry $\boldsymbol{\Sigma}^{\mathrm{T}} = \boldsymbol{\Sigma}$ of the covariance matrix implies that $\boldsymbol{\Sigma}_{aa}$ and $\boldsymbol{\Sigma}_{bb}$ are symmetric, while $\boldsymbol{\Sigma}_{ba} = \boldsymbol{\Sigma}_{ab}^{\mathrm{T}}$.

In many situations, it will be convenient to work with the inverse of the covariance matrix

$$\boldsymbol{\Lambda} \equiv \boldsymbol{\Sigma}^{-1} \tag{2.68}$$

which is known as the *precision matrix*. In fact, we shall see that some properties of Gaussian distributions are most naturally expressed in terms of the covariance, whereas others take a simpler form when viewed in terms of the precision. We therefore also introduce the partitioned form of the precision matrix

$$\boldsymbol{\Lambda} = \begin{pmatrix} \boldsymbol{\Lambda}_{aa} & \boldsymbol{\Lambda}_{ab} \\ \boldsymbol{\Lambda}_{ba} & \boldsymbol{\Lambda}_{bb} \end{pmatrix} \tag{2.69}$$

corresponding to the partitioning (2.65) of the vector $\mathbf{x}$. Because the inverse of a symmetric matrix is also symmetric, we see that $\boldsymbol{\Lambda}_{aa}$ and $\boldsymbol{\Lambda}_{bb}$ are symmetric, while $\boldsymbol{\Lambda}_{ab}^{\mathrm{T}} = \boldsymbol{\Lambda}_{ba}$. It should be stressed at this point that, for instance, $\boldsymbol{\Lambda}_{aa}$ is not simply given by the inverse of $\boldsymbol{\Sigma}_{aa}$. In fact, we shall shortly examine the relation between the inverse of a partitioned matrix and the inverses of its partitions.

Exercise 2.22

Let us begin by finding an expression for the conditional distribution $p(\mathbf{x}_a|\mathbf{x}_b)$. From the product rule of probability, we see that this conditional distribution can be

evaluated from the joint distribution $p(\mathbf{x}) = p(\mathbf{x}_a, \mathbf{x}_b)$ simply by fixing $\mathbf{x}_b$ to the observed value and normalizing the resulting expression to obtain a valid probability distribution over $\mathbf{x}_a$. Instead of performing this normalization explicitly, we can obtain the solution more efficiently by considering the quadratic form in the exponent of the Gaussian distribution given by (2.44) and then reinstating the normalization coefficient at the end of the calculation. If we make use of the partitioning (2.65), (2.66), and (2.69), we obtain

$$\begin{aligned}-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu}) = \\ -\frac{1}{2}(\mathbf{x}_a-\boldsymbol{\mu}_a)^{\mathrm{T}}\boldsymbol{\Lambda}_{aa}(\mathbf{x}_a-\boldsymbol{\mu}_a) - \frac{1}{2}(\mathbf{x}_a-\boldsymbol{\mu}_a)^{\mathrm{T}}\boldsymbol{\Lambda}_{ab}(\mathbf{x}_b-\boldsymbol{\mu}_b) \\ -\frac{1}{2}(\mathbf{x}_b-\boldsymbol{\mu}_b)^{\mathrm{T}}\boldsymbol{\Lambda}_{ba}(\mathbf{x}_a-\boldsymbol{\mu}_a) - \frac{1}{2}(\mathbf{x}_b-\boldsymbol{\mu}_b)^{\mathrm{T}}\boldsymbol{\Lambda}_{bb}(\mathbf{x}_b-\boldsymbol{\mu}_b). \quad (2.70)\end{aligned}$$

We see that as a function of $\mathbf{x}_a$, this is again a quadratic form, and hence the corresponding conditional distribution $p(\mathbf{x}_a|\mathbf{x}_b)$ will be Gaussian. Because this distribution is completely characterized by its mean and its covariance, our goal will be to identify expressions for the mean and covariance of $p(\mathbf{x}_a|\mathbf{x}_b)$ by inspection of (2.70).

This is an example of a rather common operation associated with Gaussian distributions, sometimes called 'completing the square', in which we are given a quadratic form defining the exponent terms in a Gaussian distribution, and we need to determine the corresponding mean and covariance. Such problems can be solved straightforwardly by noting that the exponent in a general Gaussian distribution $\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ can be written

$$-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu}) = -\frac{1}{2}\mathbf{x}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\mathbf{x} + \mathbf{x}^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu} + \mathrm{const} \quad (2.71)$$

where 'const' denotes terms which are independent of $\mathbf{x}$, and we have made use of the symmetry of $\boldsymbol{\Sigma}$. Thus if we take our general quadratic form and express it in the form given by the right-hand side of (2.71), then we can immediately equate the matrix of coefficients entering the second order term in $\mathbf{x}$ to the inverse covariance matrix $\boldsymbol{\Sigma}^{-1}$ and the coefficient of the linear term in $\mathbf{x}$ to $\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}$, from which we can obtain $\boldsymbol{\mu}$.

Now let us apply this procedure to the conditional Gaussian distribution $p(\mathbf{x}_a|\mathbf{x}_b)$ for which the quadratic form in the exponent is given by (2.70). We will denote the mean and covariance of this distribution by $\boldsymbol{\mu}_{a|b}$ and $\boldsymbol{\Sigma}_{a|b}$, respectively. Consider the functional dependence of (2.70) on $\mathbf{x}_a$ in which $\mathbf{x}_b$ is regarded as a constant. If we pick out all terms that are second order in $\mathbf{x}_a$, we have

$$-\frac{1}{2}\mathbf{x}_a^{\mathrm{T}}\boldsymbol{\Lambda}_{aa}\mathbf{x}_a \quad (2.72)$$

from which we can immediately conclude that the covariance (inverse precision) of $p(\mathbf{x}_a|\mathbf{x}_b)$ is given by

$$\boldsymbol{\Sigma}_{a|b} = \boldsymbol{\Lambda}_{aa}^{-1}. \quad (2.73)$$

Now consider all of the terms in (2.70) that are linear in $\mathbf{x}_a$

$$\mathbf{x}_a^{\mathrm{T}} \{\boldsymbol{\Lambda}_{aa}\boldsymbol{\mu}_a - \boldsymbol{\Lambda}_{ab}(\mathbf{x}_b - \boldsymbol{\mu}_b)\} \tag{2.74}$$

where we have used $\boldsymbol{\Lambda}_{ba}^{\mathrm{T}} = \boldsymbol{\Lambda}_{ab}$. From our discussion of the general form (2.71), the coefficient of $\mathbf{x}_a$ in this expression must equal $\boldsymbol{\Sigma}_{a|b}^{-1}\boldsymbol{\mu}_{a|b}$ and hence

$$\begin{aligned}\boldsymbol{\mu}_{a|b} &= \boldsymbol{\Sigma}_{a|b}\{\boldsymbol{\Lambda}_{aa}\boldsymbol{\mu}_a - \boldsymbol{\Lambda}_{ab}(\mathbf{x}_b - \boldsymbol{\mu}_b)\} \\ &= \boldsymbol{\mu}_a - \boldsymbol{\Lambda}_{aa}^{-1}\boldsymbol{\Lambda}_{ab}(\mathbf{x}_b - \boldsymbol{\mu}_b)\end{aligned} \tag{2.75}$$

where we have made use of (2.73).

The results (2.73) and (2.75) are expressed in terms of the partitioned precision matrix of the original joint distribution $p(\mathbf{x}_a, \mathbf{x}_b)$. We can also express these results in terms of the corresponding partitioned covariance matrix. To do this, we make use of the following identity for the inverse of a partitioned matrix

Exercise 2.24

$$\begin{pmatrix}\mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D}\end{pmatrix}^{-1} = \begin{pmatrix}\mathbf{M} & -\mathbf{M}\mathbf{B}\mathbf{D}^{-1} \\ -\mathbf{D}^{-1}\mathbf{C}\mathbf{M} & \mathbf{D}^{-1} + \mathbf{D}^{-1}\mathbf{C}\mathbf{M}\mathbf{B}\mathbf{D}^{-1}\end{pmatrix} \tag{2.76}$$

where we have defined

$$\mathbf{M} = (\mathbf{A} - \mathbf{B}\mathbf{D}^{-1}\mathbf{C})^{-1}. \tag{2.77}$$

The quantity $\mathbf{M}^{-1}$ is known as the *Schur complement* of the matrix on the left-hand side of (2.76) with respect to the submatrix $\mathbf{D}$. Using the definition

$$\begin{pmatrix}\boldsymbol{\Sigma}_{aa} & \boldsymbol{\Sigma}_{ab} \\ \boldsymbol{\Sigma}_{ba} & \boldsymbol{\Sigma}_{bb}\end{pmatrix}^{-1} = \begin{pmatrix}\boldsymbol{\Lambda}_{aa} & \boldsymbol{\Lambda}_{ab} \\ \boldsymbol{\Lambda}_{ba} & \boldsymbol{\Lambda}_{bb}\end{pmatrix} \tag{2.78}$$

and making use of (2.76), we have

$$\boldsymbol{\Lambda}_{aa} = (\boldsymbol{\Sigma}_{aa} - \boldsymbol{\Sigma}_{ab}\boldsymbol{\Sigma}_{bb}^{-1}\boldsymbol{\Sigma}_{ba})^{-1} \tag{2.79}$$

$$\boldsymbol{\Lambda}_{ab} = -(\boldsymbol{\Sigma}_{aa} - \boldsymbol{\Sigma}_{ab}\boldsymbol{\Sigma}_{bb}^{-1}\boldsymbol{\Sigma}_{ba})^{-1}\boldsymbol{\Sigma}_{ab}\boldsymbol{\Sigma}_{bb}^{-1}. \tag{2.80}$$

From these we obtain the following expressions for the mean and covariance of the conditional distribution $p(\mathbf{x}_a|\mathbf{x}_b)$

$$\boldsymbol{\mu}_{a|b} = \boldsymbol{\mu}_a + \boldsymbol{\Sigma}_{ab}\boldsymbol{\Sigma}_{bb}^{-1}(\mathbf{x}_b - \boldsymbol{\mu}_b) \tag{2.81}$$

$$\boldsymbol{\Sigma}_{a|b} = \boldsymbol{\Sigma}_{aa} - \boldsymbol{\Sigma}_{ab}\boldsymbol{\Sigma}_{bb}^{-1}\boldsymbol{\Sigma}_{ba}. \tag{2.82}$$

Comparing (2.73) and (2.82), we see that the conditional distribution $p(\mathbf{x}_a|\mathbf{x}_b)$ takes a simpler form when expressed in terms of the partitioned precision matrix than when it is expressed in terms of the partitioned covariance matrix. Note that the mean of the conditional distribution $p(\mathbf{x}_a|\mathbf{x}_b)$, given by (2.81), is a linear function of $\mathbf{x}_b$ and that the covariance, given by (2.82), is independent of $\mathbf{x}_a$. This represents an example of a *linear-Gaussian* model.

Section 8.1.4

2.3.2 Marginal Gaussian distributions

We have seen that if a joint distribution $p(\mathbf{x}_a, \mathbf{x}_b)$ is Gaussian, then the conditional distribution $p(\mathbf{x}_a|\mathbf{x}_b)$ will again be Gaussian. Now we turn to a discussion of the marginal distribution given by

$$p(\mathbf{x}_a) = \int p(\mathbf{x}_a, \mathbf{x}_b)\, \mathrm{d}\mathbf{x}_b \tag{2.83}$$

which, as we shall see, is also Gaussian. Once again, our strategy for evaluating this distribution efficiently will be to focus on the quadratic form in the exponent of the joint distribution and thereby to identify the mean and covariance of the marginal distribution $p(\mathbf{x}_a)$.

The quadratic form for the joint distribution can be expressed, using the partitioned precision matrix, in the form (2.70). Because our goal is to integrate out $\mathbf{x}_b$, this is most easily achieved by first considering the terms involving $\mathbf{x}_b$ and then completing the square in order to facilitate integration. Picking out just those terms that involve $\mathbf{x}_b$, we have

$$-\frac{1}{2}\mathbf{x}_b^{\mathrm{T}}\boldsymbol{\Lambda}_{bb}\mathbf{x}_b + \mathbf{x}_b^{T}\mathbf{m} = -\frac{1}{2}(\mathbf{x}_b - \boldsymbol{\Lambda}_{bb}^{-1}\mathbf{m})^{\mathrm{T}}\boldsymbol{\Lambda}_{bb}(\mathbf{x}_b - \boldsymbol{\Lambda}_{bb}^{-1}\mathbf{m}) + \frac{1}{2}\mathbf{m}^{\mathrm{T}}\boldsymbol{\Lambda}_{bb}^{-1}\mathbf{m} \tag{2.84}$$

where we have defined

$$\mathbf{m} = \boldsymbol{\Lambda}_{bb}\boldsymbol{\mu}_b - \boldsymbol{\Lambda}_{ba}(\mathbf{x}_a - \boldsymbol{\mu}_a). \tag{2.85}$$

We see that the dependence on $\mathbf{x}_b$ has been cast into the standard quadratic form of a Gaussian distribution corresponding to the first term on the right-hand side of (2.84), plus a term that does not depend on $\mathbf{x}_b$ (but that does depend on $\mathbf{x}_a$). Thus, when we take the exponential of this quadratic form, we see that the integration over $\mathbf{x}_b$ required by (2.83) will take the form

$$\int \exp\left\{-\frac{1}{2}(\mathbf{x}_b - \boldsymbol{\Lambda}_{bb}^{-1}\mathbf{m})^{\mathrm{T}}\boldsymbol{\Lambda}_{bb}(\mathbf{x}_b - \boldsymbol{\Lambda}_{bb}^{-1}\mathbf{m})\right\} \mathrm{d}\mathbf{x}_b. \tag{2.86}$$

This integration is easily performed by noting that it is the integral over an unnormalized Gaussian, and so the result will be the reciprocal of the normalization coefficient. We know from the form of the normalized Gaussian given by (2.43), that this coefficient is independent of the mean and depends only on the determinant of the covariance matrix. Thus, by completing the square with respect to $\mathbf{x}_b$, we can integrate out $\mathbf{x}_b$ and the only term remaining from the contributions on the left-hand side of (2.84) that depends on $\mathbf{x}_a$ is the last term on the right-hand side of (2.84) in which $\mathbf{m}$ is given by (2.85). Combining this term with the remaining terms from

(2.70) that depend on $\mathbf{x}_a$, we obtain

$$\begin{aligned}
&\frac{1}{2}\left[\boldsymbol{\Lambda}_{bb}\boldsymbol{\mu}_b - \boldsymbol{\Lambda}_{ba}(\mathbf{x}_a - \boldsymbol{\mu}_a)\right]^{\mathrm{T}} \boldsymbol{\Lambda}_{bb}^{-1}\left[\boldsymbol{\Lambda}_{bb}\boldsymbol{\mu}_b - \boldsymbol{\Lambda}_{ba}(\mathbf{x}_a - \boldsymbol{\mu}_a)\right] \\
&\quad -\frac{1}{2}\mathbf{x}_a^{\mathrm{T}}\boldsymbol{\Lambda}_{aa}\mathbf{x}_a + \mathbf{x}_a^{\mathrm{T}}(\boldsymbol{\Lambda}_{aa}\boldsymbol{\mu}_a + \boldsymbol{\Lambda}_{ab}\boldsymbol{\mu}_b) + \text{const} \\
= \quad &-\frac{1}{2}\mathbf{x}_a^{\mathrm{T}}(\boldsymbol{\Lambda}_{aa} - \boldsymbol{\Lambda}_{ab}\boldsymbol{\Lambda}_{bb}^{-1}\boldsymbol{\Lambda}_{ba})\mathbf{x}_a \\
&+\mathbf{x}_a^{\mathrm{T}}(\boldsymbol{\Lambda}_{aa} - \boldsymbol{\Lambda}_{ab}\boldsymbol{\Lambda}_{bb}^{-1}\boldsymbol{\Lambda}_{ba})^{-1}\boldsymbol{\mu}_a + \text{const} \qquad (2.87)
\end{aligned}$$

where 'const' denotes quantities independent of $\mathbf{x}_a$. Again, by comparison with (2.71), we see that the covariance of the marginal distribution of $p(\mathbf{x}_a)$ is given by

$$\boldsymbol{\Sigma}_a = (\boldsymbol{\Lambda}_{aa} - \boldsymbol{\Lambda}_{ab}\boldsymbol{\Lambda}_{bb}^{-1}\boldsymbol{\Lambda}_{ba})^{-1}. \qquad (2.88)$$

Similarly, the mean is given by

$$\boldsymbol{\Sigma}_a(\boldsymbol{\Lambda}_{aa} - \boldsymbol{\Lambda}_{ab}\boldsymbol{\Lambda}_{bb}^{-1}\boldsymbol{\Lambda}_{ba})\boldsymbol{\mu}_a = \boldsymbol{\mu}_a \qquad (2.89)$$

where we have used (2.88). The covariance in (2.88) is expressed in terms of the partitioned precision matrix given by (2.69). We can rewrite this in terms of the corresponding partitioning of the covariance matrix given by (2.67), as we did for the conditional distribution. These partitioned matrices are related by

$$\begin{pmatrix} \boldsymbol{\Lambda}_{aa} & \boldsymbol{\Lambda}_{ab} \\ \boldsymbol{\Lambda}_{ba} & \boldsymbol{\Lambda}_{bb} \end{pmatrix}^{-1} = \begin{pmatrix} \boldsymbol{\Sigma}_{aa} & \boldsymbol{\Sigma}_{ab} \\ \boldsymbol{\Sigma}_{ba} & \boldsymbol{\Sigma}_{bb} \end{pmatrix} \qquad (2.90)$$

Making use of (2.76), we then have

$$\left(\boldsymbol{\Lambda}_{aa} - \boldsymbol{\Lambda}_{ab}\boldsymbol{\Lambda}_{bb}^{-1}\boldsymbol{\Lambda}_{ba}\right)^{-1} = \boldsymbol{\Sigma}_{aa}. \qquad (2.91)$$

Thus we obtain the intuitively satisfying result that the marginal distribution $p(\mathbf{x}_a)$ has mean and covariance given by

$$\begin{aligned}
\mathbb{E}[\mathbf{x}_a] &= \boldsymbol{\mu}_a \qquad (2.92) \\
\mathrm{cov}[\mathbf{x}_a] &= \boldsymbol{\Sigma}_{aa}. \qquad (2.93)
\end{aligned}$$

We see that for a marginal distribution, the mean and covariance are most simply expressed in terms of the partitioned covariance matrix, in contrast to the conditional distribution for which the partitioned precision matrix gives rise to simpler expressions.

Our results for the marginal and conditional distributions of a partitioned Gaussian are summarized below.

Partitioned Gaussians

Given a joint Gaussian distribution $\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ with $\boldsymbol{\Lambda} \equiv \boldsymbol{\Sigma}^{-1}$ and

$$\mathbf{x} = \begin{pmatrix} \mathbf{x}_a \\ \mathbf{x}_b \end{pmatrix}, \qquad \boldsymbol{\mu} = \begin{pmatrix} \boldsymbol{\mu}_a \\ \boldsymbol{\mu}_b \end{pmatrix} \qquad (2.94)$$

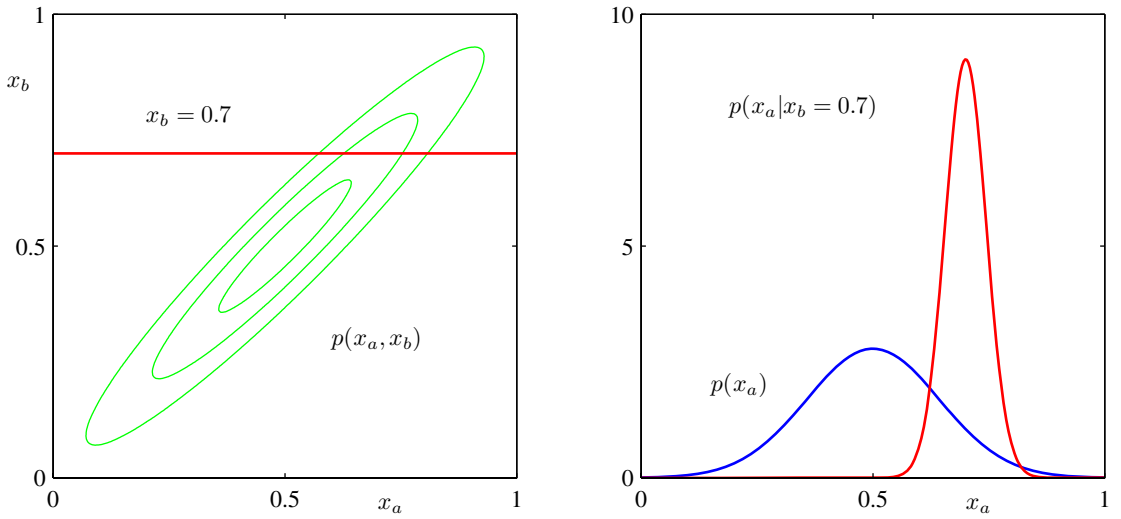


Figure 2.9 The plot on the left shows the contours of a Gaussian distribution $p(x_a, x_b)$ over two variables, and the plot on the right shows the marginal distribution $p(x_a)$ (blue curve) and the conditional distribution $p(x_a|x_b)$ for $x_b = 0.7$ (red curve).

$$\boldsymbol{\Sigma} = \begin{pmatrix} \boldsymbol{\Sigma}_{aa} & \boldsymbol{\Sigma}_{ab} \\ \boldsymbol{\Sigma}_{ba} & \boldsymbol{\Sigma}_{bb} \end{pmatrix}, \qquad \boldsymbol{\Lambda} = \begin{pmatrix} \boldsymbol{\Lambda}_{aa} & \boldsymbol{\Lambda}_{ab} \\ \boldsymbol{\Lambda}_{ba} & \boldsymbol{\Lambda}_{bb} \end{pmatrix}. \tag{2.95}$$

Conditional distribution:

$$p(\mathbf{x}_a|\mathbf{x}_b) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_{a|b}, \boldsymbol{\Lambda}_{aa}^{-1}) \tag{2.96}$$

$$\boldsymbol{\mu}_{a|b} = \boldsymbol{\mu}_a - \boldsymbol{\Lambda}_{aa}^{-1}\boldsymbol{\Lambda}_{ab}(\mathbf{x}_b - \boldsymbol{\mu}_b). \tag{2.97}$$

Marginal distribution:

$$p(\mathbf{x}_a) = \mathcal{N}(\mathbf{x}_a|\boldsymbol{\mu}_a, \boldsymbol{\Sigma}_{aa}). \tag{2.98}$$

We illustrate the idea of conditional and marginal distributions associated with a multivariate Gaussian using an example involving two variables in Figure 2.9.

2.3.3 Bayes' theorem for Gaussian variables

In Sections 2.3.1 and 2.3.2, we considered a Gaussian $p(\mathbf{x})$ in which we partitioned the vector $\mathbf{x}$ into two subvectors $\mathbf{x} = (\mathbf{x}_a, \mathbf{x}_b)$ and then found expressions for the conditional distribution $p(\mathbf{x}_a|\mathbf{x}_b)$ and the marginal distribution $p(\mathbf{x}_a)$. We noted that the mean of the conditional distribution $p(\mathbf{x}_a|\mathbf{x}_b)$ was a linear function of $\mathbf{x}_b$. Here we shall suppose that we are given a Gaussian marginal distribution $p(\mathbf{x})$ and a Gaussian conditional distribution $p(\mathbf{y}|\mathbf{x})$ in which $p(\mathbf{y}|\mathbf{x})$ has a mean that is a linear function of $\mathbf{x}$, and a covariance which is independent of $\mathbf{x}$. This is an example of

a *linear Gaussian model* (Roweis and Ghahramani, 1999), which we shall study in greater generality in Section 8.1.4. We wish to find the marginal distribution $p(\mathbf{y})$ and the conditional distribution $p(\mathbf{x}|\mathbf{y})$. This is a problem that will arise frequently in subsequent chapters, and it will prove convenient to derive the general results here.

We shall take the marginal and conditional distributions to be

$$p(\mathbf{x}) = \mathcal{N}\left(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Lambda}^{-1}\right) \tag{2.99}$$

$$p(\mathbf{y}|\mathbf{x}) = \mathcal{N}\left(\mathbf{y}|\mathbf{A}\mathbf{x}+\mathbf{b}, \mathbf{L}^{-1}\right) \tag{2.100}$$

where $\boldsymbol{\mu}$, $\mathbf{A}$, and $\mathbf{b}$ are parameters governing the means, and $\boldsymbol{\Lambda}$ and $\mathbf{L}$ are precision matrices. If $\mathbf{x}$ has dimensionality M and $\mathbf{y}$ has dimensionality D, then the matrix $\mathbf{A}$ has size $D \times M$.

First we find an expression for the joint distribution over $\mathbf{x}$ and $\mathbf{y}$. To do this, we define

$$\mathbf{z} = \begin{pmatrix} \mathbf{x} \\ \mathbf{y} \end{pmatrix} \tag{2.101}$$

and then consider the log of the joint distribution

$$\begin{aligned} \ln p(\mathbf{z}) &= \ln p(\mathbf{x}) + \ln p(\mathbf{y}|\mathbf{x}) \\ &= -\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Lambda}(\mathbf{x}-\boldsymbol{\mu}) \\ &\quad -\frac{1}{2}(\mathbf{y}-\mathbf{A}\mathbf{x}-\mathbf{b})^{\mathrm{T}}\mathbf{L}(\mathbf{y}-\mathbf{A}\mathbf{x}-\mathbf{b}) + \text{const} \end{aligned} \tag{2.102}$$

where 'const' denotes terms independent of $\mathbf{x}$ and $\mathbf{y}$. As before, we see that this is a quadratic function of the components of $\mathbf{z}$, and hence $p(\mathbf{z})$ is Gaussian distribution. To find the precision of this Gaussian, we consider the second order terms in (2.102), which can be written as

$$\begin{aligned} &-\frac{1}{2}\mathbf{x}^{\mathrm{T}}(\boldsymbol{\Lambda}+\mathbf{A}^{\mathrm{T}}\mathbf{L}\mathbf{A})\mathbf{x} - \frac{1}{2}\mathbf{y}^{\mathrm{T}}\mathbf{L}\mathbf{y} + \frac{1}{2}\mathbf{y}^{\mathrm{T}}\mathbf{L}\mathbf{A}\mathbf{x} + \frac{1}{2}\mathbf{x}^{\mathrm{T}}\mathbf{A}^{\mathrm{T}}\mathbf{L}\mathbf{y} \\ &= -\frac{1}{2}\begin{pmatrix} \mathbf{x} \\ \mathbf{y} \end{pmatrix}^{\mathrm{T}} \begin{pmatrix} \boldsymbol{\Lambda}+\mathbf{A}^{\mathrm{T}}\mathbf{L}\mathbf{A} & -\mathbf{A}^{\mathrm{T}}\mathbf{L} \\ -\mathbf{L}\mathbf{A} & \mathbf{L} \end{pmatrix} \begin{pmatrix} \mathbf{x} \\ \mathbf{y} \end{pmatrix} = -\frac{1}{2}\mathbf{z}^{\mathrm{T}}\mathbf{R}\mathbf{z} \end{aligned} \tag{2.103}$$

and so the Gaussian distribution over $\mathbf{z}$ has precision (inverse covariance) matrix given by

$$\mathbf{R} = \begin{pmatrix} \boldsymbol{\Lambda}+\mathbf{A}^{\mathrm{T}}\mathbf{L}\mathbf{A} & -\mathbf{A}^{\mathrm{T}}\mathbf{L} \\ -\mathbf{L}\mathbf{A} & \mathbf{L} \end{pmatrix}. \tag{2.104}$$

The covariance matrix is found by taking the inverse of the precision, which can be done using the matrix inversion formula (2.76) to give

Exercise 2.29

$$\mathrm{cov}[\mathbf{z}] = \mathbf{R}^{-1} = \begin{pmatrix} \boldsymbol{\Lambda}^{-1} & \boldsymbol{\Lambda}^{-1}\mathbf{A}^{\mathrm{T}} \\ \mathbf{A}\boldsymbol{\Lambda}^{-1} & \mathbf{L}^{-1}+\mathbf{A}\boldsymbol{\Lambda}^{-1}\mathbf{A}^{\mathrm{T}} \end{pmatrix}. \tag{2.105}$$

Similarly, we can find the mean of the Gaussian distribution over $\mathbf{z}$ by identifying the linear terms in (2.102), which are given by

$$\mathbf{x}^{\mathrm{T}}\boldsymbol{\Lambda}\boldsymbol{\mu} - \mathbf{x}^{\mathrm{T}}\mathbf{A}^{\mathrm{T}}\mathbf{L}\mathbf{b} + \mathbf{y}^{\mathrm{T}}\mathbf{L}\mathbf{b} = \begin{pmatrix}\mathbf{x}\\ \mathbf{y}\end{pmatrix}^{\mathrm{T}} \begin{pmatrix}\boldsymbol{\Lambda}\boldsymbol{\mu} - \mathbf{A}^{\mathrm{T}}\mathbf{L}\mathbf{b}\\ \mathbf{L}\mathbf{b}\end{pmatrix}. \tag{2.106}$$

Using our earlier result (2.71) obtained by completing the square over the quadratic form of a multivariate Gaussian, we find that the mean of $\mathbf{z}$ is given by

$$\mathbb{E}[\mathbf{z}] = \mathbf{R}^{-1}\begin{pmatrix}\boldsymbol{\Lambda}\boldsymbol{\mu} - \mathbf{A}^{\mathrm{T}}\mathbf{L}\mathbf{b}\\ \mathbf{L}\mathbf{b}\end{pmatrix}. \tag{2.107}$$

Exercise 2.30 Making use of (2.105), we then obtain

$$\mathbb{E}[\mathbf{z}] = \begin{pmatrix}\boldsymbol{\mu}\\ \mathbf{A}\boldsymbol{\mu} + \mathbf{b}\end{pmatrix}. \tag{2.108}$$

Next we find an expression for the marginal distribution $p(\mathbf{y})$ in which we have marginalized over $\mathbf{x}$. Recall that the marginal distribution over a subset of the components of a Gaussian random vector takes a particularly simple form when expressed in terms of the partitioned covariance matrix. Specifically, its mean and covariance are given by (2.92) and (2.93), respectively. Making use of (2.105) and (2.108) we see that the mean and covariance of the marginal distribution $p(\mathbf{y})$ are given by

Section 2.3

$$\mathbb{E}[\mathbf{y}] = \mathbf{A}\boldsymbol{\mu} + \mathbf{b} \tag{2.109}$$

$$\mathrm{cov}[\mathbf{y}] = \mathbf{L}^{-1} + \mathbf{A}\boldsymbol{\Lambda}^{-1}\mathbf{A}^{\mathrm{T}}. \tag{2.110}$$

A special case of this result is when $\mathbf{A} = \mathbf{I}$, in which case it reduces to the convolution of two Gaussians, for which we see that the mean of the convolution is the sum of the mean of the two Gaussians, and the covariance of the convolution is the sum of their covariances.

Finally, we seek an expression for the conditional $p(\mathbf{x}|\mathbf{y})$. Recall that the results for the conditional distribution are most easily expressed in terms of the partitioned precision matrix, using (2.73) and (2.75). Applying these results to (2.105) and (2.108) we see that the conditional distribution $p(\mathbf{x}|\mathbf{y})$ has mean and covariance given by

Section 2.3

$$\mathbb{E}[\mathbf{x}|\mathbf{y}] = (\boldsymbol{\Lambda} + \mathbf{A}^{\mathrm{T}}\mathbf{L}\mathbf{A})^{-1}\left\{\mathbf{A}^{\mathrm{T}}\mathbf{L}(\mathbf{y} - \mathbf{b}) + \boldsymbol{\Lambda}\boldsymbol{\mu}\right\} \tag{2.111}$$

$$\mathrm{cov}[\mathbf{x}|\mathbf{y}] = (\boldsymbol{\Lambda} + \mathbf{A}^{\mathrm{T}}\mathbf{L}\mathbf{A})^{-1}. \tag{2.112}$$

The evaluation of this conditional can be seen as an example of Bayes' theorem. We can interpret the distribution $p(\mathbf{x})$ as a prior distribution over $\mathbf{x}$. If the variable $\mathbf{y}$ is observed, then the conditional distribution $p(\mathbf{x}|\mathbf{y})$ represents the corresponding posterior distribution over $\mathbf{x}$. Having found the marginal and conditional distributions, we effectively expressed the joint distribution $p(\mathbf{z}) = p(\mathbf{x})p(\mathbf{y}|\mathbf{x})$ in the form $p(\mathbf{x}|\mathbf{y})p(\mathbf{y})$. These results are summarized below.

Marginal and Conditional Gaussians

Given a marginal Gaussian distribution for $\mathbf{x}$ and a conditional Gaussian distribution for $\mathbf{y}$ given $\mathbf{x}$ in the form

$$p(\mathbf{x}) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Lambda}^{-1}) \tag{2.113}$$
$$p(\mathbf{y}|\mathbf{x}) = \mathcal{N}(\mathbf{y}|\mathbf{A}\mathbf{x}+\mathbf{b}, \mathbf{L}^{-1}) \tag{2.114}$$

the marginal distribution of $\mathbf{y}$ and the conditional distribution of $\mathbf{x}$ given $\mathbf{y}$ are given by

$$p(\mathbf{y}) = \mathcal{N}(\mathbf{y}|\mathbf{A}\boldsymbol{\mu}+\mathbf{b}, \mathbf{L}^{-1}+\mathbf{A}\boldsymbol{\Lambda}^{-1}\mathbf{A}^{\mathrm{T}}) \tag{2.115}$$
$$p(\mathbf{x}|\mathbf{y}) = \mathcal{N}(\mathbf{x}|\boldsymbol{\Sigma}\{\mathbf{A}^{\mathrm{T}}\mathbf{L}(\mathbf{y}-\mathbf{b})+\boldsymbol{\Lambda}\boldsymbol{\mu}\}, \boldsymbol{\Sigma}) \tag{2.116}$$

where

$$\boldsymbol{\Sigma} = (\boldsymbol{\Lambda}+\mathbf{A}^{\mathrm{T}}\mathbf{L}\mathbf{A})^{-1}. \tag{2.117}$$

2.3.4 Maximum likelihood for the Gaussian

Given a data set $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_N)^{\mathrm{T}}$ in which the observations $\{\mathbf{x}_n\}$ are assumed to be drawn independently from a multivariate Gaussian distribution, we can estimate the parameters of the distribution by maximum likelihood. The log likelihood function is given by

$$\ln p(\mathbf{X}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = -\frac{ND}{2}\ln(2\pi) - \frac{N}{2}\ln|\boldsymbol{\Sigma}| - \frac{1}{2}\sum_{n=1}^{N}(\mathbf{x}_n-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}_n-\boldsymbol{\mu}). \tag{2.118}$$

By simple rearrangement, we see that the likelihood function depends on the data set only through the two quantities

$$\sum_{n=1}^{N}\mathbf{x}_n, \qquad \sum_{n=1}^{N}\mathbf{x}_n\mathbf{x}_n^{\mathrm{T}}. \tag{2.119}$$

These are known as the *sufficient statistics* for the Gaussian distribution. Using (C.19), the derivative of the log likelihood with respect to $\boldsymbol{\mu}$ is given by

Appendix C

$$\frac{\partial}{\partial\boldsymbol{\mu}}\ln p(\mathbf{X}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{n=1}^{N}\boldsymbol{\Sigma}^{-1}(\mathbf{x}_n-\boldsymbol{\mu}) \tag{2.120}$$

and setting this derivative to zero, we obtain the solution for the maximum likelihood estimate of the mean given by

$$\boldsymbol{\mu}_{\mathrm{ML}} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n \tag{2.121}$$

which is the mean of the observed set of data points. The maximization of (2.118) with respect to $\boldsymbol{\Sigma}$ is rather more involved. The simplest approach is to ignore the symmetry constraint and show that the resulting solution is symmetric as required. Alternative derivations of this result, which impose the symmetry and positive definiteness constraints explicitly, can be found in Magnus and Neudecker (1999). The result is as expected and takes the form

Exercise 2.34

$$\boldsymbol{\Sigma}_{\mathrm{ML}} = \frac{1}{N}\sum_{n=1}^{N}(\mathbf{x}_n - \boldsymbol{\mu}_{\mathrm{ML}})(\mathbf{x}_n - \boldsymbol{\mu}_{\mathrm{ML}})^{\mathrm{T}} \tag{2.122}$$

which involves $\boldsymbol{\mu}_{\mathrm{ML}}$ because this is the result of a joint maximization with respect to $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$. Note that the solution (2.121) for $\boldsymbol{\mu}_{\mathrm{ML}}$ does not depend on $\boldsymbol{\Sigma}_{\mathrm{ML}}$, and so we can first evaluate $\boldsymbol{\mu}_{\mathrm{ML}}$ and then use this to evaluate $\boldsymbol{\Sigma}_{\mathrm{ML}}$.

If we evaluate the expectations of the maximum likelihood solutions under the true distribution, we obtain the following results

Exercise 2.35

$$\mathbb{E}[\boldsymbol{\mu}_{\mathrm{ML}}] = \boldsymbol{\mu} \tag{2.123}$$

$$\mathbb{E}[\boldsymbol{\Sigma}_{\mathrm{ML}}] = \frac{N-1}{N}\boldsymbol{\Sigma}. \tag{2.124}$$

We see that the expectation of the maximum likelihood estimate for the mean is equal to the true mean. However, the maximum likelihood estimate for the covariance has an expectation that is less than the true value, and hence it is biased. We can correct this bias by defining a different estimator $\widetilde{\boldsymbol{\Sigma}}$ given by

$$\widetilde{\boldsymbol{\Sigma}} = \frac{1}{N-1}\sum_{n=1}^{N}(\mathbf{x}_n - \boldsymbol{\mu}_{\mathrm{ML}})(\mathbf{x}_n - \boldsymbol{\mu}_{\mathrm{ML}})^{\mathrm{T}}. \tag{2.125}$$

Clearly from (2.122) and (2.124), the expectation of $\widetilde{\boldsymbol{\Sigma}}$ is equal to $\boldsymbol{\Sigma}$.

2.3.5 Sequential estimation

Our discussion of the maximum likelihood solution for the parameters of a Gaussian distribution provides a convenient opportunity to give a more general discussion of the topic of sequential estimation for maximum likelihood. Sequential methods allow data points to be processed one at a time and then discarded and are important for on-line applications, and also where large data sets are involved so that batch processing of all data points at once is infeasible.

Consider the result (2.121) for the maximum likelihood estimator of the mean $\boldsymbol{\mu}_{\mathrm{ML}}$, which we will denote by $\boldsymbol{\mu}_{\mathrm{ML}}^{(N)}$ when it is based on N observations. If we

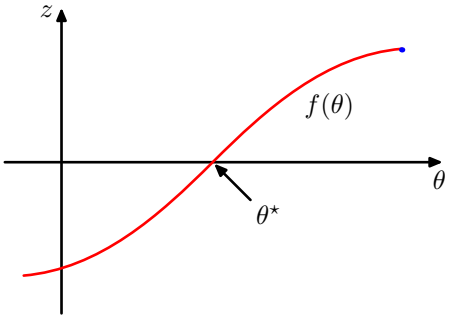


Figure 2.10 A schematic illustration of two correlated random variables z and θ, together with the regression function $f(\theta)$ given by the conditional expectation $\mathbb{E}[z|\theta]$. The Robbins-Monro algorithm provides a general sequential procedure for finding the root $\theta^\star$ of such functions.

dissect out the contribution from the final data point $\mathbf{x}_N$, we obtain

$$\begin{aligned}\boldsymbol{\mu}_{\mathrm{ML}}^{(N)} &= \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n \\ &= \frac{1}{N}\mathbf{x}_N + \frac{1}{N}\sum_{n=1}^{N-1}\mathbf{x}_n \\ &= \frac{1}{N}\mathbf{x}_N + \frac{N-1}{N}\boldsymbol{\mu}_{\mathrm{ML}}^{(N-1)} \\ &= \boldsymbol{\mu}_{\mathrm{ML}}^{(N-1)} + \frac{1}{N}(\mathbf{x}_N - \boldsymbol{\mu}_{\mathrm{ML}}^{(N-1)}). \end{aligned} \tag{2.126}$$

This result has a nice interpretation, as follows. After observing $N-1$ data points we have estimated $\boldsymbol{\mu}$ by $\boldsymbol{\mu}_{\mathrm{ML}}^{(N-1)}$. We now observe data point $\mathbf{x}_N$, and we obtain our revised estimate $\boldsymbol{\mu}_{\mathrm{ML}}^{(N)}$ by moving the old estimate a small amount, proportional to $1/N$, in the direction of the 'error signal' $(\mathbf{x}_N - \boldsymbol{\mu}_{\mathrm{ML}}^{(N-1)})$. Note that, as N increases, so the contribution from successive data points gets smaller.

The result (2.126) will clearly give the same answer as the batch result (2.121) because the two formulae are equivalent. However, we will not always be able to derive a sequential algorithm by this route, and so we seek a more general formulation of sequential learning, which leads us to the *Robbins-Monro* algorithm. Consider a pair of random variables θ and z governed by a joint distribution $p(z, \theta)$. The conditional expectation of z given θ defines a deterministic function $f(\theta)$ that is given by

$$f(\theta) \equiv \mathbb{E}[z|\theta] = \int z p(z|\theta)\,\mathrm{d}z \tag{2.127}$$

and is illustrated schematically in Figure 2.10. Functions defined in this way are called *regression functions*.

Our goal is to find the root $\theta^\star$ at which $f(\theta^\star) = 0$. If we had a large data set of observations of z and θ, then we could model the regression function directly and then obtain an estimate of its root. Suppose, however, that we observe values of z one at a time and we wish to find a corresponding sequential estimation scheme for $\theta^\star$. The following general procedure for solving such problems was given by

Robbins and Monro (1951). We shall assume that the conditional variance of z is finite so that

$$\mathbb{E}\left[(z-f)^2 \mid \theta\right] < \infty \tag{2.128}$$

and we shall also, without loss of generality, consider the case where $f(\theta) > 0$ for $\theta > \theta^\star$ and $f(\theta) < 0$ for $\theta < \theta^\star$, as is the case in Figure 2.10. The Robbins-Monro procedure then defines a sequence of successive estimates of the root $\theta^\star$ given by

$$\theta^{(N)} = \theta^{(N-1)} + a_{N-1} z(\theta^{(N-1)}) \tag{2.129}$$

where $z(\theta^{(N)})$ is an observed value of z when θ takes the value $\theta^{(N)}$. The coefficients $\{a_N\}$ represent a sequence of positive numbers that satisfy the conditions

$$\lim_{N\to\infty} a_N = 0 \tag{2.130}$$

$$\sum_{N=1}^{\infty} a_N = \infty \tag{2.131}$$

$$\sum_{N=1}^{\infty} a_N^2 < \infty. \tag{2.132}$$

It can then be shown (Robbins and Monro, 1951; Fukunaga, 1990) that the sequence of estimates given by (2.129) does indeed converge to the root with probability one. Note that the first condition (2.130) ensures that the successive corrections decrease in magnitude so that the process can converge to a limiting value. The second condition (2.131) is required to ensure that the algorithm does not converge short of the root, and the third condition (2.132) is needed to ensure that the accumulated noise has finite variance and hence does not spoil convergence.

Now let us consider how a general maximum likelihood problem can be solved sequentially using the Robbins-Monro algorithm. By definition, the maximum likelihood solution θ_{ML} is a stationary point of the log likelihood function and hence satisfies

$$\left.\frac{\partial}{\partial\theta}\left\{\frac{1}{N}\sum_{n=1}^{N}\ln p(\mathbf{x}_n|\theta)\right\}\right|_{\theta_{\mathrm{ML}}} = 0. \tag{2.133}$$

Exchanging the derivative and the summation, and taking the limit $N \to \infty$ we have

$$\lim_{N\to\infty}\frac{1}{N}\sum_{n=1}^{N}\frac{\partial}{\partial\theta}\ln p(x_n|\theta) = \mathbb{E}_x\left[\frac{\partial}{\partial\theta}\ln p(x|\theta)\right] \tag{2.134}$$

and so we see that finding the maximum likelihood solution corresponds to finding the root of a regression function. We can therefore apply the Robbins-Monro procedure, which now takes the form

$$\theta^{(N)} = \theta^{(N-1)} + a_{N-1}\frac{\partial}{\partial\theta^{(N-1)}}\ln p(x_N|\theta^{(N-1)}). \tag{2.135}$$

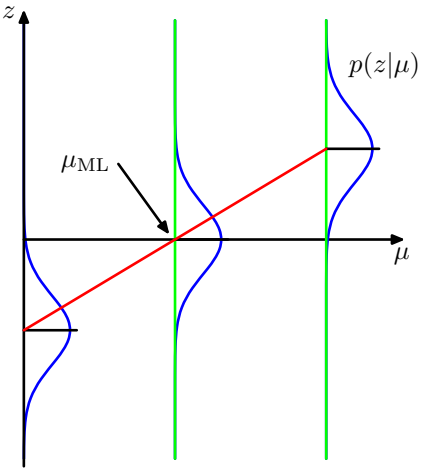


Figure 2.11 In the case of a Gaussian distribution, with θ corresponding to the mean μ, the regression function illustrated in Figure 2.10 takes the form of a straight line, as shown in red. In this case, the random variable z corresponds to the derivative of the log likelihood function and is given by $(x - \mu_{\mathrm{ML}})/\sigma^2$, and its expectation that defines the regression function is a straight line given by $(\mu - \mu_{\mathrm{ML}})/\sigma^2$. The root of the regression function corresponds to the maximum likelihood estimator μ_{ML}.

As a specific example, we consider once again the sequential estimation of the mean of a Gaussian distribution, in which case the parameter $\theta^{(N)}$ is the estimate $\mu_{\mathrm{ML}}^{(N)}$ of the mean of the Gaussian, and the random variable z is given by

$$z = \frac{\partial}{\partial \mu_{\mathrm{ML}}} \ln p(x|\mu_{\mathrm{ML}}, \sigma^2) = \frac{1}{\sigma^2}(x - \mu_{\mathrm{ML}}). \tag{2.136}$$

Thus the distribution of z is Gaussian with mean $\mu - \mu_{\mathrm{ML}}$, as illustrated in Figure 2.11. Substituting (2.136) into (2.135), we obtain the univariate form of (2.126), provided we choose the coefficients a_N to have the form $a_N = \sigma^2/N$. Note that although we have focussed on the case of a single variable, the same technique, together with the same restrictions (2.130)–(2.132) on the coefficients a_N, apply equally to the multivariate case (Blum, 1965).

2.3.6 Bayesian inference for the Gaussian

The maximum likelihood framework gave point estimates for the parameters $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$. Now we develop a Bayesian treatment by introducing prior distributions over these parameters. Let us begin with a simple example in which we consider a single Gaussian random variable x. We shall suppose that the variance σ^2 is known, and we consider the task of inferring the mean μ given a set of N observations $\mathbf{X} = \{x_1, \ldots, x_N\}$. The likelihood function, that is the probability of the observed data given μ, viewed as a function of μ, is given by

$$p(\mathbf{X}|\mu) = \prod_{n=1}^{N} p(x_n|\mu) = \frac{1}{(2\pi\sigma^2)^{N/2}} \exp\left\{-\frac{1}{2\sigma^2}\sum_{n=1}^{N}(x_n - \mu)^2\right\}. \tag{2.137}$$

Again we emphasize that the likelihood function $p(\mathbf{X}|\mu)$ is not a probability distribution over μ and is not normalized.

We see that the likelihood function takes the form of the exponential of a quadratic form in μ. Thus if we choose a prior $p(\mu)$ given by a Gaussian, it will be a

conjugate distribution for this likelihood function because the corresponding posterior will be a product of two exponentials of quadratic functions of μ and hence will also be Gaussian. We therefore take our prior distribution to be

$$p(\mu) = \mathcal{N}\left(\mu|\mu_0, \sigma_0^2\right) \tag{2.138}$$

and the posterior distribution is given by

$$p(\mu|\mathbf{X}) \propto p(\mathbf{X}|\mu)p(\mu). \tag{2.139}$$

Exercise 2.38

Simple manipulation involving completing the square in the exponent shows that the posterior distribution is given by

$$p(\mu|\mathbf{X}) = \mathcal{N}\left(\mu|\mu_N, \sigma_N^2\right) \tag{2.140}$$

where

$$\mu_N = \frac{\sigma^2}{N\sigma_0^2 + \sigma^2}\mu_0 + \frac{N\sigma_0^2}{N\sigma_0^2 + \sigma^2}\mu_{\mathrm{ML}} \tag{2.141}$$

$$\frac{1}{\sigma_N^2} = \frac{1}{\sigma_0^2} + \frac{N}{\sigma^2} \tag{2.142}$$

in which μ_{ML} is the maximum likelihood solution for μ given by the sample mean

$$\mu_{\mathrm{ML}} = \frac{1}{N}\sum_{n=1}^{N} x_n. \tag{2.143}$$

It is worth spending a moment studying the form of the posterior mean and variance. First of all, we note that the mean of the posterior distribution given by (2.141) is a compromise between the prior mean μ_0 and the maximum likelihood solution μ_{ML}. If the number of observed data points $N = 0$, then (2.141) reduces to the prior mean as expected. For $N \to \infty$, the posterior mean is given by the maximum likelihood solution. Similarly, consider the result (2.142) for the variance of the posterior distribution. We see that this is most naturally expressed in terms of the inverse variance, which is called the precision. Furthermore, the precisions are additive, so that the precision of the posterior is given by the precision of the prior plus one contribution of the data precision from each of the observed data points. As we increase the number of observed data points, the precision steadily increases, corresponding to a posterior distribution with steadily decreasing variance. With no observed data points, we have the prior variance, whereas if the number of data points $N \to \infty$, the variance σ_N^2 goes to zero and the posterior distribution becomes infinitely peaked around the maximum likelihood solution. We therefore see that the maximum likelihood result of a point estimate for μ given by (2.143) is recovered precisely from the Bayesian formalism in the limit of an infinite number of observations. Note also that for finite N, if we take the limit $\sigma_0^2 \to \infty$ in which the prior has infinite variance then the posterior mean (2.141) reduces to the maximum likelihood result, while from (2.142) the posterior variance is given by $\sigma_N^2 = \sigma^2/N$.

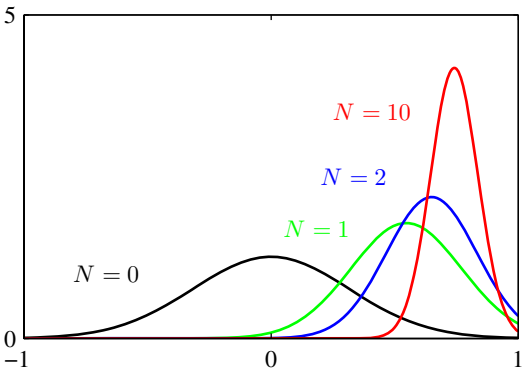


Figure 2.12 Illustration of Bayesian inference for the mean μ of a Gaussian distribution, in which the variance is assumed to be known. The curves show the prior distribution over μ (the curve labelled $N = 0$), which in this case is itself Gaussian, along with the posterior distribution given by (2.140) for increasing numbers N of data points. The data points are generated from a Gaussian of mean 0.8 and variance 0.1, and the prior is chosen to have mean 0. In both the prior and the likelihood function, the variance is set to the true value.

We illustrate our analysis of Bayesian inference for the mean of a Gaussian distribution in Figure 2.12. The generalization of this result to the case of a D-dimensional Gaussian random variable $\mathbf{x}$ with known covariance and unknown mean is straightforward.

Exercise 2.40

We have already seen how the maximum likelihood expression for the mean of a Gaussian can be re-cast as a sequential update formula in which the mean after observing N data points was expressed in terms of the mean after observing $N-1$ data points together with the contribution from data point $\mathbf{x}_N$. In fact, the Bayesian paradigm leads very naturally to a sequential view of the inference problem. To see this in the context of the inference of the mean of a Gaussian, we write the posterior distribution with the contribution from the final data point $\mathbf{x}_N$ separated out so that

Section 2.3.5

$$p(\boldsymbol{\mu}|D) \propto \left[p(\boldsymbol{\mu}) \prod_{n=1}^{N-1} p(\mathbf{x}_n|\boldsymbol{\mu}) \right] p(\mathbf{x}_N|\boldsymbol{\mu}). \tag{2.144}$$

The term in square brackets is (up to a normalization coefficient) just the posterior distribution after observing $N-1$ data points. We see that this can be viewed as a prior distribution, which is combined using Bayes' theorem with the likelihood function associated with data point $\mathbf{x}_N$ to arrive at the posterior distribution after observing N data points. This sequential view of Bayesian inference is very general and applies to any problem in which the observed data are assumed to be independent and identically distributed.

So far, we have assumed that the variance of the Gaussian distribution over the data is known and our goal is to infer the mean. Now let us suppose that the mean is known and we wish to infer the variance. Again, our calculations will be greatly simplified if we choose a conjugate form for the prior distribution. It turns out to be most convenient to work with the precision $\lambda \equiv 1/\sigma^2$. The likelihood function for λ takes the form

$$p(\mathbf{X}|\lambda) = \prod_{n=1}^{N} \mathcal{N}(x_n|\mu, \lambda^{-1}) \propto \lambda^{N/2} \exp\left\{ -\frac{\lambda}{2} \sum_{n=1}^{N} (x_n - \mu)^2 \right\}. \tag{2.145}$$

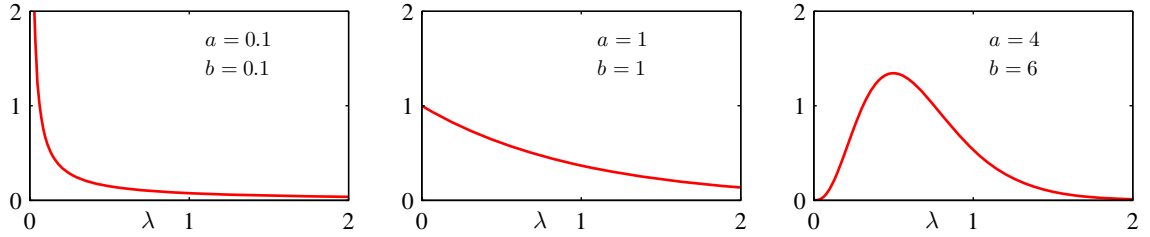


Figure 2.13 Plot of the gamma distribution $\mathrm{Gam}(\lambda|a,b)$ defined by (2.146) for various values of the parameters a and b.

The corresponding conjugate prior should therefore be proportional to the product of a power of λ and the exponential of a linear function of λ. This corresponds to the *gamma* distribution which is defined by

$$\mathrm{Gam}(\lambda|a,b) = \frac{1}{\Gamma(a)} b^a \lambda^{a-1} \exp(-b\lambda). \tag{2.146}$$

Exercise 2.41 *Exercise 2.42*

Here $\Gamma(a)$ is the gamma function that is defined by (1.141) and that ensures that (2.146) is correctly normalized. The gamma distribution has a finite integral if $a > 0$, and the distribution itself is finite if $a \geqslant 1$. It is plotted, for various values of a and b, in Figure 2.13. The mean and variance of the gamma distribution are given by

$$\mathbb{E}[\lambda] = \frac{a}{b} \tag{2.147}$$

$$\mathrm{var}[\lambda] = \frac{a}{b^2}. \tag{2.148}$$

Consider a prior distribution $\mathrm{Gam}(\lambda|a_0,b_0)$. If we multiply by the likelihood function (2.145), then we obtain a posterior distribution

$$p(\lambda|\mathbf{X}) \propto \lambda^{a_0-1}\lambda^{N/2} \exp\left\{-b_0\lambda - \frac{\lambda}{2}\sum_{n=1}^{N}(x_n-\mu)^2\right\} \tag{2.149}$$

which we recognize as a gamma distribution of the form $\mathrm{Gam}(\lambda|a_N,b_N)$ where

$$a_N = a_0 + \frac{N}{2} \tag{2.150}$$

$$b_N = b_0 + \frac{1}{2}\sum_{n=1}^{N}(x_n-\mu)^2 = b_0 + \frac{N}{2}\sigma^2_{\mathrm{ML}} \tag{2.151}$$

where σ^2_{ML} is the maximum likelihood estimator of the variance. Note that in (2.149) there is no need to keep track of the normalization constants in the prior and the likelihood function because, if required, the correct coefficient can be found at the end using the normalized form (2.146) for the gamma distribution.

From (2.150), we see that the effect of observing N data points is to increase the value of the coefficient a by $N/2$. Thus we can interpret the parameter a_0 in the prior in terms of $2a_0$ 'effective' prior observations. Similarly, from (2.151) we see that the N data points contribute $N\sigma^2_{\mathrm{ML}}/2$ to the parameter b, where σ^2_{ML} is the variance, and so we can interpret the parameter b_0 in the prior as arising from the $2a_0$ 'effective' prior observations having variance $2b_0/(2a_0) = b_0/a_0$. Recall that we made an analogous interpretation for the Dirichlet prior. These distributions are examples of the exponential family, and we shall see that the interpretation of a conjugate prior in terms of effective fictitious data points is a general one for the exponential family of distributions.

Section 2.2

Instead of working with the precision, we can consider the variance itself. The conjugate prior in this case is called the *inverse gamma* distribution, although we shall not discuss this further because we will find it more convenient to work with the precision.

Now suppose that both the mean and the precision are unknown. To find a conjugate prior, we consider the dependence of the likelihood function on μ and λ

$$\begin{aligned} p(\mathbf{X}|\mu,\lambda) &= \prod_{n=1}^{N}\left(\frac{\lambda}{2\pi}\right)^{1/2}\exp\left\{-\frac{\lambda}{2}(x_n-\mu)^2\right\} \\ &\propto \left[\lambda^{1/2}\exp\left(-\frac{\lambda\mu^2}{2}\right)\right]^N \exp\left\{\lambda\mu\sum_{n=1}^{N}x_n - \frac{\lambda}{2}\sum_{n=1}^{N}x_n^2\right\}. \end{aligned} \tag{2.152}$$

We now wish to identify a prior distribution $p(\mu,\lambda)$ that has the same functional dependence on μ and λ as the likelihood function and that should therefore take the form

$$\begin{aligned} p(\mu,\lambda) &\propto \left[\lambda^{1/2}\exp\left(-\frac{\lambda\mu^2}{2}\right)\right]^\beta \exp\left\{c\lambda\mu - d\lambda\right\} \\ &= \exp\left\{-\frac{\beta\lambda}{2}(\mu-c/\beta)^2\right\}\lambda^{\beta/2}\exp\left\{-\left(d-\frac{c^2}{2\beta}\right)\lambda\right\} \end{aligned} \tag{2.153}$$

where c, d, and β are constants. Since we can always write $p(\mu,\lambda) = p(\mu|\lambda)p(\lambda)$, we can find $p(\mu|\lambda)$ and $p(\lambda)$ by inspection. In particular, we see that $p(\mu|\lambda)$ is a Gaussian whose precision is a linear function of λ and that $p(\lambda)$ is a gamma distribution, so that the normalized prior takes the form

$$p(\mu,\lambda) = \mathcal{N}(\mu|\mu_0,(\beta\lambda)^{-1})\mathrm{Gam}(\lambda|a,b) \tag{2.154}$$

where we have defined new constants given by $\mu_0 = c/\beta$, $a = 1+\beta/2$, $b = d-c^2/2\beta$. The distribution (2.154) is called the *normal-gamma* or *Gaussian-gamma* distribution and is plotted in Figure 2.14. Note that this is not simply the product of an independent Gaussian prior over μ and a gamma prior over λ, because the precision of μ is a linear function of λ. Even if we chose a prior in which μ and λ were independent, the posterior distribution would exhibit a coupling between the precision of μ and the value of λ.

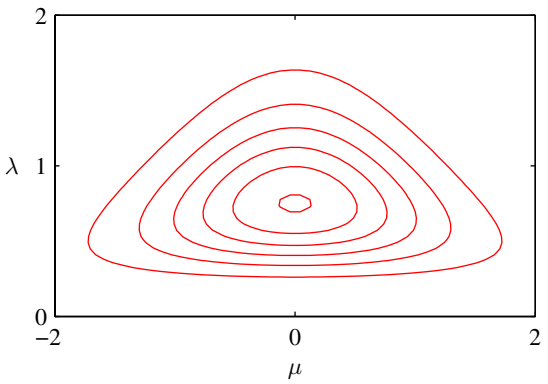


Figure 2.14 Contour plot of the normal-gamma distribution (2.154) for parameter values $\mu_0 = 0$, $\beta = 2$, $a = 5$ and $b = 6$.

In the case of the multivariate Gaussian distribution $\mathcal{N}\left(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Lambda}^{-1}\right)$ for a D-dimensional variable $\mathbf{x}$, the conjugate prior distribution for the mean $\boldsymbol{\mu}$, assuming the precision is known, is again a Gaussian. For known mean and unknown precision matrix $\boldsymbol{\Lambda}$, the conjugate prior is the *Wishart* distribution given by

Exercise 2.45

$$\mathcal{W}(\boldsymbol{\Lambda}|\mathbf{W}, \nu) = B|\boldsymbol{\Lambda}|^{(\nu-D-1)/2} \exp\left(-\frac{1}{2}\mathrm{Tr}(\mathbf{W}^{-1}\boldsymbol{\Lambda})\right) \tag{2.155}$$

where ν is called the number of *degrees of freedom* of the distribution, $\mathbf{W}$ is a $D \times D$ scale matrix, and $\mathrm{Tr}(\cdot)$ denotes the trace. The normalization constant B is given by

$$B(\mathbf{W}, \nu) = |\mathbf{W}|^{-\nu/2} \left(2^{\nu D/2} \pi^{D(D-1)/4} \prod_{i=1}^{D} \Gamma\left(\frac{\nu + 1 - i}{2}\right)\right)^{-1}. \tag{2.156}$$

Again, it is also possible to define a conjugate prior over the covariance matrix itself, rather than over the precision matrix, which leads to the *inverse Wishart* distribution, although we shall not discuss this further. If both the mean and the precision are unknown, then, following a similar line of reasoning to the univariate case, the conjugate prior is given by

$$p(\boldsymbol{\mu}, \boldsymbol{\Lambda}|\boldsymbol{\mu}_0, \beta, \mathbf{W}, \nu) = \mathcal{N}(\boldsymbol{\mu}|\boldsymbol{\mu}_0, (\beta\boldsymbol{\Lambda})^{-1})\, \mathcal{W}(\boldsymbol{\Lambda}|\mathbf{W}, \nu) \tag{2.157}$$

which is known as the *normal-Wishart* or *Gaussian-Wishart* distribution.

2.3.7 Student's t-distribution

We have seen that the conjugate prior for the precision of a Gaussian is given by a gamma distribution. If we have a univariate Gaussian $\mathcal{N}(x|\mu, \tau^{-1})$ together with a Gamma prior $\mathrm{Gam}(\tau|a, b)$ and we integrate out the precision, we obtain the marginal distribution of x in the form

Section 2.3.6

Exercise 2.46

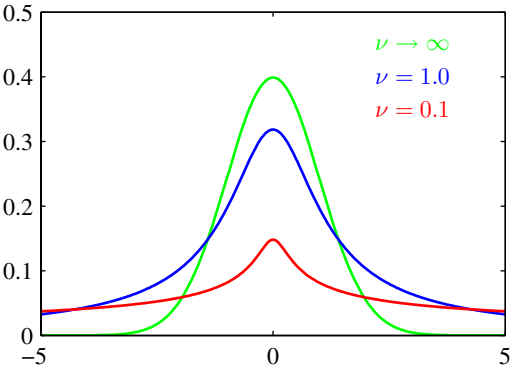


Figure 2.15 Plot of Student's t-distribution (2.159) for $\mu = 0$ and $\lambda = 1$ for various values of ν. The limit $\nu \to \infty$ corresponds to a Gaussian distribution with mean μ and precision λ.

$$
\begin{aligned}
p(x|\mu, a, b) &= \int_0^\infty \mathcal{N}(x|\mu, \tau^{-1})\mathrm{Gam}(\tau|a, b)\,\mathrm{d}\tau \qquad (2.158)\\
&= \int_0^\infty \frac{b^a e^{(-b\tau)}\tau^{a-1}}{\Gamma(a)} \left(\frac{\tau}{2\pi}\right)^{1/2} \exp\left\{-\frac{\tau}{2}(x-\mu)^2\right\}\,\mathrm{d}\tau\\
&= \frac{b^a}{\Gamma(a)} \left(\frac{1}{2\pi}\right)^{1/2} \left[b + \frac{(x-\mu)^2}{2}\right]^{-a-1/2} \Gamma(a + 1/2)
\end{aligned}
$$

where we have made the change of variable $z = \tau[b + (x-\mu)^2/2]$. By convention we define new parameters given by $\nu = 2a$ and $\lambda = a/b$, in terms of which the distribution $p(x|\mu, a, b)$ takes the form

$$
\mathrm{St}(x|\mu, \lambda, \nu) = \frac{\Gamma(\nu/2 + 1/2)}{\Gamma(\nu/2)} \left(\frac{\lambda}{\pi\nu}\right)^{1/2} \left[1 + \frac{\lambda(x-\mu)^2}{\nu}\right]^{-\nu/2-1/2} \qquad (2.159)
$$

which is known as *Student's t-distribution*. The parameter λ is sometimes called the *precision* of the t-distribution, even though it is not in general equal to the inverse of the variance. The parameter ν is called the *degrees of freedom*, and its effect is illustrated in Figure 2.15. For the particular case of $\nu = 1$, the t-distribution reduces to the *Cauchy* distribution, while in the limit $\nu \to \infty$ the t-distribution $\mathrm{St}(x|\mu, \lambda, \nu)$ becomes a Gaussian $\mathcal{N}(x|\mu, \lambda^{-1})$ with mean μ and precision λ.

Exercise 2.47

From (2.158), we see that Student's t-distribution is obtained by adding up an infinite number of Gaussian distributions having the same mean but different precisions. This can be interpreted as an infinite mixture of Gaussians (Gaussian mixtures will be discussed in detail in Section 2.3.9. The result is a distribution that in general has longer 'tails' than a Gaussian, as was seen in Figure 2.15. This gives the t-distribution an important property called *robustness*, which means that it is much less sensitive than the Gaussian to the presence of a few data points which are *outliers*. The robustness of the t-distribution is illustrated in Figure 2.16, which compares the maximum likelihood solutions for a Gaussian and a t-distribution. Note that the maximum likelihood solution for the t-distribution can be found using the expectation-maximization (EM) algorithm. Here we see that the effect of a small number of

Exercise 12.24

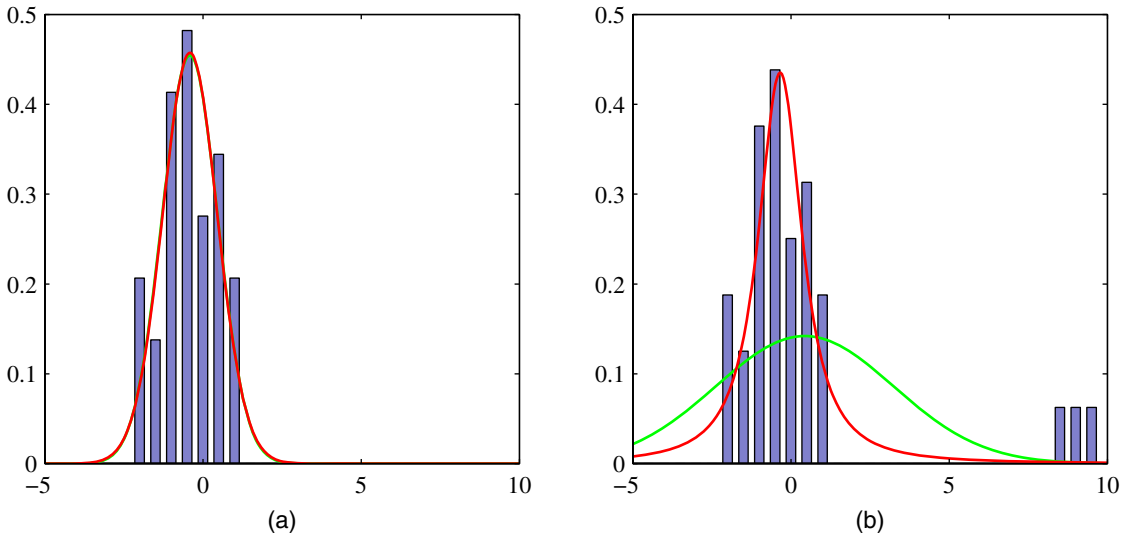


Figure 2.16 Illustration of the robustness of Student's t-distribution compared to a Gaussian. (a) Histogram distribution of 30 data points drawn from a Gaussian distribution, together with the maximum likelihood fit obtained from a t-distribution (red curve) and a Gaussian (green curve, largely hidden by the red curve). Because the t-distribution contains the Gaussian as a special case it gives almost the same solution as the Gaussian. (b) The same data set but with three additional outlying data points showing how the Gaussian (green curve) is strongly distorted by the outliers, whereas the t-distribution (red curve) is relatively unaffected.

outliers is much less significant for the t-distribution than for the Gaussian. Outliers can arise in practical applications either because the process that generates the data corresponds to a distribution having a heavy tail or simply through mislabelled data. Robustness is also an important property for regression problems. Unsurprisingly, the least squares approach to regression does not exhibit robustness, because it corresponds to maximum likelihood under a (conditional) Gaussian distribution. By basing a regression model on a heavy-tailed distribution such as a t-distribution, we obtain a more robust model.

If we go back to (2.158) and substitute the alternative parameters $\nu = 2a$, $\lambda = a/b$, and $\eta = \tau b/a$, we see that the t-distribution can be written in the form

$$\mathrm{St}(x|\mu,\lambda,\nu) = \int_0^\infty \mathcal{N}\left(x|\mu,(\eta\lambda)^{-1}\right)\mathrm{Gam}(\eta|\nu/2,\nu/2)\,\mathrm{d}\eta. \tag{2.160}$$

We can then generalize this to a multivariate Gaussian $\mathcal{N}(\mathbf{x}|\boldsymbol{\mu},\boldsymbol{\Lambda})$ to obtain the corresponding multivariate Student's t-distribution in the form

$$\mathrm{St}(\mathbf{x}|\boldsymbol{\mu},\boldsymbol{\Lambda},\nu) = \int_0^\infty \mathcal{N}(\mathbf{x}|\boldsymbol{\mu},(\eta\boldsymbol{\Lambda})^{-1})\mathrm{Gam}(\eta|\nu/2,\nu/2)\,\mathrm{d}\eta. \tag{2.161}$$

Using the same technique as for the univariate case, we can evaluate this integral to give

Exercise 2.48

$$\mathrm{St}(\mathbf{x}|\boldsymbol{\mu},\boldsymbol{\Lambda},\nu) = \frac{\Gamma(D/2+\nu/2)}{\Gamma(\nu/2)}\frac{|\boldsymbol{\Lambda}|^{1/2}}{(\pi\nu)^{D/2}}\left[1+\frac{\Delta^2}{\nu}\right]^{-D/2-\nu/2} \tag{2.162}$$

where D is the dimensionality of $\mathbf{x}$, and Δ^2 is the squared Mahalanobis distance defined by

$$\Delta^2 = (\mathbf{x}-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Lambda}(\mathbf{x}-\boldsymbol{\mu}). \tag{2.163}$$

This is the multivariate form of Student's t-distribution and satisfies the following properties

Exercise 2.49

$$\mathbb{E}[\mathbf{x}] = \boldsymbol{\mu}, \qquad \text{if} \quad \nu > 1 \tag{2.164}$$

$$\mathrm{cov}[\mathbf{x}] = \frac{\nu}{(\nu-2)}\boldsymbol{\Lambda}^{-1}, \qquad \text{if} \quad \nu > 2 \tag{2.165}$$

$$\mathrm{mode}[\mathbf{x}] = \boldsymbol{\mu} \tag{2.166}$$

with corresponding results for the univariate case.

2.3.8 Periodic variables

Although Gaussian distributions are of great practical significance, both in their own right and as building blocks for more complex probabilistic models, there are situations in which they are inappropriate as density models for continuous variables. One important case, which arises in practical applications, is that of periodic variables.

An example of a periodic variable would be the wind direction at a particular geographical location. We might, for instance, measure values of wind direction on a number of days and wish to summarize this using a parametric distribution. Another example is calendar time, where we may be interested in modelling quantities that are believed to be periodic over 24 hours or over an annual cycle. Such quantities can conveniently be represented using an angular (polar) coordinate $0 \leqslant \theta < 2\pi$.

We might be tempted to treat periodic variables by choosing some direction as the origin and then applying a conventional distribution such as the Gaussian. Such an approach, however, would give results that were strongly dependent on the arbitrary choice of origin. Suppose, for instance, that we have two observations at $\theta_1 = 1^\circ$ and $\theta_2 = 359^\circ$, and we model them using a standard univariate Gaussian distribution. If we choose the origin at 0°, then the sample mean of this data set will be 180° with standard deviation 179°, whereas if we choose the origin at 180°, then the mean will be 0° and the standard deviation will be 1°. We clearly need to develop a special approach for the treatment of periodic variables.

Let us consider the problem of evaluating the mean of a set of observations $\mathcal{D} = \{\theta_1, \ldots, \theta_N\}$ of a periodic variable. From now on, we shall assume that θ is measured in radians. We have already seen that the simple average $(\theta_1 + \cdots + \theta_N)/N$ will be strongly coordinate dependent. To find an invariant measure of the mean, we note that the observations can be viewed as points on the unit circle and can therefore be described instead by two-dimensional unit vectors $\mathbf{x}_1, \ldots, \mathbf{x}_N$ where $\|\mathbf{x}_n\| = 1$ for $n = 1, \ldots, N$, as illustrated in Figure 2.17. We can average the vectors $\{\mathbf{x}_n\}$

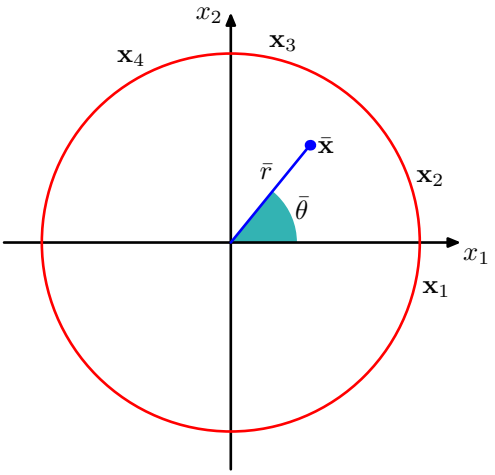


Figure 2.17 Illustration of the representation of values θ_n of a periodic variable as two-dimensional vectors $\mathbf{x}_n$ living on the unit circle. Also shown is the average $\overline{\mathbf{x}}$ of those vectors.

instead to give

$$\overline{\mathbf{x}} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n \tag{2.167}$$

and then find the corresponding angle $\overline{\theta}$ of this average. Clearly, this definition will ensure that the location of the mean is independent of the origin of the angular coordinate. Note that $\overline{\mathbf{x}}$ will typically lie inside the unit circle. The Cartesian coordinates of the observations are given by $\mathbf{x}_n = (\cos\theta_n, \sin\theta_n)$, and we can write the Cartesian coordinates of the sample mean in the form $\overline{\mathbf{x}} = (\overline{r}\cos\overline{\theta}, \overline{r}\sin\overline{\theta})$. Substituting into (2.167) and equating the x_1 and x_2 components then gives

$$\overline{r}\cos\overline{\theta} = \frac{1}{N}\sum_{n=1}^{N}\cos\theta_n, \qquad \overline{r}\sin\overline{\theta} = \frac{1}{N}\sum_{n=1}^{N}\sin\theta_n. \tag{2.168}$$

Taking the ratio, and using the identity $\tan\theta = \sin\theta/\cos\theta$, we can solve for $\overline{\theta}$ to give

$$\overline{\theta} = \tan^{-1}\left\{\frac{\sum_n \sin\theta_n}{\sum_n \cos\theta_n}\right\}. \tag{2.169}$$

Shortly, we shall see how this result arises naturally as the maximum likelihood estimator for an appropriately defined distribution over a periodic variable.

We now consider a periodic generalization of the Gaussian called the *von Mises* distribution. Here we shall limit our attention to univariate distributions, although periodic distributions can also be found over hyperspheres of arbitrary dimension. For an extensive discussion of periodic distributions, see Mardia and Jupp (2000).

By convention, we will consider distributions $p(\theta)$ that have period 2π. Any probability density $p(\theta)$ defined over θ must not only be nonnegative and integrate

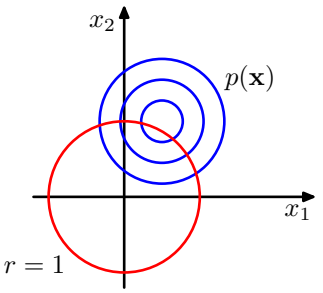


Figure 2.18 The von Mises distribution can be derived by considering a two-dimensional Gaussian of the form (2.173), whose density contours are shown in blue and conditioning on the unit circle shown in red.

to one, but it must also be periodic. Thus $p(\theta)$ must satisfy the three conditions

$$p(\theta) \geqslant 0 \tag{2.170}$$

$$\int_0^{2\pi} p(\theta)\,\mathrm{d}\theta = 1 \tag{2.171}$$

$$p(\theta + 2\pi) = p(\theta). \tag{2.172}$$

From (2.172), it follows that $p(\theta + M2\pi) = p(\theta)$ for any integer M.

We can easily obtain a Gaussian-like distribution that satisfies these three properties as follows. Consider a Gaussian distribution over two variables $\mathbf{x} = (x_1, x_2)$ having mean $\boldsymbol{\mu} = (\mu_1, \mu_2)$ and a covariance matrix $\boldsymbol{\Sigma} = \sigma^2\mathbf{I}$ where $\mathbf{I}$ is the 2×2 identity matrix, so that

$$p(x_1, x_2) = \frac{1}{2\pi\sigma^2} \exp\left\{-\frac{(x_1 - \mu_1)^2 + (x_2 - \mu_2)^2}{2\sigma^2}\right\}. \tag{2.173}$$

The contours of constant $p(\mathbf{x})$ are circles, as illustrated in Figure 2.18. Now suppose we consider the value of this distribution along a circle of fixed radius. Then by construction this distribution will be periodic, although it will not be normalized. We can determine the form of this distribution by transforming from Cartesian coordinates (x_1, x_2) to polar coordinates (r, θ) so that

$$x_1 = r\cos\theta, \qquad x_2 = r\sin\theta. \tag{2.174}$$

We also map the mean $\boldsymbol{\mu}$ into polar coordinates by writing

$$\mu_1 = r_0\cos\theta_0, \qquad \mu_2 = r_0\sin\theta_0. \tag{2.175}$$

Next we substitute these transformations into the two-dimensional Gaussian distribution (2.173), and then condition on the unit circle $r = 1$, noting that we are interested only in the dependence on θ. Focussing on the exponent in the Gaussian distribution we have

$$\begin{aligned} &-\frac{1}{2\sigma^2}\left\{(r\cos\theta - r_0\cos\theta_0)^2 + (r\sin\theta - r_0\sin\theta_0)^2\right\} \\ &= -\frac{1}{2\sigma^2}\left\{1 + r_0^2 - 2r_0\cos\theta\cos\theta_0 - 2r_0\sin\theta\sin\theta_0\right\} \\ &= \frac{r_0}{\sigma^2}\cos(\theta - \theta_0) + \text{const} \end{aligned} \tag{2.176}$$

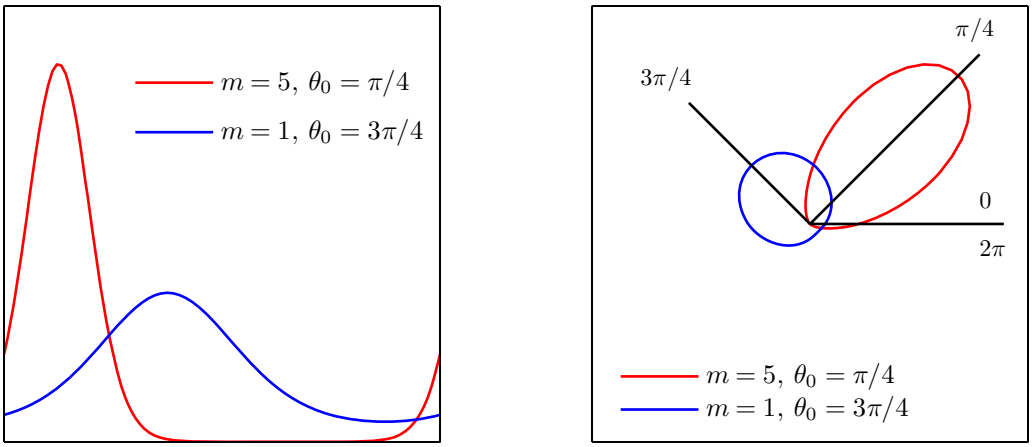


Figure 2.19 The von Mises distribution plotted for two different parameter values, shown as a Cartesian plot on the left and as the corresponding polar plot on the right.

Exercise 2.51

where 'const' denotes terms independent of θ, and we have made use of the following trigonometrical identities

$$\cos^2 A + \sin^2 A = 1 \tag{2.177}$$

$$\cos A \cos B + \sin A \sin B = \cos(A - B). \tag{2.178}$$

If we now define $m = r_0/\sigma^2$, we obtain our final expression for the distribution of $p(\theta)$ along the unit circle $r = 1$ in the form

$$p(\theta|\theta_0, m) = \frac{1}{2\pi I_0(m)} \exp\{m \cos(\theta - \theta_0)\} \tag{2.179}$$

which is called the *von Mises* distribution, or the *circular normal*. Here the parameter θ_0 corresponds to the mean of the distribution, while m, which is known as the *concentration* parameter, is analogous to the inverse variance (precision) for the Gaussian. The normalization coefficient in (2.179) is expressed in terms of $I_0(m)$, which is the zeroth-order Bessel function of the first kind (Abramowitz and Stegun, 1965) and is defined by

$$I_0(m) = \frac{1}{2\pi} \int_0^{2\pi} \exp\{m \cos\theta\}\, \mathrm{d}\theta. \tag{2.180}$$

Exercise 2.52

For large m, the distribution becomes approximately Gaussian. The von Mises distribution is plotted in Figure 2.19, and the function $I_0(m)$ is plotted in Figure 2.20.

Now consider the maximum likelihood estimators for the parameters θ_0 and m for the von Mises distribution. The log likelihood function is given by

$$\ln p(\mathcal{D}|\theta_0, m) = -N \ln(2\pi) - N \ln I_0(m) + m \sum_{n=1}^{N} \cos(\theta_n - \theta_0). \tag{2.181}$$

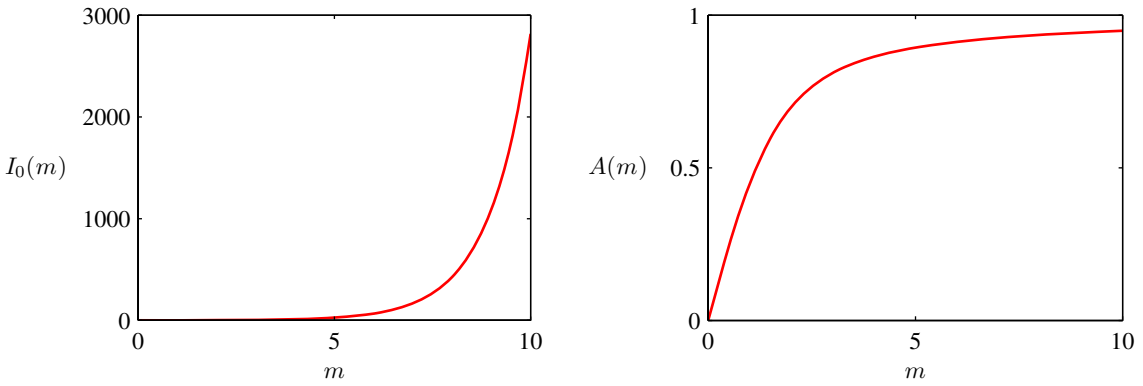


Figure 2.20 Plot of the Bessel function $I_0(m)$ defined by (2.180), together with the function $A(m)$ defined by (2.186).

Setting the derivative with respect to θ_0 equal to zero gives

$$\sum_{n=1}^{N} \sin(\theta_n - \theta_0) = 0. \tag{2.182}$$

To solve for θ_0, we make use of the trigonometric identity

$$\sin(A - B) = \cos B \sin A - \cos A \sin B \tag{2.183}$$

Exercise 2.53

from which we obtain

$$\theta_0^{\mathrm{ML}} = \tan^{-1}\left\{\frac{\sum_n \sin\theta_n}{\sum_n \cos\theta_n}\right\} \tag{2.184}$$

which we recognize as the result (2.169) obtained earlier for the mean of the observations viewed in a two-dimensional Cartesian space.

Similarly, maximizing (2.181) with respect to m, and making use of $I_0'(m) = I_1(m)$ (Abramowitz and Stegun, 1965), we have

$$A(m) = \frac{1}{N}\sum_{n=1}^{N} \cos(\theta_n - \theta_0^{\mathrm{ML}}) \tag{2.185}$$

where we have substituted for the maximum likelihood solution for θ_0^{ML} (recalling that we are performing a joint optimization over θ and m), and we have defined

$$A(m) = \frac{I_1(m)}{I_0(m)}. \tag{2.186}$$

The function $A(m)$ is plotted in Figure 2.20. Making use of the trigonometric identity (2.178), we can write (2.185) in the form

$$A(m_{\mathrm{ML}}) = \left(\frac{1}{N}\sum_{n=1}^{N} \cos\theta_n\right)\cos\theta_0^{\mathrm{ML}} - \left(\frac{1}{N}\sum_{n=1}^{N} \sin\theta_n\right)\sin\theta_0^{\mathrm{ML}}. \tag{2.187}$$

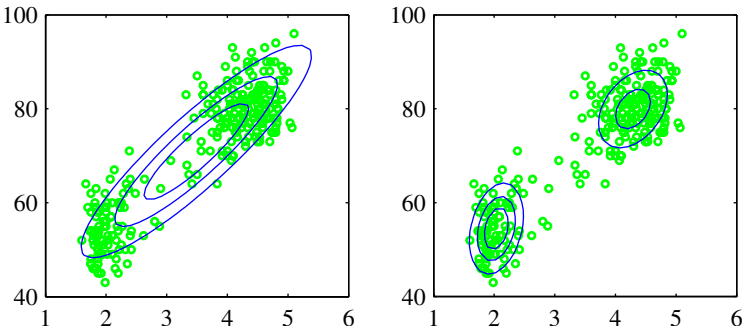


Figure 2.21 Plots of the 'old faithful' data in which the blue curves show contours of constant probability density. On the left is a single Gaussian distribution which has been fitted to the data using maximum likelihood. Note that this distribution fails to capture the two clumps in the data and indeed places much of its probability mass in the central region between the clumps where the data are relatively sparse. On the right the distribution is given by a linear combination of two Gaussians which has been fitted to the data by maximum likelihood using techniques discussed Chapter 9, and which gives a better representation of the data.

The right-hand side of (2.187) is easily evaluated, and the function $A(m)$ can be inverted numerically.

For completeness, we mention briefly some alternative techniques for the construction of periodic distributions. The simplest approach is to use a histogram of observations in which the angular coordinate is divided into fixed bins. This has the virtue of simplicity and flexibility but also suffers from significant limitations, as we shall see when we discuss histogram methods in more detail in Section 2.5. Another approach starts, like the von Mises distribution, from a Gaussian distribution over a Euclidean space but now marginalizes onto the unit circle rather than conditioning (Mardia and Jupp, 2000). However, this leads to more complex forms of distribution and will not be discussed further. Finally, any valid distribution over the real axis (such as a Gaussian) can be turned into a periodic distribution by mapping successive intervals of width 2π onto the periodic variable $(0, 2\pi)$, which corresponds to 'wrapping' the real axis around unit circle. Again, the resulting distribution is more complex to handle than the von Mises distribution.

One limitation of the von Mises distribution is that it is unimodal. By forming *mixtures* of von Mises distributions, we obtain a flexible framework for modelling periodic variables that can handle multimodality. For an example of a machine learning application that makes use of von Mises distributions, see Lawrence *et al.* (2002), and for extensions to modelling conditional densities for regression problems, see Bishop and Nabney (1996).

2.3.9 Mixtures of Gaussians

While the Gaussian distribution has some important analytical properties, it suffers from significant limitations when it comes to modelling real data sets. Consider the example shown in Figure 2.21. This is known as the 'Old Faithful' data set, and comprises 272 measurements of the eruption of the Old Faithful geyser at Yellowstone National Park in the USA. Each measurement comprises the duration of

Appendix A

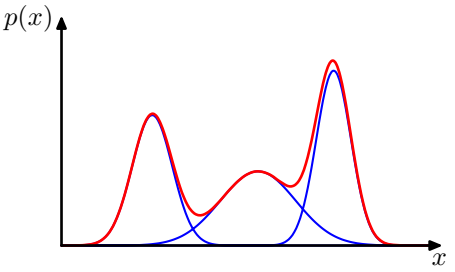


Figure 2.22 Example of a Gaussian mixture distribution in one dimension showing three Gaussians (each scaled by a coefficient) in blue and their sum in red.

the eruption in minutes (horizontal axis) and the time in minutes to the next eruption (vertical axis). We see that the data set forms two dominant clumps, and that a simple Gaussian distribution is unable to capture this structure, whereas a linear superposition of two Gaussians gives a better characterization of the data set.

Such superpositions, formed by taking linear combinations of more basic distributions such as Gaussians, can be formulated as probabilistic models known as *mixture distributions* (McLachlan and Basford, 1988; McLachlan and Peel, 2000). In Figure 2.22 we see that a linear combination of Gaussians can give rise to very complex densities. By using a sufficient number of Gaussians, and by adjusting their means and covariances as well as the coefficients in the linear combination, almost any continuous density can be approximated to arbitrary accuracy.

We therefore consider a superposition of K Gaussian densities of the form

$$p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \tag{2.188}$$

which is called a *mixture of Gaussians*. Each Gaussian density $\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ is called a *component* of the mixture and has its own mean $\boldsymbol{\mu}_k$ and covariance $\boldsymbol{\Sigma}_k$. Contour and surface plots for a Gaussian mixture having 3 components are shown in Figure 2.23.

In this section we shall consider Gaussian components to illustrate the framework of mixture models. More generally, mixture models can comprise linear combinations of other distributions. For instance, in Section 9.3.3 we shall consider mixtures of Bernoulli distributions as an example of a mixture model for discrete variables.

Section 9.3.3

The parameters π_k in (2.188) are called *mixing coefficients*. If we integrate both sides of (2.188) with respect to $\mathbf{x}$, and note that both $p(\mathbf{x})$ and the individual Gaussian components are normalized, we obtain

$$\sum_{k=1}^{K} \pi_k = 1. \tag{2.189}$$

Also, the requirement that $p(\mathbf{x}) \geqslant 0$, together with $\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \geqslant 0$, implies $\pi_k \geqslant 0$ for all k. Combining this with the condition (2.189) we obtain

$$0 \leqslant \pi_k \leqslant 1. \tag{2.190}$$

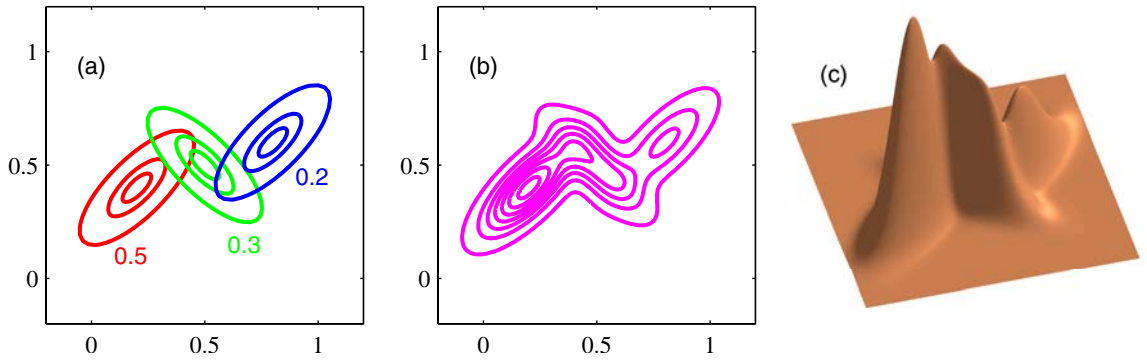


Figure 2.23 Illustration of a mixture of 3 Gaussians in a two-dimensional space. (a) Contours of constant density for each of the mixture components, in which the 3 components are denoted red, blue and green, and the values of the mixing coefficients are shown below each component. (b) Contours of the marginal probability density $p(\mathbf{x})$ of the mixture distribution. (c) A surface plot of the distribution $p(\mathbf{x})$.

We therefore see that the mixing coefficients satisfy the requirements to be probabilities.

From the sum and product rules, the marginal density is given by

$$p(\mathbf{x}) = \sum_{k=1}^{K} p(k)p(\mathbf{x}|k) \tag{2.191}$$

which is equivalent to (2.188) in which we can view $\pi_k = p(k)$ as the prior probability of picking the k^{th} component, and the density $\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) = p(\mathbf{x}|k)$ as the probability of $\mathbf{x}$ conditioned on k. As we shall see in later chapters, an important role is played by the posterior probabilities $p(k|\mathbf{x})$, which are also known as *responsibilities*. From Bayes' theorem these are given by

$$\begin{aligned} \gamma_k(\mathbf{x}) &\equiv p(k|\mathbf{x}) \\ &= \frac{p(k)p(\mathbf{x}|k)}{\sum_l p(l)p(\mathbf{x}|l)} \\ &= \frac{\pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_l \pi_l \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l)}. \end{aligned} \tag{2.192}$$

We shall discuss the probabilistic interpretation of the mixture distribution in greater detail in Chapter 9.

The form of the Gaussian mixture distribution is governed by the parameters $\boldsymbol{\pi}$, $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$, where we have used the notation $\boldsymbol{\pi} \equiv \{\pi_1, \ldots, \pi_K\}$, $\boldsymbol{\mu} \equiv \{\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K\}$ and $\boldsymbol{\Sigma} \equiv \{\boldsymbol{\Sigma}_1, \ldots \boldsymbol{\Sigma}_K\}$. One way to set the values of these parameters is to use maximum likelihood. From (2.188) the log of the likelihood function is given by

$$\ln p(\mathbf{X}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{n=1}^{N} \ln \left\{ \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right\} \tag{2.193}$$

where $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$. We immediately see that the situation is now much more complex than with a single Gaussian, due to the presence of the summation over k inside the logarithm. As a result, the maximum likelihood solution for the parameters no longer has a closed-form analytical solution. One approach to maximizing the likelihood function is to use iterative numerical optimization techniques (Fletcher, 1987; Nocedal and Wright, 1999; Bishop and Nabney, 2008). Alternatively we can employ a powerful framework called *expectation maximization*, which will be discussed at length in Chapter 9.

2.4. The Exponential Family

The probability distributions that we have studied so far in this chapter (with the exception of the Gaussian mixture) are specific examples of a broad class of distributions called the *exponential family* (Duda and Hart, 1973; Bernardo and Smith, 1994). Members of the exponential family have many important properties in common, and it is illuminating to discuss these properties in some generality.

The exponential family of distributions over $\mathbf{x}$, given parameters $\boldsymbol{\eta}$, is defined to be the set of distributions of the form

$$p(\mathbf{x}|\boldsymbol{\eta}) = h(\mathbf{x})g(\boldsymbol{\eta}) \exp\left\{\boldsymbol{\eta}^{\mathrm{T}}\mathbf{u}(\mathbf{x})\right\} \tag{2.194}$$

where $\mathbf{x}$ may be scalar or vector, and may be discrete or continuous. Here $\boldsymbol{\eta}$ are called the *natural parameters* of the distribution, and $\mathbf{u}(\mathbf{x})$ is some function of $\mathbf{x}$. The function $g(\boldsymbol{\eta})$ can be interpreted as the coefficient that ensures that the distribution is normalized and therefore satisfies

$$g(\boldsymbol{\eta}) \int h(\mathbf{x}) \exp\left\{\boldsymbol{\eta}^{\mathrm{T}}\mathbf{u}(\mathbf{x})\right\} \, \mathrm{d}\mathbf{x} = 1 \tag{2.195}$$

where the integration is replaced by summation if $\mathbf{x}$ is a discrete variable.

We begin by taking some examples of the distributions introduced earlier in the chapter and showing that they are indeed members of the exponential family. Consider first the Bernoulli distribution

$$p(x|\mu) = \mathrm{Bern}(x|\mu) = \mu^x(1-\mu)^{1-x}. \tag{2.196}$$

Expressing the right-hand side as the exponential of the logarithm, we have

$$\begin{aligned} p(x|\mu) &= \exp\{x \ln \mu + (1-x)\ln(1-\mu)\} \\ &= (1-\mu)\exp\left\{\ln\left(\frac{\mu}{1-\mu}\right)x\right\}. \end{aligned} \tag{2.197}$$

Comparison with (2.194) allows us to identify

$$\eta = \ln\left(\frac{\mu}{1-\mu}\right) \tag{2.198}$$

which we can solve for μ to give $\mu = \sigma(\eta)$, where

$$\sigma(\eta) = \frac{1}{1 + \exp(-\eta)} \tag{2.199}$$

is called the *logistic sigmoid* function. Thus we can write the Bernoulli distribution using the standard representation (2.194) in the form

$$p(x|\eta) = \sigma(-\eta)\exp(\eta x) \tag{2.200}$$

where we have used $1 - \sigma(\eta) = \sigma(-\eta)$, which is easily proved from (2.199). Comparison with (2.194) shows that

$$u(x) = x \tag{2.201}$$
$$h(x) = 1 \tag{2.202}$$
$$g(\eta) = \sigma(-\eta). \tag{2.203}$$

Next consider the multinomial distribution that, for a single observation $\mathbf{x}$, takes the form

$$p(\mathbf{x}|\boldsymbol{\mu}) = \prod_{k=1}^{M} \mu_k^{x_k} = \exp\left\{\sum_{k=1}^{M} x_k \ln \mu_k\right\} \tag{2.204}$$

where $\mathbf{x} = (x_1, \ldots, x_N)^{\mathrm{T}}$. Again, we can write this in the standard representation (2.194) so that

$$p(\mathbf{x}|\boldsymbol{\eta}) = \exp(\boldsymbol{\eta}^{\mathrm{T}}\mathbf{x}) \tag{2.205}$$

where $\eta_k = \ln \mu_k$, and we have defined $\boldsymbol{\eta} = (\eta_1, \ldots, \eta_M)^{\mathrm{T}}$. Again, comparing with (2.194) we have

$$\mathbf{u}(\mathbf{x}) = \mathbf{x} \tag{2.206}$$
$$h(\mathbf{x}) = 1 \tag{2.207}$$
$$g(\boldsymbol{\eta}) = 1. \tag{2.208}$$

Note that the parameters η_k are not independent because the parameters μ_k are subject to the constraint

$$\sum_{k=1}^{M} \mu_k = 1 \tag{2.209}$$

so that, given any $M - 1$ of the parameters μ_k, the value of the remaining parameter is fixed. In some circumstances, it will be convenient to remove this constraint by expressing the distribution in terms of only $M - 1$ parameters. This can be achieved by using the relationship (2.209) to eliminate μ_M by expressing it in terms of the remaining $\{\mu_k\}$ where $k = 1, \ldots, M - 1$, thereby leaving $M - 1$ parameters. Note that these remaining parameters are still subject to the constraints

$$0 \leqslant \mu_k \leqslant 1, \qquad \sum_{k=1}^{M-1} \mu_k \leqslant 1. \tag{2.210}$$

Making use of the constraint (2.209), the multinomial distribution in this representation then becomes

$$\begin{aligned}
&\exp\left\{\sum_{k=1}^{M} x_k \ln \mu_k\right\} \\
&= \exp\left\{\sum_{k=1}^{M-1} x_k \ln \mu_k + \left(1 - \sum_{k=1}^{M-1} x_k\right) \ln\left(1 - \sum_{k=1}^{M-1} \mu_k\right)\right\} \\
&= \exp\left\{\sum_{k=1}^{M-1} x_k \ln\left(\frac{\mu_k}{1 - \sum_{j=1}^{M-1} \mu_j}\right) + \ln\left(1 - \sum_{k=1}^{M-1} \mu_k\right)\right\}. \qquad (2.211)
\end{aligned}$$

We now identify

$$\ln\left(\frac{\mu_k}{1 - \sum_j \mu_j}\right) = \eta_k \qquad (2.212)$$

which we can solve for μ_k by first summing both sides over k and then rearranging and back-substituting to give

$$\mu_k = \frac{\exp(\eta_k)}{1 + \sum_j \exp(\eta_j)}. \qquad (2.213)$$

This is called the *softmax* function, or the *normalized exponential*. In this representation, the multinomial distribution therefore takes the form

$$p(\mathbf{x}|\boldsymbol{\eta}) = \left(1 + \sum_{k=1}^{M-1} \exp(\eta_k)\right)^{-1} \exp(\boldsymbol{\eta}^{\mathrm{T}}\mathbf{x}). \qquad (2.214)$$

This is the standard form of the exponential family, with parameter vector $\boldsymbol{\eta} = (\eta_1, \ldots, \eta_{M-1})^{\mathrm{T}}$ in which

$$\mathbf{u}(\mathbf{x}) = \mathbf{x} \qquad (2.215)$$

$$h(\mathbf{x}) = 1 \qquad (2.216)$$

$$g(\boldsymbol{\eta}) = \left(1 + \sum_{k=1}^{M-1} \exp(\eta_k)\right)^{-1}. \qquad (2.217)$$

Finally, let us consider the Gaussian distribution. For the univariate Gaussian, we have

$$p(x|\mu, \sigma^2) = \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left\{-\frac{1}{2\sigma^2}(x - \mu)^2\right\} \qquad (2.218)$$

$$= \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left\{-\frac{1}{2\sigma^2}x^2 + \frac{\mu}{\sigma^2}x - \frac{1}{2\sigma^2}\mu^2\right\} \qquad (2.219)$$

Exercise 2.57

which, after some simple rearrangement, can be cast in the standard exponential family form (2.194) with

$$\boldsymbol{\eta} = \begin{pmatrix} \mu/\sigma^2 \\ -1/2\sigma^2 \end{pmatrix} \tag{2.220}$$

$$\mathbf{u}(x) = \begin{pmatrix} x \\ x^2 \end{pmatrix} \tag{2.221}$$

$$h(\mathbf{x}) = (2\pi)^{-1/2} \tag{2.222}$$

$$g(\boldsymbol{\eta}) = (-2\eta_2)^{1/2} \exp\left(\frac{\eta_1^2}{4\eta_2}\right). \tag{2.223}$$

2.4.1 Maximum likelihood and sufficient statistics

Let us now consider the problem of estimating the parameter vector $\boldsymbol{\eta}$ in the general exponential family distribution (2.194) using the technique of maximum likelihood. Taking the gradient of both sides of (2.195) with respect to $\boldsymbol{\eta}$, we have

$$\nabla g(\boldsymbol{\eta}) \int h(\mathbf{x}) \exp\left\{\boldsymbol{\eta}^{\mathrm{T}}\mathbf{u}(\mathbf{x})\right\} \mathrm{d}\mathbf{x} + g(\boldsymbol{\eta}) \int h(\mathbf{x}) \exp\left\{\boldsymbol{\eta}^{\mathrm{T}}\mathbf{u}(\mathbf{x})\right\} \mathbf{u}(\mathbf{x}) \mathrm{d}\mathbf{x} = 0. \tag{2.224}$$

Rearranging, and making use again of (2.195) then gives

$$-\frac{1}{g(\boldsymbol{\eta})}\nabla g(\boldsymbol{\eta}) = g(\boldsymbol{\eta}) \int h(\mathbf{x}) \exp\left\{\boldsymbol{\eta}^{\mathrm{T}}\mathbf{u}(\mathbf{x})\right\} \mathbf{u}(\mathbf{x}) \mathrm{d}\mathbf{x} = \mathbb{E}[\mathbf{u}(\mathbf{x})] \tag{2.225}$$

where we have used (2.194). We therefore obtain the result

$$-\nabla \ln g(\boldsymbol{\eta}) = \mathbb{E}[\mathbf{u}(\mathbf{x})]. \tag{2.226}$$

Exercise 2.58

Note that the covariance of $\mathbf{u}(\mathbf{x})$ can be expressed in terms of the second derivatives of $g(\boldsymbol{\eta})$, and similarly for higher order moments. Thus, provided we can normalize a distribution from the exponential family, we can always find its moments by simple differentiation.

Now consider a set of independent identically distributed data denoted by $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$, for which the likelihood function is given by

$$p(\mathbf{X}|\boldsymbol{\eta}) = \left(\prod_{n=1}^{N} h(\mathbf{x}_n)\right) g(\boldsymbol{\eta})^N \exp\left\{\boldsymbol{\eta}^{\mathrm{T}} \sum_{n=1}^{N} \mathbf{u}(\mathbf{x}_n)\right\}. \tag{2.227}$$

Setting the gradient of $\ln p(\mathbf{X}|\boldsymbol{\eta})$ with respect to $\boldsymbol{\eta}$ to zero, we get the following condition to be satisfied by the maximum likelihood estimator $\boldsymbol{\eta}_{\mathrm{ML}}$

$$-\nabla \ln g(\boldsymbol{\eta}_{\mathrm{ML}}) = \frac{1}{N} \sum_{n=1}^{N} \mathbf{u}(\mathbf{x}_n) \tag{2.228}$$

which can in principle be solved to obtain $\boldsymbol{\eta}_{\mathrm{ML}}$. We see that the solution for the maximum likelihood estimator depends on the data only through $\sum_n \mathbf{u}(\mathbf{x}_n)$, which is therefore called the *sufficient statistic* of the distribution (2.194). We do not need to store the entire data set itself but only the value of the sufficient statistic. For the Bernoulli distribution, for example, the function $\mathbf{u}(x)$ is given just by x and so we need only keep the sum of the data points $\{x_n\}$, whereas for the Gaussian $\mathbf{u}(x) = (x, x^2)^{\mathrm{T}}$, and so we should keep both the sum of $\{x_n\}$ and the sum of $\{x_n^2\}$.

If we consider the limit $N \to \infty$, then the right-hand side of (2.228) becomes $\mathbb{E}[\mathbf{u}(\mathbf{x})]$, and so by comparing with (2.226) we see that in this limit $\boldsymbol{\eta}_{\mathrm{ML}}$ will equal the true value $\boldsymbol{\eta}$.

In fact, this sufficiency property holds also for Bayesian inference, although we shall defer discussion of this until Chapter 8 when we have equipped ourselves with the tools of graphical models and can thereby gain a deeper insight into these important concepts.

2.4.2 Conjugate priors

We have already encountered the concept of a conjugate prior several times, for example in the context of the Bernoulli distribution (for which the conjugate prior is the beta distribution) or the Gaussian (where the conjugate prior for the mean is a Gaussian, and the conjugate prior for the precision is the Wishart distribution). In general, for a given probability distribution $p(\mathbf{x}|\boldsymbol{\eta})$, we can seek a prior $p(\boldsymbol{\eta})$ that is conjugate to the likelihood function, so that the posterior distribution has the same functional form as the prior. For any member of the exponential family (2.194), there exists a conjugate prior that can be written in the form

$$p(\boldsymbol{\eta}|\boldsymbol{\chi}, \nu) = f(\boldsymbol{\chi}, \nu) g(\boldsymbol{\eta})^{\nu} \exp\left\{\nu \boldsymbol{\eta}^{\mathrm{T}} \boldsymbol{\chi}\right\} \tag{2.229}$$

where $f(\boldsymbol{\chi}, \nu)$ is a normalization coefficient, and $g(\boldsymbol{\eta})$ is the same function as appears in (2.194). To see that this is indeed conjugate, let us multiply the prior (2.229) by the likelihood function (2.227) to obtain the posterior distribution, up to a normalization coefficient, in the form

$$p(\boldsymbol{\eta}|\mathbf{X}, \boldsymbol{\chi}, \nu) \propto g(\boldsymbol{\eta})^{\nu+N} \exp\left\{\boldsymbol{\eta}^{\mathrm{T}} \left(\sum_{n=1}^{N} \mathbf{u}(\mathbf{x}_n) + \nu \boldsymbol{\chi}\right)\right\}. \tag{2.230}$$

This again takes the same functional form as the prior (2.229), confirming conjugacy. Furthermore, we see that the parameter ν can be interpreted as a effective number of pseudo-observations in the prior, each of which has a value for the sufficient statistic $\mathbf{u}(\mathbf{x})$ given by $\boldsymbol{\chi}$.

2.4.3 Noninformative priors

In some applications of probabilistic inference, we may have prior knowledge that can be conveniently expressed through the prior distribution. For example, if the prior assigns zero probability to some value of variable, then the posterior distribution will necessarily also assign zero probability to that value, irrespective of

any subsequent observations of data. In many cases, however, we may have little idea of what form the distribution should take. We may then seek a form of prior distribution, called a *noninformative prior*, which is intended to have as little influence on the posterior distribution as possible (Jeffries, 1946; Box and Tao, 1973; Bernardo and Smith, 1994). This is sometimes referred to as 'letting the data speak for themselves'.

If we have a distribution $p(x|\lambda)$ governed by a parameter λ, we might be tempted to propose a prior distribution $p(\lambda) = \text{const}$ as a suitable prior. If λ is a discrete variable with K states, this simply amounts to setting the prior probability of each state to $1/K$. In the case of continuous parameters, however, there are two potential difficulties with this approach. The first is that, if the domain of λ is unbounded, this prior distribution cannot be correctly normalized because the integral over λ diverges. Such priors are called *improper*. In practice, improper priors can often be used provided the corresponding posterior distribution is *proper*, i.e., that it can be correctly normalized. For instance, if we put a uniform prior distribution over the mean of a Gaussian, then the posterior distribution for the mean, once we have observed at least one data point, will be proper.

A second difficulty arises from the transformation behaviour of a probability density under a nonlinear change of variables, given by (1.27). If a function $h(\lambda)$ is constant, and we change variables to $\lambda = \eta^2$, then $\widehat{h}(\eta) = h(\eta^2)$ will also be constant. However, if we choose the density $p_\lambda(\lambda)$ to be constant, then the density of η will be given, from (1.27), by

$$p_\eta(\eta) = p_\lambda(\lambda)\left|\frac{\mathrm{d}\lambda}{\mathrm{d}\eta}\right| = p_\lambda(\eta^2)2\eta \propto \eta \tag{2.231}$$

and so the density over η will not be constant. This issue does not arise when we use maximum likelihood, because the likelihood function $p(x|\lambda)$ is a simple function of λ and so we are free to use any convenient parameterization. If, however, we are to choose a prior distribution that is constant, we must take care to use an appropriate representation for the parameters.

Here we consider two simple examples of noninformative priors (Berger, 1985). First of all, if a density takes the form

$$p(x|\mu) = f(x - \mu) \tag{2.232}$$

then the parameter μ is known as a *location parameter*. This family of densities exhibits *translation invariance* because if we shift x by a constant to give $\widehat{x} = x + c$, then

$$p(\widehat{x}|\widehat{\mu}) = f(\widehat{x} - \widehat{\mu}) \tag{2.233}$$

where we have defined $\widehat{\mu} = \mu + c$. Thus the density takes the same form in the new variable as in the original one, and so the density is independent of the choice of origin. We would like to choose a prior distribution that reflects this translation invariance property, and so we choose a prior that assigns equal probability mass to

an interval $A \leqslant \mu \leqslant B$ as to the shifted interval $A - c \leqslant \mu \leqslant B - c$. This implies

$$\int_A^B p(\mu)\,\mathrm{d}\mu = \int_{A-c}^{B-c} p(\mu)\,\mathrm{d}\mu = \int_A^B p(\mu - c)\,\mathrm{d}\mu \tag{2.234}$$

and because this must hold for all choices of A and B, we have

$$p(\mu - c) = p(\mu) \tag{2.235}$$

which implies that $p(\mu)$ is constant. An example of a location parameter would be the mean μ of a Gaussian distribution. As we have seen, the conjugate prior distribution for μ in this case is a Gaussian $p(\mu|\mu_0, \sigma_0^2) = \mathcal{N}(\mu|\mu_0, \sigma_0^2)$, and we obtain a noninformative prior by taking the limit $\sigma_0^2 \to \infty$. Indeed, from (2.141) and (2.142) we see that this gives a posterior distribution over μ in which the contributions from the prior vanish.

As a second example, consider a density of the form

$$p(x|\sigma) = \frac{1}{\sigma} f\left(\frac{x}{\sigma}\right) \tag{2.236}$$

Exercise 2.59

where $\sigma > 0$. Note that this will be a normalized density provided $f(x)$ is correctly normalized. The parameter σ is known as a *scale parameter*, and the density exhibits *scale invariance* because if we scale x by a constant to give $\widehat{x} = cx$, then

$$p(\widehat{x}|\widehat{\sigma}) = \frac{1}{\widehat{\sigma}} f\left(\frac{\widehat{x}}{\widehat{\sigma}}\right) \tag{2.237}$$

where we have defined $\widehat{\sigma} = c\sigma$. This transformation corresponds to a change of scale, for example from meters to kilometers if x is a length, and we would like to choose a prior distribution that reflects this scale invariance. If we consider an interval $A \leqslant \sigma \leqslant B$, and a scaled interval $A/c \leqslant \sigma \leqslant B/c$, then the prior should assign equal probability mass to these two intervals. Thus we have

$$\int_A^B p(\sigma)\,\mathrm{d}\sigma = \int_{A/c}^{B/c} p(\sigma)\,\mathrm{d}\sigma = \int_A^B p\left(\frac{1}{c}\sigma\right)\frac{1}{c}\,\mathrm{d}\sigma \tag{2.238}$$

and because this must hold for choices of A and B, we have

$$p(\sigma) = p\left(\frac{1}{c}\sigma\right)\frac{1}{c} \tag{2.239}$$

and hence $p(\sigma) \propto 1/\sigma$. Note that again this is an improper prior because the integral of the distribution over $0 \leqslant \sigma \leqslant \infty$ is divergent. It is sometimes also convenient to think of the prior distribution for a scale parameter in terms of the density of the log of the parameter. Using the transformation rule (1.27) for densities we see that $p(\ln \sigma) = \text{const}$. Thus, for this prior there is the same probability mass in the range $1 \leqslant \sigma \leqslant 10$ as in the range $10 \leqslant \sigma \leqslant 100$ and in $100 \leqslant \sigma \leqslant 1000$.

An example of a scale parameter would be the standard deviation σ of a Gaussian distribution, after we have taken account of the location parameter μ, because

$$\mathcal{N}(x|\mu,\sigma^2) \propto \sigma^{-1}\exp\left\{-(\widetilde{x}/\sigma)^2\right\} \tag{2.240}$$

where $\widetilde{x} = x - \mu$. As discussed earlier, it is often more convenient to work in terms of the precision $\lambda = 1/\sigma^2$ rather than σ itself. Using the transformation rule for densities, we see that a distribution $p(\sigma) \propto 1/\sigma$ corresponds to a distribution over λ of the form $p(\lambda) \propto 1/\lambda$. We have seen that the conjugate prior for λ was the gamma distribution $\mathrm{Gam}(\lambda|a_0,b_0)$ given by (2.146). The noninformative prior is obtained as the special case $a_0 = b_0 = 0$. Again, if we examine the results (2.150) and (2.151) for the posterior distribution of λ, we see that for $a_0 = b_0 = 0$, the posterior depends only on terms arising from the data and not from the prior.

Section 2.3

2.5. Nonparametric Methods

Throughout this chapter, we have focussed on the use of probability distributions having specific functional forms governed by a small number of parameters whose values are to be determined from a data set. This is called the *parametric* approach to density modelling. An important limitation of this approach is that the chosen density might be a poor model of the distribution that generates the data, which can result in poor predictive performance. For instance, if the process that generates the data is multimodal, then this aspect of the distribution can never be captured by a Gaussian, which is necessarily unimodal.

In this final section, we consider some *nonparametric* approaches to density estimation that make few assumptions about the form of the distribution. Here we shall focus mainly on simple frequentist methods. The reader should be aware, however, that nonparametric Bayesian methods are attracting increasing interest (Walker *et al.*, 1999; Neal, 2000; Müller and Quintana, 2004; Teh *et al.*, 2006).

Let us start with a discussion of histogram methods for density estimation, which we have already encountered in the context of marginal and conditional distributions in Figure 1.11 and in the context of the central limit theorem in Figure 2.6. Here we explore the properties of histogram density models in more detail, focussing on the case of a single continuous variable x. Standard histograms simply partition x into distinct bins of width Δ_i and then count the number n_i of observations of x falling in bin i. In order to turn this count into a normalized probability density, we simply divide by the total number N of observations and by the width Δ_i of the bins to obtain probability values for each bin given by

$$p_i = \frac{n_i}{N\Delta_i} \tag{2.241}$$

for which it is easily seen that $\int p(x)\,\mathrm{d}x = 1$. This gives a model for the density $p(x)$ that is constant over the width of each bin, and often the bins are chosen to have the same width $\Delta_i = \Delta$.

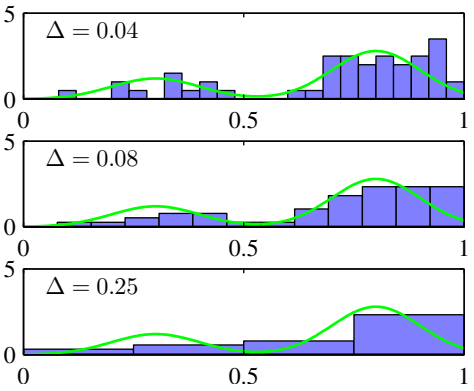


Figure 2.24 An illustration of the histogram approach to density estimation, in which a data set of 50 data points is generated from the distribution shown by the green curve. Histogram density estimates, based on (2.241), with a common bin width Δ are shown for various values of Δ.

In Figure 2.24, we show an example of histogram density estimation. Here the data is drawn from the distribution, corresponding to the green curve, which is formed from a mixture of two Gaussians. Also shown are three examples of histogram density estimates corresponding to three different choices for the bin width Δ. We see that when Δ is very small (top figure), the resulting density model is very spiky, with a lot of structure that is not present in the underlying distribution that generated the data set. Conversely, if Δ is too large (bottom figure) then the result is a model that is too smooth and that consequently fails to capture the bimodal property of the green curve. The best results are obtained for some intermediate value of Δ (middle figure). In principle, a histogram density model is also dependent on the choice of edge location for the bins, though this is typically much less significant than the value of Δ.

Note that the histogram method has the property (unlike the methods to be discussed shortly) that, once the histogram has been computed, the data set itself can be discarded, which can be advantageous if the data set is large. Also, the histogram approach is easily applied if the data points are arriving sequentially.

In practice, the histogram technique can be useful for obtaining a quick visualization of data in one or two dimensions but is unsuited to most density estimation applications. One obvious problem is that the estimated density has discontinuities that are due to the bin edges rather than any property of the underlying distribution that generated the data. Another major limitation of the histogram approach is its scaling with dimensionality. If we divide each variable in a D-dimensional space into M bins, then the total number of bins will be M^D. This exponential scaling with D is an example of the curse of dimensionality. In a space of high dimensionality, the quantity of data needed to provide meaningful estimates of local probability density would be prohibitive.

Section 1.4

The histogram approach to density estimation does, however, teach us two important lessons. First, to estimate the probability density at a particular location, we should consider the data points that lie within some local neighbourhood of that point. Note that the concept of locality requires that we assume some form of distance measure, and here we have been assuming Euclidean distance. For histograms,

this neighbourhood property was defined by the bins, and there is a natural 'smoothing' parameter describing the spatial extent of the local region, in this case the bin width. Second, the value of the smoothing parameter should be neither too large nor too small in order to obtain good results. This is reminiscent of the choice of model complexity in polynomial curve fitting discussed in Chapter 1 where the degree M of the polynomial, or alternatively the value α of the regularization parameter, was optimal for some intermediate value, neither too large nor too small. Armed with these insights, we turn now to a discussion of two widely used nonparametric techniques for density estimation, kernel estimators and nearest neighbours, which have better scaling with dimensionality than the simple histogram model.

2.5.1 Kernel density estimators

Let us suppose that observations are being drawn from some unknown probability density $p(\mathbf{x})$ in some D-dimensional space, which we shall take to be Euclidean, and we wish to estimate the value of $p(\mathbf{x})$. From our earlier discussion of locality, let us consider some small region $\mathcal{R}$ containing $\mathbf{x}$. The probability mass associated with this region is given by

$$P = \int_{\mathcal{R}} p(\mathbf{x})\,\mathrm{d}\mathbf{x}. \tag{2.242}$$

Now suppose that we have collected a data set comprising N observations drawn from $p(\mathbf{x})$. Because each data point has a probability P of falling within $\mathcal{R}$, the total number K of points that lie inside $\mathcal{R}$ will be distributed according to the binomial distribution

Section 2.1

$$\mathrm{Bin}(K|N,P) = \frac{N!}{K!(N-K)!}P^K(1-P)^{1-K}. \tag{2.243}$$

Using (2.11), we see that the mean fraction of points falling inside the region is $\mathbb{E}[K/N] = P$, and similarly using (2.12) we see that the variance around this mean is $\mathrm{var}[K/N] = P(1-P)/N$. For large N, this distribution will be sharply peaked around the mean and so

$$K \simeq NP. \tag{2.244}$$

If, however, we also assume that the region $\mathcal{R}$ is sufficiently small that the probability density $p(\mathbf{x})$ is roughly constant over the region, then we have

$$P \simeq p(\mathbf{x})V \tag{2.245}$$

where V is the volume of $\mathcal{R}$. Combining (2.244) and (2.245), we obtain our density estimate in the form

$$p(\mathbf{x}) = \frac{K}{NV}. \tag{2.246}$$

Note that the validity of (2.246) depends on two contradictory assumptions, namely that the region $\mathcal{R}$ be sufficiently small that the density is approximately constant over the region and yet sufficiently large (in relation to the value of that density) that the number K of points falling inside the region is sufficient for the binomial distribution to be sharply peaked.

We can exploit the result (2.246) in two different ways. Either we can fix K and determine the value of V from the data, which gives rise to the K-nearest-neighbour technique discussed shortly, or we can fix V and determine K from the data, giving rise to the kernel approach. It can be shown that both the K-nearest-neighbour density estimator and the kernel density estimator converge to the true probability density in the limit $N \to \infty$ provided V shrinks suitably with N, and K grows with N (Duda and Hart, 1973).

We begin by discussing the kernel method in detail, and to start with we take the region $\mathcal{R}$ to be a small hypercube centred on the point $\mathbf{x}$ at which we wish to determine the probability density. In order to count the number K of points falling within this region, it is convenient to define the following function

$$k(\mathbf{u}) = \begin{cases} 1, & |u_i| \leqslant 1/2, \quad i = 1, \ldots, D, \\ 0, & \text{otherwise} \end{cases} \tag{2.247}$$

which represents a unit cube centred on the origin. The function $k(\mathbf{u})$ is an example of a *kernel function*, and in this context is also called a *Parzen window*. From (2.247), the quantity $k((\mathbf{x} - \mathbf{x}_n)/h)$ will be one if the data point $\mathbf{x}_n$ lies inside a cube of side h centred on $\mathbf{x}$, and zero otherwise. The total number of data points lying inside this cube will therefore be

$$K = \sum_{n=1}^{N} k\left(\frac{\mathbf{x} - \mathbf{x}_n}{h}\right). \tag{2.248}$$

Substituting this expression into (2.246) then gives the following result for the estimated density at $\mathbf{x}$

$$p(\mathbf{x}) = \frac{1}{N} \sum_{n=1}^{N} \frac{1}{h^D} k\left(\frac{\mathbf{x} - \mathbf{x}_n}{h}\right) \tag{2.249}$$

where we have used $V = h^D$ for the volume of a hypercube of side h in D dimensions. Using the symmetry of the function $k(\mathbf{u})$, we can now re-interpret this equation, not as a single cube centred on $\mathbf{x}$ but as the sum over N cubes centred on the N data points $\mathbf{x}_n$.

As it stands, the kernel density estimator (2.249) will suffer from one of the same problems that the histogram method suffered from, namely the presence of artificial discontinuities, in this case at the boundaries of the cubes. We can obtain a smoother density model if we choose a smoother kernel function, and a common choice is the Gaussian, which gives rise to the following kernel density model

$$p(\mathbf{x}) = \frac{1}{N} \sum_{n=1}^{N} \frac{1}{(2\pi h^2)^{1/2}} \exp\left\{-\frac{\|\mathbf{x} - \mathbf{x}_n\|^2}{2h^2}\right\} \tag{2.250}$$

where h represents the standard deviation of the Gaussian components. Thus our density model is obtained by placing a Gaussian over each data point and then adding up the contributions over the whole data set, and then dividing by N so that the density is correctly normalized. In Figure 2.25, we apply the model (2.250) to the data

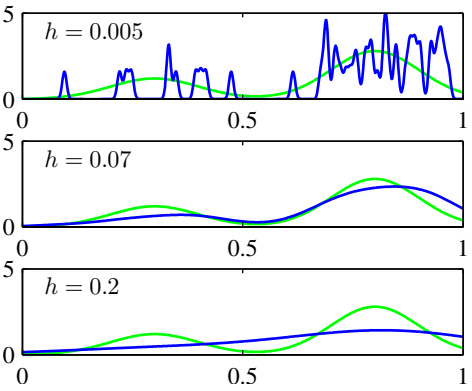


Figure 2.25 Illustration of the kernel density model (2.250) applied to the same data set used to demonstrate the histogram approach in Figure 2.24. We see that h acts as a smoothing parameter and that if it is set too small (top panel), the result is a very noisy density model, whereas if it is set too large (bottom panel), then the bimodal nature of the underlying distribution from which the data is generated (shown by the green curve) is washed out. The best density model is obtained for some intermediate value of h (middle panel).

set used earlier to demonstrate the histogram technique. We see that, as expected, the parameter h plays the role of a smoothing parameter, and there is a trade-off between sensitivity to noise at small h and over-smoothing at large h. Again, the optimization of h is a problem in model complexity, analogous to the choice of bin width in histogram density estimation, or the degree of the polynomial used in curve fitting.

We can choose any other kernel function $k(\mathbf{u})$ in (2.249) subject to the conditions

$$k(\mathbf{u}) \geqslant 0, \tag{2.251}$$

$$\int k(\mathbf{u})\,\mathrm{d}\mathbf{u} = 1 \tag{2.252}$$

which ensure that the resulting probability distribution is nonnegative everywhere and integrates to one. The class of density model given by (2.249) is called a kernel density estimator, or *Parzen* estimator. It has a great merit that there is no computation involved in the 'training' phase because this simply requires storage of the training set. However, this is also one of its great weaknesses because the computational cost of evaluating the density grows linearly with the size of the data set.

2.5.2 Nearest-neighbour methods

One of the difficulties with the kernel approach to density estimation is that the parameter h governing the kernel width is fixed for all kernels. In regions of high data density, a large value of h may lead to over-smoothing and a washing out of structure that might otherwise be extracted from the data. However, reducing h may lead to noisy estimates elsewhere in data space where the density is smaller. Thus the optimal choice for h may be dependent on location within the data space. This issue is addressed by nearest-neighbour methods for density estimation.

We therefore return to our general result (2.246) for local density estimation, and instead of fixing V and determining the value of K from the data, we consider a fixed value of K and use the data to find an appropriate value for V. To do this, we consider a small sphere centred on the point $\mathbf{x}$ at which we wish to estimate the

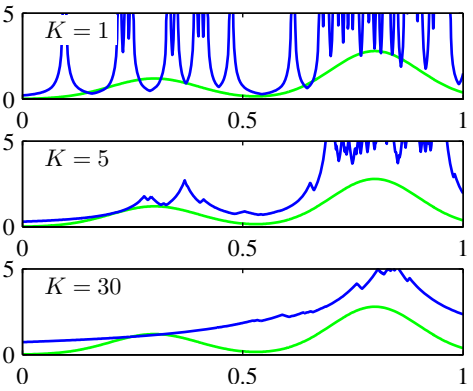


Figure 2.26 Illustration of K-nearest-neighbour density estimation using the same data set as in Figures 2.25 and 2.24. We see that the parameter K governs the degree of smoothing, so that a small value of K leads to a very noisy density model (top panel), whereas a large value (bottom panel) smoothes out the bimodal nature of the true distribution (shown by the green curve) from which the data set was generated.

density $p(\mathbf{x})$, and we allow the radius of the sphere to grow until it contains precisely K data points. The estimate of the density $p(\mathbf{x})$ is then given by (2.246) with V set to the volume of the resulting sphere. This technique is known as K *nearest neighbours* and is illustrated in Figure 2.26, for various choices of the parameter K, using the same data set as used in Figure 2.24 and Figure 2.25. We see that the value of K now governs the degree of smoothing and that again there is an optimum choice for K that is neither too large nor too small. Note that the model produced by K nearest neighbours is not a true density model because the integral over all space diverges.

Exercise 2.61

We close this chapter by showing how the K-nearest-neighbour technique for density estimation can be extended to the problem of classification. To do this, we apply the K-nearest-neighbour density estimation technique to each class separately and then make use of Bayes' theorem. Let us suppose that we have a data set comprising N_k points in class $\mathcal{C}_k$ with N points in total, so that $\sum_k N_k = N$. If we wish to classify a new point $\mathbf{x}$, we draw a sphere centred on $\mathbf{x}$ containing precisely K points irrespective of their class. Suppose this sphere has volume V and contains K_k points from class $\mathcal{C}_k$. Then (2.246) provides an estimate of the density associated with each class

$$p(\mathbf{x}|\mathcal{C}_k) = \frac{K_k}{N_k V}. \tag{2.253}$$

Similarly, the unconditional density is given by

$$p(\mathbf{x}) = \frac{K}{NV} \tag{2.254}$$

while the class priors are given by

$$p(\mathcal{C}_k) = \frac{N_k}{N}. \tag{2.255}$$

We can now combine (2.253), (2.254), and (2.255) using Bayes' theorem to obtain the posterior probability of class membership

$$p(\mathcal{C}_k|\mathbf{x}) = \frac{p(\mathbf{x}|\mathcal{C}_k)p(\mathcal{C}_k)}{p(\mathbf{x})} = \frac{K_k}{K}. \tag{2.256}$$

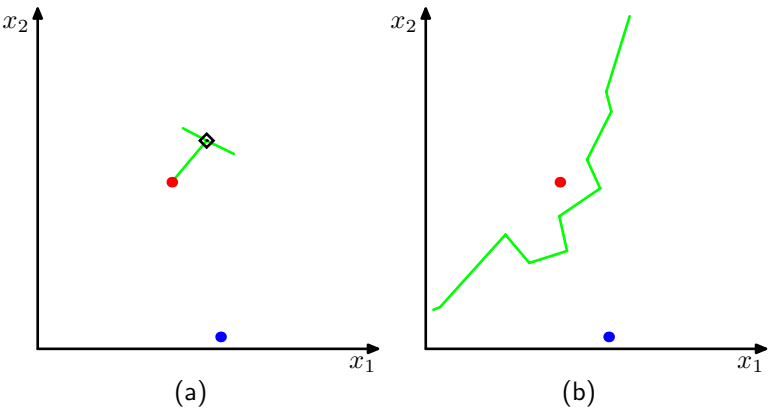


Figure 2.27 (a) In the K-nearest-neighbour classifier, a new point, shown by the black diamond, is classified according to the majority class membership of the K closest training data points, in this case $K = 3$. (b) In the nearest-neighbour ($K = 1$) approach to classification, the resulting decision boundary is composed of hyperplanes that form perpendicular bisectors of pairs of points from different classes.

If we wish to minimize the probability of misclassification, this is done by assigning the test point $\mathbf{x}$ to the class having the largest posterior probability, corresponding to the largest value of K_k/K. Thus to classify a new point, we identify the K nearest points from the training data set and then assign the new point to the class having the largest number of representatives amongst this set. Ties can be broken at random. The particular case of $K = 1$ is called the *nearest-neighbour* rule, because a test point is simply assigned to the same class as the nearest point from the training set. These concepts are illustrated in Figure 2.27.

In Figure 2.28, we show the results of applying the K-nearest-neighbour algorithm to the oil flow data, introduced in Chapter 1, for various values of K. As expected, we see that K controls the degree of smoothing, so that small K produces many small regions of each class, whereas large K leads to fewer larger regions.

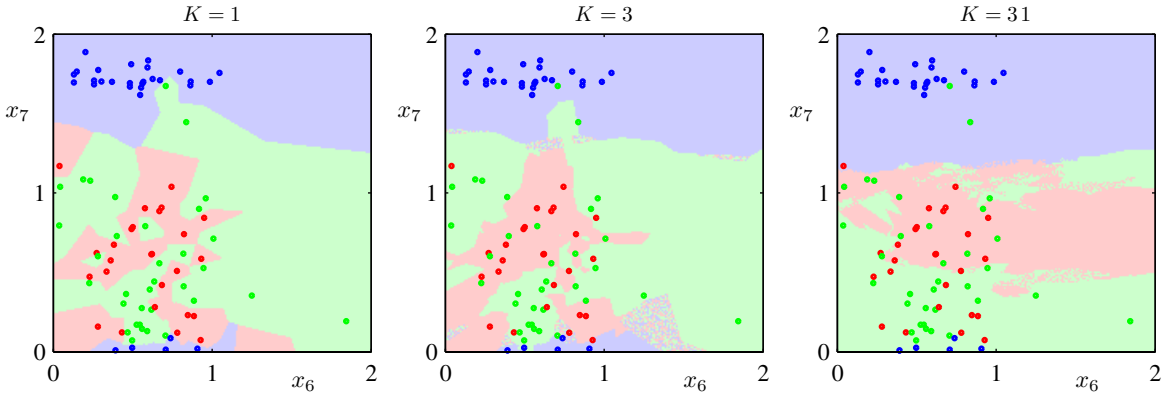


Figure 2.28 Plot of 200 data points from the oil data set showing values of x_6 plotted against x_7, where the red, green, and blue points correspond to the 'laminar', 'annular', and 'homogeneous' classes, respectively. Also shown are the classifications of the input space given by the K-nearest-neighbour algorithm for various values of K.

An interesting property of the nearest-neighbour ($K = 1$) classifier is that, in the limit $N \to \infty$, the error rate is never more than twice the minimum achievable error rate of an optimal classifier, i.e., one that uses the true class distributions (Cover and Hart, 1967) .

As discussed so far, both the K-nearest-neighbour method, and the kernel density estimator, require the entire training data set to be stored, leading to expensive computation if the data set is large. This effect can be offset, at the expense of some additional one-off computation, by constructing tree-based search structures to allow (approximate) near neighbours to be found efficiently without doing an exhaustive search of the data set. Nevertheless, these nonparametric methods are still severely limited. On the other hand, we have seen that simple parametric models are very restricted in terms of the forms of distribution that they can represent. We therefore need to find density models that are very flexible and yet for which the complexity of the models can be controlled independently of the size of the training set, and we shall see in subsequent chapters how to achieve this.

Exercises

2.1 (⋆) **www** Verify that the Bernoulli distribution (2.2) satisfies the following properties

$$\sum_{x=0}^{1} p(x|\mu) = 1 \tag{2.257}$$

$$\mathbb{E}[x] = \mu \tag{2.258}$$

$$\mathrm{var}[x] = \mu(1-\mu). \tag{2.259}$$

Show that the entropy $\mathrm{H}[x]$ of a Bernoulli distributed random binary variable x is given by

$$\mathrm{H}[x] = -\mu \ln \mu - (1-\mu)\ln(1-\mu). \tag{2.260}$$

2.2 (⋆⋆) The form of the Bernoulli distribution given by (2.2) is not symmetric between the two values of x. In some situations, it will be more convenient to use an equivalent formulation for which $x \in \{-1, 1\}$, in which case the distribution can be written

$$p(x|\mu) = \left(\frac{1-\mu}{2}\right)^{(1-x)/2} \left(\frac{1+\mu}{2}\right)^{(1+x)/2} \tag{2.261}$$

where $\mu \in [-1, 1]$. Show that the distribution (2.261) is normalized, and evaluate its mean, variance, and entropy.

2.3 (⋆⋆) **www** In this exercise, we prove that the binomial distribution (2.9) is normalized. First use the definition (2.10) of the number of combinations of m identical objects chosen from a total of N to show that

$$\binom{N}{m} + \binom{N}{m-1} = \binom{N+1}{m}. \tag{2.262}$$

Use this result to prove by induction the following result

$$(1+x)^N = \sum_{m=0}^{N} \binom{N}{m} x^m \tag{2.263}$$

which is known as the *binomial theorem*, and which is valid for all real values of x. Finally, show that the binomial distribution is normalized, so that

$$\sum_{m=0}^{N} \binom{N}{m} \mu^m (1-\mu)^{N-m} = 1 \tag{2.264}$$

which can be done by first pulling out a factor $(1-\mu)^N$ out of the summation and then making use of the binomial theorem.

2.4 ($\star\star$) Show that the mean of the binomial distribution is given by (2.11). To do this, differentiate both sides of the normalization condition (2.264) with respect to μ and then rearrange to obtain an expression for the mean of n. Similarly, by differentiating (2.264) twice with respect to μ and making use of the result (2.11) for the mean of the binomial distribution prove the result (2.12) for the variance of the binomial.

2.5 ($\star\star$) **www** In this exercise, we prove that the beta distribution, given by (2.13), is correctly normalized, so that (2.14) holds. This is equivalent to showing that

$$\int_0^1 \mu^{a-1}(1-\mu)^{b-1}\,\mathrm{d}\mu = \frac{\Gamma(a)\Gamma(b)}{\Gamma(a+b)}. \tag{2.265}$$

From the definition (1.141) of the gamma function, we have

$$\Gamma(a)\Gamma(b) = \int_0^\infty \exp(-x)x^{a-1}\,\mathrm{d}x \int_0^\infty \exp(-y)y^{b-1}\,\mathrm{d}y. \tag{2.266}$$

Use this expression to prove (2.265) as follows. First bring the integral over y inside the integrand of the integral over x, next make the change of variable $t = y + x$ where x is fixed, then interchange the order of the x and t integrations, and finally make the change of variable $x = t\mu$ where t is fixed.

2.6 ($\star$) Make use of the result (2.265) to show that the mean, variance, and mode of the beta distribution (2.13) are given respectively by

$$\mathbb{E}[\mu] = \frac{a}{a+b} \tag{2.267}$$

$$\mathrm{var}[\mu] = \frac{ab}{(a+b)^2(a+b+1)} \tag{2.268}$$

$$\mathrm{mode}[\mu] = \frac{a-1}{a+b-2}. \tag{2.269}$$

2.7 ($\star\star$) Consider a binomial random variable x given by (2.9), with prior distribution for μ given by the beta distribution (2.13), and suppose we have observed m occurrences of $x = 1$ and l occurrences of $x = 0$. Show that the posterior mean value of x lies between the prior mean and the maximum likelihood estimate for μ. To do this, show that the posterior mean can be written as λ times the prior mean plus $(1 - \lambda)$ times the maximum likelihood estimate, where $0 \leqslant \lambda \leqslant 1$. This illustrates the concept of the posterior distribution being a compromise between the prior distribution and the maximum likelihood solution.

2.8 ($\star$) Consider two variables x and y with joint distribution $p(x, y)$. Prove the following two results

$$\mathbb{E}[x] = \mathbb{E}_y\left[\mathbb{E}_x[x|y]\right] \tag{2.270}$$

$$\mathrm{var}[x] = \mathbb{E}_y\left[\mathrm{var}_x[x|y]\right] + \mathrm{var}_y\left[\mathbb{E}_x[x|y]\right]. \tag{2.271}$$

Here $\mathbb{E}_x[x|y]$ denotes the expectation of x under the conditional distribution $p(x|y)$, with a similar notation for the conditional variance.

2.9 ($\star\star\star$) **www** . In this exercise, we prove the normalization of the Dirichlet distribution (2.38) using induction. We have already shown in Exercise 2.5 that the beta distribution, which is a special case of the Dirichlet for $M = 2$, is normalized. We now assume that the Dirichlet distribution is normalized for $M - 1$ variables and prove that it is normalized for M variables. To do this, consider the Dirichlet distribution over M variables, and take account of the constraint $\sum_{k=1}^{M} \mu_k = 1$ by eliminating μ_M, so that the Dirichlet is written

$$p_M(\mu_1, \ldots, \mu_{M-1}) = C_M \prod_{k=1}^{M-1} \mu_k^{\alpha_k - 1} \left(1 - \sum_{j=1}^{M-1} \mu_j\right)^{\alpha_M - 1} \tag{2.272}$$

and our goal is to find an expression for C_M. To do this, integrate over μ_{M-1}, taking care over the limits of integration, and then make a change of variable so that this integral has limits 0 and 1. By assuming the correct result for C_{M-1} and making use of (2.265), derive the expression for C_M.

2.10 ($\star\star$) Using the property $\Gamma(x + 1) = x\Gamma(x)$ of the gamma function, derive the following results for the mean, variance, and covariance of the Dirichlet distribution given by (2.38)

$$\mathbb{E}[\mu_j] = \frac{\alpha_j}{\alpha_0} \tag{2.273}$$

$$\mathrm{var}[\mu_j] = \frac{\alpha_j(\alpha_0 - \alpha_j)}{\alpha_0^2(\alpha_0 + 1)} \tag{2.274}$$

$$\mathrm{cov}[\mu_j \mu_l] = -\frac{\alpha_j \alpha_l}{\alpha_0^2(\alpha_0 + 1)}, \qquad j \neq l \tag{2.275}$$

where α_0 is defined by (2.39).

2.11 (⋆) **www** By expressing the expectation of $\ln \mu_j$ under the Dirichlet distribution (2.38) as a derivative with respect to α_j, show that

$$\mathbb{E}[\ln \mu_j] = \psi(\alpha_j) - \psi(\alpha_0) \tag{2.276}$$

where α_0 is given by (2.39) and

$$\psi(a) \equiv \frac{d}{da} \ln \Gamma(a) \tag{2.277}$$

is the *digamma* function.

2.12 (⋆) The uniform distribution for a continuous variable x is defined by

$$\mathrm{U}(x|a, b) = \frac{1}{b-a}, \qquad a \leqslant x \leqslant b. \tag{2.278}$$

Verify that this distribution is normalized, and find expressions for its mean and variance.

2.13 (⋆⋆) Evaluate the Kullback-Leibler divergence (1.113) between two Gaussians $p(\mathbf{x}) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ and $q(\mathbf{x}) = \mathcal{N}(\mathbf{x}|\mathbf{m}, \mathbf{L})$.

2.14 (⋆⋆) **www** This exercise demonstrates that the multivariate distribution with maximum entropy, for a given covariance, is a Gaussian. The entropy of a distribution $p(\mathbf{x})$ is given by

$$\mathrm{H}[\mathbf{x}] = -\int p(\mathbf{x}) \ln p(\mathbf{x}) \, \mathrm{d}\mathbf{x}. \tag{2.279}$$

We wish to maximize $\mathrm{H}[\mathbf{x}]$ over all distributions $p(\mathbf{x})$ subject to the constraints that $p(\mathbf{x})$ be normalized and that it have a specific mean and covariance, so that

$$\int p(\mathbf{x}) \, \mathrm{d}\mathbf{x} = 1 \tag{2.280}$$

$$\int p(\mathbf{x}) \mathbf{x} \, \mathrm{d}\mathbf{x} = \boldsymbol{\mu} \tag{2.281}$$

$$\int p(\mathbf{x})(\mathbf{x} - \boldsymbol{\mu})(\mathbf{x} - \boldsymbol{\mu})^{\mathrm{T}} \, \mathrm{d}\mathbf{x} = \boldsymbol{\Sigma}. \tag{2.282}$$

By performing a variational maximization of (2.279) and using Lagrange multipliers to enforce the constraints (2.280), (2.281), and (2.282), show that the maximum likelihood distribution is given by the Gaussian (2.43).

2.15 (⋆⋆) Show that the entropy of the multivariate Gaussian $\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ is given by

$$\mathrm{H}[\mathbf{x}] = \frac{1}{2} \ln |\boldsymbol{\Sigma}| + \frac{D}{2} (1 + \ln(2\pi)) \tag{2.283}$$

where D is the dimensionality of $\mathbf{x}$.

2.16 (⋆⋆⋆) **www** Consider two random variables x_1 and x_2 having Gaussian distributions with means μ_1, μ_2 and precisions τ_1, τ_2 respectively. Derive an expression for the differential entropy of the variable $x = x_1 + x_2$. To do this, first find the distribution of x by using the relation

$$p(x) = \int_{-\infty}^{\infty} p(x|x_2)p(x_2)\,\mathrm{d}x_2 \tag{2.284}$$

and completing the square in the exponent. Then observe that this represents the convolution of two Gaussian distributions, which itself will be Gaussian, and finally make use of the result (1.110) for the entropy of the univariate Gaussian.

2.17 (⋆) **www** Consider the multivariate Gaussian distribution given by (2.43). By writing the precision matrix (inverse covariance matrix) $\boldsymbol{\Sigma}^{-1}$ as the sum of a symmetric and an anti-symmetric matrix, show that the anti-symmetric term does not appear in the exponent of the Gaussian, and hence that the precision matrix may be taken to be symmetric without loss of generality. Because the inverse of a symmetric matrix is also symmetric (see Exercise 2.22), it follows that the covariance matrix may also be chosen to be symmetric without loss of generality.

2.18 (⋆⋆⋆) Consider a real, symmetric matrix $\boldsymbol{\Sigma}$ whose eigenvalue equation is given by (2.45). By taking the complex conjugate of this equation and subtracting the original equation, and then forming the inner product with eigenvector $\mathbf{u}_i$, show that the eigenvalues λ_i are real. Similarly, use the symmetry property of $\boldsymbol{\Sigma}$ to show that two eigenvectors $\mathbf{u}_i$ and $\mathbf{u}_j$ will be orthogonal provided $\lambda_j \neq \lambda_i$. Finally, show that without loss of generality, the set of eigenvectors can be chosen to be orthonormal, so that they satisfy (2.46), even if some of the eigenvalues are zero.

2.19 (⋆⋆) Show that a real, symmetric matrix $\boldsymbol{\Sigma}$ having the eigenvector equation (2.45) can be expressed as an expansion in the eigenvectors, with coefficients given by the eigenvalues, of the form (2.48). Similarly, show that the inverse matrix $\boldsymbol{\Sigma}^{-1}$ has a representation of the form (2.49).

2.20 (⋆⋆) **www** A positive definite matrix $\boldsymbol{\Sigma}$ can be defined as one for which the quadratic form

$$\mathbf{a}^{\mathrm{T}}\boldsymbol{\Sigma}\mathbf{a} \tag{2.285}$$

is positive for any real value of the vector $\mathbf{a}$. Show that a necessary and sufficient condition for $\boldsymbol{\Sigma}$ to be positive definite is that all of the eigenvalues λ_i of $\boldsymbol{\Sigma}$, defined by (2.45), are positive.

2.21 (⋆) Show that a real, symmetric matrix of size $D \times D$ has $D(D+1)/2$ independent parameters.

2.22 (⋆) **www** Show that the inverse of a symmetric matrix is itself symmetric.

2.23 (⋆⋆) By diagonalizing the coordinate system using the eigenvector expansion (2.45), show that the volume contained within the hyperellipsoid corresponding to a constant

Mahalanobis distance Δ is given by

$$V_D|\boldsymbol{\Sigma}|^{1/2}\Delta^D \tag{2.286}$$

where V_D is the volume of the unit sphere in D dimensions, and the Mahalanobis distance is defined by (2.44).

2.24 ($\star\star$) **WWW** Prove the identity (2.76) by multiplying both sides by the matrix

$$\begin{pmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{pmatrix} \tag{2.287}$$

and making use of the definition (2.77).

2.25 ($\star\star$) In Sections 2.3.1 and 2.3.2, we considered the conditional and marginal distributions for a multivariate Gaussian. More generally, we can consider a partitioning of the components of $\mathbf{x}$ into three groups $\mathbf{x}_a$, $\mathbf{x}_b$, and $\mathbf{x}_c$, with a corresponding partitioning of the mean vector $\boldsymbol{\mu}$ and of the covariance matrix $\boldsymbol{\Sigma}$ in the form

$$\boldsymbol{\mu} = \begin{pmatrix} \boldsymbol{\mu}_a \\ \boldsymbol{\mu}_b \\ \boldsymbol{\mu}_c \end{pmatrix}, \qquad \boldsymbol{\Sigma} = \begin{pmatrix} \boldsymbol{\Sigma}_{aa} & \boldsymbol{\Sigma}_{ab} & \boldsymbol{\Sigma}_{ac} \\ \boldsymbol{\Sigma}_{ba} & \boldsymbol{\Sigma}_{bb} & \boldsymbol{\Sigma}_{bc} \\ \boldsymbol{\Sigma}_{ca} & \boldsymbol{\Sigma}_{cb} & \boldsymbol{\Sigma}_{cc} \end{pmatrix}. \tag{2.288}$$

By making use of the results of Section 2.3, find an expression for the conditional distribution $p(\mathbf{x}_a|\mathbf{x}_b)$ in which $\mathbf{x}_c$ has been marginalized out.

2.26 ($\star\star$) A very useful result from linear algebra is the *Woodbury* matrix inversion formula given by

$$(\mathbf{A}+\mathbf{BCD})^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{B}(\mathbf{C}^{-1}+\mathbf{DA}^{-1}\mathbf{B})^{-1}\mathbf{DA}^{-1}. \tag{2.289}$$

By multiplying both sides by $(\mathbf{A}+\mathbf{BCD})$ prove the correctness of this result.

2.27 ($\star$) Let $\mathbf{x}$ and $\mathbf{z}$ be two independent random vectors, so that $p(\mathbf{x},\mathbf{z}) = p(\mathbf{x})p(\mathbf{z})$. Show that the mean of their sum $\mathbf{y} = \mathbf{x}+\mathbf{z}$ is given by the sum of the means of each of the variable separately. Similarly, show that the covariance matrix of $\mathbf{y}$ is given by the sum of the covariance matrices of $\mathbf{x}$ and $\mathbf{z}$. Confirm that this result agrees with that of Exercise 1.10.

2.28 ($\star\star\star$) **WWW** Consider a joint distribution over the variable

$$\mathbf{z} = \begin{pmatrix} \mathbf{x} \\ \mathbf{y} \end{pmatrix} \tag{2.290}$$

whose mean and covariance are given by (2.108) and (2.105) respectively. By making use of the results (2.92) and (2.93) show that the marginal distribution $p(\mathbf{x})$ is given (2.99). Similarly, by making use of the results (2.81) and (2.82) show that the conditional distribution $p(\mathbf{y}|\mathbf{x})$ is given by (2.100).

2.29 ($\star\star$) Using the partitioned matrix inversion formula (2.76), show that the inverse of the precision matrix (2.104) is given by the covariance matrix (2.105).

2.30 ($\star$) By starting from (2.107) and making use of the result (2.105), verify the result (2.108).

2.31 ($\star\star$) Consider two multidimensional random vectors $\mathbf{x}$ and $\mathbf{z}$ having Gaussian distributions $p(\mathbf{x}) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_{\mathbf{x}}, \boldsymbol{\Sigma}_{\mathbf{x}})$ and $p(\mathbf{z}) = \mathcal{N}(\mathbf{z}|\boldsymbol{\mu}_{\mathbf{z}}, \boldsymbol{\Sigma}_{\mathbf{z}})$ respectively, together with their sum $\mathbf{y} = \mathbf{x}+\mathbf{z}$. Use the results (2.109) and (2.110) to find an expression for the marginal distribution $p(\mathbf{y})$ by considering the linear-Gaussian model comprising the product of the marginal distribution $p(\mathbf{x})$ and the conditional distribution $p(\mathbf{y}|\mathbf{x})$.

2.32 ($\star\star\star$) **www** This exercise and the next provide practice at manipulating the quadratic forms that arise in linear-Gaussian models, as well as giving an independent check of results derived in the main text. Consider a joint distribution $p(\mathbf{x}, \mathbf{y})$ defined by the marginal and conditional distributions given by (2.99) and (2.100). By examining the quadratic form in the exponent of the joint distribution, and using the technique of 'completing the square' discussed in Section 2.3, find expressions for the mean and covariance of the marginal distribution $p(\mathbf{y})$ in which the variable $\mathbf{x}$ has been integrated out. To do this, make use of the Woodbury matrix inversion formula (2.289). Verify that these results agree with (2.109) and (2.110) obtained using the results of Chapter 2.

2.33 ($\star\star\star$) Consider the same joint distribution as in Exercise 2.32, but now use the technique of completing the square to find expressions for the mean and covariance of the conditional distribution $p(\mathbf{x}|\mathbf{y})$. Again, verify that these agree with the corresponding expressions (2.111) and (2.112).

2.34 ($\star\star$) **www** To find the maximum likelihood solution for the covariance matrix of a multivariate Gaussian, we need to maximize the log likelihood function (2.118) with respect to $\boldsymbol{\Sigma}$, noting that the covariance matrix must be symmetric and positive definite. Here we proceed by ignoring these constraints and doing a straightforward maximization. Using the results (C.21), (C.26), and (C.28) from Appendix C, show that the covariance matrix $\boldsymbol{\Sigma}$ that maximizes the log likelihood function (2.118) is given by the sample covariance (2.122). We note that the final result is necessarily symmetric and positive definite (provided the sample covariance is nonsingular).

2.35 ($\star\star$) Use the result (2.59) to prove (2.62). Now, using the results (2.59), and (2.62), show that

$$\mathbb{E}[\mathbf{x}_n \mathbf{x}_m] = \boldsymbol{\mu}\boldsymbol{\mu}^{\mathrm{T}} + I_{nm}\boldsymbol{\Sigma} \tag{2.291}$$

where $\mathbf{x}_n$ denotes a data point sampled from a Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$, and I_{nm} denotes the (n, m) element of the identity matrix. Hence prove the result (2.124).

2.36 ($\star\star$) **www** Using an analogous procedure to that used to obtain (2.126), derive an expression for the sequential estimation of the variance of a univariate Gaussian

distribution, by starting with the maximum likelihood expression

$$\sigma^2_{\mathrm{ML}} = \frac{1}{N}\sum_{n=1}^{N}(x_n - \mu)^2. \tag{2.292}$$

Verify that substituting the expression for a Gaussian distribution into the Robbins-Monro sequential estimation formula (2.135) gives a result of the same form, and hence obtain an expression for the corresponding coefficients a_N.

2.37 ($\star\star$) Using an analogous procedure to that used to obtain (2.126), derive an expression for the sequential estimation of the covariance of a multivariate Gaussian distribution, by starting with the maximum likelihood expression (2.122). Verify that substituting the expression for a Gaussian distribution into the Robbins-Monro sequential estimation formula (2.135) gives a result of the same form, and hence obtain an expression for the corresponding coefficients a_N.

2.38 ($\star$) Use the technique of completing the square for the quadratic form in the exponent to derive the results (2.141) and (2.142).

2.39 ($\star\star$) Starting from the results (2.141) and (2.142) for the posterior distribution of the mean of a Gaussian random variable, dissect out the contributions from the first $N-1$ data points and hence obtain expressions for the sequential update of μ_N and σ^2_N. Now derive the same results starting from the posterior distribution $p(\mu|x_1,\ldots,x_{N-1}) = \mathcal{N}(\mu|\mu_{N-1},\sigma^2_{N-1})$ and multiplying by the likelihood function $p(x_N|\mu) = \mathcal{N}(x_N|\mu,\sigma^2)$ and then completing the square and normalizing to obtain the posterior distribution after N observations.

2.40 ($\star\star$) **www** Consider a D-dimensional Gaussian random variable $\mathbf{x}$ with distribution $\mathcal{N}(\mathbf{x}|\boldsymbol{\mu},\boldsymbol{\Sigma})$ in which the covariance $\boldsymbol{\Sigma}$ is known and for which we wish to infer the mean $\boldsymbol{\mu}$ from a set of observations $\mathbf{X} = \{\mathbf{x}_1,\ldots,\mathbf{x}_N\}$. Given a prior distribution $p(\boldsymbol{\mu}) = \mathcal{N}(\boldsymbol{\mu}|\boldsymbol{\mu}_0,\boldsymbol{\Sigma}_0)$, find the corresponding posterior distribution $p(\boldsymbol{\mu}|\mathbf{X})$.

2.41 ($\star$) Use the definition of the gamma function (1.141) to show that the gamma distribution (2.146) is normalized.

2.42 ($\star\star$) Evaluate the mean, variance, and mode of the gamma distribution (2.146).

2.43 ($\star$) The following distribution

$$p(x|\sigma^2,q) = \frac{q}{2(2\sigma^2)^{1/q}\Gamma(1/q)}\exp\left(-\frac{|x|^q}{2\sigma^2}\right) \tag{2.293}$$

is a generalization of the univariate Gaussian distribution. Show that this distribution is normalized so that

$$\int_{-\infty}^{\infty} p(x|\sigma^2,q)\,\mathrm{d}x = 1 \tag{2.294}$$

and that it reduces to the Gaussian when $q = 2$. Consider a regression model in which the target variable is given by $t = y(\mathbf{x},\mathbf{w}) + \epsilon$ and ϵ is a random noise

variable drawn from the distribution (2.293). Show that the log likelihood function over $\mathbf{w}$ and σ^2, for an observed data set of input vectors $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ and corresponding target variables $\mathbf{t} = (t_1, \ldots, t_N)^{\mathrm{T}}$, is given by

$$\ln p(\mathbf{t}|\mathbf{X}, \mathbf{w}, \sigma^2) = -\frac{1}{2\sigma^2} \sum_{n=1}^{N} |y(\mathbf{x}_n, \mathbf{w}) - t_n|^q - \frac{N}{q} \ln(2\sigma^2) + \text{const} \tag{2.295}$$

where 'const' denotes terms independent of both $\mathbf{w}$ and σ^2. Note that, as a function of $\mathbf{w}$, this is the L_q error function considered in Section 1.5.5.

2.44 ($\star\star$) Consider a univariate Gaussian distribution $\mathcal{N}(x|\mu, \tau^{-1})$ having conjugate Gaussian-gamma prior given by (2.154), and a data set $\mathbf{x} = \{x_1, \ldots, x_N\}$ of i.i.d. observations. Show that the posterior distribution is also a Gaussian-gamma distribution of the same functional form as the prior, and write down expressions for the parameters of this posterior distribution.

2.45 ($\star$) Verify that the Wishart distribution defined by (2.155) is indeed a conjugate prior for the precision matrix of a multivariate Gaussian.

2.46 ($\star$) **www** Verify that evaluating the integral in (2.158) leads to the result (2.159).

2.47 ($\star$) **www** Show that in the limit $\nu \to \infty$, the t-distribution (2.159) becomes a Gaussian. Hint: ignore the normalization coefficient, and simply look at the dependence on x.

2.48 ($\star$) By following analogous steps to those used to derive the univariate Student's t-distribution (2.159), verify the result (2.162) for the multivariate form of the Student's t-distribution, by marginalizing over the variable η in (2.161). Using the definition (2.161), show by exchanging integration variables that the multivariate t-distribution is correctly normalized.

2.49 ($\star\star$) By using the definition (2.161) of the multivariate Student's t-distribution as a convolution of a Gaussian with a gamma distribution, verify the properties (2.164), (2.165), and (2.166) for the multivariate t-distribution defined by (2.162).

2.50 ($\star$) Show that in the limit $\nu \to \infty$, the multivariate Student's t-distribution (2.162) reduces to a Gaussian with mean $\boldsymbol{\mu}$ and precision $\boldsymbol{\Lambda}$.

2.51 ($\star$) **www** The various trigonometric identities used in the discussion of periodic variables in this chapter can be proven easily from the relation

$$\exp(iA) = \cos A + i \sin A \tag{2.296}$$

in which i is the square root of minus one. By considering the identity

$$\exp(iA) \exp(-iA) = 1 \tag{2.297}$$

prove the result (2.177). Similarly, using the identity

$$\cos(A - B) = \Re \exp\{i(A - B)\} \tag{2.298}$$

where $\Re$ denotes the real part, prove (2.178). Finally, by using $\sin(A-B) = \Im \exp\{i(A-B)\}$, where $\Im$ denotes the imaginary part, prove the result (2.183).

2.52 ($\star\star$) For large m, the von Mises distribution (2.179) becomes sharply peaked around the mode θ_0. By defining $\xi = m^{1/2}(\theta - \theta_0)$ and making the Taylor expansion of the cosine function given by

$$\cos\alpha = 1 - \frac{\alpha^2}{2} + O(\alpha^4) \tag{2.299}$$

show that as $m \to \infty$, the von Mises distribution tends to a Gaussian.

2.53 ($\star$) Using the trigonometric identity (2.183), show that solution of (2.182) for θ_0 is given by (2.184).

2.54 ($\star$) By computing first and second derivatives of the von Mises distribution (2.179), and using $I_0(m) > 0$ for $m > 0$, show that the maximum of the distribution occurs when $\theta = \theta_0$ and that the minimum occurs when $\theta = \theta_0 + \pi \pmod{2\pi}$.

2.55 ($\star$) By making use of the result (2.168), together with (2.184) and the trigonometric identity (2.178), show that the maximum likelihood solution m_{ML} for the concentration of the von Mises distribution satisfies $A(m_{\mathrm{ML}}) = \overline{r}$ where $\overline{r}$ is the radius of the mean of the observations viewed as unit vectors in the two-dimensional Euclidean plane, as illustrated in Figure 2.17.

2.56 ($\star\star$) **www** Express the beta distribution (2.13), the gamma distribution (2.146), and the von Mises distribution (2.179) as members of the exponential family (2.194) and thereby identify their natural parameters.

2.57 ($\star$) Verify that the multivariate Gaussian distribution can be cast in exponential family form (2.194) and derive expressions for $\boldsymbol{\eta}$, $\mathbf{u}(\mathbf{x})$, $h(\mathbf{x})$ and $g(\boldsymbol{\eta})$ analogous to (2.220)–(2.223).

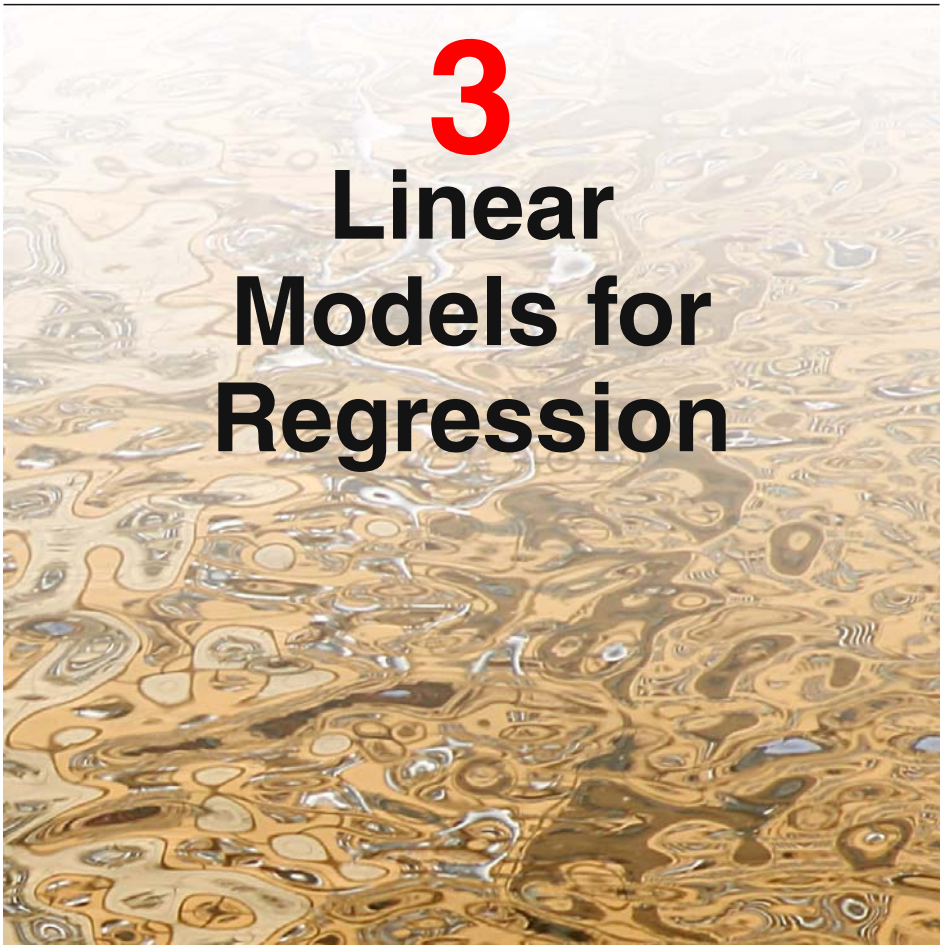
2.58 ($\star$) The result (2.226) showed that the negative gradient of $\ln g(\boldsymbol{\eta})$ for the exponential family is given by the expectation of $\mathbf{u}(\mathbf{x})$. By taking the second derivatives of (2.195), show that

$$-\nabla\nabla \ln g(\boldsymbol{\eta}) = \mathbb{E}[\mathbf{u}(\mathbf{x})\mathbf{u}(\mathbf{x})^{\mathrm{T}}] - \mathbb{E}[\mathbf{u}(\mathbf{x})]\mathbb{E}[\mathbf{u}(\mathbf{x})^{\mathrm{T}}] = \mathrm{cov}[\mathbf{u}(\mathbf{x})]. \tag{2.300}$$

2.59 ($\star$) By changing variables using $y = x/\sigma$, show that the density (2.236) will be correctly normalized, provided $f(x)$ is correctly normalized.

2.60 ($\star\star$) **www** Consider a histogram-like density model in which the space $\mathbf{x}$ is divided into fixed regions for which the density $p(\mathbf{x})$ takes the constant value h_i over the i^{th} region, and that the volume of region i is denoted Δ_i. Suppose we have a set of N observations of $\mathbf{x}$ such that n_i of these observations fall in region i. Using a Lagrange multiplier to enforce the normalization constraint on the density, derive an expression for the maximum likelihood estimator for the $\{h_i\}$.

2.61 ($\star$) Show that the K-nearest-neighbour density model defines an improper distribution whose integral over all space is divergent.



3 Linear Models for Regression

The focus so far in this book has been on unsupervised learning, including topics such as density estimation and data clustering. We turn now to a discussion of supervised learning, starting with regression. The goal of regression is to predict the value of one or more continuous *target* variables t given the value of a D-dimensional vector $\mathbf{x}$ of *input* variables. We have already encountered an example of a regression problem when we considered polynomial curve fitting in Chapter 1. The polynomial is a specific example of a broad class of functions called linear regression models, which share the property of being linear functions of the adjustable parameters, and which will form the focus of this chapter. The simplest form of linear regression models are also linear functions of the input variables. However, we can obtain a much more useful class of functions by taking linear combinations of a fixed set of nonlinear functions of the input variables, known as *basis functions*. Such models are linear functions of the parameters, which gives them simple analytical properties, and yet can be nonlinear with respect to the input variables.

Given a training data set comprising N observations $\{\mathbf{x}_n\}$, where $n = 1, \ldots, N$, together with corresponding target values $\{t_n\}$, the goal is to predict the value of t for a new value of $\mathbf{x}$. In the simplest approach, this can be done by directly constructing an appropriate function $y(\mathbf{x})$ whose values for new inputs $\mathbf{x}$ constitute the predictions for the corresponding values of t. More generally, from a probabilistic perspective, we aim to model the predictive distribution $p(t|\mathbf{x})$ because this expresses our uncertainty about the value of t for each value of $\mathbf{x}$. From this conditional distribution we can make predictions of t, for any new value of $\mathbf{x}$, in such a way as to minimize the expected value of a suitably chosen loss function. As discussed in Section 1.5.5, a common choice of loss function for real-valued variables is the squared loss, for which the optimal solution is given by the conditional expectation of t.

Although linear models have significant limitations as practical techniques for pattern recognition, particularly for problems involving input spaces of high dimensionality, they have nice analytical properties and form the foundation for more sophisticated models to be discussed in later chapters.

3.1. Linear Basis Function Models

The simplest linear model for regression is one that involves a linear combination of the input variables

$$y(\mathbf{x}, \mathbf{w}) = w_0 + w_1 x_1 + \ldots + w_D x_D \tag{3.1}$$

where $\mathbf{x} = (x_1, \ldots, x_D)^{\mathrm{T}}$. This is often simply known as *linear regression*. The key property of this model is that it is a linear function of the parameters $w_0, \ldots, w_D$. It is also, however, a linear function of the input variables x_i, and this imposes significant limitations on the model. We therefore extend the class of models by considering linear combinations of fixed nonlinear functions of the input variables, of the form

$$y(\mathbf{x}, \mathbf{w}) = w_0 + \sum_{j=1}^{M-1} w_j \phi_j(\mathbf{x}) \tag{3.2}$$

where $\phi_j(\mathbf{x})$ are known as *basis functions*. By denoting the maximum value of the index j by $M - 1$, the total number of parameters in this model will be M.

The parameter w_0 allows for any fixed offset in the data and is sometimes called a *bias* parameter (not to be confused with 'bias' in a statistical sense). It is often convenient to define an additional dummy 'basis function' $\phi_0(\mathbf{x}) = 1$ so that

$$y(\mathbf{x}, \mathbf{w}) = \sum_{j=0}^{M-1} w_j \phi_j(\mathbf{x}) = \mathbf{w}^{\mathrm{T}} \boldsymbol{\phi}(\mathbf{x}) \tag{3.3}$$

where $\mathbf{w} = (w_0, \ldots, w_{M-1})^{\mathrm{T}}$ and $\boldsymbol{\phi} = (\phi_0, \ldots, \phi_{M-1})^{\mathrm{T}}$. In many practical applications of pattern recognition, we will apply some form of fixed pre-processing,

or feature extraction, to the original data variables. If the original variables comprise the vector $\mathbf{x}$, then the features can be expressed in terms of the basis functions $\{\phi_j(\mathbf{x})\}$.

By using nonlinear basis functions, we allow the function $y(\mathbf{x}, \mathbf{w})$ to be a nonlinear function of the input vector $\mathbf{x}$. Functions of the form (3.2) are called linear models, however, because this function is linear in $\mathbf{w}$. It is this linearity in the parameters that will greatly simplify the analysis of this class of models. However, it also leads to some significant limitations, as we discuss in Section 3.6.

The example of polynomial regression considered in Chapter 1 is a particular example of this model in which there is a single input variable x, and the basis functions take the form of powers of x so that $\phi_j(x) = x^j$. One limitation of polynomial basis functions is that they are global functions of the input variable, so that changes in one region of input space affect all other regions. This can be resolved by dividing the input space up into regions and fit a different polynomial in each region, leading to *spline functions* (Hastie *et al.*, 2001).

There are many other possible choices for the basis functions, for example

$$\phi_j(x) = \exp\left\{-\frac{(x-\mu_j)^2}{2s^2}\right\} \tag{3.4}$$

where the μ_j govern the locations of the basis functions in input space, and the parameter s governs their spatial scale. These are usually referred to as 'Gaussian' basis functions, although it should be noted that they are not required to have a probabilistic interpretation, and in particular the normalization coefficient is unimportant because these basis functions will be multiplied by adaptive parameters w_j.

Another possibility is the sigmoidal basis function of the form

$$\phi_j(x) = \sigma\left(\frac{x-\mu_j}{s}\right) \tag{3.5}$$

where $\sigma(a)$ is the logistic sigmoid function defined by

$$\sigma(a) = \frac{1}{1+\exp(-a)}. \tag{3.6}$$

Equivalently, we can use the 'tanh' function because this is related to the logistic sigmoid by $\tanh(a) = 2\sigma(a) - 1$, and so a general linear combination of logistic sigmoid functions is equivalent to a general linear combination of 'tanh' functions. These various choices of basis function are illustrated in Figure 3.1.

Yet another possible choice of basis function is the Fourier basis, which leads to an expansion in sinusoidal functions. Each basis function represents a specific frequency and has infinite spatial extent. By contrast, basis functions that are localized to finite regions of input space necessarily comprise a spectrum of different spatial frequencies. In many signal processing applications, it is of interest to consider basis functions that are localized in both space and frequency, leading to a class of functions known as *wavelets*. These are also defined to be mutually orthogonal, to simplify their application. Wavelets are most applicable when the input values live

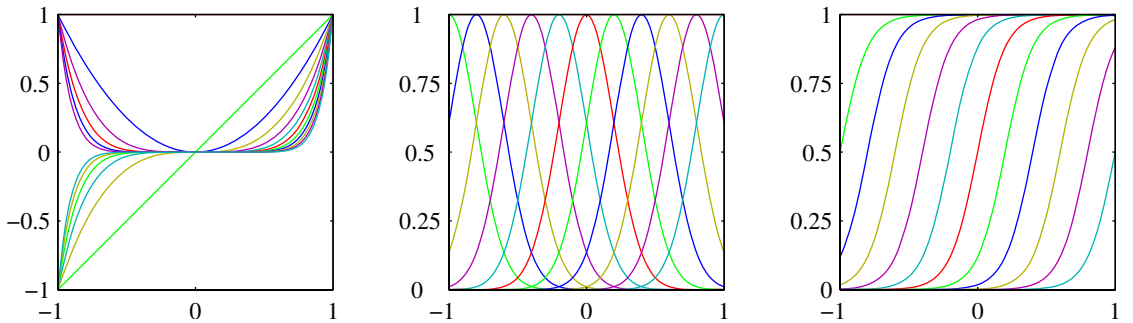


Figure 3.1 Examples of basis functions, showing polynomials on the left, Gaussians of the form (3.4) in the centre, and sigmoidal of the form (3.5) on the right.

on a regular lattice, such as the successive time points in a temporal sequence, or the pixels in an image. Useful texts on wavelets include Ogden (1997), Mallat (1999), and Vidakovic (1999).

Most of the discussion in this chapter, however, is independent of the particular choice of basis function set, and so for most of our discussion we shall not specify the particular form of the basis functions, except for the purposes of numerical illustration. Indeed, much of our discussion will be equally applicable to the situation in which the vector $\boldsymbol{\phi}(\mathbf{x})$ of basis functions is simply the identity $\boldsymbol{\phi}(\mathbf{x}) = \mathbf{x}$. Furthermore, in order to keep the notation simple, we shall focus on the case of a single target variable t. However, in Section 3.1.5, we consider briefly the modifications needed to deal with multiple target variables.

3.1.1 Maximum likelihood and least squares

In Chapter 1, we fitted polynomial functions to data sets by minimizing a sum-of-squares error function. We also showed that this error function could be motivated as the maximum likelihood solution under an assumed Gaussian noise model. Let us return to this discussion and consider the least squares approach, and its relation to maximum likelihood, in more detail.

As before, we assume that the target variable t is given by a deterministic function $y(\mathbf{x}, \mathbf{w})$ with additive Gaussian noise so that

$$t = y(\mathbf{x}, \mathbf{w}) + \epsilon \tag{3.7}$$

where ϵ is a zero mean Gaussian random variable with precision (inverse variance) β. Thus we can write

$$p(t|\mathbf{x}, \mathbf{w}, \beta) = \mathcal{N}(t|y(\mathbf{x}, \mathbf{w}), \beta^{-1}). \tag{3.8}$$

Section 1.5.5

Recall that, if we assume a squared loss function, then the optimal prediction, for a new value of $\mathbf{x}$, will be given by the conditional mean of the target variable. In the case of a Gaussian conditional distribution of the form (3.8), the conditional mean

will be simply

$$\mathbb{E}[t|\mathbf{x}] = \int t p(t|\mathbf{x})\, \mathrm{d}t = y(\mathbf{x}, \mathbf{w}). \tag{3.9}$$

Note that the Gaussian noise assumption implies that the conditional distribution of t given $\mathbf{x}$ is unimodal, which may be inappropriate for some applications. An extension to mixtures of conditional Gaussian distributions, which permit multimodal conditional distributions, will be discussed in Section 14.5.1.

Now consider a data set of inputs $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ with corresponding target values $t_1, \ldots, t_N$. We group the target variables $\{t_n\}$ into a column vector that we denote by $\mathbf{t}$ where the typeface is chosen to distinguish it from a single observation of a multivariate target, which would be denoted $\mathbf{t}$. Making the assumption that these data points are drawn independently from the distribution (3.8), we obtain the following expression for the likelihood function, which is a function of the adjustable parameters $\mathbf{w}$ and β, in the form

$$p(\mathbf{t}|\mathbf{X}, \mathbf{w}, \beta) = \prod_{n=1}^{N} \mathcal{N}(t_n|\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n), \beta^{-1}) \tag{3.10}$$

where we have used (3.3). Note that in supervised learning problems such as regression (and classification), we are not seeking to model the distribution of the input variables. Thus $\mathbf{x}$ will always appear in the set of conditioning variables, and so from now on we will drop the explicit $\mathbf{x}$ from expressions such as $p(\mathbf{t}|\mathbf{x}, \mathbf{w}, \beta)$ in order to keep the notation uncluttered. Taking the logarithm of the likelihood function, and making use of the standard form (1.46) for the univariate Gaussian, we have

$$\begin{aligned} \ln p(\mathbf{t}|\mathbf{w}, \beta) &= \sum_{n=1}^{N} \ln \mathcal{N}(t_n|\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n), \beta^{-1}) \\ &= \frac{N}{2} \ln \beta - \frac{N}{2} \ln(2\pi) - \beta E_D(\mathbf{w}) \end{aligned} \tag{3.11}$$

where the sum-of-squares error function is defined by

$$E_D(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^{N} \{t_n - \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n)\}^2. \tag{3.12}$$

Having written down the likelihood function, we can use maximum likelihood to determine $\mathbf{w}$ and β. Consider first the maximization with respect to $\mathbf{w}$. As observed already in Section 1.2.5, we see that maximization of the likelihood function under a conditional Gaussian noise distribution for a linear model is equivalent to minimizing a sum-of-squares error function given by $E_D(\mathbf{w})$. The gradient of the log likelihood function (3.11) takes the form

$$\nabla \ln p(\mathbf{t}|\mathbf{w}, \beta) = \sum_{n=1}^{N} \left\{t_n - \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n)\right\} \boldsymbol{\phi}(\mathbf{x}_n)^{\mathrm{T}}. \tag{3.13}$$

Setting this gradient to zero gives

$$0 = \sum_{n=1}^{N} t_n \boldsymbol{\phi}(\mathbf{x}_n)^{\mathrm{T}} - \mathbf{w}^{\mathrm{T}} \left(\sum_{n=1}^{N} \boldsymbol{\phi}(\mathbf{x}_n) \boldsymbol{\phi}(\mathbf{x}_n)^{\mathrm{T}} \right). \tag{3.14}$$

Solving for $\mathbf{w}$ we obtain

$$\mathbf{w}_{\mathrm{ML}} = \left(\boldsymbol{\Phi}^{\mathrm{T}} \boldsymbol{\Phi} \right)^{-1} \boldsymbol{\Phi}^{\mathrm{T}} \mathbf{t} \tag{3.15}$$

which are known as the *normal equations* for the least squares problem. Here $\boldsymbol{\Phi}$ is an $N \times M$ matrix, called the *design matrix*, whose elements are given by $\Phi_{nj} = \phi_j(\mathbf{x}_n)$, so that

$$\boldsymbol{\Phi} = \begin{pmatrix} \phi_0(\mathbf{x}_1) & \phi_1(\mathbf{x}_1) & \cdots & \phi_{M-1}(\mathbf{x}_1) \\ \phi_0(\mathbf{x}_2) & \phi_1(\mathbf{x}_2) & \cdots & \phi_{M-1}(\mathbf{x}_2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_0(\mathbf{x}_N) & \phi_1(\mathbf{x}_N) & \cdots & \phi_{M-1}(\mathbf{x}_N) \end{pmatrix}. \tag{3.16}$$

The quantity

$$\boldsymbol{\Phi}^{\dagger} \equiv \left(\boldsymbol{\Phi}^{\mathrm{T}} \boldsymbol{\Phi} \right)^{-1} \boldsymbol{\Phi}^{\mathrm{T}} \tag{3.17}$$

is known as the *Moore-Penrose pseudo-inverse* of the matrix $\boldsymbol{\Phi}$ (Rao and Mitra, 1971; Golub and Van Loan, 1996). It can be regarded as a generalization of the notion of matrix inverse to nonsquare matrices. Indeed, if $\boldsymbol{\Phi}$ is square and invertible, then using the property $(\mathbf{AB})^{-1} = \mathbf{B}^{-1}\mathbf{A}^{-1}$ we see that $\boldsymbol{\Phi}^{\dagger} \equiv \boldsymbol{\Phi}^{-1}$.

At this point, we can gain some insight into the role of the bias parameter w_0. If we make the bias parameter explicit, then the error function (3.12) becomes

$$E_D(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^{N} \{t_n - w_0 - \sum_{j=1}^{M-1} w_j \phi_j(\mathbf{x}_n)\}^2. \tag{3.18}$$

Setting the derivative with respect to w_0 equal to zero, and solving for w_0, we obtain

$$w_0 = \bar{t} - \sum_{j=1}^{M-1} w_j \overline{\phi_j} \tag{3.19}$$

where we have defined

$$\bar{t} = \frac{1}{N} \sum_{n=1}^{N} t_n, \qquad \overline{\phi_j} = \frac{1}{N} \sum_{n=1}^{N} \phi_j(\mathbf{x}_n). \tag{3.20}$$

Thus the bias w_0 compensates for the difference between the averages (over the training set) of the target values and the weighted sum of the averages of the basis function values.

We can also maximize the log likelihood function (3.11) with respect to the noise precision parameter β, giving

$$\frac{1}{\beta_{\mathrm{ML}}} = \frac{1}{N} \sum_{n=1}^{N} \{t_n - \mathbf{w}_{\mathrm{ML}}^{\mathrm{T}} \boldsymbol{\phi}(\mathbf{x}_n)\}^2 \tag{3.21}$$

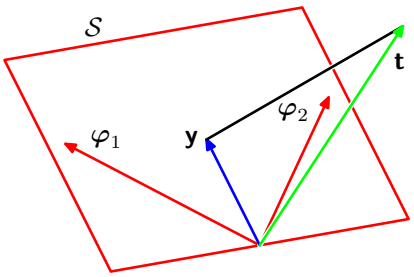


Figure 3.2 Geometrical interpretation of the least-squares solution, in an N-dimensional space whose axes are the values of $t_1, \ldots, t_N$. The least-squares regression function is obtained by finding the orthogonal projection of the data vector $\mathbf{t}$ onto the subspace spanned by the basis functions $\phi_j(\mathbf{x})$ in which each basis function is viewed as a vector $\boldsymbol{\varphi}_j$ of length N with elements $\phi_j(\mathbf{x}_n)$.

and so we see that the inverse of the noise precision is given by the residual variance of the target values around the regression function.

3.1.2 Geometry of least squares

At this point, it is instructive to consider the geometrical interpretation of the least-squares solution. To do this we consider an N-dimensional space whose axes are given by the t_n, so that $\mathbf{t} = (t_1, \ldots, t_N)^{\mathrm{T}}$ is a vector in this space. Each basis function $\phi_j(\mathbf{x}_n)$, evaluated at the N data points, can also be represented as a vector in the same space, denoted by $\boldsymbol{\varphi}_j$, as illustrated in Figure 3.2. Note that $\boldsymbol{\varphi}_j$ corresponds to the j^{th} column of $\boldsymbol{\Phi}$, whereas $\boldsymbol{\phi}(\mathbf{x}_n)$ corresponds to the n^{th} row of $\boldsymbol{\Phi}$. If the number M of basis functions is smaller than the number N of data points, then the M vectors $\phi_j(\mathbf{x}_n)$ will span a linear subspace $\mathcal{S}$ of dimensionality M. We define $\mathbf{y}$ to be an N-dimensional vector whose n^{th} element is given by $y(\mathbf{x}_n, \mathbf{w})$, where $n = 1, \ldots, N$. Because $\mathbf{y}$ is an arbitrary linear combination of the vectors $\boldsymbol{\varphi}_j$, it can live anywhere in the M-dimensional subspace. The sum-of-squares error (3.12) is then equal (up to a factor of $1/2$) to the squared Euclidean distance between $\mathbf{y}$ and $\mathbf{t}$. Thus the least-squares solution for $\mathbf{w}$ corresponds to that choice of $\mathbf{y}$ that lies in subspace $\mathcal{S}$ and that is closest to $\mathbf{t}$. Intuitively, from Figure 3.2, we anticipate that this solution corresponds to the orthogonal projection of $\mathbf{t}$ onto the subspace $\mathcal{S}$. This is indeed the case, as can easily be verified by noting that the solution for $\mathbf{y}$ is given by $\boldsymbol{\Phi}\mathbf{w}_{\mathrm{ML}}$, and then confirming that this takes the form of an orthogonal projection.

Exercise 3.2

In practice, a direct solution of the normal equations can lead to numerical difficulties when $\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}$ is close to singular. In particular, when two or more of the basis vectors $\boldsymbol{\varphi}_j$ are co-linear, or nearly so, the resulting parameter values can have large magnitudes. Such near degeneracies will not be uncommon when dealing with real data sets. The resulting numerical difficulties can be addressed using the technique of *singular value decomposition*, or *SVD* (Press *et al.*, 1992; Bishop and Nabney, 2008). Note that the addition of a regularization term ensures that the matrix is non-singular, even in the presence of degeneracies.

3.1.3 Sequential learning

Batch techniques, such as the maximum likelihood solution (3.15), which involve processing the entire training set in one go, can be computationally costly for large data sets. As we have discussed in Chapter 1, if the data set is sufficiently large, it may be worthwhile to use *sequential* algorithms, also known as *on-line* algorithms,

in which the data points are considered one at a time, and the model parameters updated after each such presentation. Sequential learning is also appropriate for real-time applications in which the data observations are arriving in a continuous stream, and predictions must be made before all of the data points are seen.

We can obtain a sequential learning algorithm by applying the technique of *stochastic gradient descent*, also known as *sequential gradient descent*, as follows. If the error function comprises a sum over data points $E = \sum_n E_n$, then after presentation of pattern n, the stochastic gradient descent algorithm updates the parameter vector $\mathbf{w}$ using

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} - \eta \nabla E_n \tag{3.22}$$

where τ denotes the iteration number, and η is a learning rate parameter. We shall discuss the choice of value for η shortly. The value of $\mathbf{w}$ is initialized to some starting vector $\mathbf{w}^{(0)}$. For the case of the sum-of-squares error function (3.12), this gives

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} + \eta(t_n - \mathbf{w}^{(\tau)\mathrm{T}}\boldsymbol{\phi}_n)\boldsymbol{\phi}_n \tag{3.23}$$

where $\boldsymbol{\phi}_n = \boldsymbol{\phi}(\mathbf{x}_n)$. This is known as *least-mean-squares* or the *LMS algorithm*. The value of η needs to be chosen with care to ensure that the algorithm converges (Bishop and Nabney, 2008).

3.1.4 Regularized least squares

In Section 1.1, we introduced the idea of adding a regularization term to an error function in order to control over-fitting, so that the total error function to be minimized takes the form

$$E_D(\mathbf{w}) + \lambda E_W(\mathbf{w}) \tag{3.24}$$

where λ is the regularization coefficient that controls the relative importance of the data-dependent error $E_D(\mathbf{w})$ and the regularization term $E_W(\mathbf{w})$. One of the simplest forms of regularizer is given by the sum-of-squares of the weight vector elements

$$E_W(\mathbf{w}) = \frac{1}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w}. \tag{3.25}$$

If we also consider the sum-of-squares error function given by

$$E(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{t_n - \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n)\}^2 \tag{3.26}$$

then the total error function becomes

$$\frac{1}{2}\sum_{n=1}^{N}\{t_n - \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n)\}^2 + \frac{\lambda}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w}. \tag{3.27}$$

This particular choice of regularizer is known in the machine learning literature as *weight decay* because in sequential learning algorithms, it encourages weight values to decay towards zero, unless supported by the data. In statistics, it provides an example of a *parameter shrinkage* method because it shrinks parameter values towards

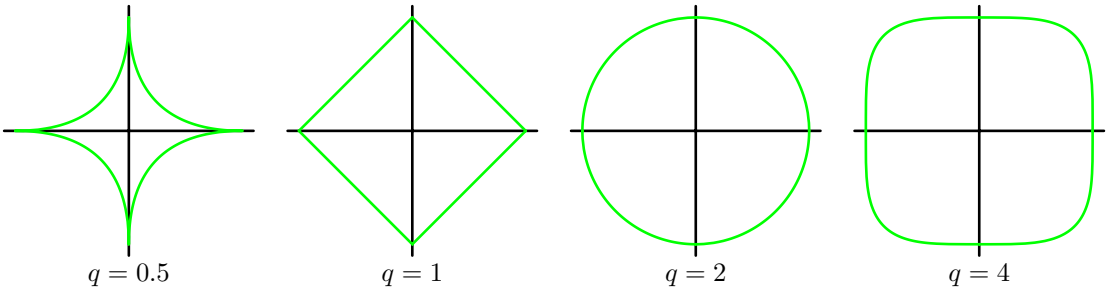


Figure 3.3 Contours of the regularization term in (3.29) for various values of the parameter q.

zero. It has the advantage that the error function remains a quadratic function of $\mathbf{w}$, and so its exact minimizer can be found in closed form. Specifically, setting the gradient of (3.27) with respect to $\mathbf{w}$ to zero, and solving for $\mathbf{w}$ as before, we obtain

$$\mathbf{w} = \left(\lambda \mathbf{I} + \boldsymbol{\Phi}^{\mathrm{T}} \boldsymbol{\Phi}\right)^{-1} \boldsymbol{\Phi}^{\mathrm{T}} \mathbf{t}. \tag{3.28}$$

This represents a simple extension of the least-squares solution (3.15).

A more general regularizer is sometimes used, for which the regularized error takes the form

$$\frac{1}{2} \sum_{n=1}^{N} \{t_n - \mathbf{w}^{\mathrm{T}} \boldsymbol{\phi}(\mathbf{x}_n)\}^2 + \frac{\lambda}{2} \sum_{j=1}^{M} |w_j|^q \tag{3.29}$$

where $q = 2$ corresponds to the quadratic regularizer (3.27). Figure 3.3 shows contours of the regularization function for different values of q.

The case of $q = 1$ is know as the *lasso* in the statistics literature (Tibshirani, 1996). It has the property that if λ is sufficiently large, some of the coefficients w_j are driven to zero, leading to a *sparse* model in which the corresponding basis functions play no role. To see this, we first note that minimizing (3.29) is equivalent to minimizing the unregularized sum-of-squares error (3.12) subject to the constraint

Exercise 3.5

$$\sum_{j=1}^{M} |w_j|^q \leqslant \eta \tag{3.30}$$

for an appropriate value of the parameter η, where the two approaches can be related using Lagrange multipliers. The origin of the sparsity can be seen from Figure 3.4, which shows that the minimum of the error function, subject to the constraint (3.30). As λ is increased, so an increasing number of parameters are driven to zero.

Appendix E

Regularization allows complex models to be trained on data sets of limited size without severe over-fitting, essentially by limiting the effective model complexity. However, the problem of determining the optimal model complexity is then shifted from one of finding the appropriate number of basis functions to one of determining a suitable value of the regularization coefficient λ. We shall return to the issue of model complexity later in this chapter.

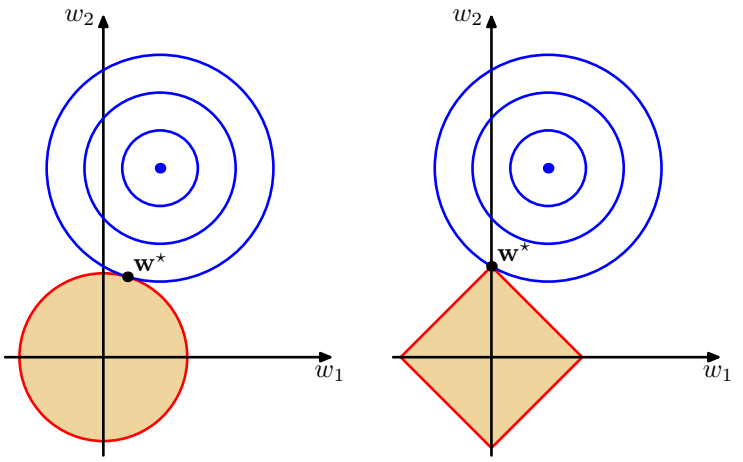


Figure 3.4 Plot of the contours of the unregularized error function (blue) along with the constraint region (3.30) for the quadratic regularizer $q = 2$ on the left and the lasso regularizer $q = 1$ on the right, in which the optimum value for the parameter vector $\mathbf{w}$ is denoted by $\mathbf{w}^\star$. The lasso gives a sparse solution in which $w_1^\star = 0$.

For the remainder of this chapter we shall focus on the quadratic regularizer (3.27) both for its practical importance and its analytical tractability.

3.1.5 Multiple outputs

So far, we have considered the case of a single target variable t. In some applications, we may wish to predict $K > 1$ target variables, which we denote collectively by the target vector $\mathbf{t}$. This could be done by introducing a different set of basis functions for each component of $\mathbf{t}$, leading to multiple, independent regression problems. However, a more interesting, and more common, approach is to use the same set of basis functions to model all of the components of the target vector so that

$$\mathbf{y}(\mathbf{x}, \mathbf{w}) = \mathbf{W}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}) \tag{3.31}$$

where $\mathbf{y}$ is a K-dimensional column vector, $\mathbf{W}$ is an $M \times K$ matrix of parameters, and $\boldsymbol{\phi}(\mathbf{x})$ is an M-dimensional column vector with elements $\phi_j(\mathbf{x})$, with $\phi_0(\mathbf{x}) = 1$ as before. Suppose we take the conditional distribution of the target vector to be an isotropic Gaussian of the form

$$p(\mathbf{t}|\mathbf{x}, \mathbf{W}, \beta) = \mathcal{N}(\mathbf{t}|\mathbf{W}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}), \beta^{-1}\mathbf{I}). \tag{3.32}$$

If we have a set of observations $\mathbf{t}_1, \ldots, \mathbf{t}_N$, we can combine these into a matrix $\mathbf{T}$ of size $N \times K$ such that the n^{th} row is given by $\mathbf{t}_n^{\mathrm{T}}$. Similarly, we can combine the input vectors $\mathbf{x}_1, \ldots, \mathbf{x}_N$ into a matrix $\mathbf{X}$. The log likelihood function is then given by

$$\begin{aligned} \ln p(\mathbf{T}|\mathbf{X}, \mathbf{W}, \beta) &= \sum_{n=1}^{N} \ln \mathcal{N}(\mathbf{t}_n|\mathbf{W}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n), \beta^{-1}\mathbf{I}) \\ &= \frac{NK}{2}\ln\left(\frac{\beta}{2\pi}\right) - \frac{\beta}{2}\sum_{n=1}^{N}\left\|\mathbf{t}_n - \mathbf{W}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n)\right\|^2. \end{aligned} \tag{3.33}$$

As before, we can maximize this function with respect to $\mathbf{W}$, giving

$$\mathbf{W}_{\mathrm{ML}} = \left(\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}\right)^{-1}\boldsymbol{\Phi}^{\mathrm{T}}\mathbf{T}. \tag{3.34}$$

If we examine this result for each target variable t_k, we have

$$\mathbf{w}_k = \left(\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}\right)^{-1}\boldsymbol{\Phi}^{\mathrm{T}}\mathbf{t}_k = \boldsymbol{\Phi}^{\dagger}\mathbf{t}_k \tag{3.35}$$

where $\mathbf{t}_k$ is an N-dimensional column vector with components t_{nk} for $n = 1, \ldots N$. Thus the solution to the regression problem decouples between the different target variables, and we need only compute a single pseudo-inverse matrix $\boldsymbol{\Phi}^{\dagger}$, which is shared by all of the vectors $\mathbf{w}_k$.

Exercise 3.6

The extension to general Gaussian noise distributions having arbitrary covariance matrices is straightforward. Again, this leads to a decoupling into K independent regression problems. This result is unsurprising because the parameters $\mathbf{W}$ define only the mean of the Gaussian noise distribution, and we know from Section 2.3.4 that the maximum likelihood solution for the mean of a multivariate Gaussian is independent of the covariance. From now on, we shall therefore consider a single target variable t for simplicity.

3.2. The Bias-Variance Decomposition

So far in our discussion of linear models for regression, we have assumed that the form and number of basis functions are both fixed. As we have seen in Chapter 1, the use of maximum likelihood, or equivalently least squares, can lead to severe over-fitting if complex models are trained using data sets of limited size. However, limiting the number of basis functions in order to avoid over-fitting has the side effect of limiting the flexibility of the model to capture interesting and important trends in the data. Although the introduction of regularization terms can control over-fitting for models with many parameters, this raises the question of how to determine a suitable value for the regularization coefficient λ. Seeking the solution that minimizes the regularized error function with respect to both the weight vector $\mathbf{w}$ and the regularization coefficient λ is clearly not the right approach since this leads to the unregularized solution with $\lambda = 0$.

As we have seen in earlier chapters, the phenomenon of over-fitting is really an unfortunate property of maximum likelihood and does not arise when we marginalize over parameters in a Bayesian setting. In this chapter, we shall consider the Bayesian view of model complexity in some depth. Before doing so, however, it is instructive to consider a frequentist viewpoint of the model complexity issue, known as the *bias-variance* trade-off. Although we shall introduce this concept in the context of linear basis function models, where it is easy to illustrate the ideas using simple examples, the discussion has more general applicability.

In Section 1.5.5, when we discussed decision theory for regression problems, we considered various loss functions each of which leads to a corresponding optimal prediction once we are given the conditional distribution $p(t|\mathbf{x})$. A popular choice is

the squared loss function, for which the optimal prediction is given by the conditional expectation, which we denote by $h(\mathbf{x})$ and which is given by

$$h(\mathbf{x}) = \mathbb{E}[t|\mathbf{x}] = \int t p(t|\mathbf{x})\,\mathrm{d}t. \tag{3.36}$$

At this point, it is worth distinguishing between the squared loss function arising from decision theory and the sum-of-squares error function that arose in the maximum likelihood estimation of model parameters. We might use more sophisticated techniques than least squares, for example regularization or a fully Bayesian approach, to determine the conditional distribution $p(t|\mathbf{x})$. These can all be combined with the squared loss function for the purpose of making predictions.

We showed in Section 1.5.5 that the expected squared loss can be written in the form

$$\mathbb{E}[L] = \int \{y(\mathbf{x}) - h(\mathbf{x})\}^2 p(\mathbf{x})\,\mathrm{d}\mathbf{x} + \int \{h(\mathbf{x}) - t\}^2 p(\mathbf{x}, t)\,\mathrm{d}\mathbf{x}\,\mathrm{d}t. \tag{3.37}$$

Recall that the second term, which is independent of $y(\mathbf{x})$, arises from the intrinsic noise on the data and represents the minimum achievable value of the expected loss. The first term depends on our choice for the function $y(\mathbf{x})$, and we will seek a solution for $y(\mathbf{x})$ which makes this term a minimum. Because it is nonnegative, the smallest that we can hope to make this term is zero. If we had an unlimited supply of data (and unlimited computational resources), we could in principle find the regression function $h(\mathbf{x})$ to any desired degree of accuracy, and this would represent the optimal choice for $y(\mathbf{x})$. However, in practice we have a data set $\mathcal{D}$ containing only a finite number N of data points, and consequently we do not know the regression function $h(\mathbf{x})$ exactly.

If we model the $h(\mathbf{x})$ using a parametric function $y(\mathbf{x}, \mathbf{w})$ governed by a parameter vector $\mathbf{w}$, then from a Bayesian perspective the uncertainty in our model is expressed through a posterior distribution over $\mathbf{w}$. A frequentist treatment, however, involves making a point estimate of $\mathbf{w}$ based on the data set $\mathcal{D}$, and tries instead to interpret the uncertainty of this estimate through the following thought experiment. Suppose we had a large number of data sets each of size N and each drawn independently from the distribution $p(t, \mathbf{x})$. For any given data set $\mathcal{D}$, we can run our learning algorithm and obtain a prediction function $y(\mathbf{x}; \mathcal{D})$. Different data sets from the ensemble will give different functions and consequently different values of the squared loss. The performance of a particular learning algorithm is then assessed by taking the average over this ensemble of data sets.

Consider the integrand of the first term in (3.37), which for a particular data set $\mathcal{D}$ takes the form

$$\{y(\mathbf{x}; \mathcal{D}) - h(\mathbf{x})\}^2. \tag{3.38}$$

Because this quantity will be dependent on the particular data set $\mathcal{D}$, we take its average over the ensemble of data sets. If we add and subtract the quantity $\mathbb{E}_{\mathcal{D}}[y(\mathbf{x}; \mathcal{D})]$

inside the braces, and then expand, we obtain

$$\begin{aligned}&\{y(\mathbf{x};\mathcal{D}) - \mathbb{E}_{\mathcal{D}}[y(\mathbf{x};\mathcal{D})] + \mathbb{E}_{\mathcal{D}}[y(\mathbf{x};\mathcal{D})] - h(\mathbf{x})\}^2 \\ &\quad = \{y(\mathbf{x};\mathcal{D}) - \mathbb{E}_{\mathcal{D}}[y(\mathbf{x};\mathcal{D})]\}^2 + \{\mathbb{E}_{\mathcal{D}}[y(\mathbf{x};\mathcal{D})] - h(\mathbf{x})\}^2 \\ &\qquad +2\{y(\mathbf{x};\mathcal{D}) - \mathbb{E}_{\mathcal{D}}[y(\mathbf{x};\mathcal{D})]\}\{\mathbb{E}_{\mathcal{D}}[y(\mathbf{x};\mathcal{D})] - h(\mathbf{x})\}. \end{aligned} \tag{3.39}$$

We now take the expectation of this expression with respect to $\mathcal{D}$ and note that the final term will vanish, giving

$$\begin{aligned}&\mathbb{E}_{\mathcal{D}}\left[\{y(\mathbf{x};\mathcal{D}) - h(\mathbf{x})\}^2\right] \\ &\quad = \underbrace{\{\mathbb{E}_{\mathcal{D}}[y(\mathbf{x};\mathcal{D})] - h(\mathbf{x})\}^2}_{(\text{bias})^2} + \underbrace{\mathbb{E}_{\mathcal{D}}\left[\{y(\mathbf{x};\mathcal{D}) - \mathbb{E}_{\mathcal{D}}[y(\mathbf{x};\mathcal{D})]\}^2\right]}_{\text{variance}}. \end{aligned} \tag{3.40}$$

We see that the expected squared difference between $y(\mathbf{x};\mathcal{D})$ and the regression function $h(\mathbf{x})$ can be expressed as the sum of two terms. The first term, called the squared *bias*, represents the extent to which the average prediction over all data sets differs from the desired regression function. The second term, called the *variance*, measures the extent to which the solutions for individual data sets vary around their average, and hence this measures the extent to which the function $y(\mathbf{x};\mathcal{D})$ is sensitive to the particular choice of data set. We shall provide some intuition to support these definitions shortly when we consider a simple example.

So far, we have considered a single input value $\mathbf{x}$. If we substitute this expansion back into (3.37), we obtain the following decomposition of the expected squared loss

$$\text{expected loss} = (\text{bias})^2 + \text{variance} + \text{noise} \tag{3.41}$$

where

$$(\text{bias})^2 = \int \{\mathbb{E}_{\mathcal{D}}[y(\mathbf{x};\mathcal{D})] - h(\mathbf{x})\}^2 p(\mathbf{x})\,\mathrm{d}\mathbf{x} \tag{3.42}$$

$$\text{variance} = \int \mathbb{E}_{\mathcal{D}}\left[\{y(\mathbf{x};\mathcal{D}) - \mathbb{E}_{\mathcal{D}}[y(\mathbf{x};\mathcal{D})]\}^2\right] p(\mathbf{x})\,\mathrm{d}\mathbf{x} \tag{3.43}$$

$$\text{noise} = \int \{h(\mathbf{x}) - t\}^2 p(\mathbf{x},t)\,\mathrm{d}\mathbf{x}\,\mathrm{d}t \tag{3.44}$$

and the bias and variance terms now refer to integrated quantities.

Our goal is to minimize the expected loss, which we have decomposed into the sum of a (squared) bias, a variance, and a constant noise term. As we shall see, there is a trade-off between bias and variance, with very flexible models having low bias and high variance, and relatively rigid models having high bias and low variance. The model with the optimal predictive capability is the one that leads to the best balance between bias and variance. This is illustrated by considering the sinusoidal data set from Chapter 1. Here we generate 100 data sets, each containing $N = 25$ data points, independently from the sinusoidal curve $h(x) = \sin(2\pi x)$. The data sets are indexed by $l = 1, \ldots, L$, where $L = 100$, and for each data set $\mathcal{D}^{(l)}$ we

Appendix A

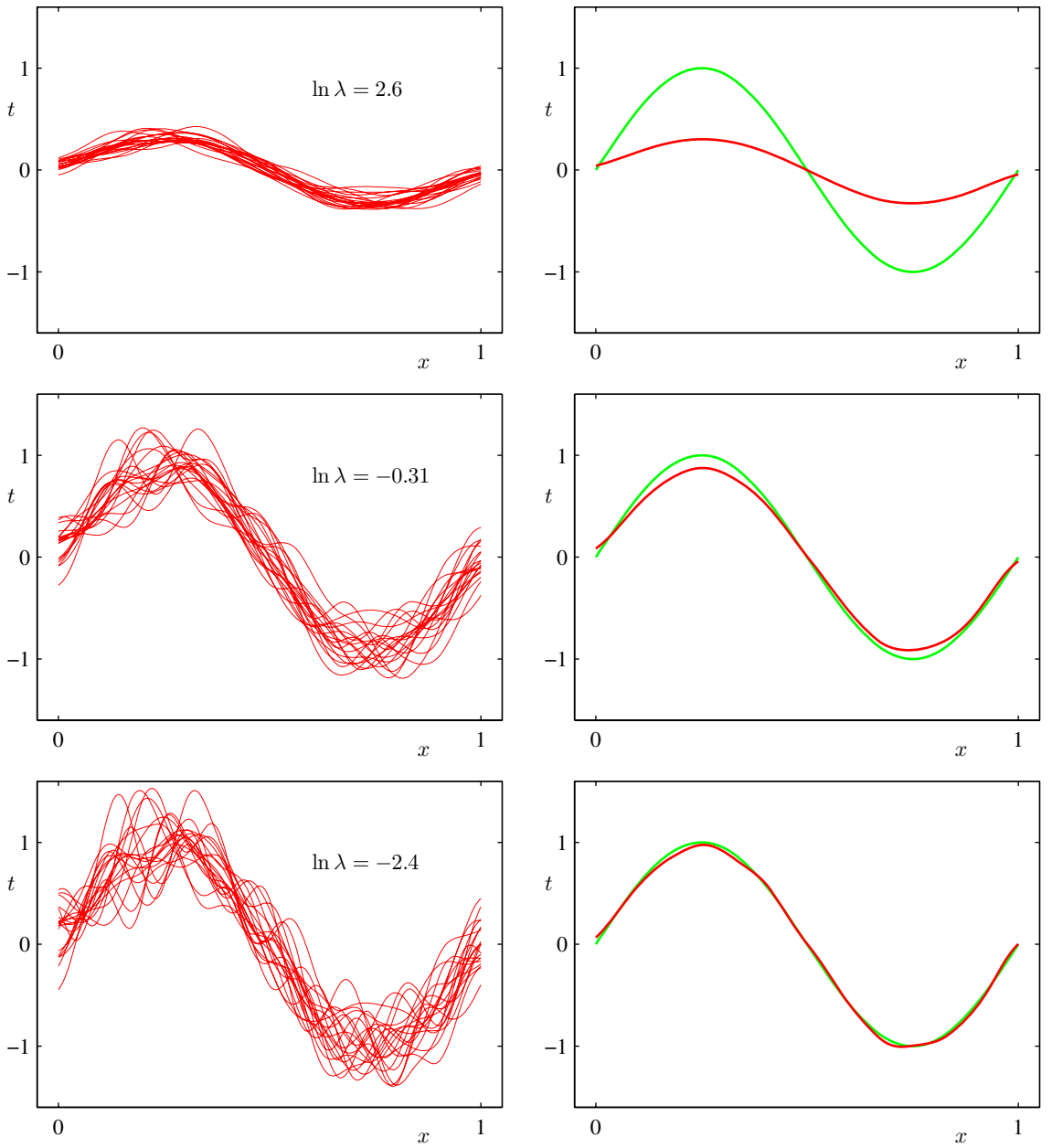


Figure 3.5 Illustration of the dependence of bias and variance on model complexity, governed by a regularization parameter λ, using the sinusoidal data set from Chapter 1. There are $L = 100$ data sets, each having $N = 25$ data points, and there are 24 Gaussian basis functions in the model so that the total number of parameters is $M = 25$ including the bias parameter. The left column shows the result of fitting the model to the data sets for various values of $\ln \lambda$ (for clarity, only 20 of the 100 fits are shown). The right column shows the corresponding average of the 100 fits (red) along with the sinusoidal function from which the data sets were generated (green).

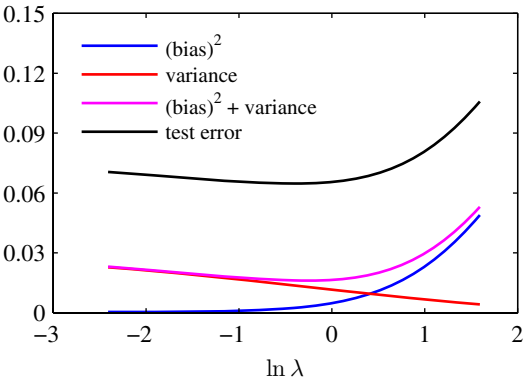


Figure 3.6 Plot of squared bias and variance, together with their sum, corresponding to the results shown in Figure 3.5. Also shown is the average test set error for a test data set size of 1000 points. The minimum value of $(\text{bias})^2 + \text{variance}$ occurs around $\ln \lambda = -0.31$, which is close to the value that gives the minimum error on the test data.

fit a model with 24 Gaussian basis functions by minimizing the regularized error function (3.27) to give a prediction function $y^{(l)}(x)$ as shown in Figure 3.5. The top row corresponds to a large value of the regularization coefficient λ that gives low variance (because the red curves in the left plot look similar) but high bias (because the two curves in the right plot are very different). Conversely on the bottom row, for which λ is small, there is large variance (shown by the high variability between the red curves in the left plot) but low bias (shown by the good fit between the average model fit and the original sinusoidal function). Note that the result of averaging many solutions for the complex model with $M = 25$ is a very good fit to the regression function, which suggests that averaging may be a beneficial procedure. Indeed, a weighted averaging of multiple solutions lies at the heart of a Bayesian approach, although the averaging is with respect to the posterior distribution of parameters, not with respect to multiple data sets.

We can also examine the bias-variance trade-off quantitatively for this example. The average prediction is estimated from

$$\overline{y}(x) = \frac{1}{L}\sum_{l=1}^{L} y^{(l)}(x) \tag{3.45}$$

and the integrated squared bias and integrated variance are then given by

$$(\text{bias})^2 = \frac{1}{N}\sum_{n=1}^{N}\{\overline{y}(x_n) - h(x_n)\}^2 \tag{3.46}$$

$$\text{variance} = \frac{1}{N}\sum_{n=1}^{N}\frac{1}{L}\sum_{l=1}^{L}\left\{y^{(l)}(x_n) - \overline{y}(x_n)\right\}^2 \tag{3.47}$$

where the integral over x weighted by the distribution $p(x)$ is approximated by a finite sum over data points drawn from that distribution. These quantities, along with their sum, are plotted as a function of $\ln \lambda$ in Figure 3.6. We see that small values of λ allow the model to become finely tuned to the noise on each individual

data set leading to large variance. Conversely, a large value of λ pulls the weight parameters towards zero leading to large bias.

Although the bias-variance decomposition may provide some interesting insights into the model complexity issue from a frequentist perspective, it is of limited practical value, because the bias-variance decomposition is based on averages with respect to ensembles of data sets, whereas in practice we have only the single observed data set. If we had a large number of independent training sets of a given size, we would be better off combining them into a single large training set, which of course would reduce the level of over-fitting for a given model complexity.

Given these limitations, we turn in the next section to a Bayesian treatment of linear basis function models, which not only provides powerful insights into the issues of over-fitting but which also leads to practical techniques for addressing the question model complexity.

3.3. Bayesian Linear Regression

In our discussion of maximum likelihood for setting the parameters of a linear regression model, we have seen that the effective model complexity, governed by the number of basis functions, needs to be controlled according to the size of the data set. Adding a regularization term to the log likelihood function means the effective model complexity can then be controlled by the value of the regularization coefficient, although the choice of the number and form of the basis functions is of course still important in determining the overall behaviour of the model.

This leaves the issue of deciding the appropriate model complexity for the particular problem, which cannot be decided simply by maximizing the likelihood function, because this always leads to excessively complex models and over-fitting. Independent hold-out data can be used to determine model complexity, as discussed in Section 1.3, but this can be both computationally expensive and wasteful of valuable data. We therefore turn to a Bayesian treatment of linear regression, which will avoid the over-fitting problem of maximum likelihood, and which will also lead to automatic methods of determining model complexity using the training data alone. Again, for simplicity we will focus on the case of a single target variable t. Extension to multiple target variables is straightforward and follows the discussion of Section 3.1.5.

3.3.1 Parameter distribution

We begin our discussion of the Bayesian treatment of linear regression by introducing a prior probability distribution over the model parameters $\mathbf{w}$. For the moment, we shall treat the noise precision parameter β as a known constant. First note that the likelihood function $p(\mathbf{t}|\mathbf{w})$ defined by (3.10) is the exponential of a quadratic function of $\mathbf{w}$. The corresponding conjugate prior is therefore given by a Gaussian distribution of the form

$$p(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\mathbf{m}_0, \mathbf{S}_0) \tag{3.48}$$

having mean $\mathbf{m}_0$ and covariance $\mathbf{S}_0$.

Next we compute the posterior distribution, which is proportional to the product of the likelihood function and the prior. Due to the choice of a conjugate Gaussian prior distribution, the posterior will also be Gaussian. We can evaluate this distribution by the usual procedure of completing the square in the exponential, and then finding the normalization coefficient using the standard result for a normalized Gaussian. However, we have already done the necessary work in deriving the general result (2.116), which allows us to write down the posterior distribution directly in the form

Exercise 3.7

$$p(\mathbf{w}|\mathbf{t}) = \mathcal{N}(\mathbf{w}|\mathbf{m}_N, \mathbf{S}_N) \tag{3.49}$$

where

$$\mathbf{m}_N = \mathbf{S}_N \left(\mathbf{S}_0^{-1}\mathbf{m}_0 + \beta\boldsymbol{\Phi}^{\mathrm{T}}\mathbf{t}\right) \tag{3.50}$$

$$\mathbf{S}_N^{-1} = \mathbf{S}_0^{-1} + \beta\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}. \tag{3.51}$$

Note that because the posterior distribution is Gaussian, its mode coincides with its mean. Thus the maximum posterior weight vector is simply given by $\mathbf{w}_{\mathrm{MAP}} = \mathbf{m}_N$. If we consider an infinitely broad prior $\mathbf{S}_0 = \alpha^{-1}\mathbf{I}$ with $\alpha \to 0$, the mean $\mathbf{m}_N$ of the posterior distribution reduces to the maximum likelihood value $\mathbf{w}_{\mathrm{ML}}$ given by (3.15). Similarly, if $N = 0$, then the posterior distribution reverts to the prior. Furthermore, if data points arrive sequentially, then the posterior distribution at any stage acts as the prior distribution for the subsequent data point, such that the new posterior distribution is again given by (3.49).

Exercise 3.8

For the remainder of this chapter, we shall consider a particular form of Gaussian prior in order to simplify the treatment. Specifically, we consider a zero-mean isotropic Gaussian governed by a single precision parameter α so that

$$p(\mathbf{w}|\alpha) = \mathcal{N}(\mathbf{w}|\mathbf{0}, \alpha^{-1}\mathbf{I}) \tag{3.52}$$

and the corresponding posterior distribution over $\mathbf{w}$ is then given by (3.49) with

$$\mathbf{m}_N = \beta\mathbf{S}_N\boldsymbol{\Phi}^{\mathrm{T}}\mathbf{t} \tag{3.53}$$

$$\mathbf{S}_N^{-1} = \alpha\mathbf{I} + \beta\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}. \tag{3.54}$$

The log of the posterior distribution is given by the sum of the log likelihood and the log of the prior and, as a function of $\mathbf{w}$, takes the form

$$\ln p(\mathbf{w}|\mathbf{t}) = -\frac{\beta}{2}\sum_{n=1}^{N}\{t_n - \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n)\}^2 - \frac{\alpha}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w} + \text{const}. \tag{3.55}$$

Maximization of this posterior distribution with respect to $\mathbf{w}$ is therefore equivalent to the minimization of the sum-of-squares error function with the addition of a quadratic regularization term, corresponding to (3.27) with $\lambda = \alpha/\beta$.

We can illustrate Bayesian learning in a linear basis function model, as well as the sequential update of a posterior distribution, using a simple example involving straight-line fitting. Consider a single input variable x, a single target variable t and

a linear model of the form $y(x, \mathbf{w}) = w_0 + w_1 x$. Because this has just two adaptive parameters, we can plot the prior and posterior distributions directly in parameter space. We generate synthetic data from the function $f(x, \mathbf{a}) = a_0 + a_1 x$ with parameter values $a_0 = -0.3$ and $a_1 = 0.5$ by first choosing values of x_n from the uniform distribution $\mathrm{U}(x|-1, 1)$, then evaluating $f(x_n, \mathbf{a})$, and finally adding Gaussian noise with standard deviation of 0.2 to obtain the target values t_n. Our goal is to recover the values of a_0 and a_1 from such data, and we will explore the dependence on the size of the data set. We assume here that the noise variance is known and hence we set the precision parameter to its true value $\beta = (1/0.2)^2 = 25$. Similarly, we fix the parameter α to 2.0. We shall shortly discuss strategies for determining α and β from the training data. Figure 3.7 shows the results of Bayesian learning in this model as the size of the data set is increased and demonstrates the sequential nature of Bayesian learning in which the current posterior distribution forms the prior when a new data point is observed. It is worth taking time to study this figure in detail as it illustrates several important aspects of Bayesian inference. The first row of this figure corresponds to the situation before any data points are observed and shows a plot of the prior distribution in $\mathbf{w}$ space together with six samples of the function $y(x, \mathbf{w})$ in which the values of $\mathbf{w}$ are drawn from the prior. In the second row, we see the situation after observing a single data point. The location (x, t) of the data point is shown by a blue circle in the right-hand column. In the left-hand column is a plot of the likelihood function $p(t|x, \mathbf{w})$ for this data point as a function of $\mathbf{w}$. Note that the likelihood function provides a soft constraint that the line must pass close to the data point, where close is determined by the noise precision β. For comparison, the true parameter values $a_0 = -0.3$ and $a_1 = 0.5$ used to generate the data set are shown by a white cross in the plots in the left column of Figure 3.7. When we multiply this likelihood function by the prior from the top row, and normalize, we obtain the posterior distribution shown in the middle plot on the second row. Samples of the regression function $y(x, \mathbf{w})$ obtained by drawing samples of $\mathbf{w}$ from this posterior distribution are shown in the right-hand plot. Note that these sample lines all pass close to the data point. The third row of this figure shows the effect of observing a second data point, again shown by a blue circle in the plot in the right-hand column. The corresponding likelihood function for this second data point alone is shown in the left plot. When we multiply this likelihood function by the posterior distribution from the second row, we obtain the posterior distribution shown in the middle plot of the third row. Note that this is exactly the same posterior distribution as would be obtained by combining the original prior with the likelihood function for the two data points. This posterior has now been influenced by two data points, and because two points are sufficient to define a line this already gives a relatively compact posterior distribution. Samples from this posterior distribution give rise to the functions shown in red in the third column, and we see that these functions pass close to both of the data points. The fourth row shows the effect of observing a total of 20 data points. The left-hand plot shows the likelihood function for the 20^{th} data point alone, and the middle plot shows the resulting posterior distribution that has now absorbed information from all 20 observations. Note how the posterior is much sharper than in the third row. In the limit of an infinite number of data points, the

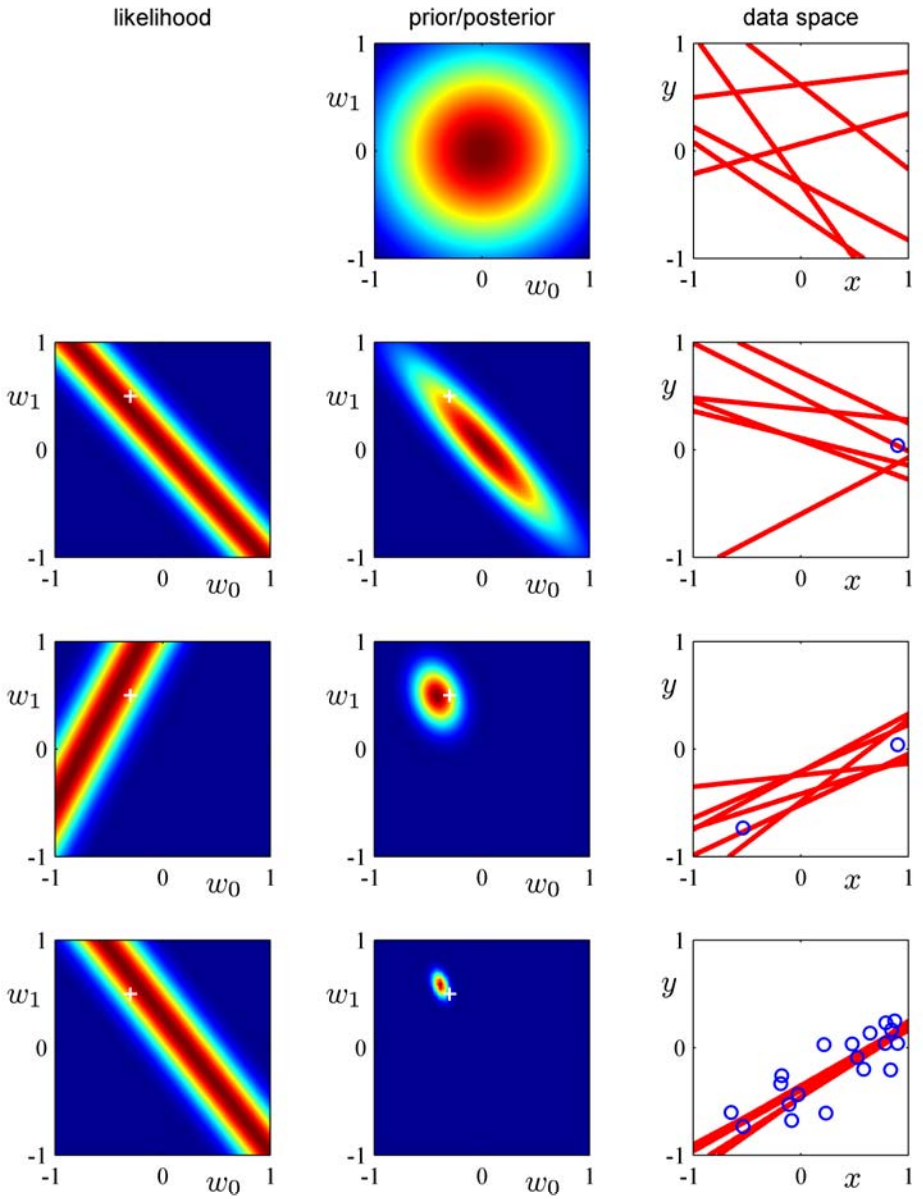


Figure 3.7 Illustration of sequential Bayesian learning for a simple linear model of the form $y(x, \mathbf{w}) = w_0 + w_1 x$. A detailed description of this figure is given in the text.

posterior distribution would become a delta function centred on the true parameter values, shown by the white cross.

Other forms of prior over the parameters can be considered. For instance, we can generalize the Gaussian prior to give

$$p(\mathbf{w}|\alpha) = \left[\frac{q}{2}\left(\frac{\alpha}{2}\right)^{1/q}\frac{1}{\Gamma(1/q)}\right]^M \exp\left(-\frac{\alpha}{2}\sum_{j=1}^{M}|w_j|^q\right) \tag{3.56}$$

in which $q = 2$ corresponds to the Gaussian distribution, and only in this case is the prior conjugate to the likelihood function (3.10). Finding the maximum of the posterior distribution over $\mathbf{w}$ corresponds to minimization of the regularized error function (3.29). In the case of the Gaussian prior, the mode of the posterior distribution was equal to the mean, although this will no longer hold if $q \neq 2$.

3.3.2 Predictive distribution

In practice, we are not usually interested in the value of $\mathbf{w}$ itself but rather in making predictions of t for new values of $\mathbf{x}$. This requires that we evaluate the *predictive distribution* defined by

$$p(t|\mathbf{t}, \alpha, \beta) = \int p(t|\mathbf{w}, \beta)p(\mathbf{w}|\mathbf{t}, \alpha, \beta)\,\mathrm{d}\mathbf{w} \tag{3.57}$$

in which $\mathbf{t}$ is the vector of target values from the training set, and we have omitted the corresponding input vectors from the right-hand side of the conditioning statements to simplify the notation. The conditional distribution $p(t|\mathbf{x}, \mathbf{w}, \beta)$ of the target variable is given by (3.8), and the posterior weight distribution is given by (3.49). We see that (3.57) involves the convolution of two Gaussian distributions, and so making use of the result (2.115) from Section 8.1.4, we see that the predictive distribution takes the form

Exercise 3.10

$$p(t|\mathbf{x}, \mathbf{t}, \alpha, \beta) = \mathcal{N}(t|\mathbf{m}_N^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}), \sigma_N^2(\mathbf{x})) \tag{3.58}$$

where the variance $\sigma_N^2(\mathbf{x})$ of the predictive distribution is given by

$$\sigma_N^2(\mathbf{x}) = \frac{1}{\beta} + \boldsymbol{\phi}(\mathbf{x})^{\mathrm{T}}\mathbf{S}_N\boldsymbol{\phi}(\mathbf{x}). \tag{3.59}$$

The first term in (3.59) represents the noise on the data whereas the second term reflects the uncertainty associated with the parameters $\mathbf{w}$. Because the noise process and the distribution of $\mathbf{w}$ are independent Gaussians, their variances are additive. Note that, as additional data points are observed, the posterior distribution becomes narrower. As a consequence it can be shown (Qazaz *et al.*, 1997) that $\sigma_{N+1}^2(\mathbf{x}) \leqslant \sigma_N^2(\mathbf{x})$. In the limit $N \to \infty$, the second term in (3.59) goes to zero, and the variance of the predictive distribution arises solely from the additive noise governed by the parameter β.

Exercise 3.11

As an illustration of the predictive distribution for Bayesian linear regression models, let us return to the synthetic sinusoidal data set of Section 1.1. In Figure 3.8,

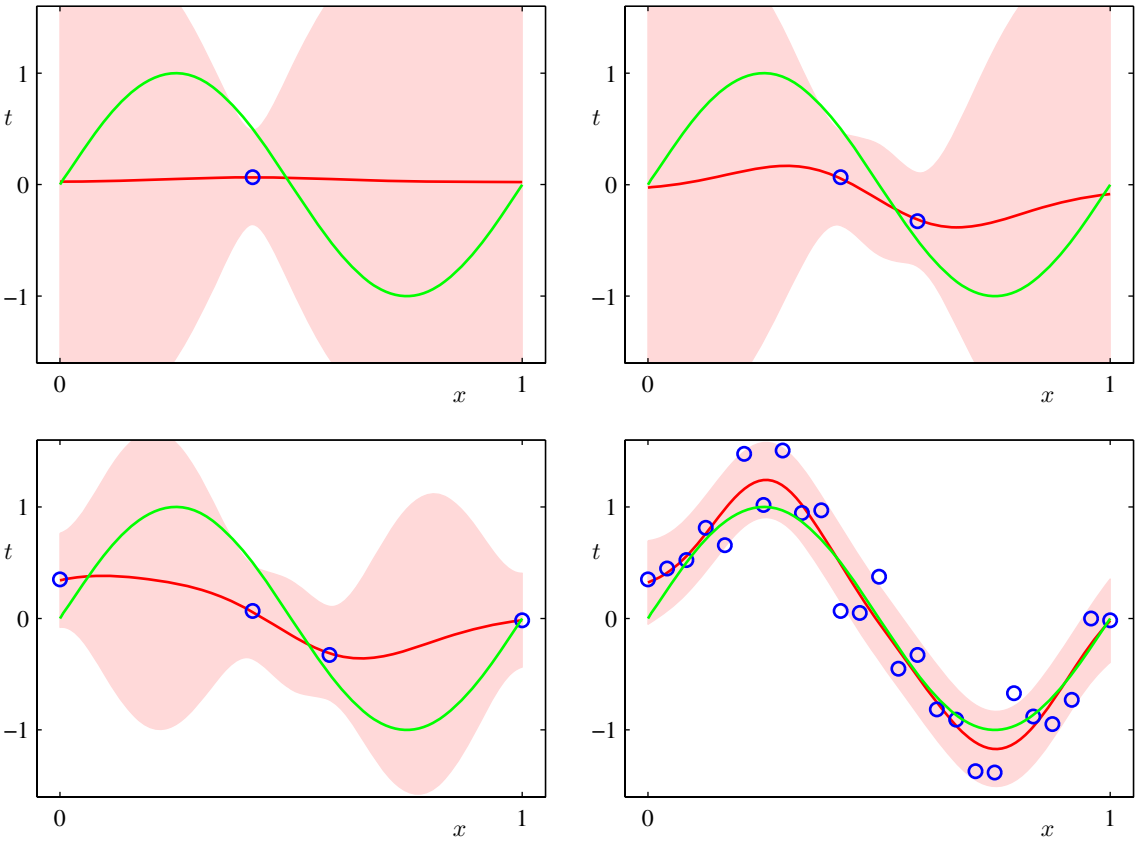


Figure 3.8 Examples of the predictive distribution (3.58) for a model consisting of 9 Gaussian basis functions of the form (3.4) using the synthetic sinusoidal data set of Section 1.1. See the text for a detailed discussion.

we fit a model comprising a linear combination of Gaussian basis functions to data sets of various sizes and then look at the corresponding posterior distributions. Here the green curves correspond to the function $\sin(2\pi x)$ from which the data points were generated (with the addition of Gaussian noise). Data sets of size $N = 1$, $N = 2$, $N = 4$, and $N = 25$ are shown in the four plots by the blue circles. For each plot, the red curve shows the mean of the corresponding Gaussian predictive distribution, and the red shaded region spans one standard deviation either side of the mean. Note that the predictive uncertainty depends on x and is smallest in the neighbourhood of the data points. Also note that the level of uncertainty decreases as more data points are observed.

The plots in Figure 3.8 only show the point-wise predictive variance as a function of x. In order to gain insight into the covariance between the predictions at different values of x, we can draw samples from the posterior distribution over $\mathbf{w}$, and then plot the corresponding functions $y(x, \mathbf{w})$, as shown in Figure 3.9.

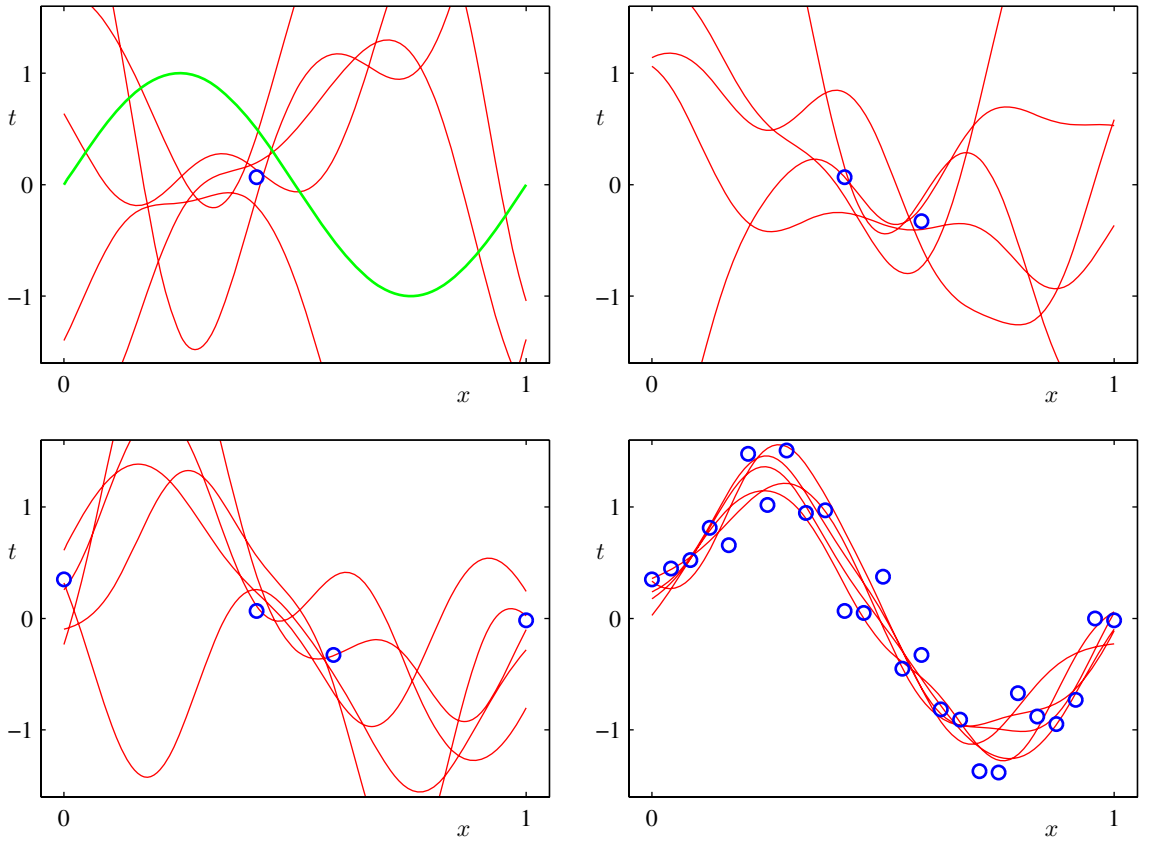


Figure 3.9 Plots of the function $y(x, \mathbf{w})$ using samples from the posterior distributions over $\mathbf{w}$ corresponding to the plots in Figure 3.8.

If we used localized basis functions such as Gaussians, then in regions away from the basis function centres, the contribution from the second term in the predictive variance (3.59) will go to zero, leaving only the noise contribution β^{-1}. Thus, the model becomes very confident in its predictions when extrapolating outside the region occupied by the basis functions, which is generally an undesirable behaviour. This problem can be avoided by adopting an alternative Bayesian approach to regression known as a Gaussian process.

Section 6.4

Note that, if both $\mathbf{w}$ and β are treated as unknown, then we can introduce a conjugate prior distribution $p(\mathbf{w}, \beta)$ that, from the discussion in Section 2.3.6, will be given by a Gaussian-gamma distribution (Denison *et al.*, 2002). In this case, the predictive distribution is a Student's t-distribution.

Exercise 3.12
Exercise 3.13

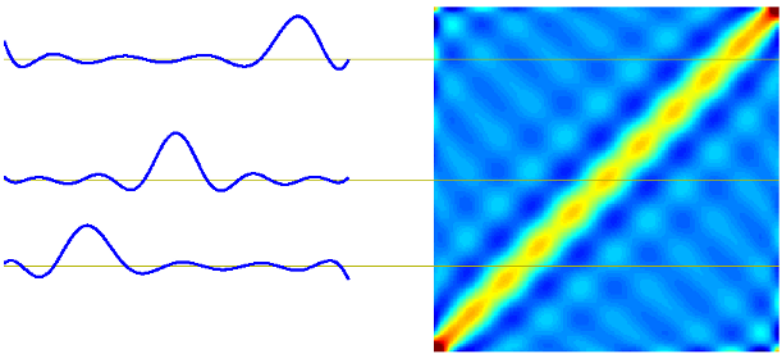


Figure 3.10 The equivalent kernel $k(x, x')$ for the Gaussian basis functions in Figure 3.1, shown as a plot of x versus x', together with three slices through this matrix corresponding to three different values of x. The data set used to generate this kernel comprised 200 values of x equally spaced over the interval $(-1, 1)$.

3.3.3 Equivalent kernel

Chapter 6

The posterior mean solution (3.53) for the linear basis function model has an interesting interpretation that will set the stage for kernel methods, including Gaussian processes. If we substitute (3.53) into the expression (3.3), we see that the predictive mean can be written in the form

$$y(\mathbf{x}, \mathbf{m}_N) = \mathbf{m}_N^{\mathrm{T}} \boldsymbol{\phi}(\mathbf{x}) = \beta \boldsymbol{\phi}(\mathbf{x})^{\mathrm{T}} \mathbf{S}_N \boldsymbol{\Phi}^{\mathrm{T}} \mathbf{t} = \sum_{n=1}^{N} \beta \boldsymbol{\phi}(\mathbf{x})^{\mathrm{T}} \mathbf{S}_N \boldsymbol{\phi}(\mathbf{x}_n) t_n \tag{3.60}$$

where $\mathbf{S}_N$ is defined by (3.51). Thus the mean of the predictive distribution at a point $\mathbf{x}$ is given by a linear combination of the training set target variables t_n, so that we can write

$$y(\mathbf{x}, \mathbf{m}_N) = \sum_{n=1}^{N} k(\mathbf{x}, \mathbf{x}_n) t_n \tag{3.61}$$

where the function

$$k(\mathbf{x}, \mathbf{x}') = \beta \boldsymbol{\phi}(\mathbf{x})^{\mathrm{T}} \mathbf{S}_N \boldsymbol{\phi}(\mathbf{x}') \tag{3.62}$$

is known as the *smoother matrix* or the *equivalent kernel*. Regression functions, such as this, which make predictions by taking linear combinations of the training set target values are known as *linear smoothers*. Note that the equivalent kernel depends on the input values $\mathbf{x}_n$ from the data set because these appear in the definition of $\mathbf{S}_N$. The equivalent kernel is illustrated for the case of Gaussian basis functions in Figure 3.10 in which the kernel functions $k(x, x')$ have been plotted as a function of x' for three different values of x. We see that they are localized around x, and so the mean of the predictive distribution at x, given by $y(x, \mathbf{m}_N)$, is obtained by forming a weighted combination of the target values in which data points close to x are given higher weight than points further removed from x. Intuitively, it seems reasonable that we should weight local evidence more strongly than distant evidence. Note that this localization property holds not only for the localized Gaussian basis functions but also for the nonlocal polynomial and sigmoidal basis functions, as illustrated in Figure 3.11.

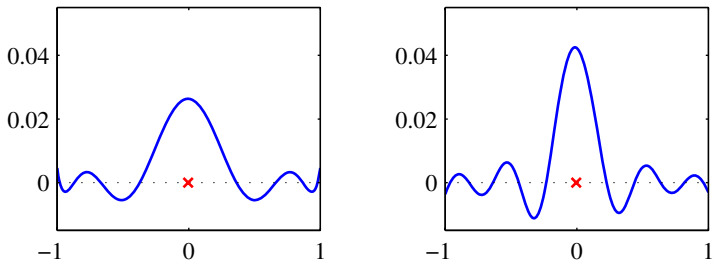


Figure 3.11 Examples of equivalent kernels $k(x, x')$ for $x = 0$ plotted as a function of x', corresponding (left) to the polynomial basis functions and (right) to the sigmoidal basis functions shown in Figure 3.1. Note that these are localized functions of x' even though the corresponding basis functions are nonlocal.

Further insight into the role of the equivalent kernel can be obtained by considering the covariance between $y(\mathbf{x})$ and $y(\mathbf{x}')$, which is given by

$$\begin{aligned} \mathrm{cov}[y(\mathbf{x}), y(\mathbf{x}')] &= \mathrm{cov}[\boldsymbol{\phi}(\mathbf{x})^{\mathrm{T}}\mathbf{w}, \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}')] \\ &= \boldsymbol{\phi}(\mathbf{x})^{\mathrm{T}}\mathbf{S}_N\boldsymbol{\phi}(\mathbf{x}') = \beta^{-1}k(\mathbf{x}, \mathbf{x}') \end{aligned} \tag{3.63}$$

where we have made use of (3.49) and (3.62). From the form of the equivalent kernel, we see that the predictive mean at nearby points will be highly correlated, whereas for more distant pairs of points the correlation will be smaller.

The predictive distribution shown in Figure 3.8 allows us to visualize the pointwise uncertainty in the predictions, governed by (3.59). However, by drawing samples from the posterior distribution over $\mathbf{w}$, and plotting the corresponding model functions $y(\mathbf{x}, \mathbf{w})$ as in Figure 3.9, we are visualizing the joint uncertainty in the posterior distribution between the y values at two (or more) x values, as governed by the equivalent kernel.

The formulation of linear regression in terms of a kernel function suggests an alternative approach to regression as follows. Instead of introducing a set of basis functions, which implicitly determines an equivalent kernel, we can instead define a localized kernel directly and use this to make predictions for new input vectors $\mathbf{x}$, given the observed training set. This leads to a practical framework for regression (and classification) called *Gaussian processes*, which will be discussed in detail in Section 6.4.

We have seen that the effective kernel defines the weights by which the training set target values are combined in order to make a prediction at a new value of $\mathbf{x}$, and it can be shown that these weights sum to one, in other words

$$\sum_{n=1}^{N} k(\mathbf{x}, \mathbf{x}_n) = 1 \tag{3.64}$$

Exercise 3.14

for all values of $\mathbf{x}$. This intuitively pleasing result can easily be proven informally by noting that the summation is equivalent to considering the predictive mean $\widehat{y}(\mathbf{x})$ for a set of target data in which $t_n = 1$ for all n. Provided the basis functions are linearly independent, that there are more data points than basis functions, and that one of the basis functions is constant (corresponding to the bias parameter), then it is clear that we can fit the training data exactly and hence that the predictive mean will

be simply $\widehat{y}(\mathbf{x}) = 1$, from which we obtain (3.64). Note that the kernel function can be negative as well as positive, so although it satisfies a summation constraint, the corresponding predictions are not necessarily convex combinations of the training set target variables.

Finally, we note that the equivalent kernel (3.62) satisfies an important property shared by kernel functions in general, namely that it can be expressed in the form an inner product with respect to a vector $\boldsymbol{\psi}(\mathbf{x})$ of nonlinear functions, so that

Chapter 6

$$k(\mathbf{x}, \mathbf{z}) = \boldsymbol{\psi}(\mathbf{x})^{\mathrm{T}} \boldsymbol{\psi}(\mathbf{z}) \tag{3.65}$$

where $\boldsymbol{\psi}(\mathbf{x}) = \beta^{1/2} \mathbf{S}_N^{1/2} \boldsymbol{\phi}(\mathbf{x})$.

3.4. Bayesian Model Comparison

In Chapter 1, we highlighted the problem of over-fitting as well as the use of cross-validation as a technique for setting the values of regularization parameters or for choosing between alternative models. Here we consider the problem of model selection from a Bayesian perspective. In this section, our discussion will be very general, and then in Section 3.5 we shall see how these ideas can be applied to the determination of regularization parameters in linear regression.

As we shall see, the over-fitting associated with maximum likelihood can be avoided by marginalizing (summing or integrating) over the model parameters instead of making point estimates of their values. Models can then be compared directly on the training data, without the need for a validation set. This allows all available data to be used for training and avoids the multiple training runs for each model associated with cross-validation. It also allows multiple complexity parameters to be determined simultaneously as part of the training process. For example, in Chapter 7 we shall introduce the *relevance vector machine*, which is a Bayesian model having one complexity parameter for every training data point.

The Bayesian view of model comparison simply involves the use of probabilities to represent uncertainty in the choice of model, along with a consistent application of the sum and product rules of probability. Suppose we wish to compare a set of L models $\{\mathcal{M}_i\}$ where $i = 1, \ldots, L$. Here a model refers to a probability distribution over the observed data $\mathcal{D}$. In the case of the polynomial curve-fitting problem, the distribution is defined over the set of target values $\mathbf{t}$, while the set of input values $\mathbf{X}$ is assumed to be known. Other types of model define a joint distributions over $\mathbf{X}$ and $\mathbf{t}$. We shall suppose that the data is generated from one of these models but we are uncertain which one. Our uncertainty is expressed through a prior probability distribution $p(\mathcal{M}_i)$. Given a training set $\mathcal{D}$, we then wish to evaluate the posterior distribution

Section 1.5.4

$$p(\mathcal{M}_i|\mathcal{D}) \propto p(\mathcal{M}_i) p(\mathcal{D}|\mathcal{M}_i). \tag{3.66}$$

The prior allows us to express a preference for different models. Let us simply assume that all models are given equal prior probability. The interesting term is the *model evidence* $p(\mathcal{D}|\mathcal{M}_i)$ which expresses the preference shown by the data for

different models, and we shall examine this term in more detail shortly. The model evidence is sometimes also called the *marginal likelihood* because it can be viewed as a likelihood function over the space of models, in which the parameters have been marginalized out. The ratio of model evidences $p(\mathcal{D}|\mathcal{M}_i)/p(\mathcal{D}|\mathcal{M}_j)$ for two models is known as a *Bayes factor* (Kass and Raftery, 1995).

Once we know the posterior distribution over models, the predictive distribution is given, from the sum and product rules, by

$$p(t|\mathbf{x}, \mathcal{D}) = \sum_{i=1}^{L} p(t|\mathbf{x}, \mathcal{M}_i, \mathcal{D}) p(\mathcal{M}_i|\mathcal{D}). \tag{3.67}$$

This is an example of a *mixture distribution* in which the overall predictive distribution is obtained by averaging the predictive distributions $p(t|\mathbf{x}, \mathcal{M}_i, \mathcal{D})$ of individual models, weighted by the posterior probabilities $p(\mathcal{M}_i|\mathcal{D})$ of those models. For instance, if we have two models that are a-posteriori equally likely and one predicts a narrow distribution around $t = a$ while the other predicts a narrow distribution around $t = b$, the overall predictive distribution will be a bimodal distribution with modes at $t = a$ and $t = b$, not a single model at $t = (a + b)/2$.

A simple approximation to model averaging is to use the single most probable model alone to make predictions. This is known as *model selection*.

For a model governed by a set of parameters $\mathbf{w}$, the model evidence is given, from the sum and product rules of probability, by

$$p(\mathcal{D}|\mathcal{M}_i) = \int p(\mathcal{D}|\mathbf{w}, \mathcal{M}_i) p(\mathbf{w}|\mathcal{M}_i) \, \mathrm{d}\mathbf{w}. \tag{3.68}$$

Chapter 11

From a sampling perspective, the marginal likelihood can be viewed as the probability of generating the data set $\mathcal{D}$ from a model whose parameters are sampled at random from the prior. It is also interesting to note that the evidence is precisely the normalizing term that appears in the denominator in Bayes' theorem when evaluating the posterior distribution over parameters because

$$p(\mathbf{w}|\mathcal{D}, \mathcal{M}_i) = \frac{p(\mathcal{D}|\mathbf{w}, \mathcal{M}_i) p(\mathbf{w}|\mathcal{M}_i)}{p(\mathcal{D}|\mathcal{M}_i)}. \tag{3.69}$$

We can obtain some insight into the model evidence by making a simple approximation to the integral over parameters. Consider first the case of a model having a single parameter w. The posterior distribution over parameters is proportional to $p(\mathcal{D}|w)p(w)$, where we omit the dependence on the model $\mathcal{M}_i$ to keep the notation uncluttered. If we assume that the posterior distribution is sharply peaked around the most probable value w_{MAP}, with width $\Delta w_{\mathrm{posterior}}$, then we can approximate the integral by the value of the integrand at its maximum times the width of the peak. If we further assume that the prior is flat with width $\Delta w_{\mathrm{prior}}$ so that $p(w) = 1/\Delta w_{\mathrm{prior}}$, then we have

$$p(\mathcal{D}) = \int p(\mathcal{D}|w) p(w) \, \mathrm{d}w \simeq p(\mathcal{D}|w_{\mathrm{MAP}}) \frac{\Delta w_{\mathrm{posterior}}}{\Delta w_{\mathrm{prior}}} \tag{3.70}$$

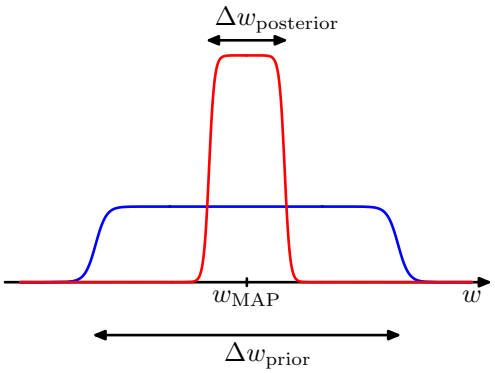


Figure 3.12 We can obtain a rough approximation to the model evidence if we assume that the posterior distribution over parameters is sharply peaked around its mode w_{MAP}.

and so taking logs we obtain

$$\ln p(\mathcal{D}) \simeq \ln p(\mathcal{D}|w_{\text{MAP}}) + \ln\left(\frac{\Delta w_{\text{posterior}}}{\Delta w_{\text{prior}}}\right). \tag{3.71}$$

This approximation is illustrated in Figure 3.12. The first term represents the fit to the data given by the most probable parameter values, and for a flat prior this would correspond to the log likelihood. The second term penalizes the model according to its complexity. Because $\Delta w_{\text{posterior}} < \Delta w_{\text{prior}}$ this term is negative, and it increases in magnitude as the ratio $\Delta w_{\text{posterior}}/\Delta w_{\text{prior}}$ gets smaller. Thus, if parameters are finely tuned to the data in the posterior distribution, then the penalty term is large.

For a model having a set of M parameters, we can make a similar approximation for each parameter in turn. Assuming that all parameters have the same ratio of $\Delta w_{\text{posterior}}/\Delta w_{\text{prior}}$, we obtain

$$\ln p(\mathcal{D}) \simeq \ln p(\mathcal{D}|\mathbf{w}_{\text{MAP}}) + M \ln\left(\frac{\Delta w_{\text{posterior}}}{\Delta w_{\text{prior}}}\right). \tag{3.72}$$

Thus, in this very simple approximation, the size of the complexity penalty increases linearly with the number M of adaptive parameters in the model. As we increase the complexity of the model, the first term will typically decrease, because a more complex model is better able to fit the data, whereas the second term will increase due to the dependence on M. The optimal model complexity, as determined by the maximum evidence, will be given by a trade-off between these two competing terms. We shall later develop a more refined version of this approximation, based on a Gaussian approximation to the posterior distribution.

Section 4.4.1

We can gain further insight into Bayesian model comparison and understand how the marginal likelihood can favour models of intermediate complexity by considering Figure 3.13. Here the horizontal axis is a one-dimensional representation of the space of possible data sets, so that each point on this axis corresponds to a specific data set. We now consider three models $\mathcal{M}_1$, $\mathcal{M}_2$ and $\mathcal{M}_3$ of successively increasing complexity. Imagine running these models generatively to produce example data sets, and then looking at the distribution of data sets that result. Any given

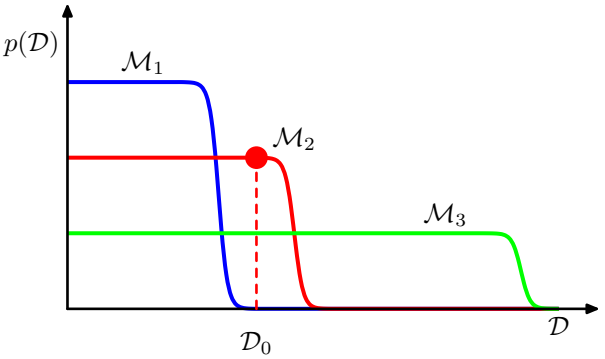


Figure 3.13 Schematic illustration of the distribution of data sets for three models of different complexity, in which $\mathcal{M}_1$ is the simplest and $\mathcal{M}_3$ is the most complex. Note that the distributions are normalized. In this example, for the particular observed data set $\mathcal{D}_0$, the model $\mathcal{M}_2$ with intermediate complexity has the largest evidence.

model can generate a variety of different data sets since the parameters are governed by a prior probability distribution, and for any choice of the parameters there may be random noise on the target variables. To generate a particular data set from a specific model, we first choose the values of the parameters from their prior distribution $p(\mathbf{w})$, and then for these parameter values we sample the data from $p(\mathcal{D}|\mathbf{w})$. A simple model (for example, based on a first order polynomial) has little variability and so will generate data sets that are fairly similar to each other. Its distribution $p(\mathcal{D})$ is therefore confined to a relatively small region of the horizontal axis. By contrast, a complex model (such as a ninth order polynomial) can generate a great variety of different data sets, and so its distribution $p(\mathcal{D})$ is spread over a large region of the space of data sets. Because the distributions $p(\mathcal{D}|\mathcal{M}_i)$ are normalized, we see that the particular data set $\mathcal{D}_0$ can have the highest value of the evidence for the model of intermediate complexity. Essentially, the simpler model cannot fit the data well, whereas the more complex model spreads its predictive probability over too broad a range of data sets and so assigns relatively small probability to any one of them.

Implicit in the Bayesian model comparison framework is the assumption that the true distribution from which the data are generated is contained within the set of models under consideration. Provided this is so, we can show that Bayesian model comparison will on average favour the correct model. To see this, consider two models $\mathcal{M}_1$ and $\mathcal{M}_2$ in which the truth corresponds to $\mathcal{M}_1$. For a given finite data set, it is possible for the Bayes factor to be larger for the incorrect model. However, if we average the Bayes factor over the distribution of data sets, we obtain the expected Bayes factor in the form

$$\int p(\mathcal{D}|\mathcal{M}_1) \ln \frac{p(\mathcal{D}|\mathcal{M}_1)}{p(\mathcal{D}|\mathcal{M}_2)} \, \mathrm{d}\mathcal{D} \tag{3.73}$$

where the average has been taken with respect to the true distribution of the data. This quantity is an example of the *Kullback-Leibler* divergence and satisfies the property of always being positive unless the two distributions are equal in which case it is zero. Thus on average the Bayes factor will always favour the correct model.

Section 1.6.1

We have seen that the Bayesian framework avoids the problem of over-fitting and allows models to be compared on the basis of the training data alone. However,

a Bayesian approach, like any approach to pattern recognition, needs to make assumptions about the form of the model, and if these are invalid then the results can be misleading. In particular, we see from Figure 3.12 that the model evidence can be sensitive to many aspects of the prior, such as the behaviour in the tails. Indeed, the evidence is not defined if the prior is improper, as can be seen by noting that an improper prior has an arbitrary scaling factor (in other words, the normalization coefficient is not defined because the distribution cannot be normalized). If we consider a proper prior and then take a suitable limit in order to obtain an improper prior (for example, a Gaussian prior in which we take the limit of infinite variance) then the evidence will go to zero, as can be seen from (3.70) and Figure 3.12. It may, however, be possible to consider the evidence ratio between two models first and then take a limit to obtain a meaningful answer.

In a practical application, therefore, it will be wise to keep aside an independent test set of data on which to evaluate the overall performance of the final system.

3.5. The Evidence Approximation

In a fully Bayesian treatment of the linear basis function model, we would introduce prior distributions over the hyperparameters α and β and make predictions by marginalizing with respect to these hyperparameters as well as with respect to the parameters $\mathbf{w}$. However, although we can integrate analytically over either $\mathbf{w}$ or over the hyperparameters, the complete marginalization over all of these variables is analytically intractable. Here we discuss an approximation in which we set the hyperparameters to specific values determined by maximizing the *marginal likelihood function* obtained by first integrating over the parameters $\mathbf{w}$. This framework is known in the statistics literature as *empirical Bayes* (Bernardo and Smith, 1994; Gelman *et al.*, 2004), or *type 2 maximum likelihood* (Berger, 1985), or *generalized maximum likelihood* (Wahba, 1975), and in the machine learning literature is also called the *evidence approximation* (Gull, 1989; MacKay, 1992a).

If we introduce hyperpriors over α and β, the predictive distribution is obtained by marginalizing over $\mathbf{w}$, α and β so that

$$p(t|\mathbf{t}) = \iiint p(t|\mathbf{w},\beta)p(\mathbf{w}|\mathbf{t},\alpha,\beta)p(\alpha,\beta|\mathbf{t})\,\mathrm{d}\mathbf{w}\,\mathrm{d}\alpha\,\mathrm{d}\beta \tag{3.74}$$

where $p(t|\mathbf{w},\beta)$ is given by (3.8) and $p(\mathbf{w}|\mathbf{t},\alpha,\beta)$ is given by (3.49) with $\mathbf{m}_N$ and $\mathbf{S}_N$ defined by (3.53) and (3.54) respectively. Here we have omitted the dependence on the input variable $\mathbf{x}$ to keep the notation uncluttered. If the posterior distribution $p(\alpha,\beta|\mathbf{t})$ is sharply peaked around values $\widehat{\alpha}$ and $\widehat{\beta}$, then the predictive distribution is obtained simply by marginalizing over $\mathbf{w}$ in which α and β are fixed to the values $\widehat{\alpha}$ and $\widehat{\beta}$, so that

$$p(t|\mathbf{t}) \simeq p(t|\mathbf{t},\widehat{\alpha},\widehat{\beta}) = \int p(t|\mathbf{w},\widehat{\beta})p(\mathbf{w}|\mathbf{t},\widehat{\alpha},\widehat{\beta})\,\mathrm{d}\mathbf{w}. \tag{3.75}$$

From Bayes' theorem, the posterior distribution for α and β is given by

$$p(\alpha, \beta|\mathbf{t}) \propto p(\mathbf{t}|\alpha, \beta)p(\alpha, \beta). \tag{3.76}$$

If the prior is relatively flat, then in the evidence framework the values of $\widehat{\alpha}$ and $\widehat{\beta}$ are obtained by maximizing the marginal likelihood function $p(\mathbf{t}|\alpha, \beta)$. We shall proceed by evaluating the marginal likelihood for the linear basis function model and then finding its maxima. This will allow us to determine values for these hyperparameters from the training data alone, without recourse to cross-validation. Recall that the ratio α/β is analogous to a regularization parameter.

As an aside it is worth noting that, if we define conjugate (Gamma) prior distributions over α and β, then the marginalization over these hyperparameters in (3.74) can be performed analytically to give a Student's t-distribution over $\mathbf{w}$ (see Section 2.3.7). Although the resulting integral over $\mathbf{w}$ is no longer analytically tractable, it might be thought that approximating this integral, for example using the Laplace approximation discussed (Section 4.4) which is based on a local Gaussian approximation centred on the mode of the posterior distribution, might provide a practical alternative to the evidence framework (Buntine and Weigend, 1991). However, the integrand as a function of $\mathbf{w}$ typically has a strongly skewed mode so that the Laplace approximation fails to capture the bulk of the probability mass, leading to poorer results than those obtained by maximizing the evidence (MacKay, 1999).

Returning to the evidence framework, we note that there are two approaches that we can take to the maximization of the log evidence. We can evaluate the evidence function analytically and then set its derivative equal to zero to obtain re-estimation equations for α and β, which we shall do in Section 3.5.2. Alternatively we use a technique called the expectation maximization (EM) algorithm, which will be discussed in Section 9.3.4 where we shall also show that these two approaches converge to the same solution.

3.5.1 Evaluation of the evidence function

The marginal likelihood function $p(\mathbf{t}|\alpha, \beta)$ is obtained by integrating over the weight parameters $\mathbf{w}$, so that

$$p(\mathbf{t}|\alpha, \beta) = \int p(\mathbf{t}|\mathbf{w}, \beta)p(\mathbf{w}|\alpha)\,\mathrm{d}\mathbf{w}. \tag{3.77}$$

Exercise 3.16 One way to evaluate this integral is to make use once again of the result (2.115) for the conditional distribution in a linear-Gaussian model. Here we shall evaluate the integral instead by completing the square in the exponent and making use of the standard form for the normalization coefficient of a Gaussian.

Exercise 3.17 From (3.11), (3.12), and (3.52), we can write the evidence function in the form

$$p(\mathbf{t}|\alpha, \beta) = \left(\frac{\beta}{2\pi}\right)^{N/2} \left(\frac{\alpha}{2\pi}\right)^{M/2} \int \exp\{-E(\mathbf{w})\}\,\mathrm{d}\mathbf{w} \tag{3.78}$$

where M is the dimensionality of $\mathbf{w}$, and we have defined

$$\begin{aligned} E(\mathbf{w}) &= \beta E_D(\mathbf{w}) + \alpha E_W(\mathbf{w}) \\ &= \frac{\beta}{2}\|\mathbf{t} - \mathbf{\Phi}\mathbf{w}\|^2 + \frac{\alpha}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w}. \end{aligned} \tag{3.79}$$

Exercise 3.18

We recognize (3.79) as being equal, up to a constant of proportionality, to the regularized sum-of-squares error function (3.27). We now complete the square over $\mathbf{w}$ giving

$$E(\mathbf{w}) = E(\mathbf{m}_N) + \frac{1}{2}(\mathbf{w} - \mathbf{m}_N)^{\mathrm{T}}\mathbf{A}(\mathbf{w} - \mathbf{m}_N) \tag{3.80}$$

where we have introduced

$$\mathbf{A} = \alpha\mathbf{I} + \beta\mathbf{\Phi}^{\mathrm{T}}\mathbf{\Phi} \tag{3.81}$$

together with

$$E(\mathbf{m}_N) = \frac{\beta}{2}\|\mathbf{t} - \mathbf{\Phi}\mathbf{m}_N\|^2 + \frac{\alpha}{2}\mathbf{m}_N^{\mathrm{T}}\mathbf{m}_N. \tag{3.82}$$

Note that $\mathbf{A}$ corresponds to the matrix of second derivatives of the error function

$$\mathbf{A} = \nabla\nabla E(\mathbf{w}) \tag{3.83}$$

and is known as the *Hessian matrix*. Here we have also defined $\mathbf{m}_N$ given by

$$\mathbf{m}_N = \beta\mathbf{A}^{-1}\mathbf{\Phi}^{\mathrm{T}}\mathbf{t}. \tag{3.84}$$

Using (3.54), we see that $\mathbf{A} = \mathbf{S}_N^{-1}$, and hence (3.84) is equivalent to the previous definition (3.53), and therefore represents the mean of the posterior distribution.

Exercise 3.19

The integral over $\mathbf{w}$ can now be evaluated simply by appealing to the standard result for the normalization coefficient of a multivariate Gaussian, giving

$$\begin{aligned} &\int \exp\{-E(\mathbf{w})\}\,\mathrm{d}\mathbf{w} \\ &= \exp\{-E(\mathbf{m}_N)\}\int \exp\left\{-\frac{1}{2}(\mathbf{w} - \mathbf{m}_N)^{\mathrm{T}}\mathbf{A}(\mathbf{w} - \mathbf{m}_N)\right\}\mathrm{d}\mathbf{w} \\ &= \exp\{-E(\mathbf{m}_N)\}(2\pi)^{M/2}|\mathbf{A}|^{-1/2}. \end{aligned} \tag{3.85}$$

Using (3.78) we can then write the log of the marginal likelihood in the form

$$\ln p(\mathbf{t}|\alpha,\beta) = \frac{M}{2}\ln\alpha + \frac{N}{2}\ln\beta - E(\mathbf{m}_N) - \frac{1}{2}\ln|\mathbf{A}| - \frac{N}{2}\ln(2\pi) \tag{3.86}$$

which is the required expression for the evidence function.

Returning to the polynomial regression problem, we can plot the model evidence against the order of the polynomial, as shown in Figure 3.14. Here we have assumed a prior of the form (1.65) with the parameter α fixed at $\alpha = 5 \times 10^{-3}$. The form of this plot is very instructive. Referring back to Figure 1.4, we see that the $M = 0$ polynomial has very poor fit to the data and consequently gives a relatively low value

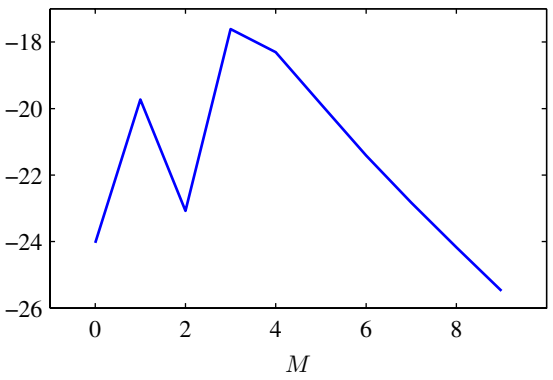


Figure 3.14 Plot of the model evidence versus the order M, for the polynomial regression model, showing that the evidence favours the model with $M = 3$.

for the evidence. Going to the $M = 1$ polynomial greatly improves the data fit, and hence the evidence is significantly higher. However, in going to $M = 2$, the data fit is improved only very marginally, due to the fact that the underlying sinusoidal function from which the data is generated is an odd function and so has no even terms in a polynomial expansion. Indeed, Figure 1.5 shows that the residual data error is reduced only slightly in going from $M = 1$ to $M = 2$. Because this richer model suffers a greater complexity penalty, the evidence actually falls in going from $M = 1$ to $M = 2$. When we go to $M = 3$ we obtain a significant further improvement in data fit, as seen in Figure 1.4, and so the evidence is increased again, giving the highest overall evidence for any of the polynomials. Further increases in the value of M produce only small improvements in the fit to the data but suffer increasing complexity penalty, leading overall to a decrease in the evidence values. Looking again at Figure 1.5, we see that the generalization error is roughly constant between $M = 3$ and $M = 8$, and it would be difficult to choose between these models on the basis of this plot alone. The evidence values, however, show a clear preference for $M = 3$, since this is the simplest model which gives a good explanation for the observed data.

3.5.2 Maximizing the evidence function

Let us first consider the maximization of $p(\mathbf{t}|\alpha, \beta)$ with respect to α. This can be done by first defining the following eigenvector equation

$$\left(\beta \boldsymbol{\Phi}^{\mathrm{T}} \boldsymbol{\Phi}\right) \mathbf{u}_i = \lambda_i \mathbf{u}_i. \tag{3.87}$$

From (3.81), it then follows that $\mathbf{A}$ has eigenvalues $\alpha + \lambda_i$. Now consider the derivative of the term involving $\ln |\mathbf{A}|$ in (3.86) with respect to α. We have

$$\frac{d}{d\alpha} \ln |\mathbf{A}| = \frac{d}{d\alpha} \ln \prod_i (\lambda_i + \alpha) = \frac{d}{d\alpha} \sum_i \ln(\lambda_i + \alpha) = \sum_i \frac{1}{\lambda_i + \alpha}. \tag{3.88}$$

Thus the stationary points of (3.86) with respect to α satisfy

$$0 = \frac{M}{2\alpha} - \frac{1}{2} \mathbf{m}_N^{\mathrm{T}} \mathbf{m}_N - \frac{1}{2} \sum_i \frac{1}{\lambda_i + \alpha}. \tag{3.89}$$

Multiplying through by 2α and rearranging, we obtain

$$\alpha \mathbf{m}_N^{\mathrm{T}} \mathbf{m}_N = M - \alpha \sum_i \frac{1}{\lambda_i + \alpha} = \gamma. \tag{3.90}$$

Since there are M terms in the sum over i, the quantity γ can be written

$$\gamma = \sum_i \frac{\lambda_i}{\alpha + \lambda_i}. \tag{3.91}$$

The interpretation of the quantity γ will be discussed shortly. From (3.90) we see
Exercise 3.20 that the value of α that maximizes the marginal likelihood satisfies

$$\alpha = \frac{\gamma}{\mathbf{m}_N^{\mathrm{T}} \mathbf{m}_N}. \tag{3.92}$$

Note that this is an implicit solution for α not only because γ depends on α, but also because the mode $\mathbf{m}_N$ of the posterior distribution itself depends on the choice of α. We therefore adopt an iterative procedure in which we make an initial choice for α and use this to find $\mathbf{m}_N$, which is given by (3.53), and also to evaluate γ, which is given by (3.91). These values are then used to re-estimate α using (3.92), and the process repeated until convergence. Note that because the matrix $\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}$ is fixed, we can compute its eigenvalues once at the start and then simply multiply these by β to obtain the λ_i.

It should be emphasized that the value of α has been determined purely by looking at the training data. In contrast to maximum likelihood methods, no independent data set is required in order to optimize the model complexity.

We can similarly maximize the log marginal likelihood (3.86) with respect to β. To do this, we note that the eigenvalues λ_i defined by (3.87) are proportional to β, and hence $d\lambda_i/d\beta = \lambda_i/\beta$ giving

$$\frac{d}{d\beta} \ln |\mathbf{A}| = \frac{d}{d\beta} \sum_i \ln(\lambda_i + \alpha) = \frac{1}{\beta} \sum_i \frac{\lambda_i}{\lambda_i + \alpha} = \frac{\gamma}{\beta}. \tag{3.93}$$

The stationary point of the marginal likelihood therefore satisfies

$$0 = \frac{N}{2\beta} - \frac{1}{2} \sum_{n=1}^{N} \left\{ t_n - \mathbf{m}_N^{\mathrm{T}} \boldsymbol{\phi}(\mathbf{x}_n) \right\}^2 - \frac{\gamma}{2\beta} \tag{3.94}$$

Exercise 3.22 and rearranging we obtain

$$\frac{1}{\beta} = \frac{1}{N - \gamma} \sum_{n=1}^{N} \left\{ t_n - \mathbf{m}_N^{\mathrm{T}} \boldsymbol{\phi}(\mathbf{x}_n) \right\}^2. \tag{3.95}$$

Again, this is an implicit solution for β and can be solved by choosing an initial value for β and then using this to calculate $\mathbf{m}_N$ and γ and then re-estimate β using (3.95), repeating until convergence. If both α and β are to be determined from the data, then their values can be re-estimated together after each update of γ.

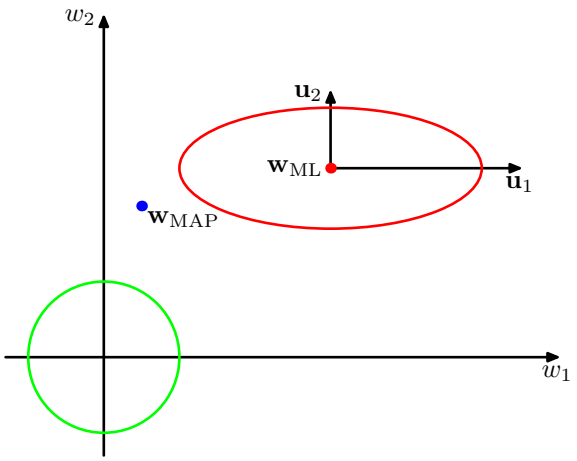


Figure 3.15 Contours of the likelihood function (red) and the prior (green) in which the axes in parameter space have been rotated to align with the eigenvectors $\mathbf{u}_i$ of the Hessian. For $\alpha = 0$, the mode of the posterior is given by the maximum likelihood solution $\mathbf{w}_{\mathrm{ML}}$, whereas for nonzero α the mode is at $\mathbf{w}_{\mathrm{MAP}} = \mathbf{m}_N$. In the direction w_1 the eigenvalue λ_1, defined by (3.87), is small compared with α and so the quantity $\lambda_1/(\lambda_1 + \alpha)$ is close to zero, and the corresponding MAP value of w_1 is also close to zero. By contrast, in the direction w_2 the eigenvalue λ_2 is large compared with α and so the quantity $\lambda_2/(\lambda_2+\alpha)$ is close to unity, and the MAP value of w_2 is close to its maximum likelihood value.

3.5.3 Effective number of parameters

The result (3.92) has an elegant interpretation (MacKay, 1992a), which provides insight into the Bayesian solution for α. To see this, consider the contours of the likelihood function and the prior as illustrated in Figure 3.15. Here we have implicitly transformed to a rotated set of axes in parameter space aligned with the eigenvectors $\mathbf{u}_i$ defined in (3.87). Contours of the likelihood function are then axis-aligned ellipses. The eigenvalues λ_i measure the curvature of the likelihood function, and so in Figure 3.15 the eigenvalue λ_1 is small compared with λ_2 (because a smaller curvature corresponds to a greater elongation of the contours of the likelihood function). Because $\beta\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}$ is a positive definite matrix, it will have positive eigenvalues, and so the ratio $\lambda_i/(\lambda_i + \alpha)$ will lie between 0 and 1. Consequently, the quantity γ defined by (3.91) will lie in the range $0 \leqslant \gamma \leqslant M$. For directions in which $\lambda_i \gg \alpha$, the corresponding parameter w_i will be close to its maximum likelihood value, and the ratio $\lambda_i/(\lambda_i + \alpha)$ will be close to 1. Such parameters are called *well determined* because their values are tightly constrained by the data. Conversely, for directions in which $\lambda_i \ll \alpha$, the corresponding parameters w_i will be close to zero, as will the ratios $\lambda_i/(\lambda_i+\alpha)$. These are directions in which the likelihood function is relatively insensitive to the parameter value and so the parameter has been set to a small value by the prior. The quantity γ defined by (3.91) therefore measures the effective total number of well determined parameters.

We can obtain some insight into the result (3.95) for re-estimating β by comparing it with the corresponding maximum likelihood result given by (3.21). Both of these formulae express the variance (the inverse precision) as an average of the squared differences between the targets and the model predictions. However, they differ in that the number of data points N in the denominator of the maximum likelihood result is replaced by $N - \gamma$ in the Bayesian result. We recall from (1.56) that the maximum likelihood estimate of the variance for a Gaussian distribution over a

single variable x is given by

$$\sigma^2_{\mathrm{ML}} = \frac{1}{N}\sum_{n=1}^{N}(x_n - \mu_{\mathrm{ML}})^2 \tag{3.96}$$

and that this estimate is biased because the maximum likelihood solution μ_{ML} for the mean has fitted some of the noise on the data. In effect, this has used up one degree of freedom in the model. The corresponding unbiased estimate is given by (1.59) and takes the form

$$\sigma^2_{\mathrm{MAP}} = \frac{1}{N-1}\sum_{n=1}^{N}(x_n - \mu_{\mathrm{ML}})^2. \tag{3.97}$$

We shall see in Section 10.1.3 that this result can be obtained from a Bayesian treatment in which we marginalize over the unknown mean. The factor of $N-1$ in the denominator of the Bayesian result takes account of the fact that one degree of freedom has been used in fitting the mean and removes the bias of maximum likelihood. Now consider the corresponding results for the linear regression model. The mean of the target distribution is now given by the function $\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x})$, which contains M parameters. However, not all of these parameters are tuned to the data. The effective number of parameters that are determined by the data is γ, with the remaining $M-\gamma$ parameters set to small values by the prior. This is reflected in the Bayesian result for the variance that has a factor $N-\gamma$ in the denominator, thereby correcting for the bias of the maximum likelihood result.

We can illustrate the evidence framework for setting hyperparameters using the sinusoidal synthetic data set from Section 1.1, together with the Gaussian basis function model comprising 9 basis functions, so that the total number of parameters in the model is given by $M = 10$ including the bias. Here, for simplicity of illustration, we have set β to its true value of 11.1 and then used the evidence framework to determine α, as shown in Figure 3.16.

We can also see how the parameter α controls the magnitude of the parameters $\{w_i\}$, by plotting the individual parameters versus the effective number γ of parameters, as shown in Figure 3.17.

If we consider the limit $N \gg M$ in which the number of data points is large in relation to the number of parameters, then from (3.87) all of the parameters will be well determined by the data because $\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}$ involves an implicit sum over data points, and so the eigenvalues λ_i increase with the size of the data set. In this case, $\gamma = M$, and the re-estimation equations for α and β become

$$\alpha = \frac{M}{2E_W(\mathbf{m}_N)} \tag{3.98}$$

$$\beta = \frac{N}{2E_D(\mathbf{m}_N)} \tag{3.99}$$

where E_W and E_D are defined by (3.25) and (3.26), respectively. These results can be used as an easy-to-compute approximation to the full evidence re-estimation

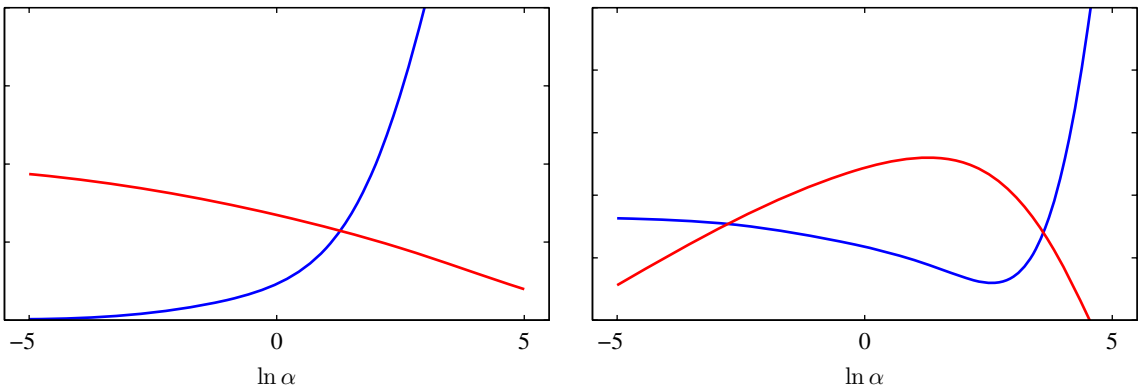
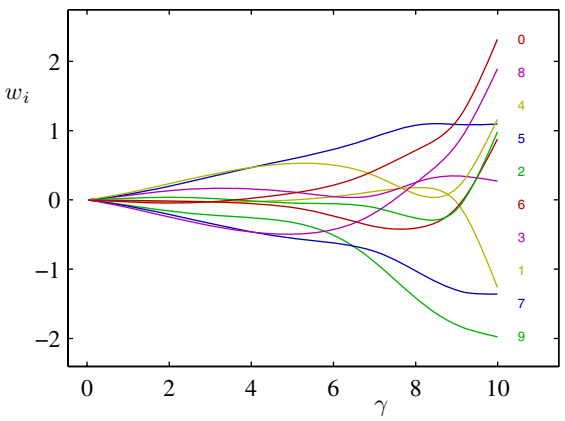


Figure 3.16 The left plot shows γ (red curve) and $2\alpha E_W(\mathbf{m}_N)$ (blue curve) versus $\ln \alpha$ for the sinusoidal synthetic data set. It is the intersection of these two curves that defines the optimum value for α given by the evidence procedure. The right plot shows the corresponding graph of log evidence $\ln p(\mathbf{t}|\alpha, \beta)$ versus $\ln \alpha$ (red curve) showing that the peak coincides with the crossing point of the curves in the left plot. Also shown is the test set error (blue curve) showing that the evidence maximum occurs close to the point of best generalization.

formulae, because they do not require evaluation of the eigenvalue spectrum of the Hessian.

Figure 3.17 Plot of the 10 parameters w_i from the Gaussian basis function model versus the effective number of parameters γ, in which the hyperparameter α is varied in the range $0 \leqslant \alpha \leqslant \infty$ causing γ to vary in the range $0 \leqslant \gamma \leqslant M$.



3.6. Limitations of Fixed Basis Functions

Throughout this chapter, we have focussed on models comprising a linear combination of fixed, nonlinear basis functions. We have seen that the assumption of linearity in the parameters led to a range of useful properties including closed-form solutions to the least-squares problem, as well as a tractable Bayesian treatment. Furthermore, for a suitable choice of basis functions, we can model arbitrary nonlinearities in the

mapping from input variables to targets. In the next chapter, we shall study an analogous class of models for classification.

It might appear, therefore, that such linear models constitute a general purpose framework for solving problems in pattern recognition. Unfortunately, there are some significant shortcomings with linear models, which will cause us to turn in later chapters to more complex models such as support vector machines and neural networks.

The difficulty stems from the assumption that the basis functions $\phi_j(\mathbf{x})$ are fixed before the training data set is observed and is a manifestation of the curse of dimensionality discussed in Section 1.4. As a consequence, the number of basis functions needs to grow rapidly, often exponentially, with the dimensionality D of the input space.

Fortunately, there are two properties of real data sets that we can exploit to help alleviate this problem. First of all, the data vectors $\{\mathbf{x}_n\}$ typically lie close to a nonlinear manifold whose intrinsic dimensionality is smaller than that of the input space as a result of strong correlations between the input variables. We will see an example of this when we consider images of handwritten digits in Chapter 12. If we are using localized basis functions, we can arrange that they are scattered in input space only in regions containing data. This approach is used in radial basis function networks and also in support vector and relevance vector machines. Neural network models, which use adaptive basis functions having sigmoidal nonlinearities, can adapt the parameters so that the regions of input space over which the basis functions vary corresponds to the data manifold. The second property is that target variables may have significant dependence on only a small number of possible directions within the data manifold. Neural networks can exploit this property by choosing the directions in input space to which the basis functions respond.

Exercises

3.1 (⋆) **www** Show that the 'tanh' function and the logistic sigmoid function (3.6) are related by

$$\tanh(a) = 2\sigma(2a) - 1. \tag{3.100}$$

Hence show that a general linear combination of logistic sigmoid functions of the form

$$y(x, \mathbf{w}) = w_0 + \sum_{j=1}^{M} w_j \sigma\left(\frac{x - \mu_j}{s}\right) \tag{3.101}$$

is equivalent to a linear combination of 'tanh' functions of the form

$$y(x, \mathbf{u}) = u_0 + \sum_{j=1}^{M} u_j \tanh\left(\frac{x - \mu_j}{s}\right) \tag{3.102}$$

and find expressions to relate the new parameters $\{u_1, \ldots, u_M\}$ to the original parameters $\{w_1, \ldots, w_M\}$.

3.2 ($\star\star$) Show that the matrix

$$\boldsymbol{\Phi}(\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi})^{-1}\boldsymbol{\Phi}^{\mathrm{T}} \tag{3.103}$$

takes any vector $\mathbf{v}$ and projects it onto the space spanned by the columns of $\boldsymbol{\Phi}$. Use this result to show that the least-squares solution (3.15) corresponds to an orthogonal projection of the vector $\mathbf{t}$ onto the manifold $\mathcal{S}$ as shown in Figure 3.2.

3.3 ($\star$) Consider a data set in which each data point t_n is associated with a weighting factor $r_n > 0$, so that the sum-of-squares error function becomes

$$E_D(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N} r_n \left\{t_n - \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n)\right\}^2. \tag{3.104}$$

Find an expression for the solution $\mathbf{w}^\star$ that minimizes this error function. Give two alternative interpretations of the weighted sum-of-squares error function in terms of (i) data dependent noise variance and (ii) replicated data points.

3.4 ($\star$) **www** Consider a linear model of the form

$$y(x, \mathbf{w}) = w_0 + \sum_{i=1}^{D} w_i x_i \tag{3.105}$$

together with a sum-of-squares error function of the form

$$E_D(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N} \{y(x_n, \mathbf{w}) - t_n\}^2. \tag{3.106}$$

Now suppose that Gaussian noise ϵ_i with zero mean and variance σ^2 is added independently to each of the input variables x_i. By making use of $\mathbb{E}[\epsilon_i] = 0$ and $\mathbb{E}[\epsilon_i\epsilon_j] = \delta_{ij}\sigma^2$, show that minimizing E_D averaged over the noise distribution is equivalent to minimizing the sum-of-squares error for noise-free input variables with the addition of a weight-decay regularization term, in which the bias parameter w_0 is omitted from the regularizer.

3.5 ($\star$) **www** Using the technique of Lagrange multipliers, discussed in Appendix E, show that minimization of the regularized error function (3.29) is equivalent to minimizing the unregularized sum-of-squares error (3.12) subject to the constraint (3.30). Discuss the relationship between the parameters η and λ.

3.6 ($\star$) **www** Consider a linear basis function regression model for a multivariate target variable $\mathbf{t}$ having a Gaussian distribution of the form

$$p(\mathbf{t}|\mathbf{W}, \boldsymbol{\Sigma}) = \mathcal{N}(\mathbf{t}|\mathbf{y}(\mathbf{x}, \mathbf{W}), \boldsymbol{\Sigma}) \tag{3.107}$$

where

$$\mathbf{y}(\mathbf{x}, \mathbf{W}) = \mathbf{W}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}) \tag{3.108}$$

together with a training data set comprising input basis vectors $\boldsymbol{\phi}(\mathbf{x}_n)$ and corresponding target vectors $\mathbf{t}_n$, with $n = 1, \ldots, N$. Show that the maximum likelihood solution $\mathbf{W}_{\mathrm{ML}}$ for the parameter matrix $\mathbf{W}$ has the property that each column is given by an expression of the form (3.15), which was the solution for an isotropic noise distribution. Note that this is independent of the covariance matrix $\boldsymbol{\Sigma}$. Show that the maximum likelihood solution for $\boldsymbol{\Sigma}$ is given by

$$\boldsymbol{\Sigma} = \frac{1}{N}\sum_{n=1}^{N}\left(\mathbf{t}_n - \mathbf{W}_{\mathrm{ML}}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n)\right)\left(\mathbf{t}_n - \mathbf{W}_{\mathrm{ML}}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n)\right)^{\mathrm{T}}. \tag{3.109}$$

3.7 ($\star$) By using the technique of completing the square, verify the result (3.49) for the posterior distribution of the parameters $\mathbf{w}$ in the linear basis function model in which $\mathbf{m}_N$ and $\mathbf{S}_N$ are defined by (3.50) and (3.51) respectively.

3.8 ($\star\star$) **www** Consider the linear basis function model in Section 3.1, and suppose that we have already observed N data points, so that the posterior distribution over $\mathbf{w}$ is given by (3.49). This posterior can be regarded as the prior for the next observation. By considering an additional data point $(\mathbf{x}_{N+1}, t_{N+1})$, and by completing the square in the exponential, show that the resulting posterior distribution is again given by (3.49) but with $\mathbf{S}_N$ replaced by $\mathbf{S}_{N+1}$ and $\mathbf{m}_N$ replaced by $\mathbf{m}_{N+1}$.

3.9 ($\star\star$) Repeat the previous exercise but instead of completing the square by hand, make use of the general result for linear-Gaussian models given by (2.116).

3.10 ($\star\star$) **www** By making use of the result (2.115) to evaluate the integral in (3.57), verify that the predictive distribution for the Bayesian linear regression model is given by (3.58) in which the input-dependent variance is given by (3.59).

3.11 ($\star\star$) We have seen that, as the size of a data set increases, the uncertainty associated with the posterior distribution over model parameters decreases. Make use of the matrix identity (Appendix C)

$$\left(\mathbf{M} + \mathbf{v}\mathbf{v}^{\mathrm{T}}\right)^{-1} = \mathbf{M}^{-1} - \frac{\left(\mathbf{M}^{-1}\mathbf{v}\right)\left(\mathbf{v}^{\mathrm{T}}\mathbf{M}^{-1}\right)}{1 + \mathbf{v}^{\mathrm{T}}\mathbf{M}^{-1}\mathbf{v}} \tag{3.110}$$

to show that the uncertainty $\sigma_N^2(\mathbf{x})$ associated with the linear regression function given by (3.59) satisfies

$$\sigma_{N+1}^2(\mathbf{x}) \leqslant \sigma_N^2(\mathbf{x}). \tag{3.111}$$

3.12 ($\star\star$) We saw in Section 2.3.6 that the conjugate prior for a Gaussian distribution with unknown mean and unknown precision (inverse variance) is a normal-gamma distribution. This property also holds for the case of the conditional Gaussian distribution $p(t|\mathbf{x}, \mathbf{w}, \beta)$ of the linear regression model. If we consider the likelihood function (3.10), then the conjugate prior for $\mathbf{w}$ and β is given by

$$p(\mathbf{w}, \beta) = \mathcal{N}(\mathbf{w}|\mathbf{m}_0, \beta^{-1}\mathbf{S}_0)\mathrm{Gam}(\beta|a_0, b_0). \tag{3.112}$$

Show that the corresponding posterior distribution takes the same functional form, so that

$$p(\mathbf{w}, \beta|\mathbf{t}) = \mathcal{N}(\mathbf{w}|\mathbf{m}_N, \beta^{-1}\mathbf{S}_N)\mathrm{Gam}(\beta|a_N, b_N) \tag{3.113}$$

and find expressions for the posterior parameters $\mathbf{m}_N$, $\mathbf{S}_N$, a_N, and b_N.

3.13 ($\star\star$) Show that the predictive distribution $p(t|\mathbf{x}, \mathbf{t})$ for the model discussed in Exercise 3.12 is given by a Student's t-distribution of the form

$$p(t|\mathbf{x}, \mathbf{t}) = \mathrm{St}(t|\mu, \lambda, \nu) \tag{3.114}$$

and obtain expressions for μ, λ and ν.

3.14 ($\star\star$) In this exercise, we explore in more detail the properties of the equivalent kernel defined by (3.62), where $\mathbf{S}_N$ is defined by (3.54). Suppose that the basis functions $\phi_j(\mathbf{x})$ are linearly independent and that the number N of data points is greater than the number M of basis functions. Furthermore, let one of the basis functions be constant, say $\phi_0(\mathbf{x}) = 1$. By taking suitable linear combinations of these basis functions, we can construct a new basis set $\psi_j(\mathbf{x})$ spanning the same space but that are orthonormal, so that

$$\sum_{n=1}^{N} \psi_j(\mathbf{x}_n)\psi_k(\mathbf{x}_n) = I_{jk} \tag{3.115}$$

where I_{jk} is defined to be 1 if $j = k$ and 0 otherwise, and we take $\psi_0(\mathbf{x}) = 1$. Show that for $\alpha = 0$, the equivalent kernel can be written as $k(\mathbf{x}, \mathbf{x}') = \boldsymbol{\psi}(\mathbf{x})^{\mathrm{T}}\boldsymbol{\psi}(\mathbf{x}')$ where $\boldsymbol{\psi} = (\psi_1, \ldots, \psi_M)^{\mathrm{T}}$. Use this result to show that the kernel satisfies the summation constraint

$$\sum_{n=1}^{N} k(\mathbf{x}, \mathbf{x}_n) = 1. \tag{3.116}$$

3.15 ($\star$) **www** Consider a linear basis function model for regression in which the parameters α and β are set using the evidence framework. Show that the function $E(\mathbf{m}_N)$ defined by (3.82) satisfies the relation $2E(\mathbf{m}_N) = N$.

3.16 ($\star\star$) Derive the result (3.86) for the log evidence function $p(\mathbf{t}|\alpha, \beta)$ of the linear regression model by making use of (2.115) to evaluate the integral (3.77) directly.

3.17 ($\star$) Show that the evidence function for the Bayesian linear regression model can be written in the form (3.78) in which $E(\mathbf{w})$ is defined by (3.79).

3.18 ($\star\star$) **www** By completing the square over $\mathbf{w}$, show that the error function (3.79) in Bayesian linear regression can be written in the form (3.80).

3.19 ($\star\star$) Show that the integration over $\mathbf{w}$ in the Bayesian linear regression model gives the result (3.85). Hence show that the log marginal likelihood is given by (3.86).

3.20 (⋆⋆) **www** Starting from (3.86) verify all of the steps needed to show that maximization of the log marginal likelihood function (3.86) with respect to α leads to the re-estimation equation (3.92).

3.21 (⋆⋆) An alternative way to derive the result (3.92) for the optimal value of α in the evidence framework is to make use of the identity

$$\frac{d}{d\alpha}\ln|\mathbf{A}| = \mathrm{Tr}\left(\mathbf{A}^{-1}\frac{d}{d\alpha}\mathbf{A}\right). \tag{3.117}$$

Prove this identity by considering the eigenvalue expansion of a real, symmetric matrix $\mathbf{A}$, and making use of the standard results for the determinant and trace of $\mathbf{A}$ expressed in terms of its eigenvalues (Appendix C). Then make use of (3.117) to derive (3.92) starting from (3.86).

3.22 (⋆⋆) Starting from (3.86) verify all of the steps needed to show that maximization of the log marginal likelihood function (3.86) with respect to β leads to the re-estimation equation (3.95).

3.23 (⋆⋆) **www** Show that the marginal probability of the data, in other words the model evidence, for the model described in Exercise 3.12 is given by

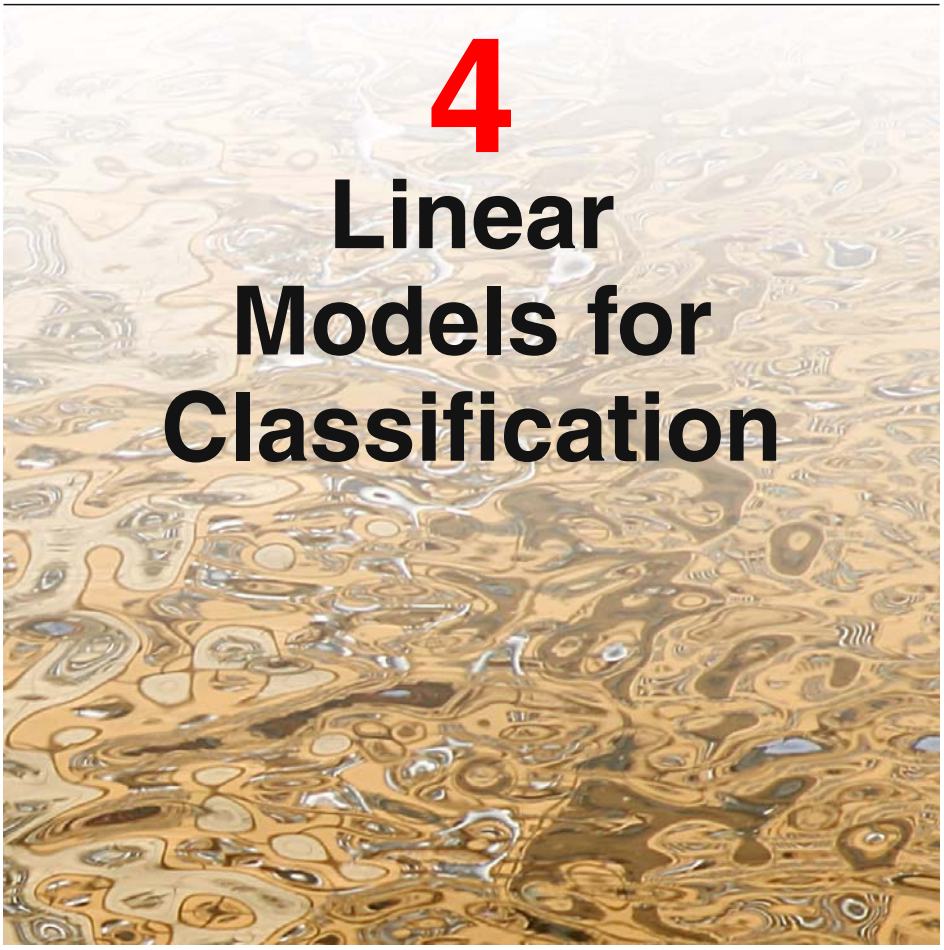
$$p(\mathbf{t}) = \frac{1}{(2\pi)^{N/2}}\frac{b_0^{a_0}}{b_N^{a_N}}\frac{\Gamma(a_N)}{\Gamma(a_0)}\frac{|\mathbf{S}_N|^{1/2}}{|\mathbf{S}_0|^{1/2}} \tag{3.118}$$

by first marginalizing with respect to $\mathbf{w}$ and then with respect to β.

3.24 (⋆⋆) Repeat the previous exercise but now use Bayes' theorem in the form

$$p(\mathbf{t}) = \frac{p(\mathbf{t}|\mathbf{w},\beta)p(\mathbf{w},\beta)}{p(\mathbf{w},\beta|\mathbf{t})} \tag{3.119}$$

and then substitute for the prior and posterior distributions and the likelihood function in order to derive the result (3.118).



4 Linear Models for Classification

In the previous chapter, we explored a class of regression models having particularly simple analytical and computational properties. We now discuss an analogous class of models for solving classification problems. The goal in classification is to take an input vector $\mathbf{x}$ and to assign it to one of K discrete classes $\mathcal{C}_k$ where $k = 1, \ldots, K$. In the most common scenario, the classes are taken to be disjoint, so that each input is assigned to one and only one class. The input space is thereby divided into *decision regions* whose boundaries are called *decision boundaries* or *decision surfaces*. In this chapter, we consider linear models for classification, by which we mean that the decision surfaces are linear functions of the input vector $\mathbf{x}$ and hence are defined by $(D - 1)$-dimensional hyperplanes within the D-dimensional input space. Data sets whose classes can be separated exactly by linear decision surfaces are said to be *linearly separable*.

For regression problems, the target variable $\mathbf{t}$ was simply the vector of real numbers whose values we wish to predict. In the case of classification, there are various

ways of using target values to represent class labels. For probabilistic models, the most convenient, in the case of two-class problems, is the binary representation in which there is a single target variable $t \in \{0, 1\}$ such that $t = 1$ represents class $\mathcal{C}_1$ and $t = 0$ represents class $\mathcal{C}_2$. We can interpret the value of t as the probability that the class is $\mathcal{C}_1$, with the values of probability taking only the extreme values of 0 and 1. For $K > 2$ classes, it is convenient to use a 1-of-K coding scheme in which $\mathbf{t}$ is a vector of length K such that if the class is $\mathcal{C}_j$, then all elements t_k of $\mathbf{t}$ are zero except element t_j, which takes the value 1. For instance, if we have $K = 5$ classes, then a pattern from class 2 would be given the target vector

$$\mathbf{t} = (0, 1, 0, 0, 0)^{\mathrm{T}}. \tag{4.1}$$

Again, we can interpret the value of t_k as the probability that the class is $\mathcal{C}_k$. For nonprobabilistic models, alternative choices of target variable representation will sometimes prove convenient.

In Chapter 1, we identified three distinct approaches to the classification problem. The simplest involves constructing a *discriminant function* that directly assigns each vector $\mathbf{x}$ to a specific class. A more powerful approach, however, models the conditional probability distribution $p(\mathcal{C}_k|\mathbf{x})$ in an inference stage, and then subsequently uses this distribution to make optimal decisions. By separating inference and decision, we gain numerous benefits, as discussed in Section 1.5.4. There are two different approaches to determining the conditional probabilities $p(\mathcal{C}_k|\mathbf{x})$. One technique is to model them directly, for example by representing them as parametric models and then optimizing the parameters using a training set. Alternatively, we can adopt a generative approach in which we model the class-conditional densities given by $p(\mathbf{x}|\mathcal{C}_k)$, together with the prior probabilities $p(\mathcal{C}_k)$ for the classes, and then we compute the required posterior probabilities using Bayes' theorem

$$p(\mathcal{C}_k|\mathbf{x}) = \frac{p(\mathbf{x}|\mathcal{C}_k)p(\mathcal{C}_k)}{p(\mathbf{x})}. \tag{4.2}$$

We shall discuss examples of all three approaches in this chapter.

In the linear regression models considered in Chapter 3, the model prediction $y(\mathbf{x}, \mathbf{w})$ was given by a linear function of the parameters $\mathbf{w}$. In the simplest case, the model is also linear in the input variables and therefore takes the form $y(\mathbf{x}) = \mathbf{w}^{\mathrm{T}}\mathbf{x} + w_0$, so that y is a real number. For classification problems, however, we wish to predict discrete class labels, or more generally posterior probabilities that lie in the range $(0, 1)$. To achieve this, we consider a generalization of this model in which we transform the linear function of $\mathbf{w}$ using a nonlinear function $f(\,\cdot\,)$ so that

$$y(\mathbf{x}) = f\left(\mathbf{w}^{\mathrm{T}}\mathbf{x} + w_0\right). \tag{4.3}$$

In the machine learning literature $f(\,\cdot\,)$ is known as an *activation function*, whereas its inverse is called a *link function* in the statistics literature. The decision surfaces correspond to $y(\mathbf{x}) = \text{constant}$, so that $\mathbf{w}^{\mathrm{T}}\mathbf{x} + w_0 = \text{constant}$ and hence the decision surfaces are linear functions of $\mathbf{x}$, even if the function $f(\cdot)$ is nonlinear. For this reason, the class of models described by (4.3) are called *generalized linear models*

(McCullagh and Nelder, 1989). Note, however, that in contrast to the models used for regression, they are no longer linear in the parameters due to the presence of the nonlinear function $f(\cdot)$. This will lead to more complex analytical and computational properties than for linear regression models. Nevertheless, these models are still relatively simple compared to the more general nonlinear models that will be studied in subsequent chapters.

The algorithms discussed in this chapter will be equally applicable if we first make a fixed nonlinear transformation of the input variables using a vector of basis functions $\boldsymbol{\phi}(\mathbf{x})$ as we did for regression models in Chapter 3. We begin by considering classification directly in the original input space $\mathbf{x}$, while in Section 4.3 we shall find it convenient to switch to a notation involving basis functions for consistency with later chapters.

4.1. Discriminant Functions

A discriminant is a function that takes an input vector $\mathbf{x}$ and assigns it to one of K classes, denoted $\mathcal{C}_k$. In this chapter, we shall restrict attention to *linear discriminants*, namely those for which the decision surfaces are hyperplanes. To simplify the discussion, we consider first the case of two classes and then investigate the extension to $K > 2$ classes.

4.1.1 Two classes

The simplest representation of a linear discriminant function is obtained by taking a linear function of the input vector so that

$$y(\mathbf{x}) = \mathbf{w}^{\mathrm{T}}\mathbf{x} + w_0 \tag{4.4}$$

where $\mathbf{w}$ is called a *weight vector*, and w_0 is a *bias* (not to be confused with bias in the statistical sense). The negative of the bias is sometimes called a *threshold*. An input vector $\mathbf{x}$ is assigned to class $\mathcal{C}_1$ if $y(\mathbf{x}) \geqslant 0$ and to class $\mathcal{C}_2$ otherwise. The corresponding decision boundary is therefore defined by the relation $y(\mathbf{x}) = 0$, which corresponds to a $(D-1)$-dimensional hyperplane within the D-dimensional input space. Consider two points $\mathbf{x}_{\mathrm{A}}$ and $\mathbf{x}_{\mathrm{B}}$ both of which lie on the decision surface. Because $y(\mathbf{x}_{\mathrm{A}}) = y(\mathbf{x}_{\mathrm{B}}) = 0$, we have $\mathbf{w}^{\mathrm{T}}(\mathbf{x}_{\mathrm{A}} - \mathbf{x}_{\mathrm{B}}) = 0$ and hence the vector $\mathbf{w}$ is orthogonal to every vector lying within the decision surface, and so $\mathbf{w}$ determines the orientation of the decision surface. Similarly, if $\mathbf{x}$ is a point on the decision surface, then $y(\mathbf{x}) = 0$, and so the normal distance from the origin to the decision surface is given by

$$\frac{\mathbf{w}^{\mathrm{T}}\mathbf{x}}{\|\mathbf{w}\|} = -\frac{w_0}{\|\mathbf{w}\|}. \tag{4.5}$$

We therefore see that the bias parameter w_0 determines the location of the decision surface. These properties are illustrated for the case of $D = 2$ in Figure 4.1.

Furthermore, we note that the value of $y(\mathbf{x})$ gives a signed measure of the perpendicular distance r of the point $\mathbf{x}$ from the decision surface. To see this, consider

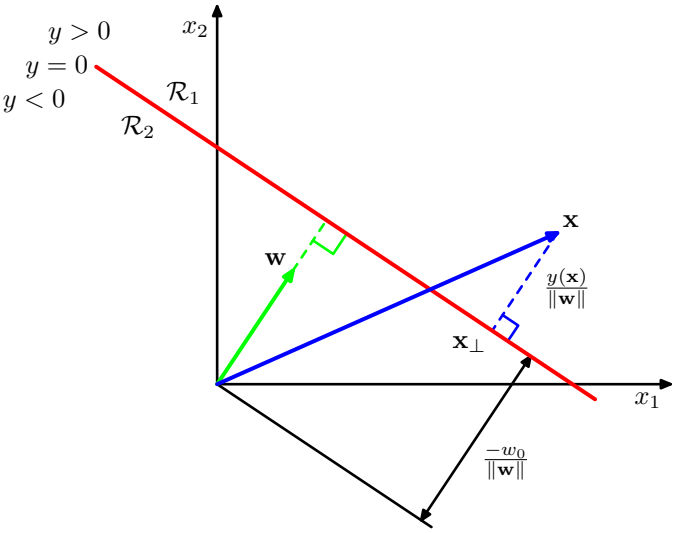


Figure 4.1 Illustration of the geometry of a linear discriminant function in two dimensions. The decision surface, shown in red, is perpendicular to $\mathbf{w}$, and its displacement from the origin is controlled by the bias parameter w_0. Also, the signed orthogonal distance of a general point $\mathbf{x}$ from the decision surface is given by $y(\mathbf{x})/\|\mathbf{w}\|$.

an arbitrary point $\mathbf{x}$ and let $\mathbf{x}_\perp$ be its orthogonal projection onto the decision surface, so that

$$\mathbf{x} = \mathbf{x}_\perp + r\frac{\mathbf{w}}{\|\mathbf{w}\|}. \tag{4.6}$$

Multiplying both sides of this result by $\mathbf{w}^{\mathrm{T}}$ and adding w_0, and making use of $y(\mathbf{x}) = \mathbf{w}^{\mathrm{T}}\mathbf{x} + w_0$ and $y(\mathbf{x}_\perp) = \mathbf{w}^{\mathrm{T}}\mathbf{x}_\perp + w_0 = 0$, we have

$$r = \frac{y(\mathbf{x})}{\|\mathbf{w}\|}. \tag{4.7}$$

This result is illustrated in Figure 4.1.

As with the linear regression models in Chapter 3, it is sometimes convenient to use a more compact notation in which we introduce an additional dummy 'input' value $x_0 = 1$ and then define $\widetilde{\mathbf{w}} = (w_0, \mathbf{w})$ and $\widetilde{\mathbf{x}} = (x_0, \mathbf{x})$ so that

$$y(\mathbf{x}) = \widetilde{\mathbf{w}}^{\mathrm{T}}\widetilde{\mathbf{x}}. \tag{4.8}$$

In this case, the decision surfaces are D-dimensional hyperplanes passing through the origin of the $D+1$-dimensional expanded input space.

4.1.2 Multiple classes

Now consider the extension of linear discriminants to $K > 2$ classes. We might be tempted be to build a K-class discriminant by combining a number of two-class discriminant functions. However, this leads to some serious difficulties (Duda and Hart, 1973) as we now show.

Consider the use of $K-1$ classifiers each of which solves a two-class problem of separating points in a particular class $\mathcal{C}_k$ from points not in that class. This is known as a *one-versus-the-rest* classifier. The left-hand example in Figure 4.2 shows an

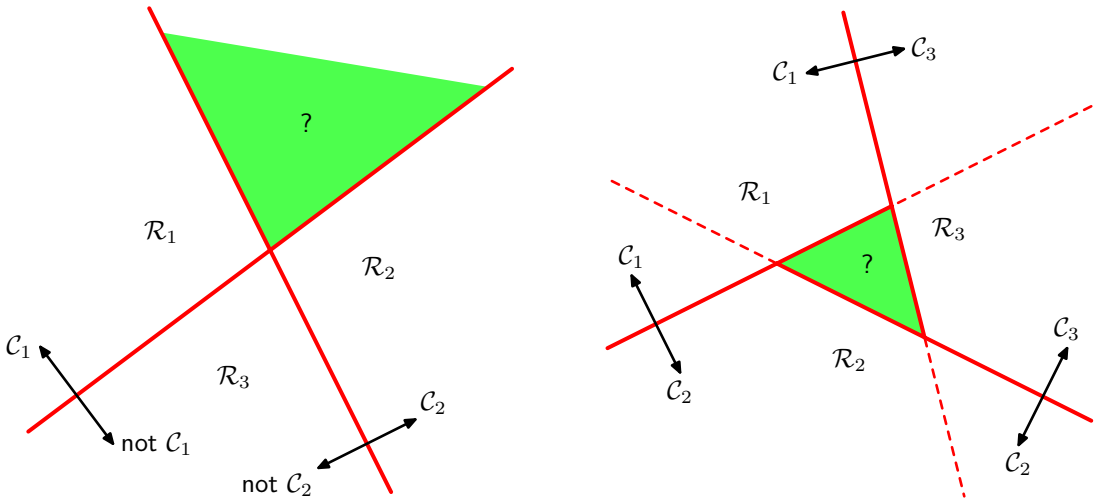


Figure 4.2 Attempting to construct a K class discriminant from a set of two class discriminants leads to ambiguous regions, shown in green. On the left is an example involving the use of two discriminants designed to distinguish points in class $\mathcal{C}_k$ from points not in class $\mathcal{C}_k$. On the right is an example involving three discriminant functions each of which is used to separate a pair of classes $\mathcal{C}_k$ and $\mathcal{C}_j$.

example involving three classes where this approach leads to regions of input space that are ambiguously classified.

An alternative is to introduce $K(K-1)/2$ binary discriminant functions, one for every possible pair of classes. This is known as a *one-versus-one* classifier. Each point is then classified according to a majority vote amongst the discriminant functions. However, this too runs into the problem of ambiguous regions, as illustrated in the right-hand diagram of Figure 4.2.

We can avoid these difficulties by considering a single K-class discriminant comprising K linear functions of the form

$$y_k(\mathbf{x}) = \mathbf{w}_k^{\mathrm{T}}\mathbf{x} + w_{k0} \tag{4.9}$$

and then assigning a point $\mathbf{x}$ to class $\mathcal{C}_k$ if $y_k(\mathbf{x}) > y_j(\mathbf{x})$ for all $j \neq k$. The decision boundary between class $\mathcal{C}_k$ and class $\mathcal{C}_j$ is therefore given by $y_k(\mathbf{x}) = y_j(\mathbf{x})$ and hence corresponds to a $(D-1)$-dimensional hyperplane defined by

$$(\mathbf{w}_k - \mathbf{w}_j)^{\mathrm{T}}\mathbf{x} + (w_{k0} - w_{j0}) = 0. \tag{4.10}$$

This has the same form as the decision boundary for the two-class case discussed in Section 4.1.1, and so analogous geometrical properties apply.

The decision regions of such a discriminant are always singly connected and convex. To see this, consider two points $\mathbf{x}_{\mathrm{A}}$ and $\mathbf{x}_{\mathrm{B}}$ both of which lie inside decision region $\mathcal{R}_k$, as illustrated in Figure 4.3. Any point $\widehat{\mathbf{x}}$ that lies on the line connecting $\mathbf{x}_{\mathrm{A}}$ and $\mathbf{x}_{\mathrm{B}}$ can be expressed in the form

$$\widehat{\mathbf{x}} = \lambda\mathbf{x}_{\mathrm{A}} + (1-\lambda)\mathbf{x}_{\mathrm{B}} \tag{4.11}$$

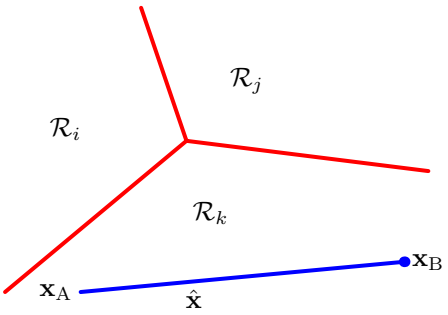


Figure 4.3 Illustration of the decision regions for a multiclass linear discriminant, with the decision boundaries shown in red. If two points $\mathbf{x}_\mathrm{A}$ and $\mathbf{x}_\mathrm{B}$ both lie inside the same decision region $\mathcal{R}_k$, then any point $\widehat{\mathbf{x}}$ that lies on the line connecting these two points must also lie in $\mathcal{R}_k$, and hence the decision region must be singly connected and convex.

where $0 \leqslant \lambda \leqslant 1$. From the linearity of the discriminant functions, it follows that

$$y_k(\widehat{\mathbf{x}}) = \lambda y_k(\mathbf{x}_\mathrm{A}) + (1-\lambda) y_k(\mathbf{x}_\mathrm{B}). \tag{4.12}$$

Because both $\mathbf{x}_\mathrm{A}$ and $\mathbf{x}_\mathrm{B}$ lie inside $\mathcal{R}_k$, it follows that $y_k(\mathbf{x}_\mathrm{A}) > y_j(\mathbf{x}_\mathrm{A})$, and $y_k(\mathbf{x}_\mathrm{B}) > y_j(\mathbf{x}_\mathrm{B})$, for all $j \neq k$, and hence $y_k(\widehat{\mathbf{x}}) > y_j(\widehat{\mathbf{x}})$, and so $\widehat{\mathbf{x}}$ also lies inside $\mathcal{R}_k$. Thus $\mathcal{R}_k$ is singly connected and convex.

Note that for two classes, we can either employ the formalism discussed here, based on two discriminant functions $y_1(\mathbf{x})$ and $y_2(\mathbf{x})$, or else use the simpler but equivalent formulation described in Section 4.1.1 based on a single discriminant function $y(\mathbf{x})$.

We now explore three approaches to learning the parameters of linear discriminant functions, based on least squares, Fisher's linear discriminant, and the perceptron algorithm.

4.1.3 Least squares for classification

In Chapter 3, we considered models that were linear functions of the parameters, and we saw that the minimization of a sum-of-squares error function led to a simple closed-form solution for the parameter values. It is therefore tempting to see if we can apply the same formalism to classification problems. Consider a general classification problem with K classes, with a 1-of-K binary coding scheme for the target vector $\mathbf{t}$. One justification for using least squares in such a context is that it approximates the conditional expectation $\mathbb{E}[\mathbf{t}|\mathbf{x}]$ of the target values given the input vector. For the binary coding scheme, this conditional expectation is given by the vector of posterior class probabilities. Unfortunately, however, these probabilities are typically approximated rather poorly, indeed the approximations can have values outside the range $(0, 1)$, due to the limited flexibility of a linear model as we shall see shortly.

Each class $\mathcal{C}_k$ is described by its own linear model so that

$$y_k(\mathbf{x}) = \mathbf{w}_k^\mathrm{T}\mathbf{x} + w_{k0} \tag{4.13}$$

where $k = 1, \ldots, K$. We can conveniently group these together using vector notation so that

$$\mathbf{y}(\mathbf{x}) = \widetilde{\mathbf{W}}^\mathrm{T}\widetilde{\mathbf{x}} \tag{4.14}$$

where $\widetilde{\mathbf{W}}$ is a matrix whose k^{th} column comprises the $D+1$-dimensional vector $\widetilde{\mathbf{w}}_k = (w_{k0}, \mathbf{w}_k^{\mathrm{T}})^{\mathrm{T}}$ and $\widetilde{\mathbf{x}}$ is the corresponding augmented input vector $(1, \mathbf{x}^{\mathrm{T}})^{\mathrm{T}}$ with a dummy input $x_0 = 1$. This representation was discussed in detail in Section 3.1. A new input $\mathbf{x}$ is then assigned to the class for which the output $y_k = \widetilde{\mathbf{w}}_k^{\mathrm{T}}\widetilde{\mathbf{x}}$ is largest.

We now determine the parameter matrix $\widetilde{\mathbf{W}}$ by minimizing a sum-of-squares error function, as we did for regression in Chapter 3. Consider a training data set $\{\mathbf{x}_n, \mathbf{t}_n\}$ where $n = 1, \ldots, N$, and define a matrix $\mathbf{T}$ whose n^{th} row is the vector $\mathbf{t}_n^{\mathrm{T}}$, together with a matrix $\widetilde{\mathbf{X}}$ whose n^{th} row is $\widetilde{\mathbf{x}}_n^{\mathrm{T}}$. The sum-of-squares error function can then be written as

$$E_D(\widetilde{\mathbf{W}}) = \frac{1}{2}\mathrm{Tr}\left\{(\widetilde{\mathbf{X}}\widetilde{\mathbf{W}} - \mathbf{T})^{\mathrm{T}}(\widetilde{\mathbf{X}}\widetilde{\mathbf{W}} - \mathbf{T})\right\}. \tag{4.15}$$

Setting the derivative with respect to $\widetilde{\mathbf{W}}$ to zero, and rearranging, we then obtain the solution for $\widetilde{\mathbf{W}}$ in the form

$$\widetilde{\mathbf{W}} = (\widetilde{\mathbf{X}}^{\mathrm{T}}\widetilde{\mathbf{X}})^{-1}\widetilde{\mathbf{X}}^{\mathrm{T}}\mathbf{T} = \widetilde{\mathbf{X}}^{\dagger}\mathbf{T} \tag{4.16}$$

where $\widetilde{\mathbf{X}}^{\dagger}$ is the pseudo-inverse of the matrix $\widetilde{\mathbf{X}}$, as discussed in Section 3.1.1. We then obtain the discriminant function in the form

$$\mathbf{y}(\mathbf{x}) = \widetilde{\mathbf{W}}^{\mathrm{T}}\widetilde{\mathbf{x}} = \mathbf{T}^{\mathrm{T}}\left(\widetilde{\mathbf{X}}^{\dagger}\right)^{\mathrm{T}}\widetilde{\mathbf{x}}. \tag{4.17}$$

An interesting property of least-squares solutions with multiple target variables is that if every target vector in the training set satisfies some linear constraint

$$\mathbf{a}^{\mathrm{T}}\mathbf{t}_n + b = 0 \tag{4.18}$$

for some constants $\mathbf{a}$ and b, then the model prediction for any value of $\mathbf{x}$ will satisfy the same constraint so that

Exercise 4.2

$$\mathbf{a}^{\mathrm{T}}\mathbf{y}(\mathbf{x}) + b = 0. \tag{4.19}$$

Thus if we use a 1-of-K coding scheme for K classes, then the predictions made by the model will have the property that the elements of $\mathbf{y}(\mathbf{x})$ will sum to 1 for any value of $\mathbf{x}$. However, this summation constraint alone is not sufficient to allow the model outputs to be interpreted as probabilities because they are not constrained to lie within the interval $(0, 1)$.

The least-squares approach gives an exact closed-form solution for the discriminant function parameters. However, even as a discriminant function (where we use it to make decisions directly and dispense with any probabilistic interpretation) it suffers from some severe problems. We have already seen that least-squares solutions lack robustness to outliers, and this applies equally to the classification application, as illustrated in Figure 4.4. Here we see that the additional data points in the right-hand figure produce a significant change in the location of the decision boundary, even though these point would be correctly classified by the original decision boundary in the left-hand figure. The sum-of-squares error function penalizes predictions that are 'too correct' in that they lie a long way on the correct side of the decision

Section 2.3.7

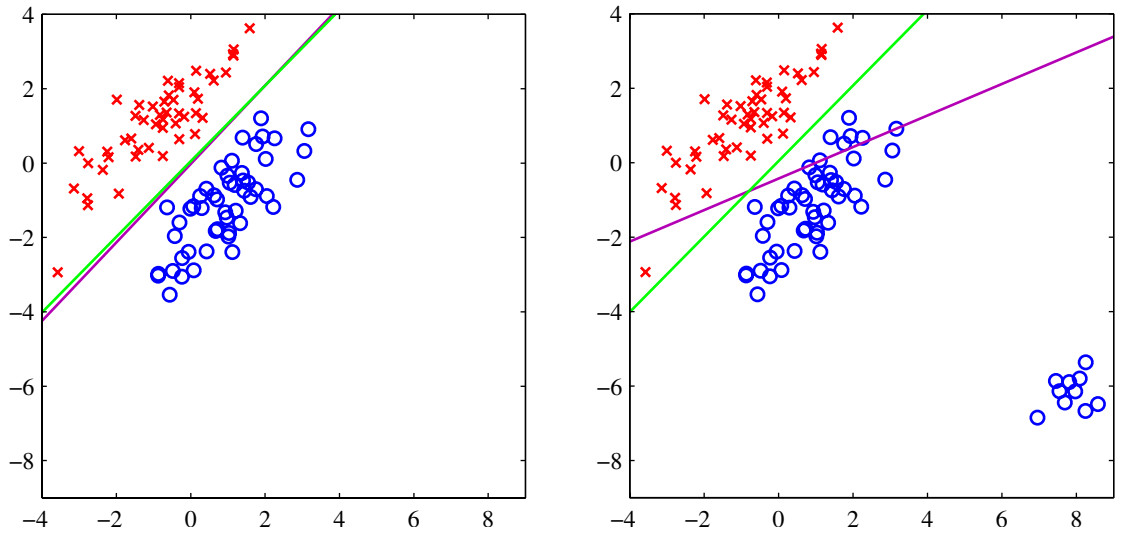


Figure 4.4 The left plot shows data from two classes, denoted by red crosses and blue circles, together with the decision boundary found by least squares (magenta curve) and also by the logistic regression model (green curve), which is discussed later in Section 4.3.2. The right-hand plot shows the corresponding results obtained when extra data points are added at the bottom left of the diagram, showing that least squares is highly sensitive to outliers, unlike logistic regression.

boundary. In Section 7.1.2, we shall consider several alternative error functions for classification and we shall see that they do not suffer from this difficulty.

However, problems with least squares can be more severe than simply lack of robustness, as illustrated in Figure 4.5. This shows a synthetic data set drawn from three classes in a two-dimensional input space (x_1, x_2), having the property that linear decision boundaries can give excellent separation between the classes. Indeed, the technique of logistic regression, described later in this chapter, gives a satisfactory solution as seen in the right-hand plot. However, the least-squares solution gives poor results, with only a small region of the input space assigned to the green class.

The failure of least squares should not surprise us when we recall that it corresponds to maximum likelihood under the assumption of a Gaussian conditional distribution, whereas binary target vectors clearly have a distribution that is far from Gaussian. By adopting more appropriate probabilistic models, we shall obtain classification techniques with much better properties than least squares. For the moment, however, we continue to explore alternative nonprobabilistic methods for setting the parameters in the linear classification models.

4.1.4 Fisher's linear discriminant

One way to view a linear classification model is in terms of dimensionality reduction. Consider first the case of two classes, and suppose we take the D-

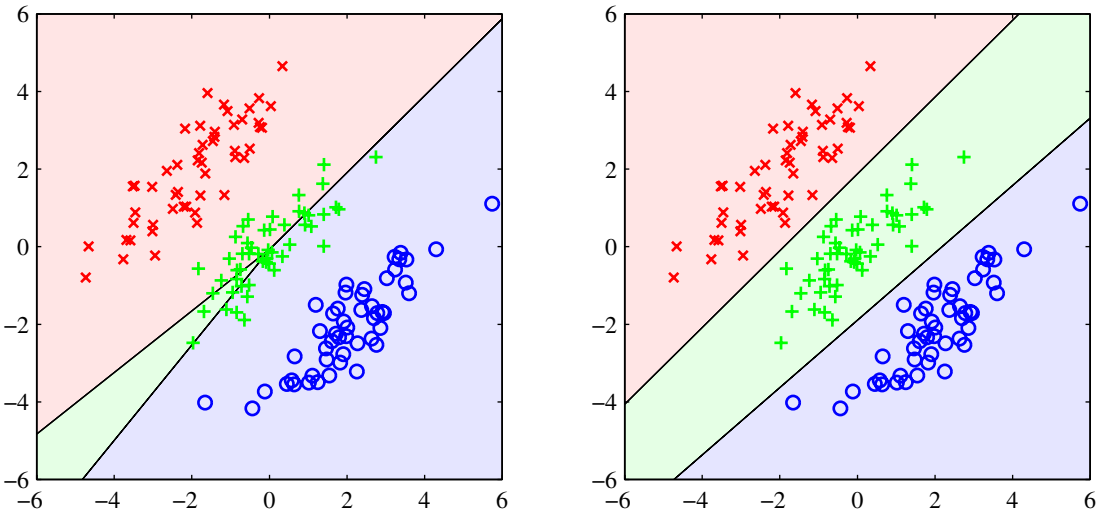


Figure 4.5 Example of a synthetic data set comprising three classes, with training data points denoted in red ($\times$), green ($+$), and blue ($\circ$). Lines denote the decision boundaries, and the background colours denote the respective classes of the decision regions. On the left is the result of using a least-squares discriminant. We see that the region of input space assigned to the green class is too small and so most of the points from this class are misclassified. On the right is the result of using logistic regressions as described in Section 4.3.2 showing correct classification of the training data.

dimensional input vector $\mathbf{x}$ and project it down to one dimension using

$$y = \mathbf{w}^{\mathrm{T}}\mathbf{x}. \tag{4.20}$$

If we place a threshold on y and classify $y \geqslant -w_0$ as class $\mathcal{C}_1$, and otherwise class $\mathcal{C}_2$, then we obtain our standard linear classifier discussed in the previous section. In general, the projection onto one dimension leads to a considerable loss of information, and classes that are well separated in the original D-dimensional space may become strongly overlapping in one dimension. However, by adjusting the components of the weight vector $\mathbf{w}$, we can select a projection that maximizes the class separation. To begin with, consider a two-class problem in which there are N_1 points of class $\mathcal{C}_1$ and N_2 points of class $\mathcal{C}_2$, so that the mean vectors of the two classes are given by

$$\mathbf{m}_1 = \frac{1}{N_1}\sum_{n \in \mathcal{C}_1} \mathbf{x}_n, \qquad \mathbf{m}_2 = \frac{1}{N_2}\sum_{n \in \mathcal{C}_2} \mathbf{x}_n. \tag{4.21}$$

The simplest measure of the separation of the classes, when projected onto $\mathbf{w}$, is the separation of the projected class means. This suggests that we might choose $\mathbf{w}$ so as to maximize

$$m_2 - m_1 = \mathbf{w}^{\mathrm{T}}(\mathbf{m}_2 - \mathbf{m}_1) \tag{4.22}$$

where

$$m_k = \mathbf{w}^{\mathrm{T}}\mathbf{m}_k \tag{4.23}$$

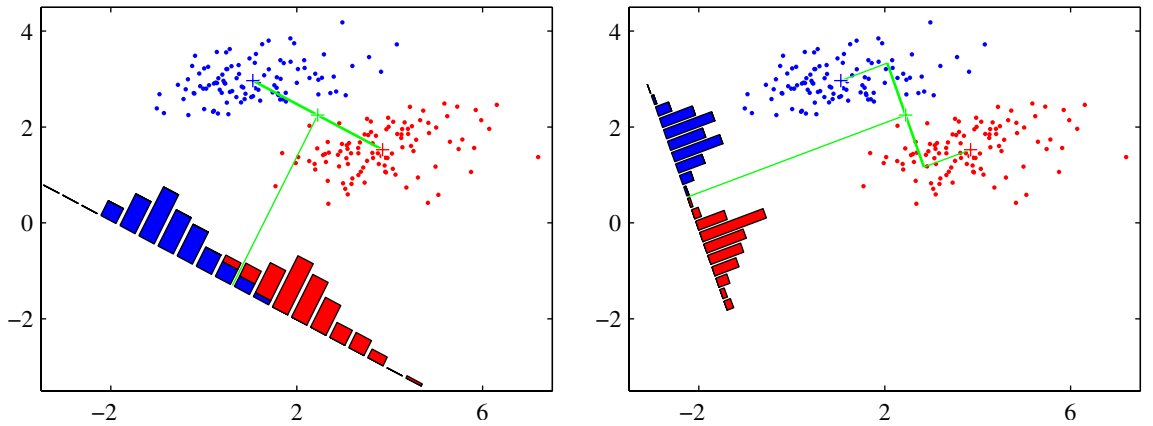


Figure 4.6 The left plot shows samples from two classes (depicted in red and blue) along with the histograms resulting from projection onto the line joining the class means. Note that there is considerable class overlap in the projected space. The right plot shows the corresponding projection based on the Fisher linear discriminant, showing the greatly improved class separation.

is the mean of the projected data from class $\mathcal{C}_k$. However, this expression can be made arbitrarily large simply by increasing the magnitude of $\mathbf{w}$. To solve this problem, we could constrain $\mathbf{w}$ to have unit length, so that $\sum_i w_i^2 = 1$. Using a Lagrange multiplier to perform the constrained maximization, we then find that $\mathbf{w} \propto (\mathbf{m}_2 - \mathbf{m}_1)$. There is still a problem with this approach, however, as illustrated in Figure 4.6. This shows two classes that are well separated in the original two-dimensional space (x_1, x_2) but that have considerable overlap when projected onto the line joining their means. This difficulty arises from the strongly nondiagonal covariances of the class distributions. The idea proposed by Fisher is to maximize a function that will give a large separation between the projected class means while also giving a small variance within each class, thereby minimizing the class overlap.

Appendix E
Exercise 4.4

The projection formula (4.20) transforms the set of labelled data points in $\mathbf{x}$ into a labelled set in the one-dimensional space y. The within-class variance of the transformed data from class $\mathcal{C}_k$ is therefore given by

$$s_k^2 = \sum_{n \in \mathcal{C}_k} (y_n - m_k)^2 \tag{4.24}$$

where $y_n = \mathbf{w}^{\mathrm{T}}\mathbf{x}_n$. We can define the total within-class variance for the whole data set to be simply $s_1^2 + s_2^2$. The Fisher criterion is defined to be the ratio of the between-class variance to the within-class variance and is given by

$$J(\mathbf{w}) = \frac{(m_2 - m_1)^2}{s_1^2 + s_2^2}. \tag{4.25}$$

Exercise 4.5

We can make the dependence on $\mathbf{w}$ explicit by using (4.20), (4.23), and (4.24) to rewrite the Fisher criterion in the form

$$J(\mathbf{w}) = \frac{\mathbf{w}^{\mathrm{T}}\mathbf{S}_{\mathrm{B}}\mathbf{w}}{\mathbf{w}^{\mathrm{T}}\mathbf{S}_{\mathrm{W}}\mathbf{w}} \tag{4.26}$$

where $\mathbf{S}_{\mathrm{B}}$ is the *between-class* covariance matrix and is given by

$$\mathbf{S}_{\mathrm{B}} = (\mathbf{m}_2 - \mathbf{m}_1)(\mathbf{m}_2 - \mathbf{m}_1)^{\mathrm{T}} \tag{4.27}$$

and $\mathbf{S}_{\mathrm{W}}$ is the total *within-class* covariance matrix, given by

$$\mathbf{S}_{\mathrm{W}} = \sum_{n\in\mathcal{C}_1}(\mathbf{x}_n - \mathbf{m}_1)(\mathbf{x}_n - \mathbf{m}_1)^{\mathrm{T}} + \sum_{n\in\mathcal{C}_2}(\mathbf{x}_n - \mathbf{m}_2)(\mathbf{x}_n - \mathbf{m}_2)^{\mathrm{T}}. \tag{4.28}$$

Differentiating (4.26) with respect to $\mathbf{w}$, we find that $J(\mathbf{w})$ is maximized when

$$(\mathbf{w}^{\mathrm{T}}\mathbf{S}_{\mathrm{B}}\mathbf{w})\mathbf{S}_{\mathrm{W}}\mathbf{w} = (\mathbf{w}^{\mathrm{T}}\mathbf{S}_{\mathrm{W}}\mathbf{w})\mathbf{S}_{\mathrm{B}}\mathbf{w}. \tag{4.29}$$

From (4.27), we see that $\mathbf{S}_{\mathrm{B}}\mathbf{w}$ is always in the direction of $(\mathbf{m}_2 - \mathbf{m}_1)$. Furthermore, we do not care about the magnitude of $\mathbf{w}$, only its direction, and so we can drop the scalar factors $(\mathbf{w}^{\mathrm{T}}\mathbf{S}_{\mathrm{B}}\mathbf{w})$ and $(\mathbf{w}^{\mathrm{T}}\mathbf{S}_{\mathrm{W}}\mathbf{w})$. Multiplying both sides of (4.29) by $\mathbf{S}_{\mathrm{W}}^{-1}$ we then obtain

$$\mathbf{w} \propto \mathbf{S}_{\mathrm{W}}^{-1}(\mathbf{m}_2 - \mathbf{m}_1). \tag{4.30}$$

Note that if the within-class covariance is isotropic, so that $\mathbf{S}_{\mathrm{W}}$ is proportional to the unit matrix, we find that $\mathbf{w}$ is proportional to the difference of the class means, as discussed above.

The result (4.30) is known as *Fisher's linear discriminant*, although strictly it is not a discriminant but rather a specific choice of direction for projection of the data down to one dimension. However, the projected data can subsequently be used to construct a discriminant, by choosing a threshold y_0 so that we classify a new point as belonging to $\mathcal{C}_1$ if $y(\mathbf{x}) \geqslant y_0$ and classify it as belonging to $\mathcal{C}_2$ otherwise. For example, we can model the class-conditional densities $p(y|\mathcal{C}_k)$ using Gaussian distributions and then use the techniques of Section 1.2.4 to find the parameters of the Gaussian distributions by maximum likelihood. Having found Gaussian approximations to the projected classes, the formalism of Section 1.5.1 then gives an expression for the optimal threshold. Some justification for the Gaussian assumption comes from the central limit theorem by noting that $y = \mathbf{w}^{\mathrm{T}}\mathbf{x}$ is the sum of a set of random variables.

4.1.5 Relation to least squares

The least-squares approach to the determination of a linear discriminant was based on the goal of making the model predictions as close as possible to a set of target values. By contrast, the Fisher criterion was derived by requiring maximum class separation in the output space. It is interesting to see the relationship between these two approaches. In particular, we shall show that, for the two-class problem, the Fisher criterion can be obtained as a special case of least squares.

So far we have considered 1-of-K coding for the target values. If, however, we adopt a slightly different target coding scheme, then the least-squares solution for

the weights becomes equivalent to the Fisher solution (Duda and Hart, 1973). In particular, we shall take the targets for class $\mathcal{C}_1$ to be N/N_1, where N_1 is the number of patterns in class $\mathcal{C}_1$, and N is the total number of patterns. This target value approximates the reciprocal of the prior probability for class $\mathcal{C}_1$. For class $\mathcal{C}_2$, we shall take the targets to be $-N/N_2$, where N_2 is the number of patterns in class $\mathcal{C}_2$.

The sum-of-squares error function can be written

$$E = \frac{1}{2}\sum_{n=1}^{N}\left(\mathbf{w}^{\mathrm{T}}\mathbf{x}_n + w_0 - t_n\right)^2. \tag{4.31}$$

Setting the derivatives of E with respect to w_0 and $\mathbf{w}$ to zero, we obtain respectively

$$\sum_{n=1}^{N}\left(\mathbf{w}^{\mathrm{T}}\mathbf{x}_n + w_0 - t_n\right) = 0 \tag{4.32}$$

$$\sum_{n=1}^{N}\left(\mathbf{w}^{\mathrm{T}}\mathbf{x}_n + w_0 - t_n\right)\mathbf{x}_n = 0. \tag{4.33}$$

From (4.32), and making use of our choice of target coding scheme for the t_n, we obtain an expression for the bias in the form

$$w_0 = -\mathbf{w}^{\mathrm{T}}\mathbf{m} \tag{4.34}$$

where we have used

$$\sum_{n=1}^{N} t_n = N_1\frac{N}{N_1} - N_2\frac{N}{N_2} = 0 \tag{4.35}$$

and where $\mathbf{m}$ is the mean of the total data set and is given by

$$\mathbf{m} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n = \frac{1}{N}(N_1\mathbf{m}_1 + N_2\mathbf{m}_2). \tag{4.36}$$

After some straightforward algebra, and again making use of the choice of t_n, the second equation (4.33) becomes

Exercise 4.6

$$\left(\mathbf{S}_{\mathrm{W}} + \frac{N_1N_2}{N}\mathbf{S}_{\mathrm{B}}\right)\mathbf{w} = N(\mathbf{m}_1 - \mathbf{m}_2) \tag{4.37}$$

where $\mathbf{S}_{\mathrm{W}}$ is defined by (4.28), $\mathbf{S}_{\mathrm{B}}$ is defined by (4.27), and we have substituted for the bias using (4.34). Using (4.27), we note that $\mathbf{S}_{\mathrm{B}}\mathbf{w}$ is always in the direction of $(\mathbf{m}_2 - \mathbf{m}_1)$. Thus we can write

$$\mathbf{w} \propto \mathbf{S}_{\mathrm{W}}^{-1}(\mathbf{m}_2 - \mathbf{m}_1) \tag{4.38}$$

where we have ignored irrelevant scale factors. Thus the weight vector coincides with that found from the Fisher criterion. In addition, we have also found an expression for the bias value w_0 given by (4.34). This tells us that a new vector $\mathbf{x}$ should be classified as belonging to class $\mathcal{C}_1$ if $y(\mathbf{x}) = \mathbf{w}^{\mathrm{T}}(\mathbf{x}-\mathbf{m}) > 0$ and class $\mathcal{C}_2$ otherwise.

4.1.6 Fisher's discriminant for multiple classes

We now consider the generalization of the Fisher discriminant to $K > 2$ classes, and we shall assume that the dimensionality D of the input space is greater than the number K of classes. Next, we introduce $D' > 1$ linear 'features' $y_k = \mathbf{w}_k^{\mathrm{T}}\mathbf{x}$, where $k = 1, \ldots, D'$. These feature values can conveniently be grouped together to form a vector $\mathbf{y}$. Similarly, the weight vectors $\{\mathbf{w}_k\}$ can be considered to be the columns of a matrix $\mathbf{W}$, so that

$$\mathbf{y} = \mathbf{W}^{\mathrm{T}}\mathbf{x}. \tag{4.39}$$

Note that again we are not including any bias parameters in the definition of $\mathbf{y}$. The generalization of the within-class covariance matrix to the case of K classes follows from (4.28) to give

$$\mathbf{S}_{\mathrm{W}} = \sum_{k=1}^{K} \mathbf{S}_k \tag{4.40}$$

where

$$\mathbf{S}_k = \sum_{n \in \mathcal{C}_k} (\mathbf{x}_n - \mathbf{m}_k)(\mathbf{x}_n - \mathbf{m}_k)^{\mathrm{T}} \tag{4.41}$$

$$\mathbf{m}_k = \frac{1}{N_k} \sum_{n \in \mathcal{C}_k} \mathbf{x}_n \tag{4.42}$$

and N_k is the number of patterns in class $\mathcal{C}_k$. In order to find a generalization of the between-class covariance matrix, we follow Duda and Hart (1973) and consider first the total covariance matrix

$$\mathbf{S}_{\mathrm{T}} = \sum_{n=1}^{N} (\mathbf{x}_n - \mathbf{m})(\mathbf{x}_n - \mathbf{m})^{\mathrm{T}} \tag{4.43}$$

where $\mathbf{m}$ is the mean of the total data set

$$\mathbf{m} = \frac{1}{N} \sum_{n=1}^{N} \mathbf{x}_n = \frac{1}{N} \sum_{k=1}^{K} N_k \mathbf{m}_k \tag{4.44}$$

and $N = \sum_k N_k$ is the total number of data points. The total covariance matrix can be decomposed into the sum of the within-class covariance matrix, given by (4.40) and (4.41), plus an additional matrix $\mathbf{S}_{\mathrm{B}}$, which we identify as a measure of the between-class covariance

$$\mathbf{S}_{\mathrm{T}} = \mathbf{S}_{\mathrm{W}} + \mathbf{S}_{\mathrm{B}} \tag{4.45}$$

where

$$\mathbf{S}_{\mathrm{B}} = \sum_{k=1}^{K} N_k (\mathbf{m}_k - \mathbf{m})(\mathbf{m}_k - \mathbf{m})^{\mathrm{T}}. \tag{4.46}$$

These covariance matrices have been defined in the original $\mathbf{x}$-space. We can now define similar matrices in the projected D'-dimensional $\mathbf{y}$-space

$$\mathbf{s}_{\mathrm{W}} = \sum_{k=1}^{K} \sum_{n \in \mathcal{C}_k} (\mathbf{y}_n - \boldsymbol{\mu}_k)(\mathbf{y}_n - \boldsymbol{\mu}_k)^{\mathrm{T}} \tag{4.47}$$

and

$$\mathbf{s}_{\mathrm{B}} = \sum_{k=1}^{K} N_k (\boldsymbol{\mu}_k - \boldsymbol{\mu})(\boldsymbol{\mu}_k - \boldsymbol{\mu})^{\mathrm{T}} \tag{4.48}$$

where

$$\boldsymbol{\mu}_k = \frac{1}{N_k} \sum_{n \in \mathcal{C}_k} \mathbf{y}_n, \qquad \boldsymbol{\mu} = \frac{1}{N} \sum_{k=1}^{K} N_k \boldsymbol{\mu}_k. \tag{4.49}$$

Again we wish to construct a scalar that is large when the between-class covariance is large and when the within-class covariance is small. There are now many possible choices of criterion (Fukunaga, 1990). One example is given by

$$J(\mathbf{W}) = \mathrm{Tr}\left\{ \mathbf{s}_{\mathrm{W}}^{-1} \mathbf{s}_{\mathrm{B}} \right\}. \tag{4.50}$$

This criterion can then be rewritten as an explicit function of the projection matrix $\mathbf{W}$ in the form

$$J(\mathbf{w}) = \mathrm{Tr}\left\{ (\mathbf{W}\mathbf{S}_{\mathrm{W}}\mathbf{W}^{\mathrm{T}})^{-1} (\mathbf{W}\mathbf{S}_{\mathrm{B}}\mathbf{W}^{\mathrm{T}}) \right\}. \tag{4.51}$$

Maximization of such criteria is straightforward, though somewhat involved, and is discussed at length in Fukunaga (1990). The weight values are determined by those eigenvectors of $\mathbf{S}_{\mathrm{W}}^{-1}\mathbf{S}_{\mathrm{B}}$ that correspond to the D' largest eigenvalues.

There is one important result that is common to all such criteria, which is worth emphasizing. We first note from (4.46) that $\mathbf{S}_{\mathrm{B}}$ is composed of the sum of K matrices, each of which is an outer product of two vectors and therefore of rank 1. In addition, only $(K-1)$ of these matrices are independent as a result of the constraint (4.44). Thus, $\mathbf{S}_{\mathrm{B}}$ has rank at most equal to $(K-1)$ and so there are at most $(K-1)$ nonzero eigenvalues. This shows that the projection onto the $(K-1)$-dimensional subspace spanned by the eigenvectors of $\mathbf{S}_{\mathrm{B}}$ does not alter the value of $J(\mathbf{w})$, and so we are therefore unable to find more than $(K-1)$ linear 'features' by this means (Fukunaga, 1990).

4.1.7 The perceptron algorithm

Another example of a linear discriminant model is the perceptron of Rosenblatt (1962), which occupies an important place in the history of pattern recognition algorithms. It corresponds to a two-class model in which the input vector $\mathbf{x}$ is first transformed using a fixed nonlinear transformation to give a feature vector $\boldsymbol{\phi}(\mathbf{x})$, and this is then used to construct a generalized linear model of the form

$$y(\mathbf{x}) = f\left(\mathbf{w}^{\mathrm{T}} \boldsymbol{\phi}(\mathbf{x})\right) \tag{4.52}$$

where the nonlinear activation function $f(\cdot)$ is given by a step function of the form

$$f(a) = \begin{cases} +1, & a \geqslant 0 \\ -1, & a < 0. \end{cases} \tag{4.53}$$

The vector $\boldsymbol{\phi}(\mathbf{x})$ will typically include a bias component $\phi_0(\mathbf{x}) = 1$. In earlier discussions of two-class classification problems, we have focussed on a target coding scheme in which $t \in \{0, 1\}$, which is appropriate in the context of probabilistic models. For the perceptron, however, it is more convenient to use target values $t = +1$ for class $\mathcal{C}_1$ and $t = -1$ for class $\mathcal{C}_2$, which matches the choice of activation function.

The algorithm used to determine the parameters $\mathbf{w}$ of the perceptron can most easily be motivated by error function minimization. A natural choice of error function would be the total number of misclassified patterns. However, this does not lead to a simple learning algorithm because the error is a piecewise constant function of $\mathbf{w}$, with discontinuities wherever a change in $\mathbf{w}$ causes the decision boundary to move across one of the data points. Methods based on changing $\mathbf{w}$ using the gradient of the error function cannot then be applied, because the gradient is zero almost everywhere.

We therefore consider an alternative error function known as the *perceptron criterion*. To derive this, we note that we are seeking a weight vector $\mathbf{w}$ such that patterns $\mathbf{x}_n$ in class $\mathcal{C}_1$ will have $\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n) > 0$, whereas patterns $\mathbf{x}_n$ in class $\mathcal{C}_2$ have $\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n) < 0$. Using the $t \in \{-1, +1\}$ target coding scheme it follows that we would like all patterns to satisfy $\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n)t_n > 0$. The perceptron criterion associates zero error with any pattern that is correctly classified, whereas for a misclassified pattern $\mathbf{x}_n$ it tries to minimize the quantity $-\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}_n)t_n$. The perceptron criterion is therefore given by

$$E_{\mathrm{P}}(\mathbf{w}) = -\sum_{n \in \mathcal{M}} \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}_n t_n \tag{4.54}$$

Frank Rosenblatt
1928–1969

Rosenblatt's perceptron played an important role in the history of machine learning. Initially, Rosenblatt simulated the perceptron on an IBM 704 computer at Cornell in 1957, but by the early 1960s he had built special-purpose hardware that provided a direct, parallel implementation of perceptron learning. Many of his ideas were encapsulated in "Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms" published in 1962. Rosenblatt's work was criticized by Marvin Minksy, whose objections were published in the book "Perceptrons", co-authored with Seymour Papert. This book was widely misinterpreted at the time as showing that neural networks were fatally flawed and could only learn solutions for linearly separable problems. In fact, it only proved such limitations in the case of single-layer networks such as the perceptron and merely conjectured (incorrectly) that they applied to more general network models. Unfortunately, however, this book contributed to the substantial decline in research funding for neural computing, a situation that was not reversed until the mid-1980s. Today, there are many hundreds, if not thousands, of applications of neural networks in widespread use, with examples in areas such as handwriting recognition and information retrieval being used routinely by millions of people.

where $\mathcal{M}$ denotes the set of all misclassified patterns. The contribution to the error associated with a particular misclassified pattern is a linear function of $\mathbf{w}$ in regions of $\mathbf{w}$ space where the pattern is misclassified and zero in regions where it is correctly classified. The total error function is therefore piecewise linear.

Section 3.1.3

We now apply the stochastic gradient descent algorithm to this error function. The change in the weight vector $\mathbf{w}$ is then given by

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} - \eta \nabla E_{\mathrm{P}}(\mathbf{w}) = \mathbf{w}^{(\tau)} + \eta \boldsymbol{\phi}_n t_n \tag{4.55}$$

where η is the learning rate parameter and τ is an integer that indexes the steps of the algorithm. Because the perceptron function $y(\mathbf{x}, \mathbf{w})$ is unchanged if we multiply $\mathbf{w}$ by a constant, we can set the learning rate parameter η equal to 1 without of generality. Note that, as the weight vector evolves during training, the set of patterns that are misclassified will change.

The perceptron learning algorithm has a simple interpretation, as follows. We cycle through the training patterns in turn, and for each pattern $\mathbf{x}_n$ we evaluate the perceptron function (4.52). If the pattern is correctly classified, then the weight vector remains unchanged, whereas if it is incorrectly classified, then for class $\mathcal{C}_1$ we add the vector $\boldsymbol{\phi}(\mathbf{x}_n)$ onto the current estimate of weight vector $\mathbf{w}$ while for class $\mathcal{C}_2$ we subtract the vector $\boldsymbol{\phi}(\mathbf{x}_n)$ from $\mathbf{w}$. The perceptron learning algorithm is illustrated in Figure 4.7.

If we consider the effect of a single update in the perceptron learning algorithm, we see that the contribution to the error from a misclassified pattern will be reduced because from (4.55) we have

$$-\mathbf{w}^{(\tau+1)\mathrm{T}} \boldsymbol{\phi}_n t_n = -\mathbf{w}^{(\tau)\mathrm{T}} \boldsymbol{\phi}_n t_n - (\boldsymbol{\phi}_n t_n)^{\mathrm{T}} \boldsymbol{\phi}_n t_n < -\mathbf{w}^{(\tau)\mathrm{T}} \boldsymbol{\phi}_n t_n \tag{4.56}$$

where we have set $\eta = 1$, and made use of $\|\boldsymbol{\phi}_n t_n\|^2 > 0$. Of course, this does not imply that the contribution to the error function from the other misclassified patterns will have been reduced. Furthermore, the change in weight vector may have caused some previously correctly classified patterns to become misclassified. Thus the perceptron learning rule is not guaranteed to reduce the total error function at each stage.

However, the *perceptron convergence theorem* states that if there exists an exact solution (in other words, if the training data set is linearly separable), then the perceptron learning algorithm is guaranteed to find an exact solution in a finite number of steps. Proofs of this theorem can be found for example in Rosenblatt (1962), Block (1962), Nilsson (1965), Minsky and Papert (1969), Hertz *et al.* (1991), and Bishop (1995a). Note, however, that the number of steps required to achieve convergence could still be substantial, and in practice, until convergence is achieved, we will not be able to distinguish between a nonseparable problem and one that is simply slow to converge.

Even when the data set is linearly separable, there may be many solutions, and which one is found will depend on the initialization of the parameters and on the order of presentation of the data points. Furthermore, for data sets that are not linearly separable, the perceptron learning algorithm will never converge.

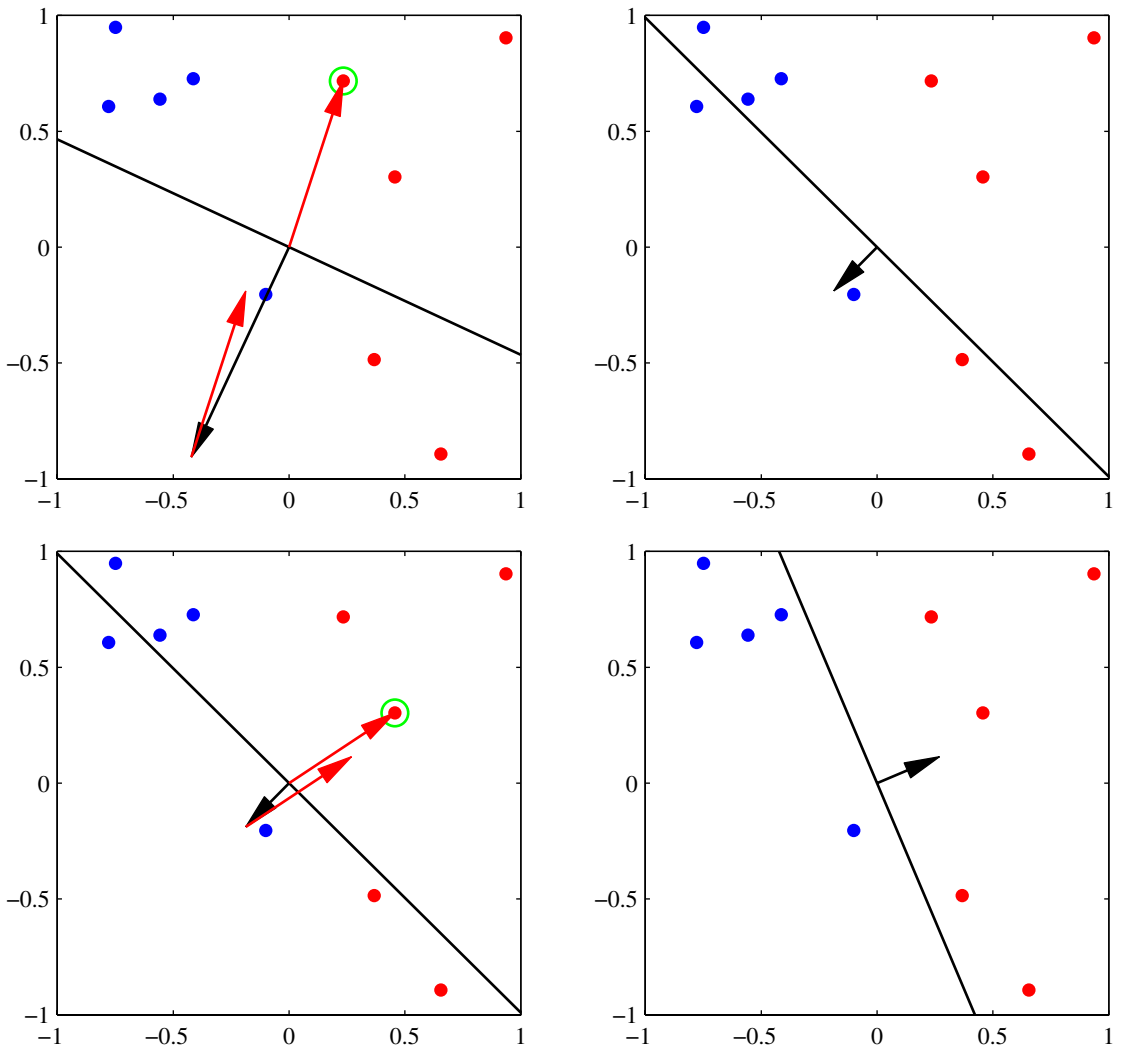


Figure 4.7 Illustration of the convergence of the perceptron learning algorithm, showing data points from two classes (red and blue) in a two-dimensional feature space (ϕ_1, ϕ_2). The top left plot shows the initial parameter vector $\mathbf{w}$ shown as a black arrow together with the corresponding decision boundary (black line), in which the arrow points towards the decision region which classified as belonging to the red class. The data point circled in green is misclassified and so its feature vector is added to the current weight vector, giving the new decision boundary shown in the top right plot. The bottom left plot shows the next misclassified point to be considered, indicated by the green circle, and its feature vector is again added to the weight vector giving the decision boundary shown in the bottom right plot for which all data points are correctly classified.

Figure 4.8 Illustration of the Mark 1 perceptron hardware. The photograph on the left shows how the inputs were obtained using a simple camera system in which an input scene, in this case a printed character, was illuminated by powerful lights, and an image focussed onto a 20×20 array of cadmium sulphide photocells, giving a primitive 400 pixel image. The perceptron also had a patch board, shown in the middle photograph, which allowed different configurations of input features to be tried. Often these were wired up at random to demonstrate the ability of the perceptron to learn without the need for precise wiring, in contrast to a modern digital computer. The photograph on the right shows one of the racks of adaptive weights. Each weight was implemented using a rotary variable resistor, also called a potentiometer, driven by an electric motor thereby allowing the value of the weight to be adjusted automatically by the learning algorithm.

Aside from difficulties with the learning algorithm, the perceptron does not provide probabilistic outputs, nor does it generalize readily to $K > 2$ classes. The most important limitation, however, arises from the fact that (in common with all of the models discussed in this chapter and the previous one) it is based on linear combinations of fixed basis functions. More detailed discussions of the limitations of perceptrons can be found in Minsky and Papert (1969) and Bishop (1995a).

Analogue hardware implementations of the perceptron were built by Rosenblatt, based on motor-driven variable resistors to implement the adaptive parameters w_j. These are illustrated in Figure 4.8. The inputs were obtained from a simple camera system based on an array of photo-sensors, while the basis functions ϕ could be chosen in a variety of ways, for example based on simple fixed functions of randomly chosen subsets of pixels from the input image. Typical applications involved learning to discriminate simple shapes or characters.

At the same time that the perceptron was being developed, a closely related system called the *adaline*, which is short for 'adaptive linear element', was being explored by Widrow and co-workers. The functional form of the model was the same as for the perceptron, but a different approach to training was adopted (Widrow and Hoff, 1960; Widrow and Lehr, 1990).

4.2. Probabilistic Generative Models

We turn next to a probabilistic view of classification and show how models with linear decision boundaries arise from simple assumptions about the distribution of the data. In Section 1.5.4, we discussed the distinction between the discriminative and the generative approaches to classification. Here we shall adopt a generative

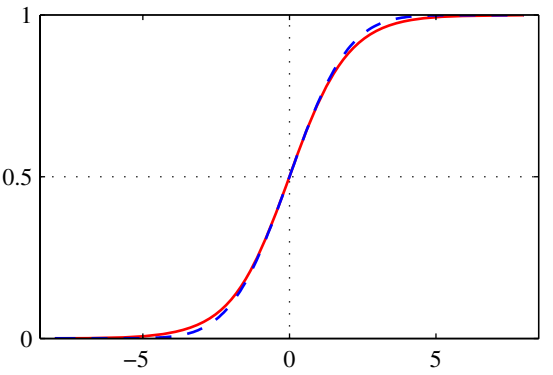


Figure 4.9 Plot of the logistic sigmoid function $\sigma(a)$ defined by (4.59), shown in red, together with the scaled probit function $\Phi(\lambda a)$, for $\lambda^2 = \pi/8$, shown in dashed blue, where $\Phi(a)$ is defined by (4.114). The scaling factor $\pi/8$ is chosen so that the derivatives of the two curves are equal for $a = 0$.

approach in which we model the class-conditional densities $p(\mathbf{x}|\mathcal{C}_k)$, as well as the class priors $p(\mathcal{C}_k)$, and then use these to compute posterior probabilities $p(\mathcal{C}_k|\mathbf{x})$ through Bayes' theorem.

Consider first of all the case of two classes. The posterior probability for class $\mathcal{C}_1$ can be written as

$$\begin{aligned} p(\mathcal{C}_1|\mathbf{x}) &= \frac{p(\mathbf{x}|\mathcal{C}_1)p(\mathcal{C}_1)}{p(\mathbf{x}|\mathcal{C}_1)p(\mathcal{C}_1) + p(\mathbf{x}|\mathcal{C}_2)p(\mathcal{C}_2)} \\ &= \frac{1}{1+\exp(-a)} = \sigma(a) \end{aligned} \tag{4.57}$$

where we have defined

$$a = \ln \frac{p(\mathbf{x}|\mathcal{C}_1)p(\mathcal{C}_1)}{p(\mathbf{x}|\mathcal{C}_2)p(\mathcal{C}_2)} \tag{4.58}$$

and $\sigma(a)$ is the *logistic sigmoid* function defined by

$$\sigma(a) = \frac{1}{1+\exp(-a)} \tag{4.59}$$

which is plotted in Figure 4.9. The term 'sigmoid' means S-shaped. This type of function is sometimes also called a 'squashing function' because it maps the whole real axis into a finite interval. The logistic sigmoid has been encountered already in earlier chapters and plays an important role in many classification algorithms. It satisfies the following symmetry property

$$\sigma(-a) = 1 - \sigma(a) \tag{4.60}$$

as is easily verified. The inverse of the logistic sigmoid is given by

$$a = \ln\left(\frac{\sigma}{1-\sigma}\right) \tag{4.61}$$

and is known as the *logit* function. It represents the log of the ratio of probabilities $\ln\left[p(\mathcal{C}_1|\mathbf{x})/p(\mathcal{C}_2|\mathbf{x})\right]$ for the two classes, also known as the *log odds*.

Note that in (4.57) we have simply rewritten the posterior probabilities in an equivalent form, and so the appearance of the logistic sigmoid may seem rather vacuous. However, it will have significance provided $a(\mathbf{x})$ takes a simple functional form. We shall shortly consider situations in which $a(\mathbf{x})$ is a linear function of $\mathbf{x}$, in which case the posterior probability is governed by a generalized linear model.

For the case of $K > 2$ classes, we have

$$\begin{aligned} p(\mathcal{C}_k|\mathbf{x}) &= \frac{p(\mathbf{x}|\mathcal{C}_k)p(\mathcal{C}_k)}{\sum_j p(\mathbf{x}|\mathcal{C}_j)p(\mathcal{C}_j)} \\ &= \frac{\exp(a_k)}{\sum_j \exp(a_j)} \end{aligned} \tag{4.62}$$

which is known as the *normalized exponential* and can be regarded as a multiclass generalization of the logistic sigmoid. Here the quantities a_k are defined by

$$a_k = \ln p(\mathbf{x}|\mathcal{C}_k)p(\mathcal{C}_k). \tag{4.63}$$

The normalized exponential is also known as the *softmax function*, as it represents a smoothed version of the 'max' function because, if $a_k \gg a_j$ for all $j \neq k$, then $p(\mathcal{C}_k|\mathbf{x}) \simeq 1$, and $p(\mathcal{C}_j|\mathbf{x}) \simeq 0$.

We now investigate the consequences of choosing specific forms for the class-conditional densities, looking first at continuous input variables $\mathbf{x}$ and then discussing briefly the case of discrete inputs.

4.2.1 Continuous inputs

Let us assume that the class-conditional densities are Gaussian and then explore the resulting form for the posterior probabilities. To start with, we shall assume that all classes share the same covariance matrix. Thus the density for class $\mathcal{C}_k$ is given by

$$p(\mathbf{x}|\mathcal{C}_k) = \frac{1}{(2\pi)^{D/2}} \frac{1}{|\boldsymbol{\Sigma}|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_k)^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu}_k)\right\}. \tag{4.64}$$

Consider first the case of two classes. From (4.57) and (4.58), we have

$$p(\mathcal{C}_1|\mathbf{x}) = \sigma(\mathbf{w}^{\mathrm{T}}\mathbf{x} + w_0) \tag{4.65}$$

where we have defined

$$\mathbf{w} = \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2) \tag{4.66}$$

$$w_0 = -\frac{1}{2}\boldsymbol{\mu}_1^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_1 + \frac{1}{2}\boldsymbol{\mu}_2^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_2 + \ln\frac{p(\mathcal{C}_1)}{p(\mathcal{C}_2)}. \tag{4.67}$$

We see that the quadratic terms in $\mathbf{x}$ from the exponents of the Gaussian densities have cancelled (due to the assumption of common covariance matrices) leading to a linear function of $\mathbf{x}$ in the argument of the logistic sigmoid. This result is illustrated for the case of a two-dimensional input space $\mathbf{x}$ in Figure 4.10. The resulting

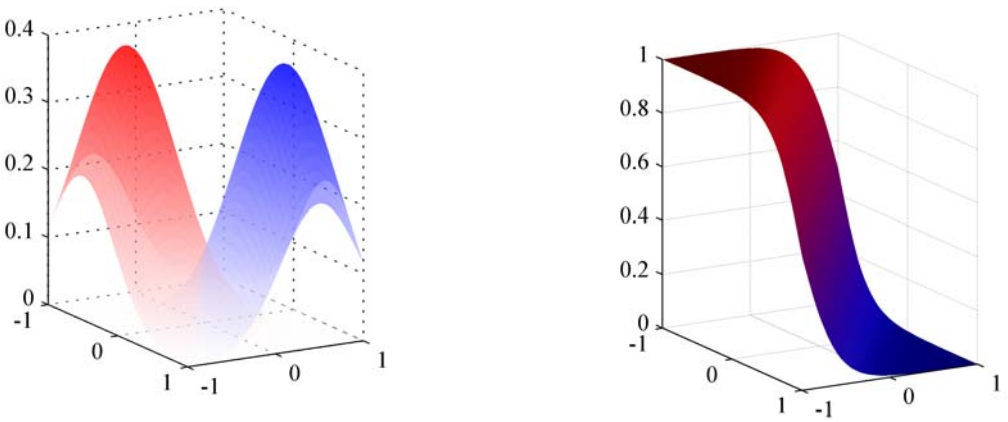


Figure 4.10 The left-hand plot shows the class-conditional densities for two classes, denoted red and blue. On the right is the corresponding posterior probability $p(\mathcal{C}_1|\mathbf{x})$, which is given by a logistic sigmoid of a linear function of $\mathbf{x}$. The surface in the right-hand plot is coloured using a proportion of red ink given by $p(\mathcal{C}_1|\mathbf{x})$ and a proportion of blue ink given by $p(\mathcal{C}_2|\mathbf{x}) = 1 - p(\mathcal{C}_1|\mathbf{x})$.

decision boundaries correspond to surfaces along which the posterior probabilities $p(\mathcal{C}_k|\mathbf{x})$ are constant and so will be given by linear functions of $\mathbf{x}$, and therefore the decision boundaries are linear in input space. The prior probabilities $p(\mathcal{C}_k)$ enter only through the bias parameter w_0 so that changes in the priors have the effect of making parallel shifts of the decision boundary and more generally of the parallel contours of constant posterior probability.

For the general case of K classes we have, from (4.62) and (4.63),

$$a_k(\mathbf{x}) = \mathbf{w}_k^{\mathrm{T}}\mathbf{x} + w_{k0} \tag{4.68}$$

where we have defined

$$\mathbf{w}_k = \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_k \tag{4.69}$$

$$w_{k0} = -\frac{1}{2}\boldsymbol{\mu}_k^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_k + \ln p(\mathcal{C}_k). \tag{4.70}$$

We see that the $a_k(\mathbf{x})$ are again linear functions of $\mathbf{x}$ as a consequence of the cancellation of the quadratic terms due to the shared covariances. The resulting decision boundaries, corresponding to the minimum misclassification rate, will occur when two of the posterior probabilities (the two largest) are equal, and so will be defined by linear functions of $\mathbf{x}$, and so again we have a generalized linear model.

If we relax the assumption of a shared covariance matrix and allow each class-conditional density $p(\mathbf{x}|\mathcal{C}_k)$ to have its own covariance matrix $\boldsymbol{\Sigma}_k$, then the earlier cancellations will no longer occur, and we will obtain quadratic functions of $\mathbf{x}$, giving rise to a *quadratic discriminant*. The linear and quadratic decision boundaries are illustrated in Figure 4.11.

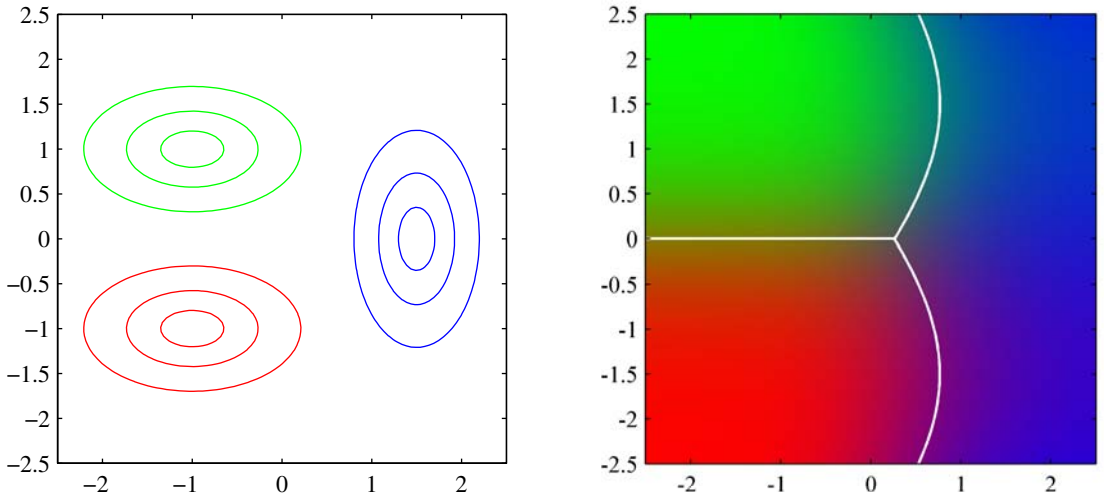


Figure 4.11 The left-hand plot shows the class-conditional densities for three classes each having a Gaussian distribution, coloured red, green, and blue, in which the red and green classes have the same covariance matrix. The right-hand plot shows the corresponding posterior probabilities, in which the RGB colour vector represents the posterior probabilities for the respective three classes. The decision boundaries are also shown. Notice that the boundary between the red and green classes, which have the same covariance matrix, is linear, whereas those between the other pairs of classes are quadratic.

4.2.2 Maximum likelihood solution

Once we have specified a parametric functional form for the class-conditional densities $p(\mathbf{x}|\mathcal{C}_k)$, we can then determine the values of the parameters, together with the prior class probabilities $p(\mathcal{C}_k)$, using maximum likelihood. This requires a data set comprising observations of $\mathbf{x}$ along with their corresponding class labels.

Consider first the case of two classes, each having a Gaussian class-conditional density with a shared covariance matrix, and suppose we have a data set $\{\mathbf{x}_n, t_n\}$ where $n = 1, \ldots, N$. Here $t_n = 1$ denotes class $\mathcal{C}_1$ and $t_n = 0$ denotes class $\mathcal{C}_2$. We denote the prior class probability $p(\mathcal{C}_1) = \pi$, so that $p(\mathcal{C}_2) = 1 - \pi$. For a data point $\mathbf{x}_n$ from class $\mathcal{C}_1$, we have $t_n = 1$ and hence

$$p(\mathbf{x}_n, \mathcal{C}_1) = p(\mathcal{C}_1)p(\mathbf{x}_n|\mathcal{C}_1) = \pi\mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_1, \boldsymbol{\Sigma}).$$

Similarly for class $\mathcal{C}_2$, we have $t_n = 0$ and hence

$$p(\mathbf{x}_n, \mathcal{C}_2) = p(\mathcal{C}_2)p(\mathbf{x}_n|\mathcal{C}_2) = (1 - \pi)\mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_2, \boldsymbol{\Sigma}).$$

Thus the likelihood function is given by

$$p(\mathbf{t}|\pi, \boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \boldsymbol{\Sigma}) = \prod_{n=1}^{N} \left[\pi\mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_1, \boldsymbol{\Sigma})\right]^{t_n} \left[(1 - \pi)\mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_2, \boldsymbol{\Sigma})\right]^{1-t_n} \tag{4.71}$$

where $\mathbf{t} = (t_1, \ldots, t_N)^{\mathrm{T}}$. As usual, it is convenient to maximize the log of the likelihood function. Consider first the maximization with respect to π. The terms in

the log likelihood function that depend on π are

$$\sum_{n=1}^{N} \{t_n \ln \pi + (1 - t_n) \ln(1 - \pi)\} . \tag{4.72}$$

Setting the derivative with respect to π equal to zero and rearranging, we obtain

$$\pi = \frac{1}{N} \sum_{n=1}^{N} t_n = \frac{N_1}{N} = \frac{N_1}{N_1 + N_2} \tag{4.73}$$

where N_1 denotes the total number of data points in class $\mathcal{C}_1$, and N_2 denotes the total number of data points in class $\mathcal{C}_2$. Thus the maximum likelihood estimate for π is simply the fraction of points in class $\mathcal{C}_1$ as expected. This result is easily generalized to the multiclass case where again the maximum likelihood estimate of the prior probability associated with class $\mathcal{C}_k$ is given by the fraction of the training set points assigned to that class.

Exercise 4.9

Now consider the maximization with respect to $\boldsymbol{\mu}_1$. Again we can pick out of the log likelihood function those terms that depend on $\boldsymbol{\mu}_1$ giving

$$\sum_{n=1}^{N} t_n \ln \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_1, \boldsymbol{\Sigma}) = -\frac{1}{2} \sum_{n=1}^{N} t_n (\mathbf{x}_n - \boldsymbol{\mu}_1)^{\mathrm{T}} \boldsymbol{\Sigma}^{-1} (\mathbf{x}_n - \boldsymbol{\mu}_1) + \text{const.} \tag{4.74}$$

Setting the derivative with respect to $\boldsymbol{\mu}_1$ to zero and rearranging, we obtain

$$\boldsymbol{\mu}_1 = \frac{1}{N_1} \sum_{n=1}^{N} t_n \mathbf{x}_n \tag{4.75}$$

which is simply the mean of all the input vectors $\mathbf{x}_n$ assigned to class $\mathcal{C}_1$. By a similar argument, the corresponding result for $\boldsymbol{\mu}_2$ is given by

$$\boldsymbol{\mu}_2 = \frac{1}{N_2} \sum_{n=1}^{N} (1 - t_n) \mathbf{x}_n \tag{4.76}$$

which again is the mean of all the input vectors $\mathbf{x}_n$ assigned to class $\mathcal{C}_2$.

Finally, consider the maximum likelihood solution for the shared covariance matrix $\boldsymbol{\Sigma}$. Picking out the terms in the log likelihood function that depend on $\boldsymbol{\Sigma}$, we have

$$\begin{aligned} &-\frac{1}{2} \sum_{n=1}^{N} t_n \ln |\boldsymbol{\Sigma}| - \frac{1}{2} \sum_{n=1}^{N} t_n (\mathbf{x}_n - \boldsymbol{\mu}_1)^{\mathrm{T}} \boldsymbol{\Sigma}^{-1} (\mathbf{x}_n - \boldsymbol{\mu}_1) \\ &-\frac{1}{2} \sum_{n=1}^{N} (1 - t_n) \ln |\boldsymbol{\Sigma}| - \frac{1}{2} \sum_{n=1}^{N} (1 - t_n) (\mathbf{x}_n - \boldsymbol{\mu}_2)^{\mathrm{T}} \boldsymbol{\Sigma}^{-1} (\mathbf{x}_n - \boldsymbol{\mu}_2) \\ &= -\frac{N}{2} \ln |\boldsymbol{\Sigma}| - \frac{N}{2} \mathrm{Tr} \left\{ \boldsymbol{\Sigma}^{-1} \mathbf{S} \right\} \end{aligned} \tag{4.77}$$

where we have defined

$$\mathbf{S} = \frac{N_1}{N}\mathbf{S}_1 + \frac{N_2}{N}\mathbf{S}_2 \tag{4.78}$$

$$\mathbf{S}_1 = \frac{1}{N_1}\sum_{n\in\mathcal{C}_1}(\mathbf{x}_n - \boldsymbol{\mu}_1)(\mathbf{x}_n - \boldsymbol{\mu}_1)^{\mathrm{T}} \tag{4.79}$$

$$\mathbf{S}_2 = \frac{1}{N_2}\sum_{n\in\mathcal{C}_2}(\mathbf{x}_n - \boldsymbol{\mu}_2)(\mathbf{x}_n - \boldsymbol{\mu}_2)^{\mathrm{T}}. \tag{4.80}$$

Using the standard result for the maximum likelihood solution for a Gaussian distribution, we see that $\boldsymbol{\Sigma} = \mathbf{S}$, which represents a weighted average of the covariance matrices associated with each of the two classes separately.

This result is easily extended to the K class problem to obtain the corresponding maximum likelihood solutions for the parameters in which each class-conditional density is Gaussian with a shared covariance matrix. Note that the approach of fitting Gaussian distributions to the classes is not robust to outliers, because the maximum likelihood estimation of a Gaussian is not robust.

Exercise 4.10

Section 2.3.7

4.2.3 Discrete features

Let us now consider the case of discrete feature values x_i. For simplicity, we begin by looking at binary feature values $x_i \in \{0, 1\}$ and discuss the extension to more general discrete features shortly. If there are D inputs, then a general distribution would correspond to a table of 2^D numbers for each class, containing $2^D - 1$ independent variables (due to the summation constraint). Because this grows exponentially with the number of features, we might seek a more restricted representation. Here we will make the *naive Bayes* assumption in which the feature values are treated as independent, conditioned on the class $\mathcal{C}_k$. Thus we have class-conditional distributions of the form

Section 8.2.2

$$p(\mathbf{x}|\mathcal{C}_k) = \prod_{i=1}^{D}\mu_{ki}^{x_i}(1 - \mu_{ki})^{1-x_i} \tag{4.81}$$

which contain D independent parameters for each class. Substituting into (4.63) then gives

$$a_k(\mathbf{x}) = \sum_{i=1}^{D}\{x_i \ln \mu_{ki} + (1 - x_i)\ln(1 - \mu_{ki})\} + \ln p(\mathcal{C}_k) \tag{4.82}$$

which again are linear functions of the input values x_i. For the case of $K = 2$ classes, we can alternatively consider the logistic sigmoid formulation given by (4.57). Analogous results are obtained for discrete variables each of which can take $M > 2$ states.

Exercise 4.11

4.2.4 Exponential family

As we have seen, for both Gaussian distributed and discrete inputs, the posterior class probabilities are given by generalized linear models with logistic sigmoid ($K =$

2 classes) or softmax ($K \geqslant 2$ classes) activation functions. These are particular cases of a more general result obtained by assuming that the class-conditional densities $p(\mathbf{x}|\mathcal{C}_k)$ are members of the exponential family of distributions.

Using the form (2.194) for members of the exponential family, we see that the distribution of $\mathbf{x}$ can be written in the form

$$p(\mathbf{x}|\boldsymbol{\lambda}_k) = h(\mathbf{x})g(\boldsymbol{\lambda}_k)\exp\left\{\boldsymbol{\lambda}_k^{\mathrm{T}}\mathbf{u}(\mathbf{x})\right\}. \tag{4.83}$$

We now restrict attention to the subclass of such distributions for which $\mathbf{u}(\mathbf{x}) = \mathbf{x}$. Then we make use of (2.236) to introduce a scaling parameter s, so that we obtain the restricted set of exponential family class-conditional densities of the form

$$p(\mathbf{x}|\boldsymbol{\lambda}_k, s) = \frac{1}{s}h\left(\frac{1}{s}\mathbf{x}\right)g(\boldsymbol{\lambda}_k)\exp\left\{\frac{1}{s}\boldsymbol{\lambda}_k^{\mathrm{T}}\mathbf{x}\right\}. \tag{4.84}$$

Note that we are allowing each class to have its own parameter vector $\boldsymbol{\lambda}_k$ but we are assuming that the classes share the same scale parameter s.

For the two-class problem, we substitute this expression for the class-conditional densities into (4.58) and we see that the posterior class probability is again given by a logistic sigmoid acting on a linear function $a(\mathbf{x})$ which is given by

$$a(\mathbf{x}) = (\boldsymbol{\lambda}_1 - \boldsymbol{\lambda}_2)^{\mathrm{T}}\mathbf{x} + \ln g(\boldsymbol{\lambda}_1) - \ln g(\boldsymbol{\lambda}_2) + \ln p(\mathcal{C}_1) - \ln p(\mathcal{C}_2). \tag{4.85}$$

Similarly, for the K-class problem, we substitute the class-conditional density expression into (4.63) to give

$$a_k(\mathbf{x}) = \boldsymbol{\lambda}_k^{\mathrm{T}}\mathbf{x} + \ln g(\boldsymbol{\lambda}_k) + \ln p(\mathcal{C}_k) \tag{4.86}$$

and so again is a linear function of $\mathbf{x}$.

4.3. Probabilistic Discriminative Models

For the two-class classification problem, we have seen that the posterior probability of class $\mathcal{C}_1$ can be written as a logistic sigmoid acting on a linear function of $\mathbf{x}$, for a wide choice of class-conditional distributions $p(\mathbf{x}|\mathcal{C}_k)$. Similarly, for the multiclass case, the posterior probability of class $\mathcal{C}_k$ is given by a softmax transformation of a linear function of $\mathbf{x}$. For specific choices of the class-conditional densities $p(\mathbf{x}|\mathcal{C}_k)$, we have used maximum likelihood to determine the parameters of the densities as well as the class priors $p(\mathcal{C}_k)$ and then used Bayes' theorem to find the posterior class probabilities.

However, an alternative approach is to use the functional form of the generalized linear model explicitly and to determine its parameters directly by using maximum likelihood. We shall see that there is an efficient algorithm finding such solutions known as *iterative reweighted least squares*, or *IRLS*.

The indirect approach to finding the parameters of a generalized linear model, by fitting class-conditional densities and class priors separately and then applying

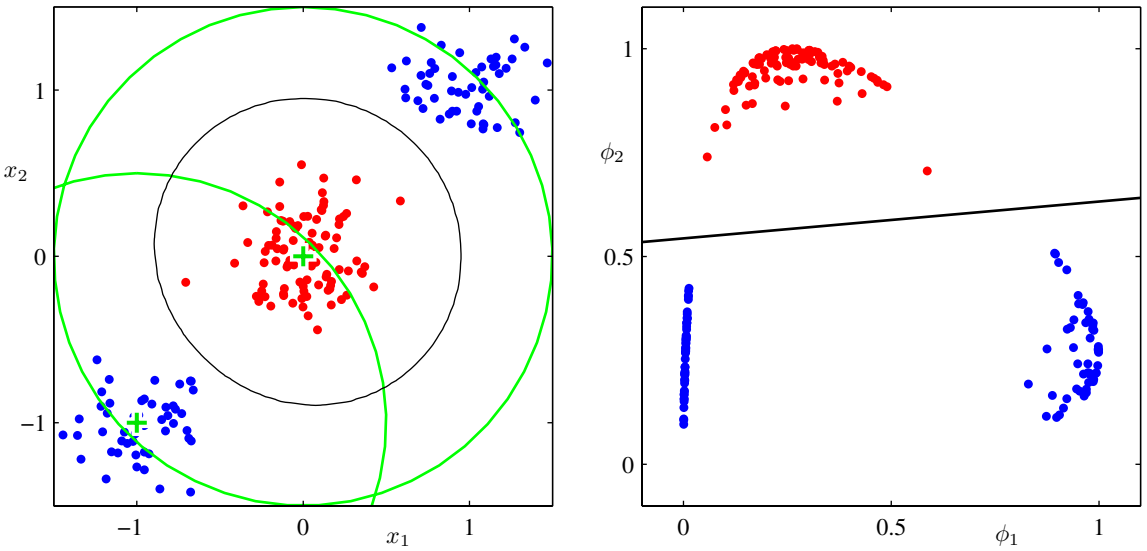


Figure 4.12 Illustration of the role of nonlinear basis functions in linear classification models. The left plot shows the original input space (x_1, x_2) together with data points from two classes labelled red and blue. Two 'Gaussian' basis functions $\phi_1(\mathbf{x})$ and $\phi_2(\mathbf{x})$ are defined in this space with centres shown by the green crosses and with contours shown by the green circles. The right-hand plot shows the corresponding feature space (ϕ_1, ϕ_2) together with the linear decision boundary obtained given by a logistic regression model of the form discussed in Section 4.3.2. This corresponds to a nonlinear decision boundary in the original input space, shown by the black curve in the left-hand plot.

Bayes' theorem, represents an example of *generative* modelling, because we could take such a model and generate synthetic data by drawing values of $\mathbf{x}$ from the marginal distribution $p(\mathbf{x})$. In the direct approach, we are maximizing a likelihood function defined through the conditional distribution $p(\mathcal{C}_k|\mathbf{x})$, which represents a form of *discriminative* training. One advantage of the discriminative approach is that there will typically be fewer adaptive parameters to be determined, as we shall see shortly. It may also lead to improved predictive performance, particularly when the class-conditional density assumptions give a poor approximation to the true distributions.

4.3.1 Fixed basis functions

So far in this chapter, we have considered classification models that work directly with the original input vector $\mathbf{x}$. However, all of the algorithms are equally applicable if we first make a fixed nonlinear transformation of the inputs using a vector of basis functions $\boldsymbol{\phi}(\mathbf{x})$. The resulting decision boundaries will be linear in the feature space $\boldsymbol{\phi}$, and these correspond to nonlinear decision boundaries in the original $\mathbf{x}$ space, as illustrated in Figure 4.12. Classes that are linearly separable in the feature space $\boldsymbol{\phi}(\mathbf{x})$ need not be linearly separable in the original observation space $\mathbf{x}$. Note that as in our discussion of linear models for regression, one of the

basis functions is typically set to a constant, say $\phi_0(\mathbf{x}) = 1$, so that the corresponding parameter w_0 plays the role of a bias. For the remainder of this chapter, we shall include a fixed basis function transformation $\boldsymbol{\phi}(\mathbf{x})$, as this will highlight some useful similarities to the regression models discussed in Chapter 3.

For many problems of practical interest, there is significant overlap between the class-conditional densities $p(\mathbf{x}|\mathcal{C}_k)$. This corresponds to posterior probabilities $p(\mathcal{C}_k|\mathbf{x})$, which, for at least some values of $\mathbf{x}$, are not 0 or 1. In such cases, the optimal solution is obtained by modelling the posterior probabilities accurately and then applying standard decision theory, as discussed in Chapter 1. Note that nonlinear transformations $\boldsymbol{\phi}(\mathbf{x})$ cannot remove such class overlap. Indeed, they can increase the level of overlap, or create overlap where none existed in the original observation space. However, suitable choices of nonlinearity can make the process of modelling the posterior probabilities easier.

Section 3.6

Such fixed basis function models have important limitations, and these will be resolved in later chapters by allowing the basis functions themselves to adapt to the data. Notwithstanding these limitations, models with fixed nonlinear basis functions play an important role in applications, and a discussion of such models will introduce many of the key concepts needed for an understanding of their more complex counterparts.

4.3.2 Logistic regression

We begin our treatment of generalized linear models by considering the problem of two-class classification. In our discussion of generative approaches in Section 4.2, we saw that under rather general assumptions, the posterior probability of class $\mathcal{C}_1$ can be written as a logistic sigmoid acting on a linear function of the feature vector $\boldsymbol{\phi}$ so that

$$p(\mathcal{C}_1|\boldsymbol{\phi}) = y(\boldsymbol{\phi}) = \sigma\left(\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}\right) \tag{4.87}$$

with $p(\mathcal{C}_2|\boldsymbol{\phi}) = 1 - p(\mathcal{C}_1|\boldsymbol{\phi})$. Here $\sigma(\cdot)$ is the *logistic sigmoid* function defined by (4.59). In the terminology of statistics, this model is known as *logistic regression*, although it should be emphasized that this is a model for classification rather than regression.

For an M-dimensional feature space $\boldsymbol{\phi}$, this model has M adjustable parameters. By contrast, if we had fitted Gaussian class conditional densities using maximum likelihood, we would have used $2M$ parameters for the means and $M(M+1)/2$ parameters for the (shared) covariance matrix. Together with the class prior $p(\mathcal{C}_1)$, this gives a total of $M(M+5)/2+1$ parameters, which grows quadratically with M, in contrast to the linear dependence on M of the number of parameters in logistic regression. For large values of M, there is a clear advantage in working with the logistic regression model directly.

We now use maximum likelihood to determine the parameters of the logistic regression model. To do this, we shall make use of the derivative of the logistic sigmoid function, which can conveniently be expressed in terms of the sigmoid function itself

Exercise 4.12

$$\frac{d\sigma}{da} = \sigma(1-\sigma). \tag{4.88}$$

For a data set $\{\boldsymbol{\phi}_n, t_n\}$, where $t_n \in \{0, 1\}$ and $\boldsymbol{\phi}_n = \boldsymbol{\phi}(\mathbf{x}_n)$, with $n = 1, \ldots, N$, the likelihood function can be written

$$p(\mathbf{t}|\mathbf{w}) = \prod_{n=1}^{N} y_n^{t_n} \{1 - y_n\}^{1-t_n} \tag{4.89}$$

where $\mathbf{t} = (t_1, \ldots, t_N)^{\mathrm{T}}$ and $y_n = p(\mathcal{C}_1|\boldsymbol{\phi}_n)$. As usual, we can define an error function by taking the negative logarithm of the likelihood, which gives the *cross-entropy* error function in the form

$$E(\mathbf{w}) = -\ln p(\mathbf{t}|\mathbf{w}) = -\sum_{n=1}^{N} \{t_n \ln y_n + (1 - t_n) \ln(1 - y_n)\} \tag{4.90}$$

where $y_n = \sigma(a_n)$ and $a_n = \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}_n$. Taking the gradient of the error function with respect to $\mathbf{w}$, we obtain

Exercise 4.13

$$\nabla E(\mathbf{w}) = \sum_{n=1}^{N} (y_n - t_n)\boldsymbol{\phi}_n \tag{4.91}$$

where we have made use of (4.88). We see that the factor involving the derivative of the logistic sigmoid has cancelled, leading to a simplified form for the gradient of the log likelihood. In particular, the contribution to the gradient from data point n is given by the 'error' $y_n - t_n$ between the target value and the prediction of the model, times the basis function vector $\boldsymbol{\phi}_n$. Furthermore, comparison with (3.13) shows that this takes precisely the same form as the gradient of the sum-of-squares error function for the linear regression model.

Section 3.1.1

If desired, we could make use of the result (4.91) to give a sequential algorithm in which patterns are presented one at a time, in which each of the weight vectors is updated using (3.22) in which ∇E_n is the n^{th} term in (4.91).

It is worth noting that maximum likelihood can exhibit severe over-fitting for data sets that are linearly separable. This arises because the maximum likelihood solution occurs when the hyperplane corresponding to $\sigma = 0.5$, equivalent to $\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi} = 0$, separates the two classes and the magnitude of $\mathbf{w}$ goes to infinity. In this case, the logistic sigmoid function becomes infinitely steep in feature space, corresponding to a Heaviside step function, so that every training point from each class k is assigned a posterior probability $p(\mathcal{C}_k|\mathbf{x}) = 1$. Furthermore, there is typically a continuum of such solutions because any separating hyperplane will give rise to the same posterior probabilities at the training data points, as will be seen later in Figure 10.13. Maximum likelihood provides no way to favour one such solution over another, and which solution is found in practice will depend on the choice of optimization algorithm and on the parameter initialization. Note that the problem will arise even if the number of data points is large compared with the number of parameters in the model, so long as the training data set is linearly separable. The singularity can be avoided by inclusion of a prior and finding a MAP solution for $\mathbf{w}$, or equivalently by adding a regularization term to the error function.

Exercise 4.14

4.3.3 Iterative reweighted least squares

In the case of the linear regression models discussed in Chapter 3, the maximum likelihood solution, on the assumption of a Gaussian noise model, leads to a closed-form solution. This was a consequence of the quadratic dependence of the log likelihood function on the parameter vector $\mathbf{w}$. For logistic regression, there is no longer a closed-form solution, due to the nonlinearity of the logistic sigmoid function. However, the departure from a quadratic form is not substantial. To be precise, the error function is concave, as we shall see shortly, and hence has a unique minimum. Furthermore, the error function can be minimized by an efficient iterative technique based on the *Newton-Raphson* iterative optimization scheme, which uses a local quadratic approximation to the log likelihood function. The Newton-Raphson update, for minimizing a function $E(\mathbf{w})$, takes the form (Fletcher, 1987; Bishop and Nabney, 2008)

$$\mathbf{w}^{(\text{new})} = \mathbf{w}^{(\text{old})} - \mathbf{H}^{-1}\nabla E(\mathbf{w}). \tag{4.92}$$

where $\mathbf{H}$ is the Hessian matrix whose elements comprise the second derivatives of $E(\mathbf{w})$ with respect to the components of $\mathbf{w}$.

Let us first of all apply the Newton-Raphson method to the linear regression model (3.3) with the sum-of-squares error function (3.12). The gradient and Hessian of this error function are given by

$$\nabla E(\mathbf{w}) = \sum_{n=1}^{N}(\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}_n - t_n)\boldsymbol{\phi}_n = \boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}\mathbf{w} - \boldsymbol{\Phi}^{\mathrm{T}}\mathbf{t} \tag{4.93}$$

$$\mathbf{H} = \nabla\nabla E(\mathbf{w}) = \sum_{n=1}^{N}\boldsymbol{\phi}_n\boldsymbol{\phi}_n^{\mathrm{T}} = \boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi} \tag{4.94}$$

Section 3.1.1

where $\boldsymbol{\Phi}$ is the $N \times M$ design matrix, whose n^{th} row is given by $\boldsymbol{\phi}_n^{\mathrm{T}}$. The Newton-Raphson update then takes the form

$$\begin{aligned}\mathbf{w}^{(\text{new})} &= \mathbf{w}^{(\text{old})} - (\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi})^{-1}\left\{\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}\mathbf{w}^{(\text{old})} - \boldsymbol{\Phi}^{\mathrm{T}}\mathbf{t}\right\} \\ &= (\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi})^{-1}\boldsymbol{\Phi}^{\mathrm{T}}\mathbf{t}\end{aligned} \tag{4.95}$$

which we recognize as the standard least-squares solution. Note that the error function in this case is quadratic and hence the Newton-Raphson formula gives the exact solution in one step.

Now let us apply the Newton-Raphson update to the cross-entropy error function (4.90) for the logistic regression model. From (4.91) we see that the gradient and Hessian of this error function are given by

$$\nabla E(\mathbf{w}) = \sum_{n=1}^{N}(y_n - t_n)\boldsymbol{\phi}_n = \boldsymbol{\Phi}^{\mathrm{T}}(\mathbf{y} - \mathbf{t}) \tag{4.96}$$

$$\mathbf{H} = \nabla\nabla E(\mathbf{w}) = \sum_{n=1}^{N} y_n(1 - y_n)\boldsymbol{\phi}_n\boldsymbol{\phi}_n^{\mathrm{T}} = \boldsymbol{\Phi}^{\mathrm{T}}\mathbf{R}\boldsymbol{\Phi} \tag{4.97}$$

where we have made use of (4.88). Also, we have introduced the $N \times N$ diagonal matrix $\mathbf{R}$ with elements

$$R_{nn} = y_n(1 - y_n). \tag{4.98}$$

We see that the Hessian is no longer constant but depends on $\mathbf{w}$ through the weighting matrix $\mathbf{R}$, corresponding to the fact that the error function is no longer quadratic. Using the property $0 < y_n < 1$, which follows from the form of the logistic sigmoid function, we see that $\mathbf{u}^{\mathrm{T}}\mathbf{H}\mathbf{u} > 0$ for an arbitrary vector $\mathbf{u}$, and so the Hessian matrix $\mathbf{H}$ is positive definite. It follows that the error function is a concave function of $\mathbf{w}$ and hence has a unique minimum.

Exercise 4.15

The Newton-Raphson update formula for the logistic regression model then becomes

$$\begin{aligned} \mathbf{w}^{(\mathrm{new})} &= \mathbf{w}^{(\mathrm{old})} - (\mathbf{\Phi}^{\mathrm{T}}\mathbf{R}\mathbf{\Phi})^{-1}\mathbf{\Phi}^{\mathrm{T}}(\mathbf{y} - \mathbf{t}) \\ &= (\mathbf{\Phi}^{\mathrm{T}}\mathbf{R}\mathbf{\Phi})^{-1}\left\{\mathbf{\Phi}^{\mathrm{T}}\mathbf{R}\mathbf{\Phi}\mathbf{w}^{(\mathrm{old})} - \mathbf{\Phi}^{\mathrm{T}}(\mathbf{y} - \mathbf{t})\right\} \\ &= (\mathbf{\Phi}^{\mathrm{T}}\mathbf{R}\mathbf{\Phi})^{-1}\mathbf{\Phi}^{\mathrm{T}}\mathbf{R}\mathbf{z} \end{aligned} \tag{4.99}$$

where $\mathbf{z}$ is an N-dimensional vector with elements

$$\mathbf{z} = \mathbf{\Phi}\mathbf{w}^{(\mathrm{old})} - \mathbf{R}^{-1}(\mathbf{y} - \mathbf{t}). \tag{4.100}$$

We see that the update formula (4.99) takes the form of a set of normal equations for a weighted least-squares problem. Because the weighing matrix $\mathbf{R}$ is not constant but depends on the parameter vector $\mathbf{w}$, we must apply the normal equations iteratively, each time using the new weight vector $\mathbf{w}$ to compute a revised weighing matrix $\mathbf{R}$. For this reason, the algorithm is known as *iterative reweighted least squares*, or *IRLS* (Rubin, 1983). As in the weighted least-squares problem, the elements of the diagonal weighting matrix $\mathbf{R}$ can be interpreted as variances because the mean and variance of t in the logistic regression model are given by

$$\mathbb{E}[t] = \sigma(\mathbf{x}) = y \tag{4.101}$$

$$\mathrm{var}[t] = \mathbb{E}[t^2] - \mathbb{E}[t]^2 = \sigma(\mathbf{x}) - \sigma(\mathbf{x})^2 = y(1 - y) \tag{4.102}$$

where we have used the property $t^2 = t$ for $t \in \{0, 1\}$. In fact, we can interpret IRLS as the solution to a linearized problem in the space of the variable $a = \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}$. The quantity z_n, which corresponds to the n^{th} element of $\mathbf{z}$, can then be given a simple interpretation as an effective target value in this space obtained by making a local linear approximation to the logistic sigmoid function around the current operating point $\mathbf{w}^{(\mathrm{old})}$

$$\begin{aligned} a_n(\mathbf{w}) &\simeq a_n(\mathbf{w}^{(\mathrm{old})}) + \left.\frac{\mathrm{d}a_n}{\mathrm{d}y_n}\right|_{\mathbf{w}^{(\mathrm{old})}}(t_n - y_n) \\ &= \boldsymbol{\phi}_n^{\mathrm{T}}\mathbf{w}^{(\mathrm{old})} - \frac{(y_n - t_n)}{y_n(1 - y_n)} = z_n. \end{aligned} \tag{4.103}$$

4.3.4 Multiclass logistic regression

Section 4.2

In our discussion of generative models for multiclass classification, we have seen that for a large class of distributions, the posterior probabilities are given by a softmax transformation of linear functions of the feature variables, so that

$$p(\mathcal{C}_k|\boldsymbol{\phi}) = y_k(\boldsymbol{\phi}) = \frac{\exp(a_k)}{\sum_j \exp(a_j)} \tag{4.104}$$

where the 'activations' a_k are given by

$$a_k = \mathbf{w}_k^{\mathrm{T}}\boldsymbol{\phi}. \tag{4.105}$$

There we used maximum likelihood to determine separately the class-conditional densities and the class priors and then found the corresponding posterior probabilities using Bayes' theorem, thereby implicitly determining the parameters $\{\mathbf{w}_k\}$. Here we consider the use of maximum likelihood to determine the parameters $\{\mathbf{w}_k\}$ of this model directly. To do this, we will require the derivatives of y_k with respect to all of the activations a_j. These are given by

Exercise 4.17

$$\frac{\partial y_k}{\partial a_j} = y_k(I_{kj} - y_j) \tag{4.106}$$

where I_{kj} are the elements of the identity matrix.

Next we write down the likelihood function. This is most easily done using the 1-of-K coding scheme in which the target vector $\mathbf{t}_n$ for a feature vector $\boldsymbol{\phi}_n$ belonging to class $\mathcal{C}_k$ is a binary vector with all elements zero except for element k, which equals one. The likelihood function is then given by

$$p(\mathbf{T}|\mathbf{w}_1, \ldots, \mathbf{w}_K) = \prod_{n=1}^{N}\prod_{k=1}^{K} p(\mathcal{C}_k|\boldsymbol{\phi}_n)^{t_{nk}} = \prod_{n=1}^{N}\prod_{k=1}^{K} y_{nk}^{t_{nk}} \tag{4.107}$$

where $y_{nk} = y_k(\boldsymbol{\phi}_n)$, and $\mathbf{T}$ is an $N \times K$ matrix of target variables with elements t_{nk}. Taking the negative logarithm then gives

$$E(\mathbf{w}_1, \ldots, \mathbf{w}_K) = -\ln p(\mathbf{T}|\mathbf{w}_1, \ldots, \mathbf{w}_K) = -\sum_{n=1}^{N}\sum_{k=1}^{K} t_{nk} \ln y_{nk} \tag{4.108}$$

which is known as the *cross-entropy* error function for the multiclass classification problem.

We now take the gradient of the error function with respect to one of the parameter vectors $\mathbf{w}_j$. Making use of the result (4.106) for the derivatives of the softmax function, we obtain

Exercise 4.18

$$\nabla_{\mathbf{w}_j} E(\mathbf{w}_1, \ldots, \mathbf{w}_K) = \sum_{n=1}^{N} (y_{nj} - t_{nj})\,\boldsymbol{\phi}_n \tag{4.109}$$

where we have made use of $\sum_k t_{nk} = 1$. Once again, we see the same form arising for the gradient as was found for the sum-of-squares error function with the linear model and the cross-entropy error for the logistic regression model, namely the product of the error $(y_{nj} - t_{nj})$ times the basis function $\boldsymbol{\phi}_n$. Again, we could use this to formulate a sequential algorithm in which patterns are presented one at a time, in which each of the weight vectors is updated using (3.22).

We have seen that the derivative of the log likelihood function for a linear regression model with respect to the parameter vector $\mathbf{w}$ for a data point n took the form of the 'error' $y_n - t_n$ times the feature vector $\boldsymbol{\phi}_n$. Similarly, for the combination of logistic sigmoid activation function and cross-entropy error function (4.90), and for the softmax activation function with the multiclass cross-entropy error function (4.108), we again obtain this same simple form. This is an example of a more general result, as we shall see in Section 4.3.6.

To find a batch algorithm, we again appeal to the Newton-Raphson update to obtain the corresponding IRLS algorithm for the multiclass problem. This requires evaluation of the Hessian matrix that comprises blocks of size $M \times M$ in which block j, k is given by

$$\nabla_{\mathbf{w}_k} \nabla_{\mathbf{w}_j} E(\mathbf{w}_1, \ldots, \mathbf{w}_K) = -\sum_{n=1}^{N} y_{nk}(I_{kj} - y_{nj})\boldsymbol{\phi}_n \boldsymbol{\phi}_n^{\mathrm{T}}. \tag{4.110}$$

Exercise 4.20

As with the two-class problem, the Hessian matrix for the multiclass logistic regression model is positive definite and so the error function again has a unique minimum. Practical details of IRLS for the multiclass case can be found in Bishop and Nabney (2008).

4.3.5 Probit regression

We have seen that, for a broad range of class-conditional distributions, described by the exponential family, the resulting posterior class probabilities are given by a logistic (or softmax) transformation acting on a linear function of the feature variables. However, not all choices of class-conditional density give rise to such a simple form for the posterior probabilities (for instance, if the class-conditional densities are modelled using Gaussian mixtures). This suggests that it might be worth exploring other types of discriminative probabilistic model. For the purposes of this chapter, however, we shall return to the two-class case, and again remain within the framework of generalized linear models so that

$$p(t = 1|a) = f(a) \tag{4.111}$$

where $a = \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}$, and $f(\cdot)$ is the activation function.

One way to motivate an alternative choice for the link function is to consider a noisy threshold model, as follows. For each input $\boldsymbol{\phi}_n$, we evaluate $a_n = \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}_n$ and then we set the target value according to

$$\begin{cases} t_n = 1 & \text{if } a_n \geqslant \theta \\ t_n = 0 & \text{otherwise.} \end{cases} \tag{4.112}$$

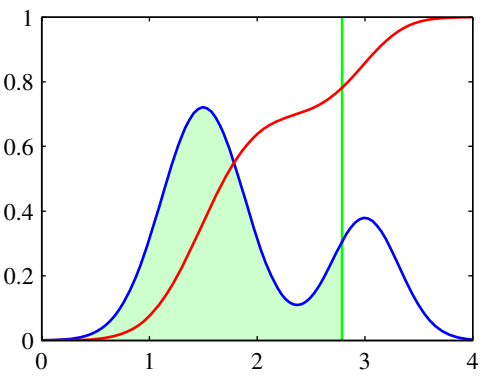


Figure 4.13 Schematic example of a probability density $p(\theta)$ shown by the blue curve, given in this example by a mixture of two Gaussians, along with its cumulative distribution function $f(a)$, shown by the red curve. Note that the value of the blue curve at any point, such as that indicated by the vertical green line, corresponds to the slope of the red curve at the same point. Conversely, the value of the red curve at this point corresponds to the area under the blue curve indicated by the shaded green region. In the stochastic threshold model, the class label takes the value $t = 1$ if the value of $a = \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}$ exceeds a threshold, otherwise it takes the value $t = 0$. This is equivalent to an activation function given by the cumulative distribution function $f(a)$.

If the value of θ is drawn from a probability density $p(\theta)$, then the corresponding activation function will be given by the cumulative distribution function

$$f(a) = \int_{-\infty}^{a} p(\theta)\,\mathrm{d}\theta \tag{4.113}$$

as illustrated in Figure 4.13.

As a specific example, suppose that the density $p(\theta)$ is given by a zero mean, unit variance Gaussian. The corresponding cumulative distribution function is given by

$$\Phi(a) = \int_{-\infty}^{a} \mathcal{N}(\theta|0,1)\,\mathrm{d}\theta \tag{4.114}$$

which is known as the *probit* function. It has a sigmoidal shape and is compared with the logistic sigmoid function in Figure 4.9. Note that the use of a more general Gaussian distribution does not change the model because this is equivalent to a re-scaling of the linear coefficients $\mathbf{w}$. Many numerical packages provide for the evaluation of a closely related function defined by

$$\mathrm{erf}(a) = \frac{2}{\sqrt{\pi}} \int_{0}^{a} \exp(-\theta^2/2)\,\mathrm{d}\theta \tag{4.115}$$

and known as the *erf function* or *error function* (not to be confused with the error function of a machine learning model). It is related to the probit function by

Exercise 4.21

$$\boldsymbol{\Phi}(a) = \frac{1}{2}\left\{1 + \frac{1}{\sqrt{2}}\mathrm{erf}(a)\right\}. \tag{4.116}$$

The generalized linear model based on a probit activation function is known as *probit regression*.

We can determine the parameters of this model using maximum likelihood, by a straightforward extension of the ideas discussed earlier. In practice, the results found using probit regression tend to be similar to those of logistic regression. We shall,

however, find another use for the probit model when we discuss Bayesian treatments of logistic regression in Section 4.5.

One issue that can occur in practical applications is that of *outliers*, which can arise for instance through errors in measuring the input vector $\mathbf{x}$ or through mislabelling of the target value t. Because such points can lie a long way to the wrong side of the ideal decision boundary, they can seriously distort the classifier. Note that the logistic and probit regression models behave differently in this respect because the tails of the logistic sigmoid decay asymptotically like $\exp(-x)$ for $x \to \infty$, whereas for the probit activation function they decay like $\exp(-x^2)$, and so the probit model can be significantly more sensitive to outliers.

However, both the logistic and the probit models assume the data is correctly labelled. The effect of mislabelling is easily incorporated into a probabilistic model by introducing a probability ϵ that the target value t has been flipped to the wrong value (Opper and Winther, 2000a), leading to a target value distribution for data point $\mathbf{x}$ of the form

$$\begin{aligned} p(t|\mathbf{x}) &= (1-\epsilon)\sigma(\mathbf{x}) + \epsilon(1-\sigma(\mathbf{x})) \\ &= \epsilon + (1-2\epsilon)\sigma(\mathbf{x}) \end{aligned} \tag{4.117}$$

where $\sigma(\mathbf{x})$ is the activation function with input vector $\mathbf{x}$. Here ϵ may be set in advance, or it may be treated as a hyperparameter whose value is inferred from the data.

4.3.6 Canonical link functions

For the linear regression model with a Gaussian noise distribution, the error function, corresponding to the negative log likelihood, is given by (3.12). If we take the derivative with respect to the parameter vector $\mathbf{w}$ of the contribution to the error function from a data point n, this takes the form of the 'error' $y_n - t_n$ times the feature vector $\boldsymbol{\phi}_n$, where $y_n = \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}_n$. Similarly, for the combination of the logistic sigmoid activation function and the cross-entropy error function (4.90), and for the softmax activation function with the multiclass cross-entropy error function (4.108), we again obtain this same simple form. We now show that this is a general result of assuming a conditional distribution for the target variable from the exponential family, along with a corresponding choice for the activation function known as the *canonical link function*.

We again make use of the restricted form (4.84) of exponential family distributions. Note that here we are applying the assumption of exponential family distribution to the target variable t, in contrast to Section 4.2.4 where we applied it to the input vector $\mathbf{x}$. We therefore consider conditional distributions of the target variable of the form

$$p(t|\eta, s) = \frac{1}{s} h\left(\frac{t}{s}\right) g(\eta) \exp\left\{\frac{\eta t}{s}\right\}. \tag{4.118}$$

Using the same line of argument as led to the derivation of the result (2.226), we see that the conditional mean of t, which we denote by y, is given by

$$y \equiv \mathbb{E}[t|\eta] = -s\frac{d}{d\eta}\ln g(\eta). \tag{4.119}$$

Thus y and η must related, and we denote this relation through $\eta = \psi(y)$.

Following Nelder and Wedderburn (1972), we define a *generalized linear model* to be one for which y is a nonlinear function of a linear combination of the input (or feature) variables so that

$$y = f(\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}) \tag{4.120}$$

where $f(\cdot)$ is known as the *activation function* in the machine learning literature, and $f^{-1}(\cdot)$ is known as the *link function* in statistics.

Now consider the log likelihood function for this model, which, as a function of η, is given by

$$\ln p(\mathbf{t}|\eta, s) = \sum_{n=1}^{N} \ln p(t_n|\eta, s) = \sum_{n=1}^{N} \left\{ \ln g(\eta_n) + \frac{\eta_n t_n}{s} \right\} + \text{const} \tag{4.121}$$

where we are assuming that all observations share a common scale parameter (which corresponds to the noise variance for a Gaussian distribution for instance) and so s is independent of n. The derivative of the log likelihood with respect to the model parameters $\mathbf{w}$ is then given by

$$\begin{aligned} \nabla_{\mathbf{w}} \ln p(\mathbf{t}|\eta, s) &= \sum_{n=1}^{N} \left\{ \frac{d}{d\eta_n} \ln g(\eta_n) + \frac{t_n}{s} \right\} \frac{d\eta_n}{dy_n} \frac{dy_n}{da_n} \nabla a_n \\ &= \sum_{n=1}^{N} \frac{1}{s} \{t_n - y_n\} \psi'(y_n) f'(a_n) \boldsymbol{\phi}_n \end{aligned} \tag{4.122}$$

where $a_n = \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}_n$, and we have used $y_n = f(a_n)$ together with the result (4.119) for $\mathbb{E}[t|\eta]$. We now see that there is a considerable simplification if we choose a particular form for the link function $f^{-1}(y)$ given by

$$f^{-1}(y) = \psi(y) \tag{4.123}$$

which gives $f(\psi(y)) = y$ and hence $f'(\psi)\psi'(y) = 1$. Also, because $a = f^{-1}(y)$, we have $a = \psi$ and hence $f'(a)\psi'(y) = 1$. In this case, the gradient of the error function reduces to

$$\nabla \ln E(\mathbf{w}) = \frac{1}{s} \sum_{n=1}^{N} \{y_n - t_n\} \boldsymbol{\phi}_n. \tag{4.124}$$

For the Gaussian $s = \beta^{-1}$, whereas for the logistic model $s = 1$.

4.4. The Laplace Approximation

In Section 4.5 we shall discuss the Bayesian treatment of logistic regression. As we shall see, this is more complex than the Bayesian treatment of linear regression models, discussed in Sections 3.3 and 3.5. In particular, we cannot integrate exactly

over the parameter vector $\mathbf{w}$ since the posterior distribution is no longer Gaussian. It is therefore necessary to introduce some form of approximation. Later in the book we shall consider a range of techniques based on analytical approximations and numerical sampling.

Chapter 10
Chapter 11

Here we introduce a simple, but widely used, framework called the Laplace approximation, that aims to find a Gaussian approximation to a probability density defined over a set of continuous variables. Consider first the case of a single continuous variable z, and suppose the distribution $p(z)$ is defined by

$$p(z) = \frac{1}{Z} f(z) \tag{4.125}$$

where $Z = \int f(z)\,\mathrm{d}z$ is the normalization coefficient. We shall suppose that the value of Z is unknown. In the Laplace method the goal is to find a Gaussian approximation $q(z)$ which is centred on a mode of the distribution $p(z)$. The first step is to find a mode of $p(z)$, in other words a point z_0 such that $p'(z_0) = 0$, or equivalently

$$\left.\frac{df(z)}{dz}\right|_{z=z_0} = 0. \tag{4.126}$$

A Gaussian distribution has the property that its logarithm is a quadratic function of the variables. We therefore consider a Taylor expansion of $\ln f(z)$ centred on the mode z_0 so that

$$\ln f(z) \simeq \ln f(z_0) - \frac{1}{2} A (z - z_0)^2 \tag{4.127}$$

where

$$A = -\left.\frac{d^2}{dz^2} \ln f(z)\right|_{z=z_0}. \tag{4.128}$$

Note that the first-order term in the Taylor expansion does not appear since z_0 is a local maximum of the distribution. Taking the exponential we obtain

$$f(z) \simeq f(z_0) \exp\left\{-\frac{A}{2}(z - z_0)^2\right\}. \tag{4.129}$$

We can then obtain a normalized distribution $q(z)$ by making use of the standard result for the normalization of a Gaussian, so that

$$q(z) = \left(\frac{A}{2\pi}\right)^{1/2} \exp\left\{-\frac{A}{2}(z - z_0)^2\right\}. \tag{4.130}$$

The Laplace approximation is illustrated in Figure 4.14. Note that the Gaussian approximation will only be well defined if its precision $A > 0$, in other words the stationary point z_0 must be a local maximum, so that the second derivative of $f(z)$ at the point z_0 is negative.

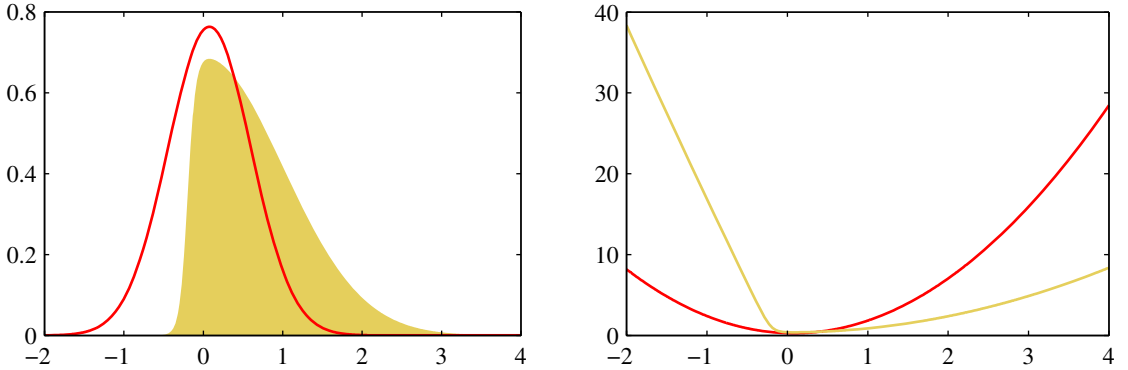


Figure 4.14 Illustration of the Laplace approximation applied to the distribution $p(z) \propto \exp(-z^2/2)\sigma(20z + 4)$ where $\sigma(z)$ is the logistic sigmoid function defined by $\sigma(z) = (1 + e^{-z})^{-1}$. The left plot shows the normalized distribution $p(z)$ in yellow, together with the Laplace approximation centred on the mode z_0 of $p(z)$ in red. The right plot shows the negative logarithms of the corresponding curves.

We can extend the Laplace method to approximate a distribution $p(\mathbf{z}) = f(\mathbf{z})/Z$ defined over an M-dimensional space $\mathbf{z}$. At a stationary point $\mathbf{z}_0$ the gradient $\nabla f(\mathbf{z})$ will vanish. Expanding around this stationary point we have

$$\ln f(\mathbf{z}) \simeq \ln f(\mathbf{z}_0) - \frac{1}{2}(\mathbf{z} - \mathbf{z}_0)^{\mathrm{T}}\mathbf{A}(\mathbf{z} - \mathbf{z}_0) \tag{4.131}$$

where the $M \times M$ Hessian matrix $\mathbf{A}$ is defined by

$$\mathbf{A} = -\nabla\nabla \ln f(\mathbf{z})|_{\mathbf{z}=\mathbf{z}_0} \tag{4.132}$$

and ∇ is the gradient operator. Taking the exponential of both sides we obtain

$$f(\mathbf{z}) \simeq f(\mathbf{z}_0) \exp\left\{-\frac{1}{2}(\mathbf{z} - \mathbf{z}_0)^{\mathrm{T}}\mathbf{A}(\mathbf{z} - \mathbf{z}_0)\right\}. \tag{4.133}$$

The distribution $q(\mathbf{z})$ is proportional to $f(\mathbf{z})$ and the appropriate normalization coefficient can be found by inspection, using the standard result (2.43) for a normalized multivariate Gaussian, giving

$$q(\mathbf{z}) = \frac{|\mathbf{A}|^{1/2}}{(2\pi)^{M/2}} \exp\left\{-\frac{1}{2}(\mathbf{z} - \mathbf{z}_0)^{\mathrm{T}}\mathbf{A}(\mathbf{z} - \mathbf{z}_0)\right\} = \mathcal{N}(\mathbf{z}|\mathbf{z}_0, \mathbf{A}^{-1}) \tag{4.134}$$

where $|\mathbf{A}|$ denotes the determinant of $\mathbf{A}$. This Gaussian distribution will be well defined provided its precision matrix, given by $\mathbf{A}$, is positive definite, which implies that the stationary point $\mathbf{z}_0$ must be a local maximum, not a minimum or a saddle point.

In order to apply the Laplace approximation we first need to find the mode $\mathbf{z}_0$, and then evaluate the Hessian matrix at that mode. In practice a mode will typically be found by running some form of numerical optimization algorithm (Bishop

and Nabney, 2008). Many of the distributions encountered in practice will be multimodal and so there will be different Laplace approximations according to which mode is being considered. Note that the normalization constant Z of the true distribution does not need to be known in order to apply the Laplace method. As a result of the central limit theorem, the posterior distribution for a model is expected to become increasingly better approximated by a Gaussian as the number of observed data points is increased, and so we would expect the Laplace approximation to be most useful in situations where the number of data points is relatively large.

One major weakness of the Laplace approximation is that, since it is based on a Gaussian distribution, it is only directly applicable to real variables. In other cases it may be possible to apply the Laplace approximation to a transformation of the variable. For instance if $0 \leqslant \tau < \infty$ then we can consider a Laplace approximation of $\ln \tau$. The most serious limitation of the Laplace framework, however, is that it is based purely on the aspects of the true distribution at a specific value of the variable, and so can fail to capture important global properties. In Chapter 10 we shall consider alternative approaches which adopt a more global perspective.

4.4.1 Model comparison and BIC

As well as approximating the distribution $p(\mathbf{z})$ we can also obtain an approximation to the normalization constant Z. Using the approximation (4.133) we have

$$\begin{aligned} Z &= \int f(\mathbf{z})\,\mathrm{d}\mathbf{z} \\ &\simeq f(\mathbf{z}_0) \int \exp\left\{-\frac{1}{2}(\mathbf{z}-\mathbf{z}_0)^{\mathrm{T}}\mathbf{A}(\mathbf{z}-\mathbf{z}_0)\right\}\mathrm{d}\mathbf{z} \\ &= f(\mathbf{z}_0)\frac{(2\pi)^{M/2}}{|\mathbf{A}|^{1/2}} \end{aligned} \tag{4.135}$$

where we have noted that the integrand is Gaussian and made use of the standard result (2.43) for a normalized Gaussian distribution. We can use the result (4.135) to obtain an approximation to the model evidence which, as discussed in Section 3.4, plays a central role in Bayesian model comparison.

Consider a data set $\mathcal{D}$ and a set of models $\{\mathcal{M}_i\}$ having parameters $\{\boldsymbol{\theta}_i\}$. For each model we define a likelihood function $p(\mathcal{D}|\boldsymbol{\theta}_i, \mathcal{M}_i)$. If we introduce a prior $p(\boldsymbol{\theta}_i|\mathcal{M}_i)$ over the parameters, then we are interested in computing the model evidence $p(\mathcal{D}|\mathcal{M}_i)$ for the various models. From now on we omit the conditioning on $\mathcal{M}_i$ to keep the notation uncluttered. From Bayes' theorem the model evidence is given by

$$p(\mathcal{D}) = \int p(\mathcal{D}|\boldsymbol{\theta})p(\boldsymbol{\theta})\,\mathrm{d}\boldsymbol{\theta}. \tag{4.136}$$

Identifying $f(\boldsymbol{\theta}) = p(\mathcal{D}|\boldsymbol{\theta})p(\boldsymbol{\theta})$ and $Z = p(\mathcal{D})$, and applying the result (4.135), we obtain

Exercise 4.22

$$\ln p(\mathcal{D}) \simeq \ln p(\mathcal{D}|\boldsymbol{\theta}_{\mathrm{MAP}}) + \underbrace{\ln p(\boldsymbol{\theta}_{\mathrm{MAP}}) + \frac{M}{2}\ln(2\pi) - \frac{1}{2}\ln|\mathbf{A}|}_{\text{Occam factor}} \tag{4.137}$$

where $\boldsymbol{\theta}_{\text{MAP}}$ is the value of $\boldsymbol{\theta}$ at the mode of the posterior distribution, and $\mathbf{A}$ is the *Hessian* matrix of second derivatives of the negative log posterior

$$\mathbf{A} = -\nabla\nabla \ln p(\mathcal{D}|\boldsymbol{\theta}_{\text{MAP}})p(\boldsymbol{\theta}_{\text{MAP}}) = -\nabla\nabla \ln p(\boldsymbol{\theta}_{\text{MAP}}|\mathcal{D}). \tag{4.138}$$

The first term on the right hand side of (4.137) represents the log likelihood evaluated using the optimized parameters, while the remaining three terms comprise the 'Occam factor' which penalizes model complexity.

If we assume that the Gaussian prior distribution over parameters is broad, and that the Hessian has full rank, then we can approximate (4.137) very roughly using

Exercise 4.23

$$\ln p(\mathcal{D}) \simeq \ln p(\mathcal{D}|\boldsymbol{\theta}_{\text{MAP}}) - \frac{1}{2}M \ln N \tag{4.139}$$

where N is the number of data points, M is the number of parameters in $\boldsymbol{\theta}$ and we have omitted additive constants. This is known as the *Bayesian Information Criterion* (BIC) or the *Schwarz criterion* (Schwarz, 1978). Note that, compared to AIC given by (1.73), this penalizes model complexity more heavily.

Complexity measures such as AIC and BIC have the virtue of being easy to evaluate, but can also give misleading results. In particular, the assumption that the Hessian matrix has full rank is often not valid since many of the parameters are not 'well-determined'. We can use the result (4.137) to obtain a more accurate estimate of the model evidence starting from the Laplace approximation, as we illustrate in the context of neural networks in Section 5.7.

Section 3.5.3

4.5. Bayesian Logistic Regression

We now turn to a Bayesian treatment of logistic regression. Exact Bayesian inference for logistic regression is intractable. In particular, evaluation of the posterior distribution would require normalization of the product of a prior distribution and a likelihood function that itself comprises a product of logistic sigmoid functions, one for every data point. Evaluation of the predictive distribution is similarly intractable. Here we consider the application of the Laplace approximation to the problem of Bayesian logistic regression (Spiegelhalter and Lauritzen, 1990; MacKay, 1992b).

4.5.1 Laplace approximation

Recall from Section 4.4 that the Laplace approximation is obtained by finding the mode of the posterior distribution and then fitting a Gaussian centred at that mode. This requires evaluation of the second derivatives of the log posterior, which is equivalent to finding the Hessian matrix.

Because we seek a Gaussian representation for the posterior distribution, it is natural to begin with a Gaussian prior, which we write in the general form

$$p(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\mathbf{m}_0, \mathbf{S}_0) \tag{4.140}$$

where $\mathbf{m}_0$ and $\mathbf{S}_0$ are fixed hyperparameters. The posterior distribution over $\mathbf{w}$ is given by

$$p(\mathbf{w}|\mathbf{t}) \propto p(\mathbf{w})p(\mathbf{t}|\mathbf{w}) \tag{4.141}$$

where $\mathbf{t} = (t_1, \ldots, t_N)^{\mathrm{T}}$. Taking the log of both sides, and substituting for the prior distribution using (4.140), and for the likelihood function using (4.89), we obtain

$$\begin{aligned} \ln p(\mathbf{w}|\mathbf{t}) &= -\frac{1}{2}(\mathbf{w}-\mathbf{m}_0)^{\mathrm{T}}\mathbf{S}_0^{-1}(\mathbf{w}-\mathbf{m}_0) \\ &\quad + \sum_{n=1}^{N}\{t_n \ln y_n + (1-t_n)\ln(1-y_n)\} + \text{const} \end{aligned} \tag{4.142}$$

where $y_n = \sigma(\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}_n)$. To obtain a Gaussian approximation to the posterior distribution, we first maximize the posterior distribution to give the MAP (maximum posterior) solution $\mathbf{w}_{\mathrm{MAP}}$, which defines the mean of the Gaussian. The covariance is then given by the inverse of the matrix of second derivatives of the negative log likelihood, which takes the form

$$\mathbf{S}_N = -\nabla\nabla \ln p(\mathbf{w}|\mathbf{t}) = \mathbf{S}_0^{-1} + \sum_{n=1}^{N} y_n(1-y_n)\boldsymbol{\phi}_n\boldsymbol{\phi}_n^{\mathrm{T}}. \tag{4.143}$$

The Gaussian approximation to the posterior distribution therefore takes the form

$$q(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\mathbf{w}_{\mathrm{MAP}}, \mathbf{S}_N). \tag{4.144}$$

Having obtained a Gaussian approximation to the posterior distribution, there remains the task of marginalizing with respect to this distribution in order to make predictions.

4.5.2 Predictive distribution

The predictive distribution for class $\mathcal{C}_1$, given a new feature vector $\boldsymbol{\phi}(\mathbf{x})$, is obtained by marginalizing with respect to the posterior distribution $p(\mathbf{w}|\mathbf{t})$, which is itself approximated by a Gaussian distribution $q(\mathbf{w})$ so that

$$p(\mathcal{C}_1|\boldsymbol{\phi}, \mathbf{t}) = \int p(\mathcal{C}_1|\boldsymbol{\phi}, \mathbf{w})p(\mathbf{w}|\mathbf{t})\,\mathrm{d}\mathbf{w} \simeq \int \sigma(\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi})q(\mathbf{w})\,\mathrm{d}\mathbf{w} \tag{4.145}$$

with the corresponding probability for class $\mathcal{C}_2$ given by $p(\mathcal{C}_2|\boldsymbol{\phi}, \mathbf{t}) = 1 - p(\mathcal{C}_1|\boldsymbol{\phi}, \mathbf{t})$. To evaluate the predictive distribution, we first note that the function $\sigma(\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi})$ depends on $\mathbf{w}$ only through its projection onto $\boldsymbol{\phi}$. Denoting $a = \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}$, we have

$$\sigma(\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}) = \int \delta(a - \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi})\sigma(a)\,\mathrm{d}a \tag{4.146}$$

where $\delta(\cdot)$ is the Dirac delta function. From this we obtain

$$\int \sigma(\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi})q(\mathbf{w})\,\mathrm{d}\mathbf{w} = \int \sigma(a)p(a)\,\mathrm{d}a \tag{4.147}$$

where

$$p(a) = \int \delta(a - \mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}) q(\mathbf{w})\, \mathrm{d}\mathbf{w}. \tag{4.148}$$

We can evaluate $p(a)$ by noting that the delta function imposes a linear constraint on $\mathbf{w}$ and so forms a marginal distribution from the joint distribution $q(\mathbf{w})$ by integrating out all directions orthogonal to $\boldsymbol{\phi}$. Because $q(\mathbf{w})$ is Gaussian, we know from Section 2.3.2 that the marginal distribution will also be Gaussian. We can evaluate the mean and covariance of this distribution by taking moments, and interchanging the order of integration over a and $\mathbf{w}$, so that

$$\mu_a = \mathbb{E}[a] = \int p(a) a \, \mathrm{d}a = \int q(\mathbf{w}) \mathbf{w}^{\mathrm{T}} \boldsymbol{\phi} \, \mathrm{d}\mathbf{w} = \mathbf{w}_{\mathrm{MAP}}^{\mathrm{T}} \boldsymbol{\phi} \tag{4.149}$$

where we have used the result (4.144) for the variational posterior distribution $q(\mathbf{w})$. Similarly

$$\begin{aligned} \sigma_a^2 &= \mathrm{var}[a] = \int p(a) \left\{ a^2 - \mathbb{E}[a]^2 \right\} \mathrm{d}a \\ &= \int q(\mathbf{w}) \left\{ (\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi})^2 - (\mathbf{m}_N^{\mathrm{T}}\boldsymbol{\phi})^2 \right\} \mathrm{d}\mathbf{w} = \boldsymbol{\phi}^{\mathrm{T}} \mathbf{S}_N \boldsymbol{\phi}. \end{aligned} \tag{4.150}$$

Note that the distribution of a takes the same form as the predictive distribution (3.58) for the linear regression model, with the noise variance set to zero. Thus our variational approximation to the predictive distribution becomes

$$p(\mathcal{C}_1|\mathbf{t}) = \int \sigma(a) p(a) \, \mathrm{d}a = \int \sigma(a) \mathcal{N}(a|\mu_a, \sigma_a^2) \, \mathrm{d}a. \tag{4.151}$$

This result can also be derived directly by making use of the results for the marginal of a Gaussian distribution given in Section 2.3.2.

Exercise 4.24

The integral over a represents the convolution of a Gaussian with a logistic sigmoid, and cannot be evaluated analytically. We can, however, obtain a good approximation (Spiegelhalter and Lauritzen, 1990; MacKay, 1992b; Barber and Bishop, 1998a) by making use of the close similarity between the logistic sigmoid function $\sigma(a)$ defined by (4.59) and the probit function $\Phi(a)$ defined by (4.114). In order to obtain the best approximation to the logistic function we need to re-scale the horizontal axis, so that we approximate $\sigma(a)$ by $\Phi(\lambda a)$. We can find a suitable value of λ by requiring that the two functions have the same slope at the origin, which gives $\lambda^2 = \pi/8$. The similarity of the logistic sigmoid and the probit function, for this choice of λ, is illustrated in Figure 4.9.

Exercise 4.25

The advantage of using a probit function is that its convolution with a Gaussian can be expressed analytically in terms of another probit function. Specifically we can show that

Exercise 4.26

$$\int \Phi(\lambda a) \mathcal{N}(a|\mu, \sigma^2) \, \mathrm{d}a = \Phi\left(\frac{\mu}{(\lambda^{-2} + \sigma^2)^{1/2}} \right). \tag{4.152}$$

We now apply the approximation $\sigma(a) \simeq \Phi(\lambda a)$ to the probit functions appearing on both sides of this equation, leading to the following approximation for the convolution of a logistic sigmoid with a Gaussian

$$\int \sigma(a)\mathcal{N}(a|\mu, \sigma^2)\,\mathrm{d}a \simeq \sigma\left(\kappa(\sigma^2)\mu\right) \tag{4.153}$$

where we have defined

$$\kappa(\sigma^2) = (1 + \pi\sigma^2/8)^{-1/2}. \tag{4.154}$$

Applying this result to (4.151) we obtain the approximate predictive distribution in the form

$$p(\mathcal{C}_1|\boldsymbol{\phi}, \mathbf{t}) = \sigma\left(\kappa(\sigma_a^2)\mu_a\right) \tag{4.155}$$

where μ_a and σ_a^2 are defined by (4.149) and (4.150), respectively, and $\kappa(\sigma_a^2)$ is defined by (4.154).

Note that the decision boundary corresponding to $p(\mathcal{C}_1|\boldsymbol{\phi}, \mathbf{t}) = 0.5$ is given by $\mu_a = 0$, which is the same as the decision boundary obtained by using the MAP value for $\mathbf{w}$. Thus if the decision criterion is based on minimizing misclassification rate, with equal prior probabilities, then the marginalization over $\mathbf{w}$ has no effect. However, for more complex decision criteria it will play an important role. Marginalization of the logistic sigmoid model under a Gaussian approximation to the posterior distribution will be illustrated in the context of variational inference in Figure 10.13.

Exercises

4.1 ($\star\star$) Given a set of data points $\{\mathbf{x}_n\}$, we can define the *convex hull* to be the set of all points $\mathbf{x}$ given by

$$\mathbf{x} = \sum_n \alpha_n \mathbf{x}_n \tag{4.156}$$

where $\alpha_n \geqslant 0$ and $\sum_n \alpha_n = 1$. Consider a second set of points $\{\mathbf{y}_n\}$ together with their corresponding convex hull. By definition, the two sets of points will be linearly separable if there exists a vector $\widehat{\mathbf{w}}$ and a scalar w_0 such that $\widehat{\mathbf{w}}^{\mathrm{T}}\mathbf{x}_n + w_0 > 0$ for all $\mathbf{x}_n$, and $\widehat{\mathbf{w}}^{\mathrm{T}}\mathbf{y}_n + w_0 < 0$ for all $\mathbf{y}_n$. Show that if their convex hulls intersect, the two sets of points cannot be linearly separable, and conversely that if they are linearly separable, their convex hulls do not intersect.

4.2 ($\star\star$) **www** Consider the minimization of a sum-of-squares error function (4.15), and suppose that all of the target vectors in the training set satisfy a linear constraint

$$\mathbf{a}^{\mathrm{T}}\mathbf{t}_n + b = 0 \tag{4.157}$$

where $\mathbf{t}_n$ corresponds to the n^{th} row of the matrix $\mathbf{T}$ in (4.15). Show that as a consequence of this constraint, the elements of the model prediction $\mathbf{y}(\mathbf{x})$ given by the least-squares solution (4.17) also satisfy this constraint, so that

$$\mathbf{a}^{\mathrm{T}}\mathbf{y}(\mathbf{x}) + b = 0. \tag{4.158}$$

To do so, assume that one of the basis functions $\phi_0(\mathbf{x}) = 1$ so that the corresponding parameter w_0 plays the role of a bias.

4.3 ($\star\star$) Extend the result of Exercise 4.2 to show that if multiple linear constraints are satisfied simultaneously by the target vectors, then the same constraints will also be satisfied by the least-squares prediction of a linear model.

4.4 ($\star$) **WWW** Show that maximization of the class separation criterion given by (4.23) with respect to $\mathbf{w}$, using a Lagrange multiplier to enforce the constraint $\mathbf{w}^{\mathrm{T}}\mathbf{w} = 1$, leads to the result that $\mathbf{w} \propto (\mathbf{m}_2 - \mathbf{m}_1)$.

4.5 ($\star$) By making use of (4.20), (4.23), and (4.24), show that the Fisher criterion (4.25) can be written in the form (4.26).

4.6 ($\star$) Using the definitions of the between-class and within-class covariance matrices given by (4.27) and (4.28), respectively, together with (4.34) and (4.36) and the choice of target values described in Section 4.1.5, show that the expression (4.33) that minimizes the sum-of-squares error function can be written in the form (4.37).

4.7 ($\star$) **WWW** Show that the logistic sigmoid function (4.59) satisfies the property $\sigma(-a) = 1 - \sigma(a)$ and that its inverse is given by $\sigma^{-1}(y) = \ln\{y/(1-y)\}$.

4.8 ($\star$) Using (4.57) and (4.58), derive the result (4.65) for the posterior class probability in the two-class generative model with Gaussian densities, and verify the results (4.66) and (4.67) for the parameters $\mathbf{w}$ and w_0.

4.9 ($\star$) **WWW** Consider a generative classification model for K classes defined by prior class probabilities $p(\mathcal{C}_k) = \pi_k$ and general class-conditional densities $p(\boldsymbol{\phi}|\mathcal{C}_k)$ where $\boldsymbol{\phi}$ is the input feature vector. Suppose we are given a training data set $\{\boldsymbol{\phi}_n, \mathbf{t}_n\}$ where $n = 1, \ldots, N$, and $\mathbf{t}_n$ is a binary target vector of length K that uses the 1-of-K coding scheme, so that it has components $t_{nj} = I_{jk}$ if pattern n is from class $\mathcal{C}_k$. Assuming that the data points are drawn independently from this model, show that the maximum-likelihood solution for the prior probabilities is given by

$$\pi_k = \frac{N_k}{N} \tag{4.159}$$

where N_k is the number of data points assigned to class $\mathcal{C}_k$.

4.10 ($\star\star$) Consider the classification model of Exercise 4.9 and now suppose that the class-conditional densities are given by Gaussian distributions with a shared covariance matrix, so that

$$p(\boldsymbol{\phi}|\mathcal{C}_k) = \mathcal{N}(\boldsymbol{\phi}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}). \tag{4.160}$$

Show that the maximum likelihood solution for the mean of the Gaussian distribution for class $\mathcal{C}_k$ is given by

$$\boldsymbol{\mu}_k = \frac{1}{N_k} \sum_{n=1}^{N} t_{nk} \boldsymbol{\phi}_n \tag{4.161}$$

which represents the mean of those feature vectors assigned to class $\mathcal{C}_k$. Similarly, show that the maximum likelihood solution for the shared covariance matrix is given by

$$\boldsymbol{\Sigma} = \sum_{k=1}^{K} \frac{N_k}{N} \mathbf{S}_k \tag{4.162}$$

where

$$\mathbf{S}_k = \frac{1}{N_k} \sum_{n=1}^{N} t_{nk} (\boldsymbol{\phi}_n - \boldsymbol{\mu}_k)(\boldsymbol{\phi}_n - \boldsymbol{\mu}_k)^{\mathrm{T}}. \tag{4.163}$$

Thus $\boldsymbol{\Sigma}$ is given by a weighted average of the covariances of the data associated with each class, in which the weighting coefficients are given by the prior probabilities of the classes.

4.11 ($\star\star$) Consider a classification problem with K classes for which the feature vector $\boldsymbol{\phi}$ has M components each of which can take L discrete states. Let the values of the components be represented by a 1-of-L binary coding scheme. Further suppose that, conditioned on the class $\mathcal{C}_k$, the M components of $\boldsymbol{\phi}$ are independent, so that the class-conditional density factorizes with respect to the feature vector components. Show that the quantities a_k given by (4.63), which appear in the argument to the softmax function describing the posterior class probabilities, are linear functions of the components of $\boldsymbol{\phi}$. Note that this represents an example of the naive Bayes model which is discussed in Section 8.2.2.

4.12 ($\star$) **www** Verify the relation (4.88) for the derivative of the logistic sigmoid function defined by (4.59).

4.13 ($\star$) **www** By making use of the result (4.88) for the derivative of the logistic sigmoid, show that the derivative of the error function (4.90) for the logistic regression model is given by (4.91).

4.14 ($\star$) Show that for a linearly separable data set, the maximum likelihood solution for the logistic regression model is obtained by finding a vector $\mathbf{w}$ whose decision boundary $\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}(\mathbf{x}) = 0$ separates the classes and then taking the magnitude of $\mathbf{w}$ to infinity.

4.15 ($\star\star$) Show that the Hessian matrix $\mathbf{H}$ for the logistic regression model, given by (4.97), is positive definite. Here $\mathbf{R}$ is a diagonal matrix with elements $y_n(1 - y_n)$, and y_n is the output of the logistic regression model for input vector $\mathbf{x}_n$. Hence show that the error function is a concave function of $\mathbf{w}$ and that it has a unique minimum.

4.16 ($\star$) Consider a binary classification problem in which each observation $\mathbf{x}_n$ is known to belong to one of two classes, corresponding to $t = 0$ and $t = 1$, and suppose that the procedure for collecting training data is imperfect, so that training points are sometimes mislabelled. For every data point $\mathbf{x}_n$, instead of having a value t for the class label, we have instead a value π_n representing the probability that $t_n = 1$. Given a probabilistic model $p(t = 1|\boldsymbol{\phi})$, write down the log likelihood function appropriate to such a data set.

4.17 (⋆) **www** Show that the derivatives of the softmax activation function (4.104), where the a_k are defined by (4.105), are given by (4.106).

4.18 (⋆) Using the result (4.91) for the derivatives of the softmax activation function, show that the gradients of the cross-entropy error (4.108) are given by (4.109).

4.19 (⋆) **www** Write down expressions for the gradient of the log likelihood, as well as the corresponding Hessian matrix, for the probit regression model defined in Section 4.3.5. These are the quantities that would be required to train such a model using IRLS.

4.20 (⋆⋆) Show that the Hessian matrix for the multiclass logistic regression problem, defined by (4.110), is positive semidefinite. Note that the full Hessian matrix for this problem is of size $MK \times MK$, where M is the number of parameters and K is the number of classes. To prove the positive semidefinite property, consider the product $\mathbf{u}^{\mathrm{T}}\mathbf{H}\mathbf{u}$ where $\mathbf{u}$ is an arbitrary vector of length MK, and then apply Jensen's inequality.

4.21 (⋆) Show that the probit function (4.114) and the erf function (4.115) are related by (4.116).

4.22 (⋆) Using the result (4.135), derive the expression (4.137) for the log model evidence under the Laplace approximation.

4.23 (⋆⋆) **www** In this exercise, we derive the BIC result (4.139) starting from the Laplace approximation to the model evidence given by (4.137). Show that if the prior over parameters is Gaussian of the form $p(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta}|\mathbf{m}, \mathbf{V}_0)$, the log model evidence under the Laplace approximation takes the form

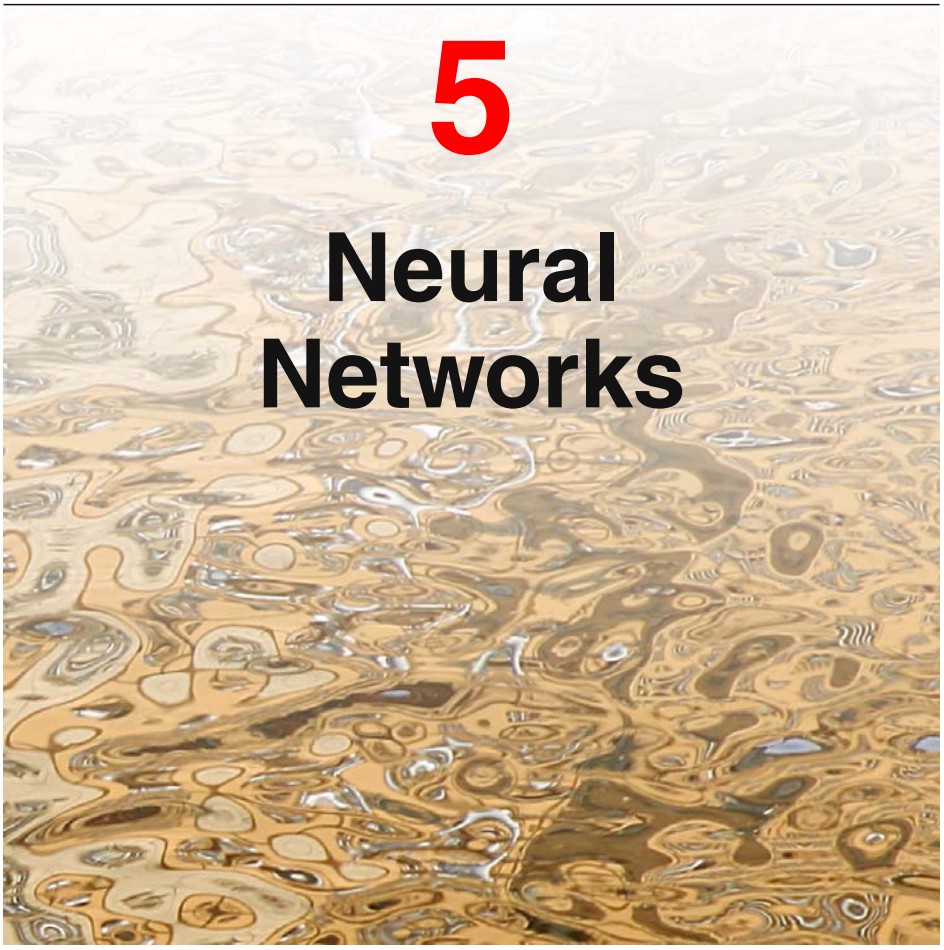
$$\ln p(\mathcal{D}) \simeq \ln p(\mathcal{D}|\boldsymbol{\theta}_{\mathrm{MAP}}) - \frac{1}{2}(\boldsymbol{\theta}_{\mathrm{MAP}} - \mathbf{m})^{\mathrm{T}}\mathbf{V}_0^{-1}(\boldsymbol{\theta}_{\mathrm{MAP}} - \mathbf{m}) - \frac{1}{2}\ln|\mathbf{H}| + \mathrm{const}$$

where $\mathbf{H}$ is the matrix of second derivatives of the log likelihood $\ln p(\mathcal{D}|\boldsymbol{\theta})$ evaluated at $\boldsymbol{\theta}_{\mathrm{MAP}}$. Now assume that the prior is broad so that $\mathbf{V}_0^{-1}$ is small and the second term on the right-hand side above can be neglected. Furthermore, consider the case of independent, identically distributed data so that $\mathbf{H}$ is the sum of terms one for each data point. Show that the log model evidence can then be written approximately in the form of the BIC expression (4.139).

4.24 (⋆⋆) Use the results from Section 2.3.2 to derive the result (4.151) for the marginalization of the logistic regression model with respect to a Gaussian posterior distribution over the parameters $\mathbf{w}$.

4.25 (⋆⋆) Suppose we wish to approximate the logistic sigmoid $\sigma(a)$ defined by (4.59) by a scaled probit function $\Phi(\lambda a)$, where $\Phi(a)$ is defined by (4.114). Show that if λ is chosen so that the derivatives of the two functions are equal at $a = 0$, then $\lambda^2 = \pi/8$.

4.26 ($\star\star$) In this exercise, we prove the relation (4.152) for the convolution of a probit function with a Gaussian distribution. To do this, show that the derivative of the left-hand side with respect to μ is equal to the derivative of the right-hand side, and then integrate both sides with respect to μ and then show that the constant of integration vanishes. Note that before differentiating the left-hand side, it is convenient first to introduce a change of variable given by $a = \mu + \sigma z$ so that the integral over a is replaced by an integral over z. When we differentiate the left-hand side of the relation (4.152), we will then obtain a Gaussian integral over z that can be evaluated analytically.



5 Neural Networks

In Chapters 3 and 4 we considered models for regression and classification that comprised linear combinations of fixed basis functions. We saw that such models have useful analytical and computational properties but that their practical applicability was limited by the curse of dimensionality. In order to apply such models to large-scale problems, it is necessary to adapt the basis functions to the data.

Support vector machines (SVMs), discussed in Chapter 7, address this by first defining basis functions that are centred on the training data points and then selecting a subset of these during training. One advantage of SVMs is that, although the training involves nonlinear optimization, the objective function is convex, and so the solution of the optimization problem is relatively straightforward. The number of basis functions in the resulting models is generally much smaller than the number of training points, although it is often still relatively large and typically increases with the size of the training set. The relevance vector machine, discussed in Section 7.2, also chooses a subset from a fixed set of basis functions and typically results in much

sparser models. Unlike the SVM it also produces probabilistic outputs, although this is at the expense of a nonconvex optimization during training.

An alternative approach is to fix the number of basis functions in advance but allow them to be adaptive, in other words to use parametric forms for the basis functions in which the parameter values are adapted during training. The most successful model of this type in the context of pattern recognition is the feed-forward neural network, also known as the *multilayer perceptron*, discussed in this chapter. In fact, 'multilayer perceptron' is really a misnomer, because the model comprises multiple layers of logistic regression models (with continuous nonlinearities) rather than multiple perceptrons (with discontinuous nonlinearities). For many applications, the resulting model can be significantly more compact, and hence faster to evaluate, than a support vector machine having the same generalization performance. The price to be paid for this compactness, as with the relevance vector machine, is that the likelihood function, which forms the basis for network training, is no longer a convex function of the model parameters. In practice, however, it is often worth investing substantial computational resources during the training phase in order to obtain a compact model that is fast at processing new data.

The term 'neural network' has its origins in attempts to find mathematical representations of information processing in biological systems (McCulloch and Pitts, 1943; Widrow and Hoff, 1960; Rosenblatt, 1962; Rumelhart *et al.*, 1986). Indeed, it has been used very broadly to cover a wide range of different models, many of which have been the subject of exaggerated claims regarding their biological plausibility. From the perspective of practical applications of pattern recognition, however, biological realism would impose entirely unnecessary constraints. Our focus in this chapter is therefore on neural networks as efficient models for statistical pattern recognition. In particular, we shall restrict our attention to the specific class of neural networks that have proven to be of greatest practical value, namely the multilayer perceptron.

We begin by considering the functional form of the network model, including the specific parameterization of the basis functions, and we then discuss the problem of determining the network parameters within a maximum likelihood framework, which involves the solution of a nonlinear optimization problem. This requires the evaluation of derivatives of the log likelihood function with respect to the network parameters, and we shall see how these can be obtained efficiently using the technique of *error backpropagation*. We shall also show how the backpropagation framework can be extended to allow other derivatives to be evaluated, such as the Jacobian and Hessian matrices. Next we discuss various approaches to regularization of neural network training and the relationships between them. We also consider some extensions to the neural network model, and in particular we describe a general framework for modelling conditional probability distributions known as *mixture density networks*. Finally, we discuss the use of Bayesian treatments of neural networks. Additional background on neural network models can be found in Bishop (1995a).

5.1. Feed-forward Network Functions

The linear models for regression and classification discussed in Chapters 3 and 4, respectively, are based on linear combinations of fixed nonlinear basis functions $\phi_j(\mathbf{x})$ and take the form

$$y(\mathbf{x}, \mathbf{w}) = f\left(\sum_{j=1}^{M} w_j \phi_j(\mathbf{x})\right) \tag{5.1}$$

where $f(\cdot)$ is a nonlinear activation function in the case of classification and is the identity in the case of regression. Our goal is to extend this model by making the basis functions $\phi_j(\mathbf{x})$ depend on parameters and then to allow these parameters to be adjusted, along with the coefficients $\{w_j\}$, during training. There are, of course, many ways to construct parametric nonlinear basis functions. Neural networks use basis functions that follow the same form as (5.1), so that each basis function is itself a nonlinear function of a linear combination of the inputs, where the coefficients in the linear combination are adaptive parameters.

This leads to the basic neural network model, which can be described a series of functional transformations. First we construct M linear combinations of the input variables $x_1, \ldots, x_D$ in the form

$$a_j = \sum_{i=1}^{D} w_{ji}^{(1)} x_i + w_{j0}^{(1)} \tag{5.2}$$

where $j = 1, \ldots, M$, and the superscript (1) indicates that the corresponding parameters are in the first 'layer' of the network. We shall refer to the parameters $w_{ji}^{(1)}$ as *weights* and the parameters $w_{j0}^{(1)}$ as *biases*, following the nomenclature of Chapter 3. The quantities a_j are known as *activations*. Each of them is then transformed using a differentiable, nonlinear *activation function* $h(\cdot)$ to give

$$z_j = h(a_j). \tag{5.3}$$

These quantities correspond to the outputs of the basis functions in (5.1) that, in the context of neural networks, are called *hidden units*. The nonlinear functions $h(\cdot)$ are generally chosen to be sigmoidal functions such as the logistic sigmoid or the 'tanh' function. Following (5.1), these values are again linearly combined to give *output unit activations*

Exercise 5.1

$$a_k = \sum_{j=1}^{M} w_{kj}^{(2)} z_j + w_{k0}^{(2)} \tag{5.4}$$

where $k = 1, \ldots, K$, and K is the total number of outputs. This transformation corresponds to the second layer of the network, and again the $w_{k0}^{(2)}$ are bias parameters. Finally, the output unit activations are transformed using an appropriate activation function to give a set of network outputs y_k. The choice of activation function is determined by the nature of the data and the assumed distribution of target variables

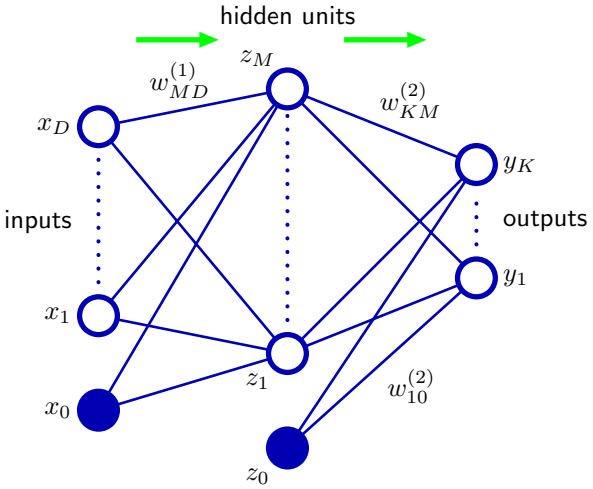


Figure 5.1 Network diagram for the two-layer neural network corresponding to (5.7). The input, hidden, and output variables are represented by nodes, and the weight parameters are represented by links between the nodes, in which the bias parameters are denoted by links coming from additional input and hidden variables x_0 and z_0. Arrows denote the direction of information flow through the network during forward propagation.

and follows the same considerations as for linear models discussed in Chapters 3 and 4. Thus for standard regression problems, the activation function is the identity so that $y_k = a_k$. Similarly, for multiple binary classification problems, each output unit activation is transformed using a logistic sigmoid function so that

$$y_k = \sigma(a_k) \tag{5.5}$$

where

$$\sigma(a) = \frac{1}{1 + \exp(-a)}. \tag{5.6}$$

Finally, for multiclass problems, a softmax activation function of the form (4.62) is used. The choice of output unit activation function is discussed in detail in Section 5.2.

We can combine these various stages to give the overall network function that, for sigmoidal output unit activation functions, takes the form

$$y_k(\mathbf{x}, \mathbf{w}) = \sigma\left(\sum_{j=1}^{M} w_{kj}^{(2)} h\left(\sum_{i=1}^{D} w_{ji}^{(1)} x_i + w_{j0}^{(1)}\right) + w_{k0}^{(2)}\right) \tag{5.7}$$

where the set of all weight and bias parameters have been grouped together into a vector $\mathbf{w}$. Thus the neural network model is simply a nonlinear function from a set of input variables $\{x_i\}$ to a set of output variables $\{y_k\}$ controlled by a vector $\mathbf{w}$ of adjustable parameters.

This function can be represented in the form of a network diagram as shown in Figure 5.1. The process of evaluating (5.7) can then be interpreted as a *forward propagation* of information through the network. It should be emphasized that these diagrams do not represent probabilistic graphical models of the kind to be considered in Chapter 8 because the internal nodes represent deterministic variables rather than stochastic ones. For this reason, we have adopted a slightly different graphical

notation for the two kinds of model. We shall see later how to give a probabilistic interpretation to a neural network.

As discussed in Section 3.1, the bias parameters in (5.2) can be absorbed into the set of weight parameters by defining an additional input variable x_0 whose value is clamped at $x_0 = 1$, so that (5.2) takes the form

$$a_j = \sum_{i=0}^{D} w_{ji}^{(1)} x_i. \tag{5.8}$$

We can similarly absorb the second-layer biases into the second-layer weights, so that the overall network function becomes

$$y_k(\mathbf{x}, \mathbf{w}) = \sigma\left(\sum_{j=0}^{M} w_{kj}^{(2)} h\left(\sum_{i=0}^{D} w_{ji}^{(1)} x_i\right)\right). \tag{5.9}$$

As can be seen from Figure 5.1, the neural network model comprises two stages of processing, each of which resembles the perceptron model of Section 4.1.7, and for this reason the neural network is also known as the *multilayer perceptron*, or MLP. A key difference compared to the perceptron, however, is that the neural network uses continuous sigmoidal nonlinearities in the hidden units, whereas the perceptron uses step-function nonlinearities. This means that the neural network function is differentiable with respect to the network parameters, and this property will play a central role in network training.

If the activation functions of all the hidden units in a network are taken to be linear, then for any such network we can always find an equivalent network without hidden units. This follows from the fact that the composition of successive linear transformations is itself a linear transformation. However, if the number of hidden units is smaller than either the number of input or output units, then the transformations that the network can generate are not the most general possible linear transformations from inputs to outputs because information is lost in the dimensionality reduction at the hidden units. In Section 12.4.2, we show that networks of linear units give rise to principal component analysis. In general, however, there is little interest in multilayer networks of linear units.

The network architecture shown in Figure 5.1 is the most commonly used one in practice. However, it is easily generalized, for instance by considering additional layers of processing each consisting of a weighted linear combination of the form (5.4) followed by an element-wise transformation using a nonlinear activation function. Note that there is some confusion in the literature regarding the terminology for counting the number of layers in such networks. Thus the network in Figure 5.1 may be described as a 3-layer network (which counts the number of layers of units, and treats the inputs as units) or sometimes as a single-hidden-layer network (which counts the number of layers of hidden units). We recommend a terminology in which Figure 5.1 is called a two-layer network, because it is the number of layers of adaptive weights that is important for determining the network properties.

Another generalization of the network architecture is to include *skip-layer* connections, each of which is associated with a corresponding adaptive parameter. For

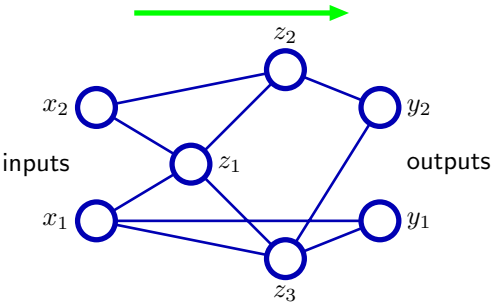


Figure 5.2 Example of a neural network having a general feed-forward topology. Note that each hidden and output unit has an associated bias parameter (omitted for clarity).

instance, in a two-layer network these would go directly from inputs to outputs. In principle, a network with sigmoidal hidden units can always mimic skip layer connections (for bounded input values) by using a sufficiently small first-layer weight that, over its operating range, the hidden unit is effectively linear, and then compensating with a large weight value from the hidden unit to the output. In practice, however, it may be advantageous to include skip-layer connections explicitly.

Furthermore, the network can be sparse, with not all possible connections within a layer being present. We shall see an example of a sparse network architecture when we consider convolutional neural networks in Section 5.5.6.

Because there is a direct correspondence between a network diagram and its mathematical function, we can develop more general network mappings by considering more complex network diagrams. However, these must be restricted to a *feed-forward* architecture, in other words to one having no closed directed cycles, to ensure that the outputs are deterministic functions of the inputs. This is illustrated with a simple example in Figure 5.2. Each (hidden or output) unit in such a network computes a function given by

$$z_k = h\left(\sum_j w_{kj} z_j\right) \tag{5.10}$$

where the sum runs over all units that send connections to unit k (and a bias parameter is included in the summation). For a given set of values applied to the inputs of the network, successive application of (5.10) allows the activations of all units in the network to be evaluated including those of the output units.

The approximation properties of feed-forward networks have been widely studied (Funahashi, 1989; Cybenko, 1989; Hornik *et al.*, 1989; Stinchecombe and White, 1989; Cotter, 1990; Ito, 1991; Hornik, 1991; Kreinovich, 1991; Ripley, 1996) and found to be very general. Neural networks are therefore said to be *universal approximators*. For example, a two-layer network with linear outputs can uniformly approximate any continuous function on a compact input domain to arbitrary accuracy provided the network has a sufficiently large number of hidden units. This result holds for a wide range of hidden unit activation functions, but excluding polynomials. Although such theorems are reassuring, the key problem is how to find suitable parameter values given a set of training data, and in later sections of this chapter we

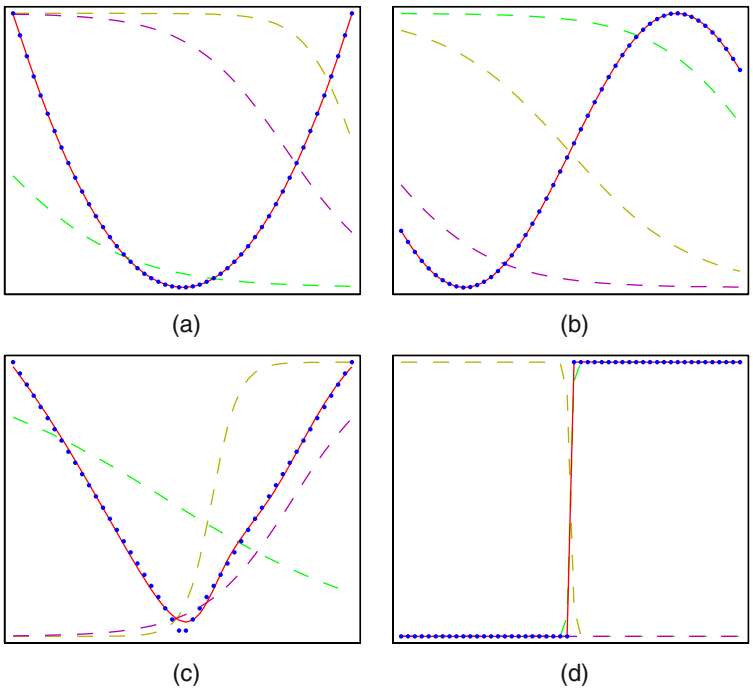


Figure 5.3 Illustration of the capability of a multilayer perceptron to approximate four different functions comprising (a) $f(x) = x^2$, (b) $f(x) = \sin(x)$, (c), $f(x) = |x|$, and (d) $f(x) = H(x)$ where $H(x)$ is the Heaviside step function. In each case, $N = 50$ data points, shown as blue dots, have been sampled uniformly in x over the interval $(-1, 1)$ and the corresponding values of $f(x)$ evaluated. These data points are then used to train a two-layer network having 3 hidden units with 'tanh' activation functions and linear output units. The resulting network functions are shown by the red curves, and the outputs of the three hidden units are shown by the three dashed curves.

will show that there exist effective solutions to this problem based on both maximum likelihood and Bayesian approaches.

The capability of a two-layer network to model a broad range of functions is illustrated in Figure 5.3. This figure also shows how individual hidden units work collaboratively to approximate the final function. The role of hidden units in a simple classification problem is illustrated in Figure 5.4 using the synthetic classification data set described in Appendix A.

5.1.1 Weight-space symmetries

One property of feed-forward networks, which will play a role when we consider Bayesian model comparison, is that multiple distinct choices for the weight vector $\mathbf{w}$ can all give rise to the same mapping function from inputs to outputs (Chen *et al.*, 1993). Consider a two-layer network of the form shown in Figure 5.1 with M hidden units having 'tanh' activation functions and full connectivity in both layers. If we change the sign of all of the weights and the bias feeding into a particular hidden unit, then, for a given input pattern, the sign of the activation of the hidden unit will be reversed, because 'tanh' is an odd function, so that $\tanh(-a) = -\tanh(a)$. This transformation can be exactly compensated by changing the sign of all of the weights leading out of that hidden unit. Thus, by changing the signs of a particular group of weights (and a bias), the input–output mapping function represented by the network is unchanged, and so we have found two different weight vectors that give rise to the same mapping function. For M hidden units, there will be M such 'sign-flip'

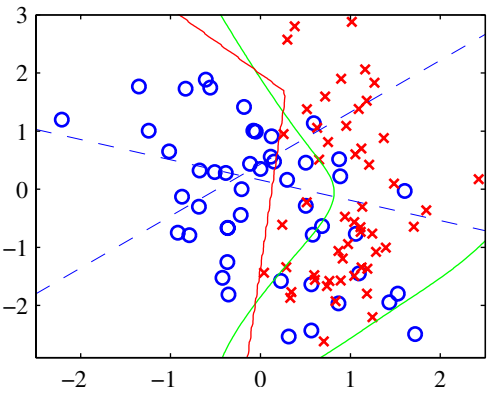


Figure 5.4 Example of the solution of a simple two-class classification problem involving synthetic data using a neural network having two inputs, two hidden units with 'tanh' activation functions, and a single output having a logistic sigmoid activation function. The dashed blue lines show the $z = 0.5$ contours for each of the hidden units, and the red line shows the $y = 0.5$ decision surface for the network. For comparison, the green line denotes the optimal decision boundary computed from the distributions used to generate the data.

symmetries, and thus any given weight vector will be one of a set 2^M equivalent weight vectors .

Similarly, imagine that we interchange the values of all of the weights (and the bias) leading both into and out of a particular hidden unit with the corresponding values of the weights (and bias) associated with a different hidden unit. Again, this clearly leaves the network input–output mapping function unchanged, but it corresponds to a different choice of weight vector. For M hidden units, any given weight vector will belong to a set of $M!$ equivalent weight vectors associated with this interchange symmetry, corresponding to the $M!$ different orderings of the hidden units. The network will therefore have an overall weight-space symmetry factor of $M!2^M$. For networks with more than two layers of weights, the total level of symmetry will be given by the product of such factors, one for each layer of hidden units.

It turns out that these factors account for all of the symmetries in weight space (except for possible accidental symmetries due to specific choices for the weight values). Furthermore, the existence of these symmetries is not a particular property of the 'tanh' function but applies to a wide range of activation functions (Kůrková and Kainen, 1994). In many cases, these symmetries in weight space are of little practical consequence, although in Section 5.7 we shall encounter a situation in which we need to take them into account.

5.2. Network Training

So far, we have viewed neural networks as a general class of parametric nonlinear functions from a vector $\mathbf{x}$ of input variables to a vector $\mathbf{y}$ of output variables. A simple approach to the problem of determining the network parameters is to make an analogy with the discussion of polynomial curve fitting in Section 1.1, and therefore to minimize a sum-of-squares error function. Given a training set comprising a set of input vectors $\{\mathbf{x}_n\}$, where $n = 1, \ldots, N$, together with a corresponding set of

target vectors $\{\mathbf{t}_n\}$, we minimize the error function

$$E(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N} \|\mathbf{y}(\mathbf{x}_n, \mathbf{w}) - \mathbf{t}_n\|^2. \tag{5.11}$$

However, we can provide a much more general view of network training by first giving a probabilistic interpretation to the network outputs. We have already seen many advantages of using probabilistic predictions in Section 1.5.4. Here it will also provide us with a clearer motivation both for the choice of output unit nonlinearity and the choice of error function.

We start by discussing regression problems, and for the moment we consider a single target variable t that can take any real value. Following the discussions in Section 1.2.5 and 3.1, we assume that t has a Gaussian distribution with an $\mathbf{x}$-dependent mean, which is given by the output of the neural network, so that

$$p(t|\mathbf{x}, \mathbf{w}) = \mathcal{N}\left(t|y(\mathbf{x}, \mathbf{w}), \beta^{-1}\right) \tag{5.12}$$

where β is the precision (inverse variance) of the Gaussian noise. Of course this is a somewhat restrictive assumption, and in Section 5.6 we shall see how to extend this approach to allow for more general conditional distributions. For the conditional distribution given by (5.12), it is sufficient to take the output unit activation function to be the identity, because such a network can approximate any continuous function from $\mathbf{x}$ to y. Given a data set of N independent, identically distributed observations $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$, along with corresponding target values $\mathbf{t} = \{t_1, \ldots, t_N\}$, we can construct the corresponding likelihood function

$$p(\mathbf{t}|\mathbf{X}, \mathbf{w}, \beta) = \prod_{n=1}^{N} p(t_n|\mathbf{x}_n, \mathbf{w}, \beta).$$

Taking the negative logarithm, we obtain the error function

$$\frac{\beta}{2}\sum_{n=1}^{N}\{y(\mathbf{x}_n, \mathbf{w}) - t_n\}^2 - \frac{N}{2}\ln\beta + \frac{N}{2}\ln(2\pi) \tag{5.13}$$

which can be used to learn the parameters $\mathbf{w}$ and β. In Section 5.7, we shall discuss the Bayesian treatment of neural networks, while here we consider a maximum likelihood approach. Note that in the neural networks literature, it is usual to consider the minimization of an error function rather than the maximization of the (log) likelihood, and so here we shall follow this convention. Consider first the determination of $\mathbf{w}$. Maximizing the likelihood function is equivalent to minimizing the sum-of-squares error function given by

$$E(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{y(\mathbf{x}_n, \mathbf{w}) - t_n\}^2 \tag{5.14}$$

where we have discarded additive and multiplicative constants. The value of $\mathbf{w}$ found by minimizing $E(\mathbf{w})$ will be denoted $\mathbf{w}_{\mathrm{ML}}$ because it corresponds to the maximum likelihood solution. In practice, the nonlinearity of the network function $y(\mathbf{x}_n, \mathbf{w})$ causes the error $E(\mathbf{w})$ to be nonconvex, and so in practice local maxima of the likelihood may be found, corresponding to local minima of the error function, as discussed in Section 5.2.1.

Having found $\mathbf{w}_{\mathrm{ML}}$, the value of β can be found by minimizing the negative log likelihood to give

$$\frac{1}{\beta_{\mathrm{ML}}} = \frac{1}{N}\sum_{n=1}^{N}\{y(\mathbf{x}_n, \mathbf{w}_{\mathrm{ML}}) - t_n\}^2. \tag{5.15}$$

Note that this can be evaluated once the iterative optimization required to find $\mathbf{w}_{\mathrm{ML}}$ is completed. If we have multiple target variables, and we assume that they are independent conditional on $\mathbf{x}$ and $\mathbf{w}$ with shared noise precision β, then the conditional distribution of the target values is given by

$$p(\mathbf{t}|\mathbf{x}, \mathbf{w}) = \mathcal{N}\left(\mathbf{t}|\mathbf{y}(\mathbf{x}, \mathbf{w}), \beta^{-1}\mathbf{I}\right). \tag{5.16}$$

Following the same argument as for a single target variable, we see that the maximum likelihood weights are determined by minimizing the sum-of-squares error function (5.11). The noise precision is then given by

Exercise 5.2

$$\frac{1}{\beta_{\mathrm{ML}}} = \frac{1}{NK}\sum_{n=1}^{N}\|\mathbf{y}(\mathbf{x}_n, \mathbf{w}_{\mathrm{ML}}) - \mathbf{t}_n\|^2 \tag{5.17}$$

where K is the number of target variables. The assumption of independence can be dropped at the expense of a slightly more complex optimization problem.

Exercise 5.3

Recall from Section 4.3.6 that there is a natural pairing of the error function (given by the negative log likelihood) and the output unit activation function. In the regression case, we can view the network as having an output activation function that is the identity, so that $y_k = a_k$. The corresponding sum-of-squares error function has the property

$$\frac{\partial E}{\partial a_k} = y_k - t_k \tag{5.18}$$

which we shall make use of when discussing error backpropagation in Section 5.3.

Now consider the case of binary classification in which we have a single target variable t such that $t = 1$ denotes class $\mathcal{C}_1$ and $t = 0$ denotes class $\mathcal{C}_2$. Following the discussion of canonical link functions in Section 4.3.6, we consider a network having a single output whose activation function is a logistic sigmoid

$$y = \sigma(a) \equiv \frac{1}{1 + \exp(-a)} \tag{5.19}$$

so that $0 \leqslant y(\mathbf{x}, \mathbf{w}) \leqslant 1$. We can interpret $y(\mathbf{x}, \mathbf{w})$ as the conditional probability $p(\mathcal{C}_1|\mathbf{x})$, with $p(\mathcal{C}_2|\mathbf{x})$ given by $1 - y(\mathbf{x}, \mathbf{w})$. The conditional distribution of targets given inputs is then a Bernoulli distribution of the form

$$p(t|\mathbf{x}, \mathbf{w}) = y(\mathbf{x}, \mathbf{w})^t \left\{1 - y(\mathbf{x}, \mathbf{w})\right\}^{1-t}. \tag{5.20}$$

If we consider a training set of independent observations, then the error function, which is given by the negative log likelihood, is then a *cross-entropy* error function of the form

$$E(\mathbf{w}) = -\sum_{n=1}^{N} \{t_n \ln y_n + (1 - t_n) \ln(1 - y_n)\} \tag{5.21}$$

where y_n denotes $y(\mathbf{x}_n, \mathbf{w})$. Note that there is no analogue of the noise precision β because the target values are assumed to be correctly labelled. However, the model is easily extended to allow for labelling errors. Simard *et al.* (2003) found that using the cross-entropy error function instead of the sum-of-squares for a classification problem leads to faster training as well as improved generalization.

Exercise 5.4

If we have K separate binary classifications to perform, then we can use a network having K outputs each of which has a logistic sigmoid activation function. Associated with each output is a binary class label $t_k \in \{0, 1\}$, where $k = 1, \ldots, K$. If we assume that the class labels are independent, given the input vector, then the conditional distribution of the targets is

$$p(\mathbf{t}|\mathbf{x}, \mathbf{w}) = \prod_{k=1}^{K} y_k(\mathbf{x}, \mathbf{w})^{t_k} \left[1 - y_k(\mathbf{x}, \mathbf{w})\right]^{1-t_k}. \tag{5.22}$$

Taking the negative logarithm of the corresponding likelihood function then gives the following error function

Exercise 5.5

$$E(\mathbf{w}) = -\sum_{n=1}^{N} \sum_{k=1}^{K} \{t_{nk} \ln y_{nk} + (1 - t_{nk}) \ln(1 - y_{nk})\} \tag{5.23}$$

where y_{nk} denotes $y_k(\mathbf{x}_n, \mathbf{w})$. Again, the derivative of the error function with respect to the activation for a particular output unit takes the form (5.18) just as in the regression case.

Exercise 5.6

It is interesting to contrast the neural network solution to this problem with the corresponding approach based on a linear classification model of the kind discussed in Chapter 4. Suppose that we are using a standard two-layer network of the kind shown in Figure 5.1. We see that the weight parameters in the first layer of the network are shared between the various outputs, whereas in the linear model each classification problem is solved independently. The first layer of the network can be viewed as performing a nonlinear feature extraction, and the sharing of features between the different outputs can save on computation and can also lead to improved generalization.

Finally, we consider the standard multiclass classification problem in which each input is assigned to one of K mutually exclusive classes. The binary target variables $t_k \in \{0, 1\}$ have a 1-of-K coding scheme indicating the class, and the network outputs are interpreted as $y_k(\mathbf{x}, \mathbf{w}) = p(t_k = 1|\mathbf{x})$, leading to the following error function

$$E(\mathbf{w}) = -\sum_{n=1}^{N} \sum_{k=1}^{K} t_{kn} \ln y_k(\mathbf{x}_n, \mathbf{w}). \tag{5.24}$$

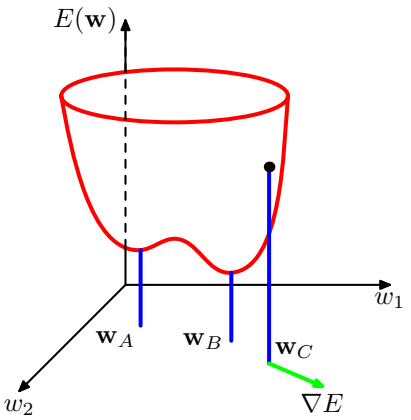


Figure 5.5 Geometrical view of the error function $E(\mathbf{w})$ as a surface sitting over weight space. Point $\mathbf{w}_A$ is a local minimum and $\mathbf{w}_B$ is the global minimum. At any point $\mathbf{w}_C$, the local gradient of the error surface is given by the vector ∇E.

Following the discussion of Section 4.3.4, we see that the output unit activation function, which corresponds to the canonical link, is given by the softmax function

$$y_k(\mathbf{x}, \mathbf{w}) = \frac{\exp(a_k(\mathbf{x}, \mathbf{w}))}{\sum_j \exp(a_j(\mathbf{x}, \mathbf{w}))} \tag{5.25}$$

which satisfies $0 \leqslant y_k \leqslant 1$ and $\sum_k y_k = 1$. Note that the $y_k(\mathbf{x}, \mathbf{w})$ are unchanged if a constant is added to all of the $a_k(\mathbf{x}, \mathbf{w})$, causing the error function to be constant for some directions in weight space. This degeneracy is removed if an appropriate regularization term (Section 5.5) is added to the error function.

Exercise 5.7

Once again, the derivative of the error function with respect to the activation for a particular output unit takes the familiar form (5.18).

In summary, there is a natural choice of both output unit activation function and matching error function, according to the type of problem being solved. For regression we use linear outputs and a sum-of-squares error, for (multiple independent) binary classifications we use logistic sigmoid outputs and a cross-entropy error function, and for multiclass classification we use softmax outputs with the corresponding multiclass cross-entropy error function. For classification problems involving two classes, we can use a single logistic sigmoid output, or alternatively we can use a network with two outputs having a softmax output activation function.

5.2.1 Parameter optimization

We turn next to the task of finding a weight vector $\mathbf{w}$ which minimizes the chosen function $E(\mathbf{w})$. At this point, it is useful to have a geometrical picture of the error function, which we can view as a surface sitting over weight space as shown in Figure 5.5. First note that if we make a small step in weight space from $\mathbf{w}$ to $\mathbf{w} + \delta\mathbf{w}$ then the change in the error function is $\delta E \simeq \delta\mathbf{w}^{\mathrm{T}} \nabla E(\mathbf{w})$, where the vector $\nabla E(\mathbf{w})$ points in the direction of greatest rate of increase of the error function. Because the error $E(\mathbf{w})$ is a smooth continuous function of $\mathbf{w}$, its smallest value will occur at a

point in weight space such that the gradient of the error function vanishes, so that

$$\nabla E(\mathbf{w}) = 0 \tag{5.26}$$

as otherwise we could make a small step in the direction of $-\nabla E(\mathbf{w})$ and thereby further reduce the error. Points at which the gradient vanishes are called stationary points, and may be further classified into minima, maxima, and saddle points.

Our goal is to find a vector $\mathbf{w}$ such that $E(\mathbf{w})$ takes its smallest value. However, the error function typically has a highly nonlinear dependence on the weights and bias parameters, and so there will be many points in weight space at which the gradient vanishes (or is numerically very small). Indeed, from the discussion in Section 5.1.1 we see that for any point $\mathbf{w}$ that is a local minimum, there will be other points in weight space that are equivalent minima. For instance, in a two-layer network of the kind shown in Figure 5.1, with M hidden units, each point in weight space is a member of a family of $M!2^M$ equivalent points.

Section 5.1.1

Furthermore, there will typically be multiple inequivalent stationary points and in particular multiple inequivalent minima. A minimum that corresponds to the smallest value of the error function for any weight vector is said to be a *global minimum*. Any other minima corresponding to higher values of the error function are said to be *local minima*. For a successful application of neural networks, it may not be necessary to find the global minimum (and in general it will not be known whether the global minimum has been found) but it may be necessary to compare several local minima in order to find a sufficiently good solution.

Because there is clearly no hope of finding an analytical solution to the equation $\nabla E(\mathbf{w}) = 0$ we resort to iterative numerical procedures. The optimization of continuous nonlinear functions is a widely studied problem and there exists an extensive literature on how to solve it efficiently. Most techniques involve choosing some initial value $\mathbf{w}^{(0)}$ for the weight vector and then moving through weight space in a succession of steps of the form

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} + \Delta\mathbf{w}^{(\tau)} \tag{5.27}$$

where τ labels the iteration step. Different algorithms involve different choices for the weight vector update $\Delta\mathbf{w}^{(\tau)}$. Many algorithms make use of gradient information and therefore require that, after each update, the value of $\nabla E(\mathbf{w})$ is evaluated at the new weight vector $\mathbf{w}^{(\tau+1)}$. In order to understand the importance of gradient information, it is useful to consider a local approximation to the error function based on a Taylor expansion.

5.2.2 Local quadratic approximation

Insight into the optimization problem, and into the various techniques for solving it, can be obtained by considering a local quadratic approximation to the error function.

Consider the Taylor expansion of $E(\mathbf{w})$ around some point $\widehat{\mathbf{w}}$ in weight space

$$E(\mathbf{w}) \simeq E(\widehat{\mathbf{w}}) + (\mathbf{w} - \widehat{\mathbf{w}})^{\mathrm{T}}\mathbf{b} + \frac{1}{2}(\mathbf{w} - \widehat{\mathbf{w}})^{\mathrm{T}}\mathbf{H}(\mathbf{w} - \widehat{\mathbf{w}}) \tag{5.28}$$

where cubic and higher terms have been omitted. Here $\mathbf{b}$ is defined to be the gradient of E evaluated at $\widehat{\mathbf{w}}$

$$\mathbf{b} \equiv \nabla E|_{\mathbf{w}=\widehat{\mathbf{w}}} \tag{5.29}$$

and the Hessian matrix $\mathbf{H} = \nabla\nabla E$ has elements

$$(\mathbf{H})_{ij} \equiv \left.\frac{\partial E}{\partial w_i \partial w_j}\right|_{\mathbf{w}=\widehat{\mathbf{w}}}. \tag{5.30}$$

From (5.28), the corresponding local approximation to the gradient is given by

$$\nabla E \simeq \mathbf{b} + \mathbf{H}(\mathbf{w} - \widehat{\mathbf{w}}). \tag{5.31}$$

For points $\mathbf{w}$ that are sufficiently close to $\widehat{\mathbf{w}}$, these expressions will give reasonable approximations for the error and its gradient.

Consider the particular case of a local quadratic approximation around a point $\mathbf{w}^\star$ that is a minimum of the error function. In this case there is no linear term, because $\nabla E = 0$ at $\mathbf{w}^\star$, and (5.28) becomes

$$E(\mathbf{w}) = E(\mathbf{w}^\star) + \frac{1}{2}(\mathbf{w} - \mathbf{w}^\star)^{\mathrm{T}}\mathbf{H}(\mathbf{w} - \mathbf{w}^\star) \tag{5.32}$$

where the Hessian $\mathbf{H}$ is evaluated at $\mathbf{w}^\star$. In order to interpret this geometrically, consider the eigenvalue equation for the Hessian matrix

$$\mathbf{H}\mathbf{u}_i = \lambda_i \mathbf{u}_i \tag{5.33}$$

where the eigenvectors $\mathbf{u}_i$ form a complete orthonormal set (Appendix C) so that

$$\mathbf{u}_i^{\mathrm{T}}\mathbf{u}_j = \delta_{ij}. \tag{5.34}$$

We now expand $(\mathbf{w} - \mathbf{w}^\star)$ as a linear combination of the eigenvectors in the form

$$\mathbf{w} - \mathbf{w}^\star = \sum_i \alpha_i \mathbf{u}_i. \tag{5.35}$$

This can be regarded as a transformation of the coordinate system in which the origin is translated to the point $\mathbf{w}^\star$, and the axes are rotated to align with the eigenvectors (through the orthogonal matrix whose columns are the $\mathbf{u}_i$), and is discussed in more detail in Appendix C. Substituting (5.35) into (5.32), and using (5.33) and (5.34), allows the error function to be written in the form

$$E(\mathbf{w}) = E(\mathbf{w}^\star) + \frac{1}{2}\sum_i \lambda_i \alpha_i^2. \tag{5.36}$$

A matrix $\mathbf{H}$ is said to be *positive definite* if, and only if,

$$\mathbf{v}^{\mathrm{T}}\mathbf{H}\mathbf{v} > 0 \qquad \text{for all } \mathbf{v}. \tag{5.37}$$

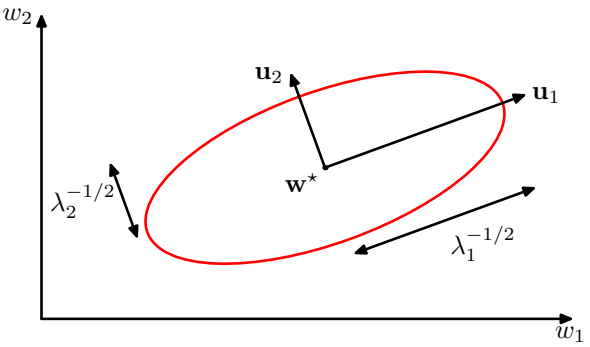


Figure 5.6 In the neighbourhood of a minimum $\mathbf{w}^\star$, the error function can be approximated by a quadratic. Contours of constant error are then ellipses whose axes are aligned with the eigenvectors $\mathbf{u}_i$ of the Hessian matrix, with lengths that are inversely proportional to the square roots of the corresponding eigenvectors λ_i.

Because the eigenvectors $\{\mathbf{u}_i\}$ form a complete set, an arbitrary vector $\mathbf{v}$ can be written in the form

$$\mathbf{v} = \sum_i c_i \mathbf{u}_i. \tag{5.38}$$

From (5.33) and (5.34), we then have

$$\mathbf{v}^{\mathrm{T}}\mathbf{H}\mathbf{v} = \sum_i c_i^2 \lambda_i \tag{5.39}$$

Exercise 5.10 and so $\mathbf{H}$ will be positive definite if, and only if, all of its eigenvalues are positive. In the new coordinate system, whose basis vectors are given by the eigenvectors *Exercise 5.11* $\{\mathbf{u}_i\}$, the contours of constant E are ellipses centred on the origin, as illustrated in Figure 5.6. For a one-dimensional weight space, a stationary point $w^\star$ will be a minimum if

$$\left.\frac{\partial^2 E}{\partial w^2}\right|_{w^\star} > 0. \tag{5.40}$$

The corresponding result in D-dimensions is that the Hessian matrix, evaluated at *Exercise 5.12* $\mathbf{w}^\star$, should be positive definite.

5.2.3 Use of gradient information

As we shall see in Section 5.3, it is possible to evaluate the gradient of an error function efficiently by means of the backpropagation procedure. The use of this gradient information can lead to significant improvements in the speed with which the minima of the error function can be located. We can see why this is so, as follows.

In the quadratic approximation to the error function, given in (5.28), the error surface is specified by the quantities $\mathbf{b}$ and $\mathbf{H}$, which contain a total of $W(W+1)/2$ independent elements (because the matrix $\mathbf{H}$ is symmetric), where W is the *Exercise 5.13* dimensionality of $\mathbf{w}$ (i.e., the total number of adaptive parameters in the network). The location of the minimum of this quadratic approximation therefore depends on $O(W^2)$ parameters, and we should not expect to be able to locate the minimum until we have gathered $O(W^2)$ independent pieces of information. If we do not make use of gradient information, we would expect to have to perform $O(W^2)$ function

evaluations, each of which would require $O(W)$ steps. Thus, the computational effort needed to find the minimum using such an approach would be $O(W^3)$.

Now compare this with an algorithm that makes use of the gradient information. Because each evaluation of ∇E brings W items of information, we might hope to find the minimum of the function in $O(W)$ gradient evaluations. As we shall see, by using error backpropagation, each such evaluation takes only $O(W)$ steps and so the minimum can now be found in $O(W^2)$ steps. For this reason, the use of gradient information forms the basis of practical algorithms for training neural networks.

5.2.4 Gradient descent optimization

The simplest approach to using gradient information is to choose the weight update in (5.27) to comprise a small step in the direction of the negative gradient, so that

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} - \eta \nabla E(\mathbf{w}^{(\tau)}) \tag{5.41}$$

where the parameter $\eta > 0$ is known as the *learning rate*. After each such update, the gradient is re-evaluated for the new weight vector and the process repeated. Note that the error function is defined with respect to a training set, and so each step requires that the entire training set be processed in order to evaluate ∇E. Techniques that use the whole data set at once are called *batch* methods. At each step the weight vector is moved in the direction of the greatest rate of decrease of the error function, and so this approach is known as *gradient descent* or *steepest descent*. Although such an approach might intuitively seem reasonable, in fact it turns out to be a poor algorithm, for reasons discussed in Bishop and Nabney (2008).

For batch optimization, there are more efficient methods, such as *conjugate gradients* and *quasi-Newton* methods, which are much more robust and much faster than simple gradient descent (Gill *et al.*, 1981; Fletcher, 1987; Nocedal and Wright, 1999). Unlike gradient descent, these algorithms have the property that the error function always decreases at each iteration unless the weight vector has arrived at a local or global minimum.

In order to find a sufficiently good minimum, it may be necessary to run a gradient-based algorithm multiple times, each time using a different randomly chosen starting point, and comparing the resulting performance on an independent validation set.

There is, however, an on-line version of gradient descent that has proved useful in practice for training neural networks on large data sets (Le Cun *et al.*, 1989). Error functions based on maximum likelihood for a set of independent observations comprise a sum of terms, one for each data point

$$E(\mathbf{w}) = \sum_{n=1}^{N} E_n(\mathbf{w}). \tag{5.42}$$

On-line gradient descent, also known as *sequential gradient descent* or *stochastic gradient descent*, makes an update to the weight vector based on one data point at a time, so that

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} - \eta \nabla E_n(\mathbf{w}^{(\tau)}). \tag{5.43}$$

This update is repeated by cycling through the data either in sequence or by selecting points at random with replacement. There are of course intermediate scenarios in which the updates are based on batches of data points.

One advantage of on-line methods compared to batch methods is that the former handle redundancy in the data much more efficiently. To see, this consider an extreme example in which we take a data set and double its size by duplicating every data point. Note that this simply multiplies the error function by a factor of 2 and so is equivalent to using the original error function. Batch methods will require double the computational effort to evaluate the batch error function gradient, whereas on-line methods will be unaffected. Another property of on-line gradient descent is the possibility of escaping from local minima, since a stationary point with respect to the error function for the whole data set will generally not be a stationary point for each data point individually.

Nonlinear optimization algorithms, and their practical application to neural network training, are discussed in detail in Bishop and Nabney (2008).

5.3. Error Backpropagation

Our goal in this section is to find an efficient technique for evaluating the gradient of an error function $E(\mathbf{w})$ for a feed-forward neural network. We shall see that this can be achieved using a local message passing scheme in which information is sent alternately forwards and backwards through the network and is known as *error backpropagation*, or sometimes simply as *backprop*.

It should be noted that the term backpropagation is used in the neural computing literature to mean a variety of different things. For instance, the multilayer perceptron architecture is sometimes called a backpropagation network. The term backpropagation is also used to describe the training of a multilayer perceptron using gradient descent applied to a sum-of-squares error function. In order to clarify the terminology, it is useful to consider the nature of the training process more carefully. Most training algorithms involve an iterative procedure for minimization of an error function, with adjustments to the weights being made in a sequence of steps. At each such step, we can distinguish between two distinct stages. In the first stage, the derivatives of the error function with respect to the weights must be evaluated. As we shall see, the important contribution of the backpropagation technique is in providing a computationally efficient method for evaluating such derivatives. Because it is at this stage that errors are propagated backwards through the network, we shall use the term backpropagation specifically to describe the evaluation of derivatives. In the second stage, the derivatives are then used to compute the adjustments to be made to the weights. The simplest such technique, and the one originally considered by Rumelhart *et al.* (1986), involves gradient descent. It is important to recognize that the two stages are distinct. Thus, the first stage, namely the propagation of errors backwards through the network in order to evaluate derivatives, can be applied to many other kinds of network and not just the multilayer perceptron. It can also be applied to error functions other that just the simple sum-of-squares, and to the eval-

uation of other derivatives such as the Jacobian and Hessian matrices, as we shall see later in this chapter. Similarly, the second stage of weight adjustment using the calculated derivatives can be tackled using a variety of optimization schemes, many of which are substantially more powerful than simple gradient descent.

5.3.1 Evaluation of error-function derivatives

We now derive the backpropagation algorithm for a general network having arbitrary feed-forward topology, arbitrary differentiable nonlinear activation functions, and a broad class of error function. The resulting formulae will then be illustrated using a simple layered network structure having a single layer of sigmoidal hidden units together with a sum-of-squares error.

Many error functions of practical interest, for instance those defined by maximum likelihood for a set of i.i.d. data, comprise a sum of terms, one for each data point in the training set, so that

$$E(\mathbf{w}) = \sum_{n=1}^{N} E_n(\mathbf{w}). \tag{5.44}$$

Here we shall consider the problem of evaluating $\nabla E_n(\mathbf{w})$ for one such term in the error function. This may be used directly for sequential optimization, or the results can be accumulated over the training set in the case of batch methods.

Consider first a simple linear model in which the outputs y_k are linear combinations of the input variables x_i so that

$$y_k = \sum_i w_{ki} x_i \tag{5.45}$$

together with an error function that, for a particular input pattern n, takes the form

$$E_n = \frac{1}{2} \sum_k (y_{nk} - t_{nk})^2 \tag{5.46}$$

where $y_{nk} = y_k(\mathbf{x}_n, \mathbf{w})$. The gradient of this error function with respect to a weight w_{ji} is given by

$$\frac{\partial E_n}{\partial w_{ji}} = (y_{nj} - t_{nj}) x_{ni} \tag{5.47}$$

which can be interpreted as a 'local' computation involving the product of an 'error signal' $y_{nj} - t_{nj}$ associated with the output end of the link w_{ji} and the variable x_{ni} associated with the input end of the link. In Section 4.3.2, we saw how a similar formula arises with the logistic sigmoid activation function together with the cross entropy error function, and similarly for the softmax activation function together with its matching cross-entropy error function. We shall now see how this simple result extends to the more complex setting of multilayer feed-forward networks.

In a general feed-forward network, each unit computes a weighted sum of its inputs of the form

$$a_j = \sum_i w_{ji} z_i \tag{5.48}$$

where z_i is the activation of a unit, or input, that sends a connection to unit j, and w_{ji} is the weight associated with that connection. In Section 5.1, we saw that biases can be included in this sum by introducing an extra unit, or input, with activation fixed at $+1$. We therefore do not need to deal with biases explicitly. The sum in (5.48) is transformed by a nonlinear activation function $h(\cdot)$ to give the activation z_j of unit j in the form

$$z_j = h(a_j). \tag{5.49}$$

Note that one or more of the variables z_i in the sum in (5.48) could be an input, and similarly, the unit j in (5.49) could be an output.

For each pattern in the training set, we shall suppose that we have supplied the corresponding input vector to the network and calculated the activations of all of the hidden and output units in the network by successive application of (5.48) and (5.49). This process is often called *forward propagation* because it can be regarded as a forward flow of information through the network.

Now consider the evaluation of the derivative of E_n with respect to a weight w_{ji}. The outputs of the various units will depend on the particular input pattern n. However, in order to keep the notation uncluttered, we shall omit the subscript n from the network variables. First we note that E_n depends on the weight w_{ji} only via the summed input a_j to unit j. We can therefore apply the chain rule for partial derivatives to give

$$\frac{\partial E_n}{\partial w_{ji}} = \frac{\partial E_n}{\partial a_j}\frac{\partial a_j}{\partial w_{ji}}. \tag{5.50}$$

We now introduce a useful notation

$$\delta_j \equiv \frac{\partial E_n}{\partial a_j} \tag{5.51}$$

where the δ's are often referred to as *errors* for reasons we shall see shortly. Using (5.48), we can write

$$\frac{\partial a_j}{\partial w_{ji}} = z_i. \tag{5.52}$$

Substituting (5.51) and (5.52) into (5.50), we then obtain

$$\frac{\partial E_n}{\partial w_{ji}} = \delta_j z_i. \tag{5.53}$$

Equation (5.53) tells us that the required derivative is obtained simply by multiplying the value of δ for the unit at the output end of the weight by the value of z for the unit at the input end of the weight (where $z = 1$ in the case of a bias). Note that this takes the same form as for the simple linear model considered at the start of this section. Thus, in order to evaluate the derivatives, we need only to calculate the value of δ_j for each hidden and output unit in the network, and then apply (5.53).

As we have seen already, for the output units, we have

$$\delta_k = y_k - t_k \tag{5.54}$$

McCullagh, P. and J. A. Nelder (1989). *Generalized Linear Models* (Second ed.). Chapman and Hall.

McCulloch, W. S. and W. Pitts (1943). A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics* **5**, 115–133. Reprinted in Anderson and Rosenfeld (1988).

McEliece, R. J., D. J. C. MacKay, and J. F. Cheng (1998). Turbo decoding as an instance of Pearl's 'Belief Ppropagation' algorithm. *IEEE Journal on Selected Areas in Communications* **16**, 140–152.

McLachlan, G. J. and K. E. Basford (1988). *Mixture Models: Inference and Applications to Clustering*. Marcel Dekker.

McLachlan, G. J. and T. Krishnan (1997). *The EM Algorithm and its Extensions*. Wiley.

McLachlan, G. J. and D. Peel (2000). *Finite Mixture Models*. Wiley.

Meng, X. L. and D. B. Rubin (1993). Maximum likelihood estimation via the ECM algorithm: a general framework. *Biometrika* **80**, 267–278.

Metropolis, N., A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller (1953). Equation of state calculations by fast computing machines. *Journal of Chemical Physics* **21**(6), 1087–1092.

Metropolis, N. and S. Ulam (1949). The Monte Carlo method. *Journal of the American Statistical Association* **44**(247), 335–341.

Mika, S., G. Rätsch, J. Weston, and B. Schölkopf (1999). Fisher discriminant analysis with kernels. In Y. H. Hu, J. Larsen, E. Wilson, and S. Douglas (Eds.), *Neural Networks for Signal Processing IX*, pp. 41–48. IEEE.

Minka, T. (2001a). Expectation propagation for approximate Bayesian inference. In J. Breese and D. Koller (Eds.), *Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence*, pp. 362–369. Morgan Kaufmann.

Minka, T. (2001b). *A family of approximate algorithms for Bayesian inference*. Ph. D. thesis, MIT.

Minka, T. (2004). Power EP. Technical Report MSR-TR-2004-149, Microsoft Research Cambridge.

Minka, T. (2005). Divergence measures and message passing. Technical Report MSR-TR-2005-173, Microsoft Research Cambridge.

Minka, T. P. (2001c). Automatic choice of dimensionality for PCA. In T. K. Leen, T. G. Dietterich, and V. Tresp (Eds.), *Advances in Neural Information Processing Systems*, Volume 13, pp. 598–604. MIT Press.

Minsky, M. L. and S. A. Papert (1969). *Perceptrons*. MIT Press. Expanded edition 1990.

Miskin, J. W. and D. J. C. MacKay (2001). Ensemble learning for blind source separation. In S. J. Roberts and R. M. Everson (Eds.), *Independent Component Analysis: Principles and Practice*. Cambridge University Press.

Møller, M. (1993). Efficient Training of Feed-Forward Neural Networks. Ph. D. thesis, Aarhus University, Denmark.

Moody, J. and C. J. Darken (1989). Fast learning in networks of locally-tuned processing units. *Neural Computation* **1**(2), 281–294.

Moore, A. W. (2000). The anchors hierarch: using the triangle inequality to survive high dimensional data. In *Proceedings of the Twelfth Conference on Uncertainty in Artificial Intelligence*, pp. 397–405.

Müller, K. R., S. Mika, G. Rätsch, K. Tsuda, and B. Schölkopf (2001). An introduction to kernel-based learning algorithms. *IEEE Transactions on Neural Networks* **12**(2), 181–202.

Müller, P. and F. A. Quintana (2004). Nonparametric Bayesian data analysis. *Statistical Science* **19**(1), 95–110.

Nabney, I. T. (2002). *Netlab: Algorithms for Pattern Recognition*. Springer.

Nadaraya, É. A. (1964). On estimating regression. *Theory of Probability and its Applications* **9**(1), 141–142.

Nag, R., K. Wong, and F. Fallside (1986). Script recognition using hidden markov models. In *ICASSP86*, pp. 2071–2074. IEEE.

Neal, R. M. (1993). Probabilistic inference using Markov chain Monte Carlo methods. Technical Report CRG-TR-93-1, Department of Computer Science, University of Toronto, Canada.

Neal, R. M. (1996). *Bayesian Learning for Neural Networks*. Springer. Lecture Notes in Statistics 118.

Neal, R. M. (1997). Monte Carlo implementation of Gaussian process models for Bayesian regression and classification. Technical Report 9702, Department of Computer Statistics, University of Toronto.

Neal, R. M. (1999). Suppressing random walks in Markov chain Monte Carlo using ordered overrelaxation. In M. I. Jordan (Ed.), *Learning in Graphical Models*, pp. 205–228. MIT Press.

Neal, R. M. (2000). Markov chain sampling for Dirichlet process mixture models. *Journal of Computational and Graphical Statistics* **9**, 249–265.

Neal, R. M. (2003). Slice sampling. *Annals of Statistics* **31**, 705–767.

Neal, R. M. and G. E. Hinton (1999). A new view of the EM algorithm that justifies incremental and other variants. In M. I. Jordan (Ed.), *Learning in Graphical Models*, pp. 355–368. MIT Press.

Nelder, J. A. and R. W. M. Wedderburn (1972). Generalized linear models. *Journal of the Royal Statistical Society, A* **135**, 370–384.

Nilsson, N. J. (1965). *Learning Machines*. McGraw-Hill. Reprinted as *The Mathematical Foundations of Learning Machines*, Morgan Kaufmann, (1990).

Nocedal, J. and S. J. Wright (1999). *Numerical Optimization*. Springer.

Nowlan, S. J. and G. E. Hinton (1992). Simplifying neural networks by soft weight sharing. *Neural Computation* **4**(4), 473–493.

Ogden, R. T. (1997). *Essential Wavelets for Statistical Applications and Data Analysis*. Birkhäuser.

Opper, M. and O. Winther (1999). A Bayesian approach to on-line learning. In D. Saad (Ed.), *On-Line Learning in Neural Networks*, pp. 363–378. Cambridge University Press.

Opper, M. and O. Winther (2000a). Gaussian processes and SVM: mean field theory and leave-one-out. In A. J. Smola, P. L. Bartlett, B. Schölkopf, and D. Shuurmans (Eds.), *Advances in Large Margin Classifiers*, pp. 311–326. MIT Press.

Opper, M. and O. Winther (2000b). Gaussian processes for classification. *Neural Computation* **12**(11), 2655–2684.

Osuna, E., R. Freund, and F. Girosi (1996). Support vector machines: training and applications. A.I. Memo AIM-1602, MIT.

Papoulis, A. (1984). *Probability, Random Variables, and Stochastic Processes* (Second ed.). McGraw-Hill.

Parisi, G. (1988). *Statistical Field Theory*. Addison-Wesley.

Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann.

Pearlmutter, B. A. (1994). Fast exact multiplication by the Hessian. *Neural Computation* **6**(1), 147–160.

Pearlmutter, B. A. and L. C. Parra (1997). Maximum likelihood source separation: a context-sensitive generalization of ICA. In M. C. Mozer, M. I. Jordan, and T. Petsche (Eds.), *Advances in Neural Information Processing Systems*, Volume 9, pp. 613–619. MIT Press.

Pearson, K. (1901). On lines and planes of closest fit to systems of points in space. *The London, Edinburgh and Dublin Philosophical Magazine and Journal of Science, Sixth Series* **2**, 559–572.

Platt, J. C. (1999). Fast training of support vector machines using sequential minimal optimization. In B. Schölkopf, C. J. C. Burges, and A. J. Smola (Eds.), *Advances in Kernel Methods – Support Vector Learning*, pp. 185–208. MIT Press.

Platt, J. C. (2000). Probabilities for SV machines. In A. J. Smola, P. L. Bartlett, B. Schölkopf, and D. Shuurmans (Eds.), *Advances in Large Margin Classifiers*, pp. 61–73. MIT Press.

Platt, J. C., N. Cristianini, and J. Shawe-Taylor (2000). Large margin DAGs for multiclass classification. In S. A. Solla, T. K. Leen, and K. R. Müller (Eds.), *Advances in Neural Information Processing Systems*, Volume 12, pp. 547–553. MIT Press.

Poggio, T. and F. Girosi (1990). Networks for approximation and learning. *Proceedings of the IEEE* **78**(9), 1481–1497.

Powell, M. J. D. (1987). Radial basis functions for multivariable interpolation: a review. In J. C. Mason and M. G. Cox (Eds.), *Algorithms for Approximation*, pp. 143–167. Oxford University Press.

Press, W. H., S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery (1992). *Numerical Recipes in C: The Art of Scientific Computing* (Second ed.). Cambridge University Press.

Qazaz, C. S., C. K. I. Williams, and C. M. Bishop (1997). An upper bound on the Bayesian error bars for generalized linear regression. In S. W. Ellacott, J. C. Mason, and I. J. Anderson (Eds.), *Mathematics of Neural Networks: Models, Algorithms and Applications*, pp. 295–299. Kluwer.

Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning* **1**(1), 81–106.

Quinlan, J. R. (1993). *C4.5: Programs for Machine Learning*. Morgan Kaufmann.

Rabiner, L. and B. H. Juang (1993). *Fundamentals of Speech Recognition*. Prentice Hall.

Rabiner, L. R. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE* **77**(2), 257–285.

Ramasubramanian, V. and K. K. Paliwal (1990). A generalized optimization of the k-d tree for fast nearest-neighbour search. In *Proceedings Fourth IEEE Region 10 International Conference (TENCON'89)*, pp. 565–568.

Ramsey, F. (1931). Truth and probability. In R. Braithwaite (Ed.), *The Foundations of Mathematics and other Logical Essays*. Humanities Press.

Rao, C. R. and S. K. Mitra (1971). *Generalized Inverse of Matrices and Its Applications*. Wiley.

Rasmussen, C. E. (1996). *Evaluation of Gaussian Processes and Other Methods for Non-Linear Regression*. Ph. D. thesis, University of Toronto.

Rasmussen, C. E. and J. Quiñonero-Candela (2005). Healing the relevance vector machine by augmentation. In L. D. Raedt and S. Wrobel (Eds.), *Proceedings of the 22nd International Conference on Machine Learning*, pp. 689–696.

Rasmussen, C. E. and C. K. I. Williams (2006). *Gaussian Processes for Machine Learning*. MIT Press.

Rauch, H. E., F. Tung, and C. T. Striebel (1965). Maximum likelihood estimates of linear dynamical systems. *AIAA Journal* **3**, 1445–1450.

Ricotti, L. P., S. Ragazzini, and G. Martinelli (1988). Learning of word stress in a sub-optimal second order backpropagation neural network. In *Proceedings of the IEEE International Conference on Neural Networks*, Volume 1, pp. 355–361. IEEE.

Ripley, B. D. (1996). *Pattern Recognition and Neural Networks*. Cambridge University Press.

Robbins, H. and S. Monro (1951). A stochastic approximation method. *Annals of Mathematical Statistics* **22**, 400–407.

Robert, C. P. and G. Casella (1999). *Monte Carlo Statistical Methods*. Springer.

Rockafellar, R. (1972). *Convex Analysis*. Princeton University Press.

Rosenblatt, F. (1962). *Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms*. Spartan.

Roth, V. and V. Steinhage (2000). Nonlinear discriminant analysis using kernel functions. In S. A.

Solla, T. K. Leen, and K. R. Müller (Eds.), *Advances in Neural Information Processing Systems*, Volume 12. MIT Press.

Roweis, S. (1998). EM algorithms for PCA and SPCA. In M. I. Jordan, M. J. Kearns, and S. A. Solla (Eds.), *Advances in Neural Information Processing Systems*, Volume 10, pp. 626–632. MIT Press.

Roweis, S. and Z. Ghahramani (1999). A unifying review of linear Gaussian models. *Neural Computation* **11**(2), 305–345.

Roweis, S. and L. Saul (2000, December). Nonlinear dimensionality reduction by locally linear embedding. *Science* **290**, 2323–2326.

Rubin, D. B. (1983). Iteratively reweighted least squares. In *Encyclopedia of Statistical Sciences*, Volume 4, pp. 272–275. Wiley.

Rubin, D. B. and D. T. Thayer (1982). EM algorithms for ML factor analysis. *Psychometrika* **47**(1), 69–76.

Rumelhart, D. E., G. E. Hinton, and R. J. Williams (1986). Learning internal representations by error propagation. In D. E. Rumelhart, J. L. McClelland, and the PDP Research Group (Eds.), *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, Volume 1: Foundations, pp. 318–362. MIT Press. Reprinted in Anderson and Rosenfeld (1988).

Rumelhart, D. E., J. L. McClelland, and the PDP Research Group (Eds.) (1986). *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, Volume 1: Foundations. MIT Press.

Sagan, H. (1969). *Introduction to the Calculus of Variations*. Dover.

Savage, L. J. (1961). The subjective basis of statistical practice. Technical report, Department of Statistics, University of Michigan, Ann Arbor.

Schölkopf, B., J. Platt, J. Shawe-Taylor, A. Smola, and R. C. Williamson (2001). Estimating the support of a high-dimensional distribution. *Neural Computation* **13**(7), 1433–1471.

Schölkopf, B., A. Smola, and K.-R. Müller (1998). Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation* **10**(5), 1299–1319.

Schölkopf, B., A. Smola, R. C. Williamson, and P. L. Bartlett (2000). New support vector algorithms. *Neural Computation* **12**(5), 1207–1245.

Schölkopf, B. and A. J. Smola (2002). *Learning with Kernels*. MIT Press.

Schwarz, G. (1978). Estimating the dimension of a model. *Annals of Statistics* **6**, 461–464.

Schwarz, H. R. (1988). *Finite element methods*. Academic Press.

Seeger, M. (2003). *Bayesian Gaussian Process Models: PAC-Bayesian Generalization Error Bounds and Sparse Approximations*. Ph. D. thesis, University of Edinburg.

Seeger, M., C. K. I. Williams, and N. Lawrence (2003). Fast forward selection to speed up sparse Gaussian processes. In C. M. Bishop and B. Frey (Eds.), *Proceedings Ninth International Workshop on Artificial Intelligence and Statistics, Key West, Florida*.

Shachter, R. D. and M. Peot (1990). Simulation approaches to general probabilistic inference on belief networks. In P. P. Bonissone, M. Henrion, L. N. Kanal, and J. F. Lemmer (Eds.), *Uncertainty in Artificial Intelligence*, Volume 5. Elsevier.

Shannon, C. E. (1948). A mathematical theory of communication. *The Bell System Technical Journal* **27**(3), 379–423 and 623–656.

Shawe-Taylor, J. and N. Cristianini (2004). *Kernel Methods for Pattern Analysis*. Cambridge University Press.

Sietsma, J. and R. J. F. Dow (1991). Creating artificial neural networks that generalize. *Neural Networks* **4**(1), 67–79.

Simard, P., Y. Le Cun, and J. Denker (1993). Efficient pattern recognition using a new transformation distance. In S. J. Hanson, J. D. Cowan, and

C. L. Giles (Eds.), *Advances in Neural Information Processing Systems*, Volume 5, pp. 50–58. Morgan Kaufmann.

Simard, P., B. Victorri, Y. Le Cun, and J. Denker (1992). Tangent prop – a formalism for specifying selected invariances in an adaptive network. In J. E. Moody, S. J. Hanson, and R. P. Lippmann (Eds.), *Advances in Neural Information Processing Systems*, Volume 4, pp. 895–903. Morgan Kaufmann.

Simard, P. Y., D. Steinkraus, and J. Platt (2003). Best practice for convolutional neural networks applied to visual document analysis. In *Proceedings International Conference on Document Analysis and Recognition (ICDAR)*, pp. 958–962. IEEE Computer Society.

Sirovich, L. (1987). Turbulence and the dynamics of coherent structures. *Quarterly Applied Mathematics* **45**(3), 561–590.

Smola, A. J. and P. Bartlett (2001). Sparse greedy Gaussian process regression. In T. K. Leen, T. G. Dietterich, and V. Tresp (Eds.), *Advances in Neural Information Processing Systems*, Volume 13, pp. 619–625. MIT Press.

Spiegelhalter, D. and S. Lauritzen (1990). Sequential updating of conditional probabilities on directed graphical structures. *Networks* **20**, 579–605.

Stinchecombe, M. and H. White (1989). Universal approximation using feed-forward networks with non-sigmoid hidden layer activation functions. In *International Joint Conference on Neural Networks*, Volume 1, pp. 613–618. IEEE.

Stone, J. V. (2004). *Independent Component Analysis: A Tutorial Introduction*. MIT Press.

Sung, K. K. and T. Poggio (1994). Example-based learning for view-based human face detection. A.I. Memo 1521, MIT.

Sutton, R. S. and A. G. Barto (1998). *Reinforcement Learning: An Introduction*. MIT Press.

Svensén, M. and C. M. Bishop (2004). Robust Bayesian mixture modelling. *Neurocomputing* **64**, 235–252.

Tarassenko, L. (1995). Novelty detection for the identification of masses in mamograms. In *Proceedings Fourth IEE International Conference on Artificial Neural Networks*, Volume 4, pp. 442–447. IEE.

Tax, D. and R. Duin (1999). Data domain description by support vectors. In M. Verleysen (Ed.), *Proceedings European Symposium on Artificial Neural Networks, ESANN*, pp. 251–256. D. Facto Press.

Teh, Y. W., M. I. Jordan, M. J. Beal, and D. M. Blei (2006). Hierarchical Dirichlet processes. *Journal of the Americal Statistical Association*. to appear.

Tenenbaum, J. B., V. de Silva, and J. C. Langford (2000, December). A global framework for nonlinear dimensionality reduction. *Science* **290**, 2319–2323.

Tesauro, G. (1994). TD-Gammon, a self-teaching backgammon program, achieves master-level play. *Neural Computation* **6**(2), 215–219.

Thiesson, B., D. M. Chickering, D. Heckerman, and C. Meek (2004). ARMA time-series modelling with graphical models. In M. Chickering and J. Halpern (Eds.), *Proceedings of the Twentieth Conference on Uncertainty in Artificial Intelligence, Banff, Canada*, pp. 552–560. AUAI Press.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, B* **58**, 267–288.

Tierney, L. (1994). Markov chains for exploring posterior distributions. *Annals of Statistics* **22**(4), 1701–1762.

Tikhonov, A. N. and V. Y. Arsenin (1977). *Solutions of Ill-Posed Problems*. V. H. Winston.

Tino, P. and I. T. Nabney (2002). Hierarchical GTM: constructing localized non-linear projection manifolds in a principled way. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **24**(5), 639–656.

Tino, P., I. T. Nabney, and Y. Sun (2001). Using directional curvatures to visualize folding patterns of the GTM projection manifolds. In

G. Dorffner, H. Bischof, and K. Hornik (Eds.), *Artificial Neural Networks – ICANN 2001*, pp. 421–428. Springer.

Tipping, M. E. (1999). Probabilistic visualisation of high-dimensional binary data. In M. S. Kearns, S. A. Solla, and D. A. Cohn (Eds.), *Advances in Neural Information Processing Systems*, Volume 11, pp. 592–598. MIT Press.

Tipping, M. E. (2001). Sparse Bayesian learning and the relevance vector machine. *Journal of Machine Learning Research* **1**, 211–244.

Tipping, M. E. and C. M. Bishop (1997). Probabilistic principal component analysis. Technical Report NCRG/97/010, Neural Computing Research Group, Aston University.

Tipping, M. E. and C. M. Bishop (1999a). Mixtures of probabilistic principal component analyzers. *Neural Computation* **11**(2), 443–482.

Tipping, M. E. and C. M. Bishop (1999b). Probabilistic principal component analysis. *Journal of the Royal Statistical Society, Series B* **21**(3), 611–622.

Tipping, M. E. and A. Faul (2003). Fast marginal likelihood maximization for sparse Bayesian models. In C. M. Bishop and B. Frey (Eds.), *Proceedings Ninth International Workshop on Artificial Intelligence and Statistics, Key West, Florida*.

Tong, S. and D. Koller (2000). Restricted Bayes optimal classifiers. In *Proceedings 17th National Conference on Artificial Intelligence*, pp. 658–664. AAAI.

Tresp, V. (2001). Scaling kernel-based systems to large data sets. *Data Mining and Knowledge Discovery* **5**(3), 197–211.

Uhlenbeck, G. E. and L. S. Ornstein (1930). On the theory of Brownian motion. *Phys. Rev.* **36**, 823–841.

Valiant, L. G. (1984). A theory of the learnable. *Communications of the Association for Computing Machinery* **27**, 1134–1142.

Vapnik, V. N. (1982). *Estimation of dependences based on empirical data*. Springer.

Vapnik, V. N. (1995). *The nature of statistical learning theory*. Springer.

Vapnik, V. N. (1998). *Statistical learning theory*. Wiley.

Veropoulos, K., C. Campbell, and N. Cristianini (1999). Controlling the sensitivity of support vector machines. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI99), Workshop ML3*, pp. 55–60.

Vidakovic, B. (1999). *Statistical Modelling by Wavelets*. Wiley.

Viola, P. and M. Jones (2004). Robust real-time face detection. *International Journal of Computer Vision* **57**(2), 137–154.

Viterbi, A. J. (1967). Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory* **IT-13**, 260–267.

Viterbi, A. J. and J. K. Omura (1979). *Principles of Digital Communication and Coding*. McGraw-Hill.

Wahba, G. (1975). A comparison of GCV and GML for choosing the smoothing parameter in the generalized spline smoothing problem. *Numerical Mathematics* **24**, 383–393.

Wainwright, M. J., T. S. Jaakkola, and A. S. Willsky (2005). A new class of upper bounds on the log partition function. *IEEE Transactions on Information Theory* **51**, 2313–2335.

Walker, A. M. (1969). On the asymptotic behaviour of posterior distributions. *Journal of the Royal Statistical Society, B* **31**(1), 80–88.

Walker, S. G., P. Damien, P. W. Laud, and A. F. M. Smith (1999). Bayesian nonparametric inference for random distributions and related functions (with discussion). *Journal of the Royal Statistical Society, B* **61**(3), 485–527.

Watson, G. S. (1964). Smooth regression analysis. *Sankhyā: The Indian Journal of Statistics. Series A* **26**, 359–372.

Webb, A. R. (1994). Functional approximation by feed-forward networks: a least-squares approach to generalisation. *IEEE Transactions on Neural Networks* **5**(3), 363–371.

Weisstein, E. W. (1999). *CRC Concise Encyclopedia of Mathematics*. Chapman and Hall, and CRC.

Weston, J. and C. Watkins (1999). Multi-class support vector machines. In M. Verlysen (Ed.), *Proceedings ESANN'99, Brussels*. D-Facto Publications.

Whittaker, J. (1990). *Graphical Models in Applied Multivariate Statistics*. Wiley.

Widrow, B. and M. E. Hoff (1960). Adaptive switching circuits. In *IRE WESCON Convention Record*, Volume 4, pp. 96–104. Reprinted in Anderson and Rosenfeld (1988).

Widrow, B. and M. A. Lehr (1990). 30 years of adaptive neural networks: perceptron, madeline, and backpropagation. *Proceedings of the IEEE* **78**(9), 1415–1442.

Wiegerinck, W. and T. Heskes (2003). Fractional belief propagation. In S. Becker, S. Thrun, and K. Obermayer (Eds.), *Advances in Neural Information Processing Systems*, Volume 15, pp. 455–462. MIT Press.

Williams, C. K. I. (1998). Computation with infinite neural networks. *Neural Computation* **10**(5), 1203–1216.

Williams, C. K. I. (1999). Prediction with Gaussian processes: from linear regression to linear prediction and beyond. In M. I. Jordan (Ed.), *Learning in Graphical Models*, pp. 599–621. MIT Press.

Williams, C. K. I. and D. Barber (1998). Bayesian classification with Gaussian processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **20**, 1342–1351.

Williams, C. K. I. and M. Seeger (2001). Using the Nystrom method to speed up kernel machines. In T. K. Leen, T. G. Dietterich, and V. Tresp (Eds.), *Advances in Neural Information Processing Systems*, Volume 13, pp. 682–688. MIT Press.

Williams, O., A. Blake, and R. Cipolla (2005). Sparse Bayesian learning for efficient visual tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **27**(8), 1292–1304.

Williams, P. M. (1996). Using neural networks to model conditional multivariate densities. *Neural Computation* **8**(4), 843–854.

Winn, J. and C. M. Bishop (2005). Variational message passing. *Journal of Machine Learning Research* **6**, 661–694.

Zarchan, P. and H. Musoff (2005). *Fundamentals of Kalman Filtering: A Practical Approach* (Second ed.). AIAA.

Index

Page numbers in **bold** indicate the primary source of information for the corresponding topic.

1-of-K coding scheme, 424

acceptance criterion, **538**, 541, 544
activation function, **180**, 213, 227
active constraint, 328, **709**
AdaBoost, 657, **658**
adaline, 196
adaptive rejection sampling, 530
ADF, *see* assumed density filtering
AIC, *see* Akaike information criterion
Akaike information criterion, **33**, 217
α family of divergences, 469
α recursion, 620
ancestral sampling, **365**, 525, 613
annular flow, 679
AR model, *see* autoregressive model
arc, 360
ARD, *see* automatic relevance determination
ARMA, *see* autoregressive moving average
assumed density filtering, 510
autoassociative networks, 592
automatic relevance determination, 259, 312, **349**, 485, 582
autoregressive hidden Markov model, 632
autoregressive model, 609
autoregressive moving average, 304

back-tracking, **415**, 630
backgammon, 3
backpropagation, 241
bagging, 656
basis function, **138**, 172, 204, 227
batch training, 240
Baum-Welch algorithm, 618
Bayes' theorem, 15
Bayes, Thomas, 21
Bayesian analysis, vii, 9, **21**
    hierarchical, 372
    model averaging, 654
Bayesian information criterion, 33, **216**
Bayesian model comparison, **161**, 473, 483
Bayesian network, 360
Bayesian probability, 21
belief propagation, 403
Bernoulli distribution, **69**, 113, 685
    mixture model, 444
Bernoulli, Jacob, 69
beta distribution, **71**, 686
beta recursion, 621
between-class covariance, 189
bias, **27**, 149
bias parameter, **138**, 181, 227, 346
bias-variance trade-off, 147
BIC, *see* Bayesian information criterion
binary entropy, 495
binomial distribution, **70**, 686

biological sequence, 610
bipartite graph, 401
bits, 49
blind source separation, 591
blocked path, 374, **378**, 384
Boltzmann distribution, 387
Boltzmann, Ludwig Eduard, 53
Boolean logic, 21
boosting, 657
bootstrap, **23**, 656
bootstrap filter, 646
box constraints, **333**, 342
Box-Muller method, 527

C4.5, 663
calculus of variations, 462
canonical correlation analysis, 565
canonical link function, 212
CART, *see* classification and regression trees
Cauchy distribution, **527**, 529, 692
causality, 366
CCA, *see* canonical correlation analysis
central differences, 246
central limit theorem, 78
chain graph, 393
chaining, 555
Chapman-Kolmogorov equations, 397
child node, 361
Cholesky decomposition, 528
chunking, 335
circular normal, *see* von Mises distribution
classical probability, 21
classification, 3
classification and regression trees, 663
clique, 385
clustering, 3
clutter problem, 511
co-parents, **383**, 492
code-book vectors, 429
combining models, 45, **653**
committee, 655
complete data set, 440
completing the square, 86
computational learning theory, 326, 344
concave function, 56
concentration parameter, **108**, 693
condensation algorithm, 646
conditional entropy, 55
conditional expectation, 20
conditional independence, 46, **372**, 383
conditional mixture model, *see* mixture model
conditional probability, 14
conjugate prior, 68, 98, **117**, 490
convex duality, 494
convex function, **55**, 493
convolutional neural network, 267
correlation matrix, 567
cost function, 41
covariance, 20
  between-class, 189
  within-class, 189
covariance matrix
  diagonal, 84
  isotropic, 84
  partitioned, **85**, 307
  positive definite, 308
Cox's axioms, 21
credit assignment, 3
cross-entropy error function, **206**, 209, 235, 631, 666
cross-validation, **32**, 161
cumulative distribution function, 18
curse of dimensionality, **33**, 36
curve fitting, 4

D map, *see* dependency map
d-separation, 373, **378**, 443
DAG, *see* directed acyclic graph
DAGSVM, 339
data augmentation, 537
data compression, 429
decision boundary, **39**, 179
decision region, **39**, 179
decision surface, *see* decision boundary
decision theory, 38
decision tree, 654, **663**, 673
decomposition methods, 335
degrees of freedom, 559
degrees-of-freedom parameter, **102**, 693
density estimation, 3, **67**

density network, 597
dependency map, 392
descendant node, 376
design matrix, **142**, 347
differential entropy, 53
digamma function, 687
directed acyclic graph, 362
directed cycle, 362
directed factorization, 381
Dirichlet distribution, **76**, 687
Dirichlet, Lejeune, 77
discriminant function, 43, 180, **181**
discriminative model, **43**, 203
distortion measure, 424
distributive law of multiplication, 396
DNA, 610
document retrieval, 299
dual representation, **293**, 329
dual-energy gamma densitometry, 678
dynamic programming, 411
dynamical system, 548

E step, *see* expectation step
early stopping, 259
ECM, *see* expectation conditional maximization
edge, 360
effective number of observations, **72**, 101
effective number of parameters, 9, **170**, 281
elliptical K-means, 444
EM, *see* expectation maximization
emission probability, 611
empirical Bayes, *see* evidence approximation
energy function, 387
entropy, 49
    conditional, 55
    differential, 53
    relative, 55
EP, *see* expectation propagation
ϵ-tube, 341
ϵ-insensitive error function, 340
equality constraint, 709
equivalent kernel, **159**, 301
erf function, 211
error backpropagation, *see* backpropagation
error function, **5**, 23
error-correcting output codes, 339
Euler, Leonhard, 465
Euler-Lagrange equations, 705
evidence approximation, **165**, 347, 581
evidence function, 161
expectation, 19
expectation conditional maximization, 454
expectation maximization, 113, **423**, 440
    Gaussian mixture, 435
    generalized, 454
    sampling methods, 536
expectation propagation, 315, 468, **505**
expectation step, 437
explaining away, 378
exploitation, 3
exploration, 3
exponential distribution, **526**, 688
exponential family, 68, **113**, 202, 490
extensive variables, 490

face detection, 2
face tracking, 355
factor analysis, 583
    mixture model, 595
factor graph, 360, **399**, 625
factor loading, 584
factorial hidden Markov model, 633
factorized distribution, **464**, 476
feature extraction, 2
feature map, 268
feature space, **292**, 586
Fisher information matrix, 298
Fisher kernel, 298
Fisher's linear discriminant, 186
flooding schedule, 417
forward kinematics, 272
forward problem, 272
forward propagation, **228**, 243
forward-backward algorithm, 618
fractional belief propagation, 517
frequentist probability, 21
fuel system, 376
function interpolation, 299
functional, 462, **703**
    derivative, 463

gamma densitometry, 678
gamma distribution, 529, **688**
gamma function, 71
gating function, 672
Gauss, Carl Friedrich, 79
Gaussian, 24, **78**, 688
  conditional, **85**, 93
  marginal, **88**, 93
  maximum likelihood, 93
  mixture, 110, 270, 273, **430**
  sequential estimation, 94
  sufficient statistics, 93
  wrapped, 110
Gaussian kernel, 296
Gaussian process, 160, **303**
Gaussian random field, 305
Gaussian-gamma distribution, **101**, 690
Gaussian-Wishart distribution, **102**, 475, 478, **690**
GEM, *see* expectation maximization, generalized
generalization, 2
generalized linear model, **180**, 213
generalized maximum likelihood, *see* evidence approximation
generative model, **43**, 196, 297, 365, 572, 631
generative topographic mapping, 597
  directional curvature, 599
  magnification factor, 599
geodesic distance, 596
Gibbs sampling, 542
  blocking, 546
Gibbs, Josiah Willard, 543
Gini index, 666
global minimum, 237
gradient descent, 240
Gram matrix, 293
graph-cut algorithm, 390
graphical model, 359
  bipartite, 401
  directed, 360
  factorization, 362, 384
  fully connected, 361
  inference, 393
  tree, 398
  treewidth, 417
  triangulated, 416
  undirected, 360
Green's function, 299
GTM, *see* generative topographic mapping

Hamilton, William Rowan, 549
Hamiltonian dynamics, 548
Hamiltonian function, 549
Hammersley-Clifford theorem, 387
handwriting recognition, 1, 610, 614
handwritten digit, 565, 614, **677**
head-to-head path, 376
head-to-tail path, 375
Heaviside step function, 206
Hellinger distance, 470
Hessian matrix, 167, 215, 217, 238, **249**
  diagonal approximation, 250
  exact evaluation, 253
  fast multiplication, 254
  finite differences, 252
  inverse, 252
  outer product approximation, 251
heteroscedastic, **273**, 311
hidden Markov model, 297, **610**
  autoregressive, 632
  factorial, 633
  forward-backward algorithm, 618
  input-output, 633
  left-to-right, 613
  maximum likelihood, 615
  scaling factor, 627
  sum-product algorithm, 625
  switching, 644
  variational inference, 625
hidden unit, 227
hidden variable, 84, **364**, 430, 559
hierarchical Bayesian model, 372
hierarchical mixture of experts, 673
hinge error function, 337
Hinton diagram, 584
histogram density estimation, 120
HME, *see* hierarchical mixture of experts
hold-out set, 11
homogeneous flow, 679
homogeneous kernel, 292
homogeneous Markov chain, **540**, 608

Hooke's law, 580
hybrid Monte Carlo, 548
hyperparameter, **71**, 280, 311, 346, 372, 502
hyperprior, 372

I map, *see* independence map
i.i.d., *see* independent identically distributed
ICA, *see* independent component analysis
ICM, *see* iterated conditional modes
ID3, 663
identifiability, 435
image de-noising, 387
importance sampling, 525, **532**
importance weights, 533
improper prior, **118**, 259, 472
imputation step, 537
imputation-posterior algorithm, 537
inactive constraint, 328, **709**
incomplete data set, 440
independence map, 392
independent component analysis, 591
independent factor analysis, 592
independent identically distributed, **26**, 379
independent variables, 17
independent, identically distributed, 605
induced factorization, 485
inequality constraint, 709
inference, 38, **42**
information criterion, 33
information geometry, 298
information theory, 48
input-output hidden Markov model, 633
intensive variables, 490
intrinsic dimensionality, 559
invariance, 261
inverse gamma distribution, 101
inverse kinematics, 272
inverse problem, 272
inverse Wishart distribution, 102
IP algorithm, *see* imputation-posterior algorithm
IRLS, *see* iterative reweighted least squares
Ising model, 389
isomap, 596
isometric feature map, 596
iterated conditional modes, **389**, 415
iterative reweighted least squares, **207**, 210, 316, 354, 672

Jacobian matrix, **247**, 264
Jensen's inequality, 56
join tree, 416
junction tree algorithm, 392, **416**

K nearest neighbours, 125
K-means clustering algorithm, **424**, 443
K-medoids algorithm, 428
Kalman filter, 304, **637**
    extended, 644
Kalman gain matrix, 639
Kalman smoother, 637
Karhunen-Loève transform, 561
Karush-Kuhn-Tucker conditions, 330, 333, 342, **710**
kernel density estimator, **122**, 326
kernel function, 123, **292**, 294
    Fisher, 298
    Gaussian, 296
    homogeneous, 292
    nonvectorial inputs, 297
    stationary, 292
kernel PCA, 586
kernel regression, 300, **302**
kernel substitution, 292
kernel trick, 292
kinetic energy, 549
KKT, *see* Karush-Kuhn-Tucker conditions
KL divergence, *see* Kullback-Leibler divergence
kriging, *see* Gaussian process
Kullback-Leibler divergence, **55**, 451, 468, 505

Lagrange multiplier, 707
Lagrange, Joseph-Louis, 329
Lagrangian, 328, 332, 341, **708**
laminar flow, 678
Laplace approximation, **213**, 217, 278, 315, 354
Laplace, Pierre-Simon, 24
large margin, *see* margin
lasso, 145
latent class analysis, 444
latent trait model, 597
latent variable, 84, **364**, 430, 559

lattice diagram, **414**, 611, 621, 629
LDS, *see* linear dynamical system
leapfrog discretization, 551
learning, 2
learning rate parameter, 240
least-mean-squares algorithm, 144
leave-one-out, 33
likelihood function, 22
likelihood weighted sampling, 534
linear discriminant, 181
  Fisher, 186
linear dynamical system, 84, **635**
  inference, 638
linear independence, 696
linear regression, 138
  EM, 448
  mixture model, 667
  variational, 486
linear smoother, 159
linear-Gaussian model, 87, **370**
linearly separable, 179
link, 360
link function, 180, 213
Liouville's Theorem, 550
LLE, *see* locally linear embedding
LMS algorithm, *see* least-mean-squares algorithm
local minimum, 237
local receptive field, 268
locally linear embedding, 596
location parameter, 118
log odds, 197
logic sampling, 525
logistic regression, **205**, 336
  Bayesian, 217, 498
  mixture model, 670
  multiclass, 209
logistic sigmoid, 114, 139, **197**, 205, 220, 227, 495
logit function, 197
loopy belief propagation, 417
loss function, 41
loss matrix, 41
lossless data compression, 429
lossy data compression, 429
lower bound, 484

M step, *see* maximization step
machine learning, vii
macrostate, 51
Mahalanobis distance, 80
manifold, **38**, 590, 595, 681
MAP, *see* maximum posterior
margin, 326, **327**, 502
  error, 334
  soft, 332
marginal likelihood, **162**, 165
marginal probability, 14
Markov blanket, **382**, 384, 545
Markov boundary, *see* Markov blanket
Markov chain, 397, **539**
  first order, 607
  homogeneous, **540**, 608
  second order, 608
Markov chain Monte Carlo, 537
Markov model, 607
  homogeneous, 612
Markov network, *see* Markov random field
Markov random field, 84, 360, **383**
max-sum algorithm, **411**, 629
maximal clique, 385
maximal spanning tree, 416
maximization step, 437
maximum likelihood, 9, **23**, 26, 116
  Gaussian mixture, 432
  singularities, 480
  type 2, *see* evidence approximation
maximum margin, *see* margin
maximum posterior, **30**, 441
MCMC, *see* Markov chain Monte Carlo
MDN, *see* mixture density network
MDS, *see* multidimensional scaling
mean, 24
mean field theory, 465
mean value theorem, 52
measure theory, 19
memory-based methods, 292
message passing, 396
  pending message, 417
  schedule, 417
  variational, 491
Metropolis algorithm, 538
Metropolis-Hastings algorithm, 541

microstate, 51
minimum risk, 44
Minkowski loss, 48
missing at random, **441**, 579
missing data, 579
mixing coefficient, 111
mixture component, 111
mixture density network, **272**, 673
mixture distribution, *see* mixture model
mixture model, 162, **423**
  conditional, 273, **666**
  linear regression, 667
  logistic regression, 670
  symmetries, 483
mixture of experts, 672
mixture of Gaussians, 110, 270, 273, **430**
MLP, *see* multilayer perceptron
MNIST data, 677
model comparison, 6, 32, **161**, 473, 483
model evidence, 161
model selection, 162
moment matching, **506**, 510
momentum variable, 548
Monte Carlo EM algorithm, 536
Monte Carlo sampling, 24, **523**
Moore-Penrose pseudo-inverse, *see* pseudo-inverse
moralization, **391**, 401
MRF, *see* Markov random field
multidimensional scaling, 596
multilayer perceptron, 226, **229**
multimodality, 272
multinomial distribution, 76, 114, **690**
multiplicity, 51
mutual information, 55, **57**

Nadaraya-Watson, *see* kernel regression
naive Bayes model, 46, **380**
nats, 50
natural language modelling, 610
natural parameters, 113
nearest-neighbour methods, 124
neural network, 225
  convolutional, 267
  regularization, 256
  relation to Gaussian process, 319
Newton-Raphson, **207**, 317
node, 360
noiseless coding theorem, 50
nonidentifiability, 585
noninformative prior, 23, **117**
nonparametric methods, 68, **120**
normal distribution, *see* Gaussian
normal equations, 142
normal-gamma distribution, **101**, 691
normal-Wishart distribution, **102**, 475, 478, 691
normalized exponential, *see* softmax function
novelty detection, 44
ν-SVM, 334

object recognition, 366
observed variable, 364
Occam factor, 217
oil flow data, 34, 560, 568, **678**
Old Faithful data, 110, 479, 484, **681**
on-line learning, *see* sequential learning
one-versus-one classifier, **183**, 339
one-versus-the-rest classifier, **182**, 338
ordered over-relaxation, 545
Ornstein-Uhlenbeck process, 305
orthogonal least squares, 301
outlier, 44, 185, 212
outliers, **103**
over-fitting, **6**, 147, 434, 464
over-relaxation, 544

PAC learning, *see* probably approximately correct
PAC-Bayesian framework, 345
parameter shrinkage, 144
parent node, 361
particle filter, 645
partition function, **386**, 554
Parzen estimator, *see* kernel density estimator
Parzen window, 123
pattern recognition, vii
PCA, *see* principal component analysis
pending message, 417
perceptron, 192
  convergence theorem, 194
  hardware, 196
perceptron criterion, 193
perfect map, 392

periodic variable, 105
phase space, 549
photon noise, 680
plate, 363
polynomial curve fitting, 4, 362
polytree, 399
position variable, 548
positive definite covariance, 81
positive definite matrix, 701
positive semidefinite covariance, 81
positive semidefinite matrix, 701
posterior probability, 17
posterior step, 537
potential energy, 549
potential function, 386
power EP, 517
power method, 563
precision matrix, 85
precision parameter, 24
predictive distribution, **30**, 156
preprocessing, 2
principal component analysis, **561**, 572, 593
  Bayesian, 580
  EM algorithm, 577
  Gibbs sampling, 583
  mixture distribution, 595
  physical analogy, 580
principal curve, 595
principal subspace, 561
principal surface, 596
prior, 17
  conjugate, 68, 98, **117**, 490
  consistent, 257
  improper, **118**, 259, 472
  noninformative, 23, **117**
probabilistic graphical model, *see* graphical model
probabilistic PCA, 570
probability, 12
  Bayesian, 21
  classical, 21
  density, 17
  frequentist, 21
  mass function, 19
  prior, 45
  product rule, 13, **14**, 359
  sum rule, 13, **14**, 359
  theory, 12
probably approximately correct, 344
probit function, **211**, 219
probit regression, 210
product rule of probability, 13, **14**, 359
proposal distribution, **528**, 532, 538
protected conjugate gradients, 335
protein sequence, 610
pseudo-inverse, **142**, 185
pseudo-random numbers, 526

quadratic discriminant, 199
quality parameter, 351

radial basis function, 292, **299**
Rauch-Tung-Striebel equations, 637
regression, 3
regression function, **47**, 95
regularization, 10
  Tikhonov, 267
regularized least squares, 144
reinforcement learning, 3
reject option, **42**, 45
rejection sampling, 528
relative entropy, 55
relevance vector, 348
relevance vector machine, 161, **345**
responsibility, 112, **432**, 477
ridge regression, 10
RMS error, *see* root-mean-square error
Robbins-Monro algorithm, 95
robot arm, 272
robustness, **103**, 185
root node, 399
root-mean-square error, 6
Rosenblatt, Frank, 193
rotation invariance, **573**, 585
RTS equations, *see* Rauch-Tung-Striebel equations
running intersection property, 416
RVM, *see* relevance vector machine

sample mean, 27
sample variance, 27
sampling-importance-resampling, 534
scale invariance, 119, **261**

scale parameter, 119
scaling factor, 627
Schwarz criterion, *see* Bayesian information criterion
self-organizing map, 598
sequential data, 605
sequential estimation, 94
sequential gradient descent, 144, 240
sequential learning, **73**, 143
sequential minimal optimization, 335
serial message passing schedule, 417
Shannon, Claude, 55
shared parameters, 368
shrinkage, 10
Shur complement, 87
sigmoid, *see* logistic sigmoid
simplex, 76
single-class support vector machine, 339
singular value decomposition, 143
sinusoidal data, 682
SIR, *see* sampling-importance-resampling
skip-layer connection, 229
slack variable, 331
slice sampling, 546
SMO, *see* sequential minimal optimization
smoother matrix, 159
smoothing parameter, 122
soft margin, 332
soft weight sharing, 269
softmax function, 115, **198**, 236, 274, 356, 497
SOM, *see* self-organizing map
sparsity, 145, 347, **349**, 582
sparsity parameter, 351
spectrogram, 606
speech recognition, **605**, 610
sphereing, 568
spline functions, 139
standard deviation, 24
standardizing, 425, **567**
state space model, 609
  switching, 644
stationary kernel, 292
statistical bias, *see* bias
statistical independence, *see* independent variables
statistical learning theory, *see* computational learning theory, 326, 344
steepest descent, 240
Stirling's approximation, 51
stochastic, 5
stochastic EM, 536
stochastic gradient descent, 144, 240
stochastic process, 305
stratified flow, 678
Student's t-distribution, **102**, 483, 691
subsampling, 268
sufficient statistics, 69, 75, **116**
sum rule of probability, 13, **14**, 359
sum-of-squares error, **5**, 29, 184, 232, 662
sum-product algorithm, 399, **402**
  for hidden Markov model, 625
supervised learning, 3
support vector, 330
support vector machine, 225
  for regression, 339
  multiclass, 338
survival of the fittest, 646
SVD, *see* singular value decomposition
SVM, *see* support vector machine
switching hidden Markov model, 644
switching state space model, 644
synthetic data sets, 682

tail-to-tail path, 374
tangent distance, 265
tangent propagation, 262, **263**
tapped delay line, 609
target vector, 2
test set, 2, **32**
threshold parameter, 181
tied parameters, 368
Tikhonov regularization, 267
time warping, 615
tomography, 679
training, 2
training set, 2
transition probability, **540**, 610
translation invariance, 118, **261**
tree-reweighted message passing, 517
treewidth, 417

trellis diagram, *see* lattice diagram
triangulated graph, 416
type 2 maximum likelihood, *see* evidence approximation

undetermined multiplier, *see* Lagrange multiplier
undirected graph, *see* Markov random field
uniform distribution, 692
uniform sampling, 534
uniquenesses, 584
unobserved variable, *see* latent variable
unsupervised learning, 3
utility function, 41

validation set, 11, **32**
Vapnik-Chervonenkis dimension, 344
variance, **20**, 24, 149
variational inference, 315, **462**, 635
  for Gaussian mixture, 474
  for hidden Markov model, 625
  local, 493
VC dimension, *see* Vapnik-Chervonenkis dimension
vector quantization, 429
vertex, *see* node
visualization, 3
Viterbi algorithm, 415, **629**
von Mises distribution, **108**, 693

wavelets, 139
weak learner, 657
weight decay, 10, **144**, 257
weight parameter, 227
weight sharing, 268
  soft, 269
weight vector, 181
weight-space symmetry, **231**, 281
weighted least squares, 668
well-determined parameters, 170
whitening, 299, **568**
Wishart distribution, **102**, 693
within-class covariance, 189
Woodbury identity, 696
wrapped distribution, 110

Yellowstone National Park, 110, **681**