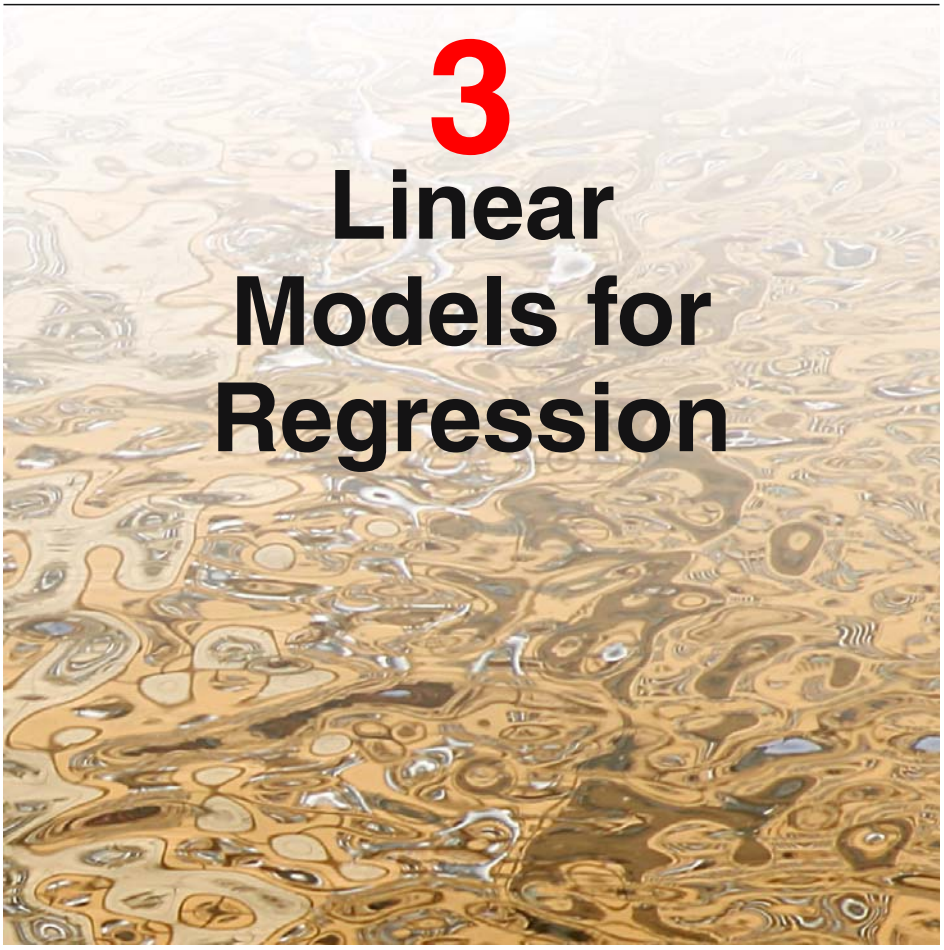


3 Linear Models for Regression

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

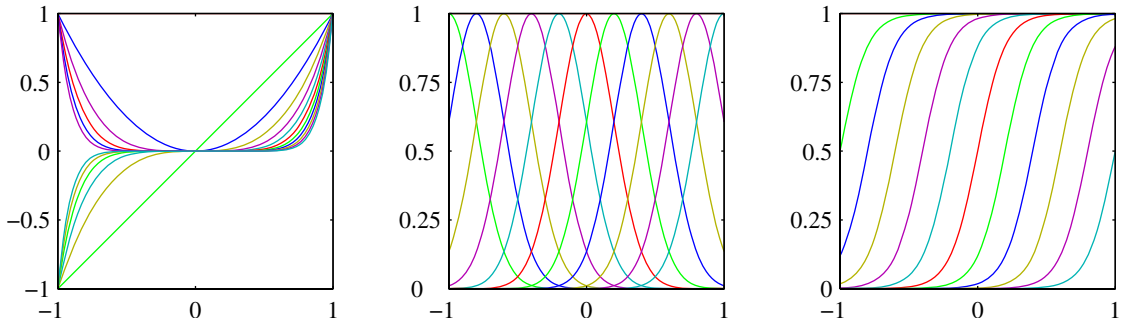


Figure 3.1 Examples of basis functions, showing polynomials on the left, Gaussians of the form (3.4) in the centre, and sigmoidal of the form (3.5) on the right.

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

$$\frac{1}{\beta_{\mathrm{ML}}} = \frac{1}{N} \sum_{n=1}^{N} \{t_n - \mathbf{w}_{\mathrm{ML}}^{\mathrm{T}} \boldsymbol{\phi}(\mathbf{x}_n)\}^2 \tag{3.21}$$

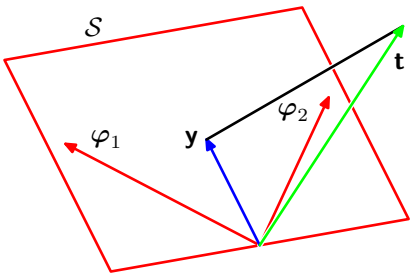


Figure 3.2 Geometrical interpretation of the least-squares solution, in an N-dimensional space whose axes are the values of $t_1, \ldots, t_N$. The least-squares regression function is obtained by finding the orthogonal projection of the data vector $\mathbf{t}$ onto the subspace spanned by the basis 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

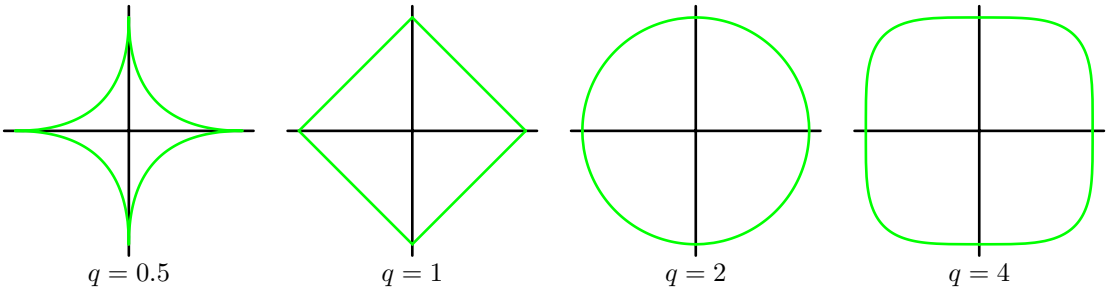


Figure 3.3 Contours of the regularization term in (3.29) for various values of the parameter q.

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.

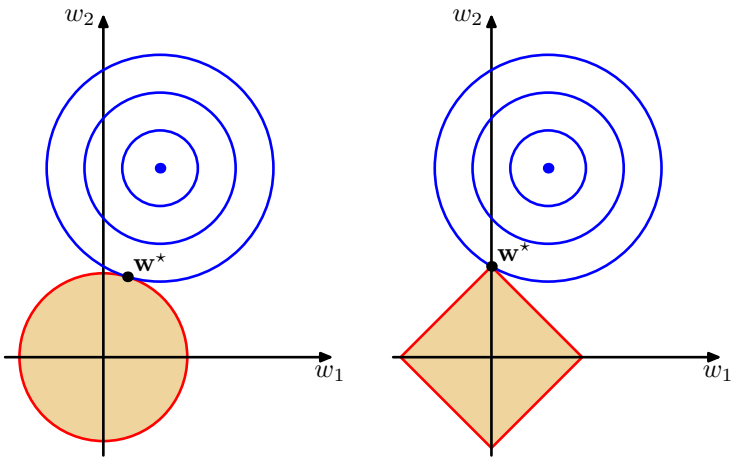


Figure 3.4 Plot of the contours of the unregularized error function (blue) along with the constraint region (3.30) for the quadratic regularizer $q = 2$ on the left and the lasso regularizer $q = 1$ on the right, in which the optimum value for the parameter vector $\mathbf{w}$ is denoted by $\mathbf{w}^\star$. 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

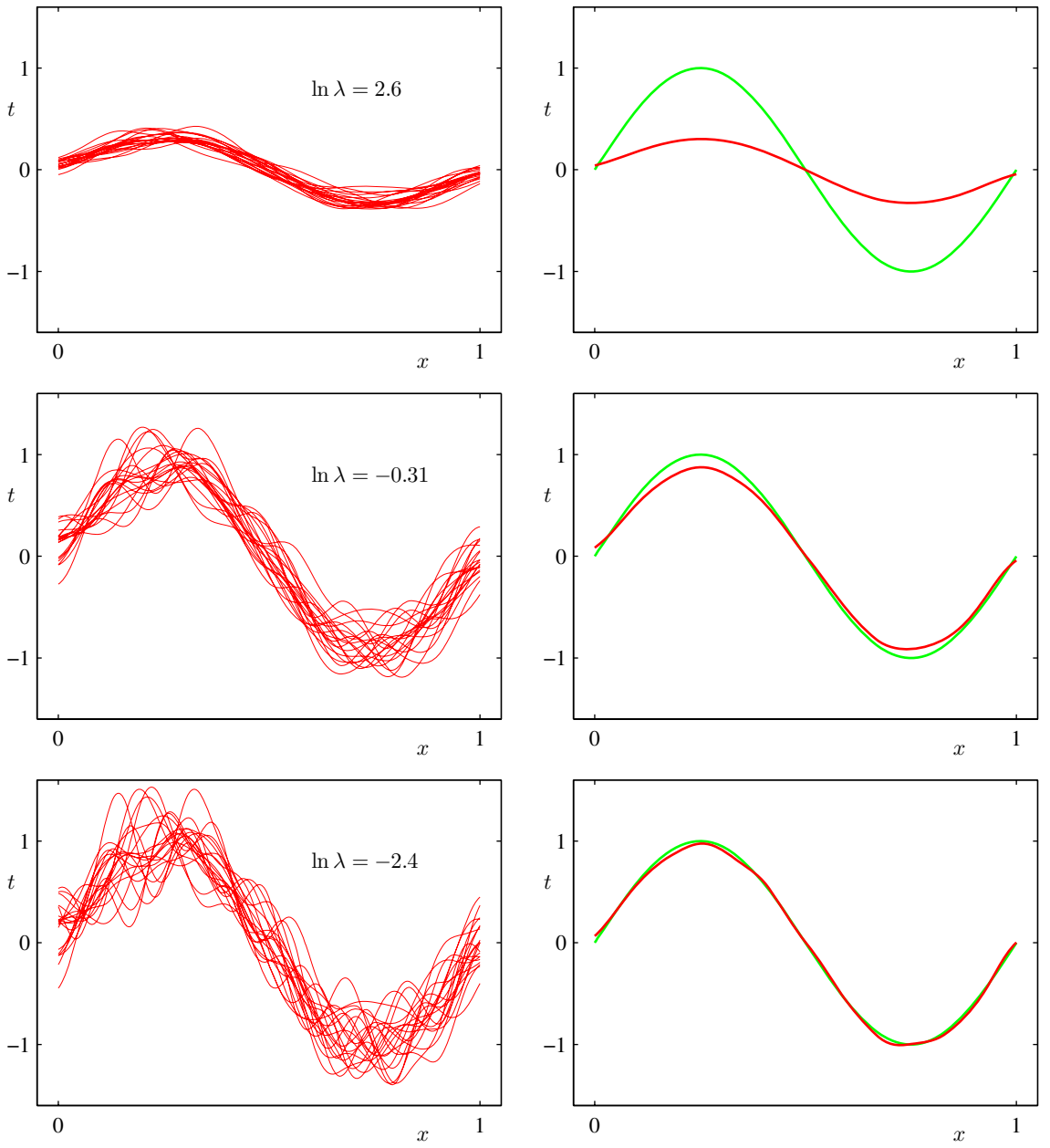


Figure 3.5 Illustration of the dependence of bias and variance on model complexity, governed by a regularization parameter λ, using the sinusoidal data set from Chapter 1. There are $L = 100$ data sets, each having $N = 25$ data points, and there are 24 Gaussian basis functions in the model so that the 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).

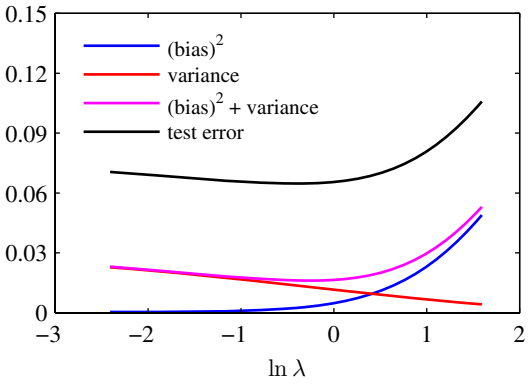


Figure 3.6 Plot of squared bias and variance, together with their sum, corresponding to the results shown in Figure 3.5. Also shown is the average test set error for a test data set size of 1000 points. The minimum value of $(\text{bias})^2 + \text{variance}$ occurs around $\ln \lambda = -0.31$, which 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

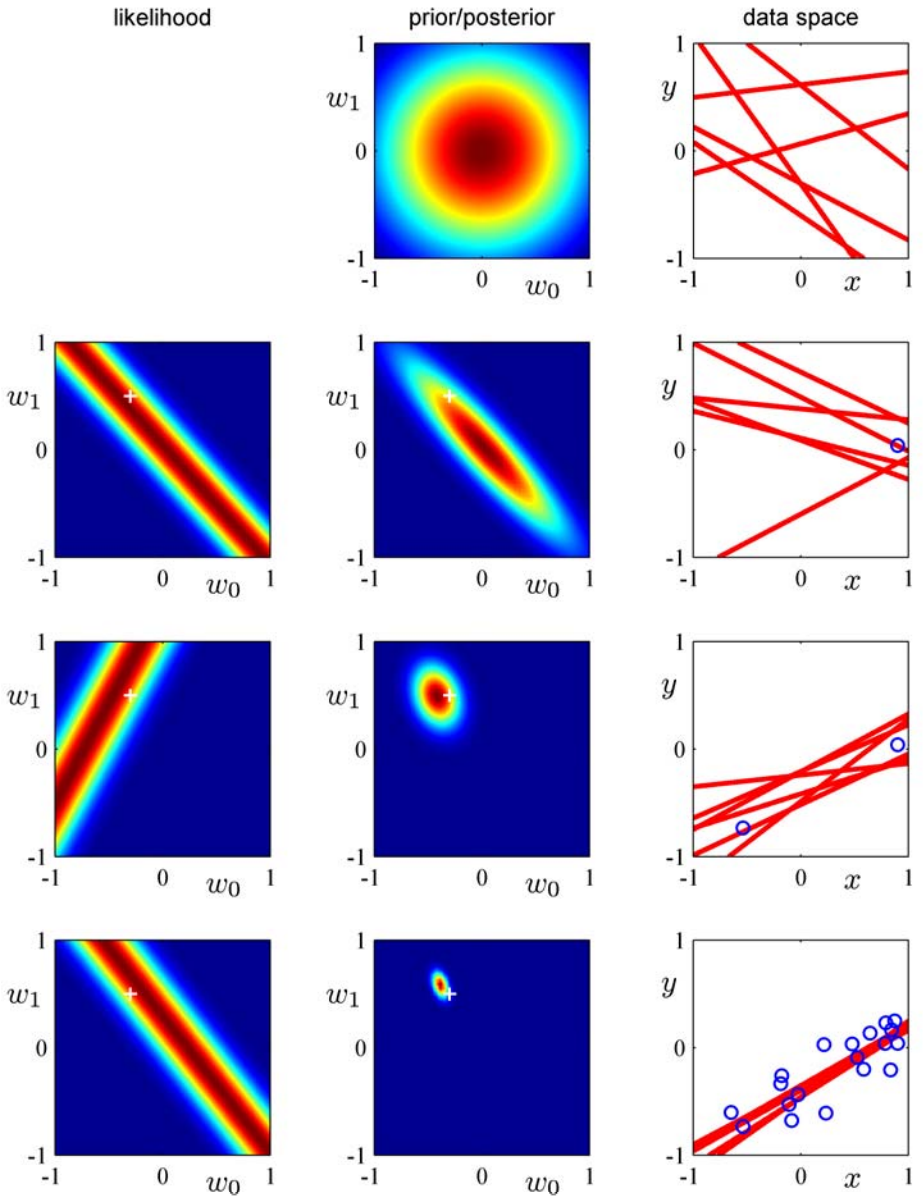


Figure 3.7 Illustration of sequential Bayesian learning for a simple linear model of the form $y(x, \mathbf{w}) = w_0 + w_1 x$. A detailed description of this figure is given in the text.

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,

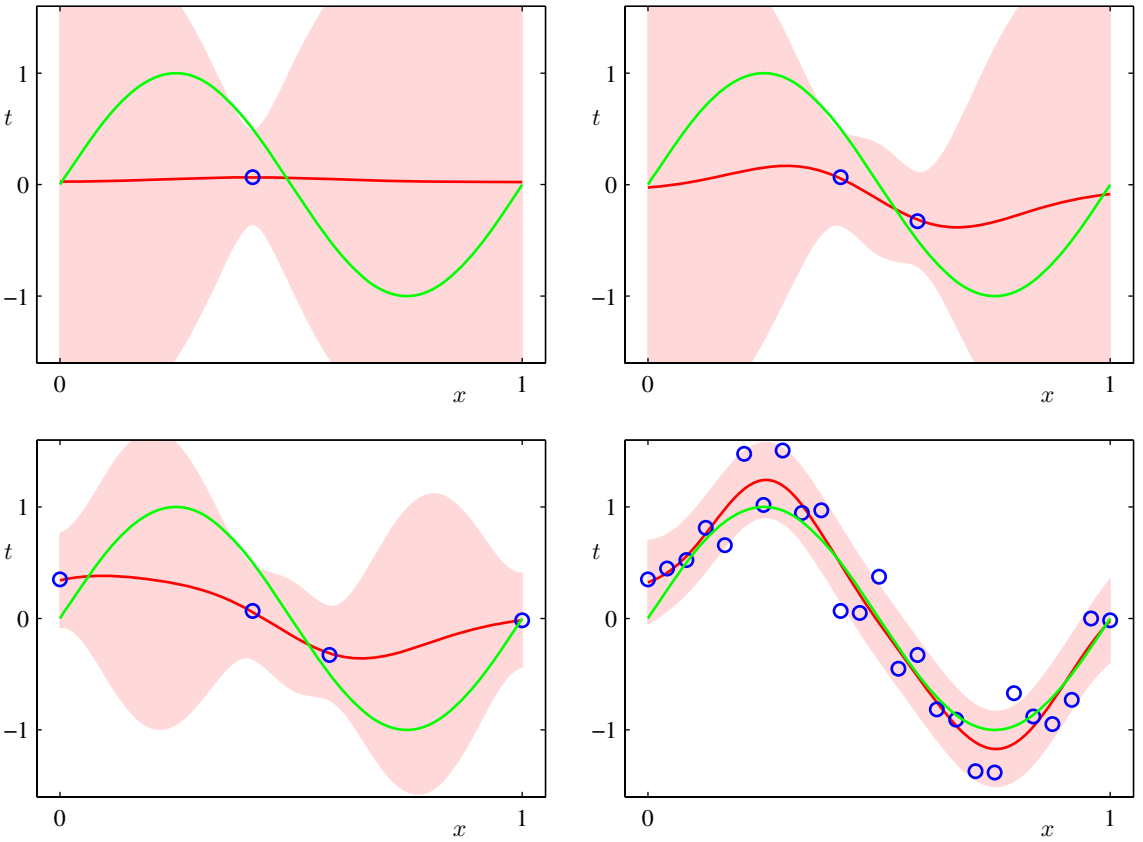


Figure 3.8 Examples of the predictive distribution (3.58) for a model consisting of 9 Gaussian basis functions of the form (3.4) using the synthetic sinusoidal data set of Section 1.1. See the text for a detailed discussion.

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.

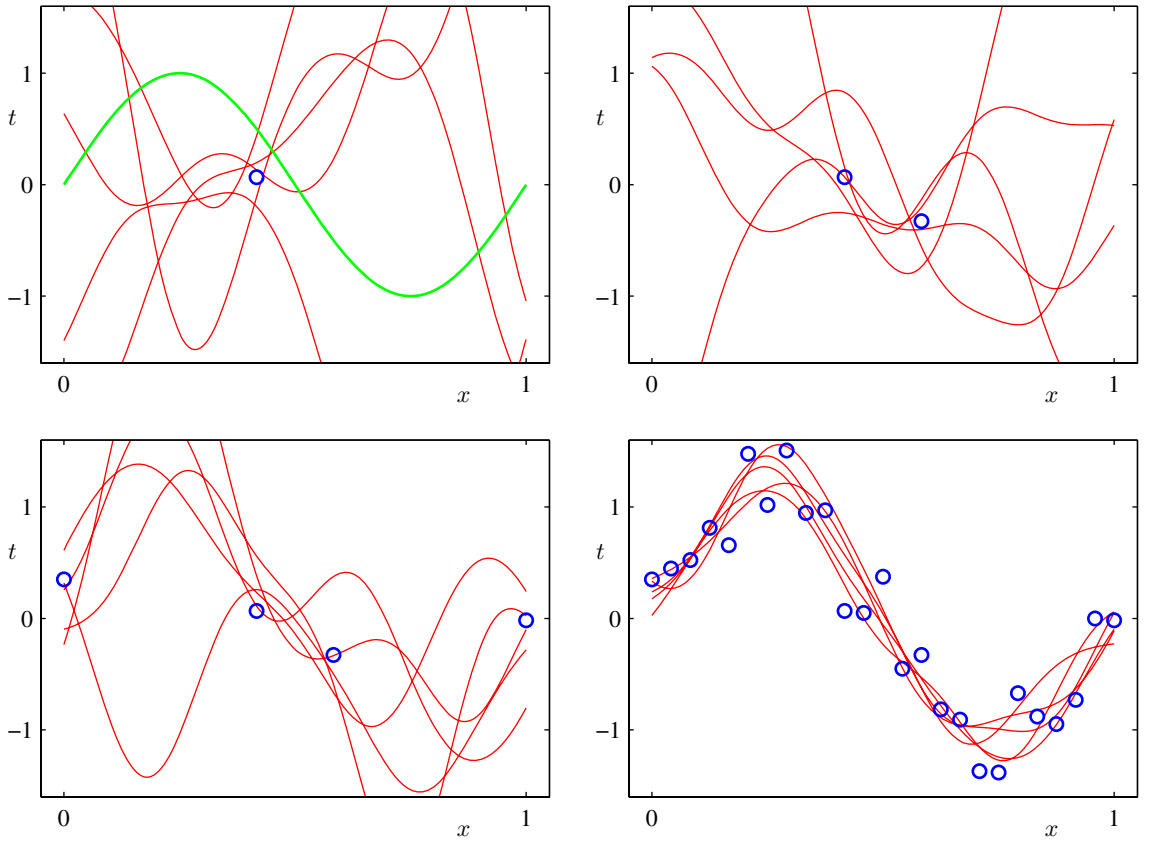


Figure 3.9 Plots of the function $y(x, \mathbf{w})$ using samples from the posterior distributions over $\mathbf{w}$ corresponding to the plots in Figure 3.8.

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

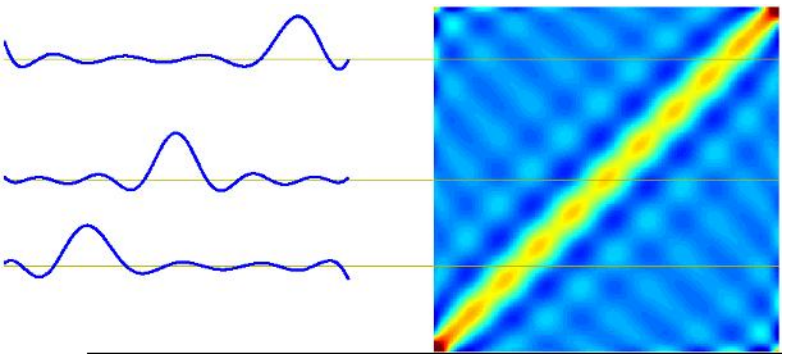


Figure 3.10 The equivalent kernel $k(x, x')$ for the Gaussian basis functions in Figure 3.1, shown as a plot of x versus x', together with three slices through this matrix corresponding to three different values of x. The data set used to generate this kernel comprised 200 values of x equally spaced 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.

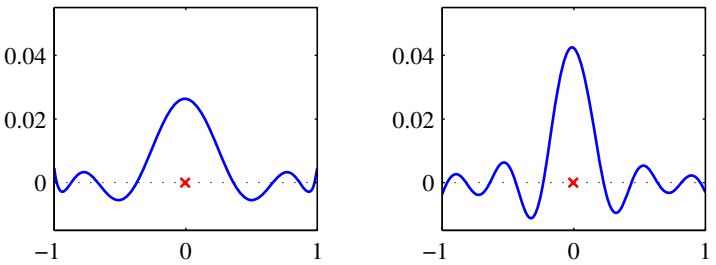


Figure 3.11 Examples of equivalent kernels $k(x, x')$ for $x = 0$ plotted as a function of x', corresponding (left) to the polynomial basis functions and (right) to the sigmoidal basis functions shown in Figure 3.1. Note that these are localized functions of x' even though the corresponding basis functions 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

$$p(\mathcal{D}) = \int p(\mathcal{D}|w) p(w) \, \mathrm{d}w \simeq p(\mathcal{D}|w_{\mathrm{MAP}}) \frac{\Delta w_{\mathrm{posterior}}}{\Delta w_{\mathrm{prior}}} \tag{3.70}$$

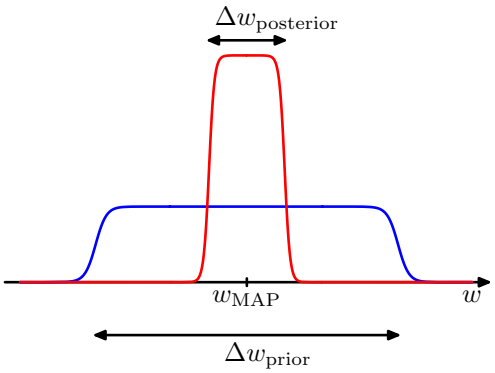


Figure 3.12 We can obtain a rough approximation to the model evidence if we assume that the posterior distribution over parameters is sharply peaked around its mode w_{MAP}.

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

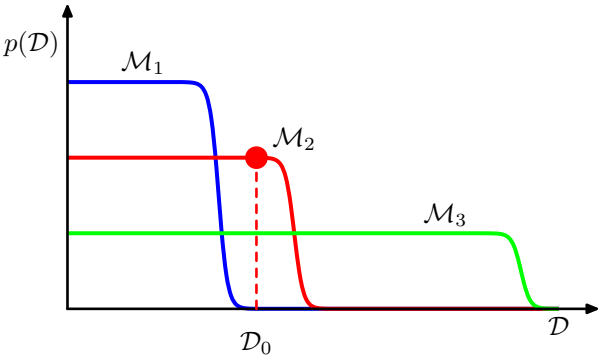


Figure 3.13 Schematic illustration of the distribution of data sets for three models of different complexity, in which $\mathcal{M}_1$ is the simplest and $\mathcal{M}_3$ is the most complex. Note that the distributions are normalized. In this example, for the particular observed data set $\mathcal{D}_0$, 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

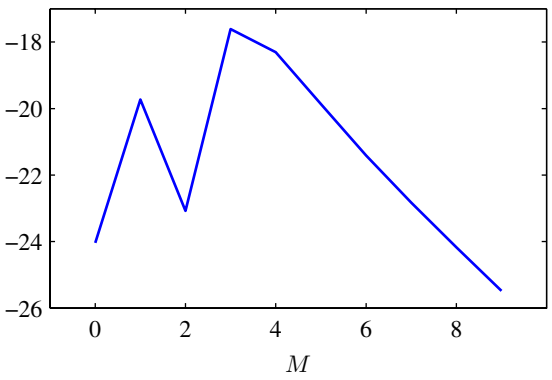


Figure 3.14 Plot of the model evidence versus the order M, for the polynomial regression model, showing that the evidence favours the model with $M = 3$.

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.

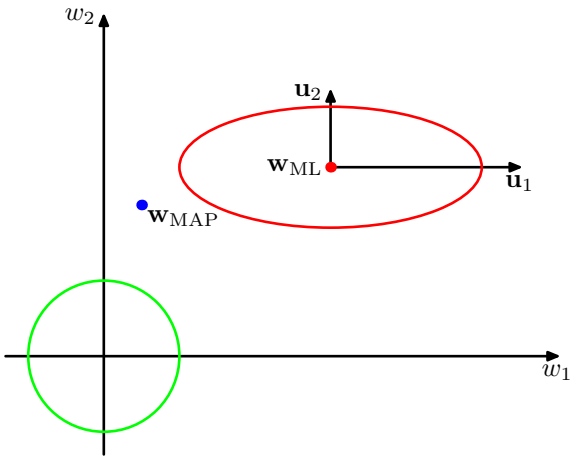


Figure 3.15 Contours of the likelihood function (red) and the prior (green) in which the axes in parameter space have been rotated to align with the eigenvectors $\mathbf{u}_i$ of the Hessian. For $\alpha = 0$, the mode of the posterior is given by the maximum likelihood solution $\mathbf{w}_{\mathrm{ML}}$, 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

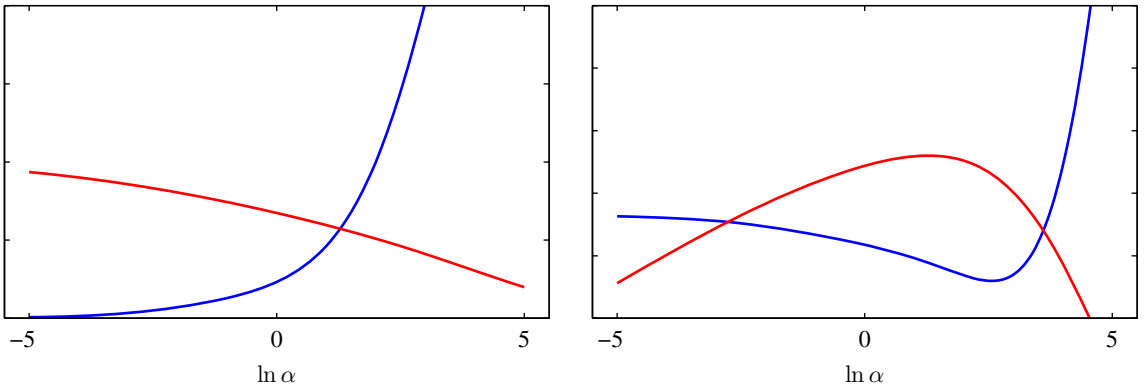


Figure 3.16 The left plot shows γ (red curve) and $2\alpha E_W(\mathbf{m}_N)$ (blue curve) versus $\ln\alpha$ for the sinusoidal synthetic data set. It is the intersection of these two curves that defines the optimum value for α given by the evidence procedure. The right plot shows the corresponding 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.

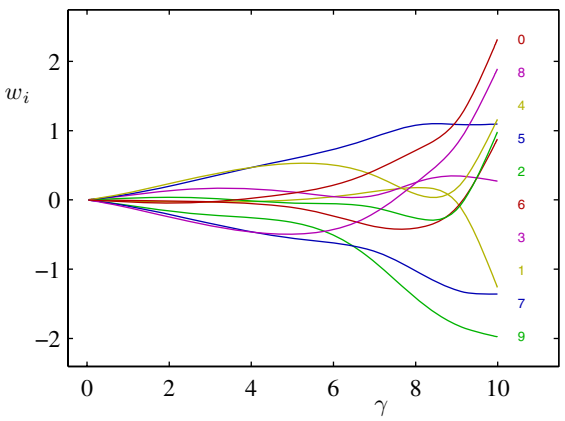


Figure 3.17 Plot of the 10 parameters w_i from the Gaussian basis function model versus the effective number of parameters γ, in which the hyperparameter α is varied in the range $0 \leqslant \alpha \leqslant \infty$ causing γ to vary in the range $0 \leqslant \gamma \leqslant M$.

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

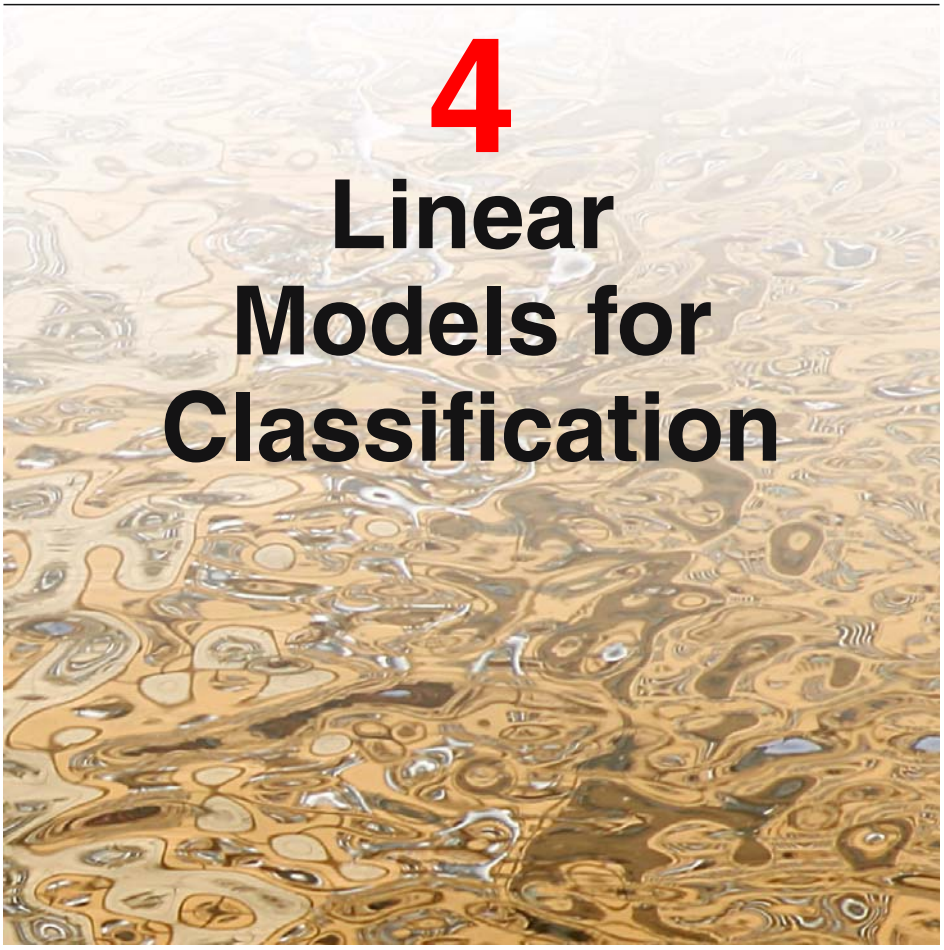
$$p(\mathbf{t}) = \frac{1}{(2\pi)^{N/2}}\frac{b_0^{a_0}}{b_N^{a_N}}\frac{\Gamma(a_N)}{\Gamma(a_0)}\frac{|\mathbf{S}_N|^{1/2}}{|\mathbf{S}_0|^{1/2}} \tag{3.118}$$

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

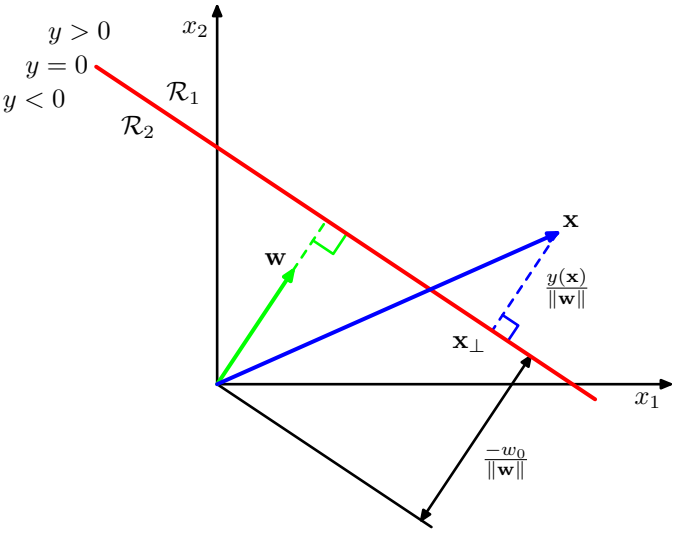


Figure 4.1 Illustration of the geometry of a linear discriminant function in two dimensions. The decision surface, shown in red, is perpendicular to $\mathbf{w}$, and its displacement from the origin is controlled by the bias parameter w_0. Also, the signed orthogonal distance of a general point $\mathbf{x}$ 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

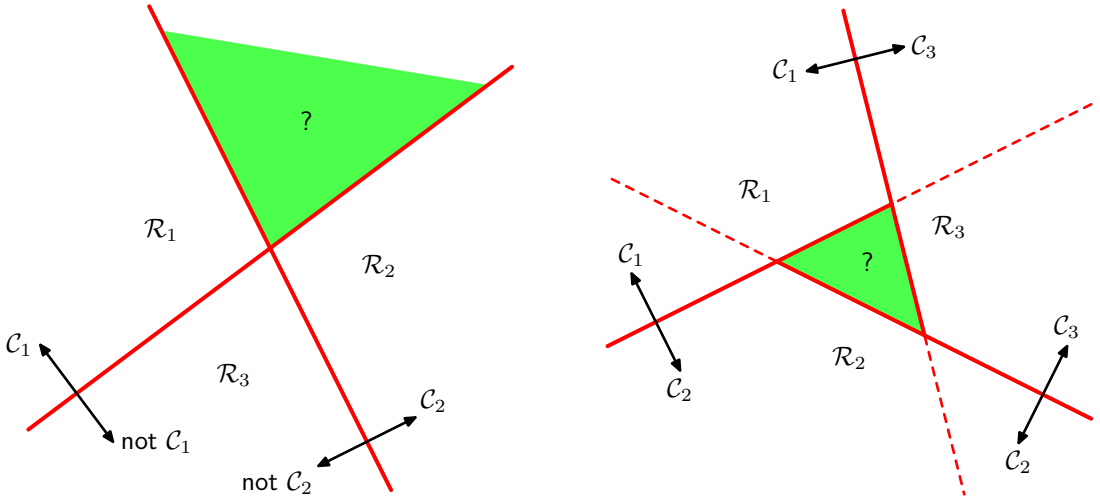


Figure 4.2 Attempting to construct a K class discriminant from a set of two class discriminants leads to ambiguous regions, shown in green. On the left is an example involving the use of two discriminants designed to distinguish points in class $\mathcal{C}_k$ from points not in class $\mathcal{C}_k$. 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

$$\widehat{\mathbf{x}} = \lambda\mathbf{x}_{\mathrm{A}} + (1-\lambda)\mathbf{x}_{\mathrm{B}} \tag{4.11}$$

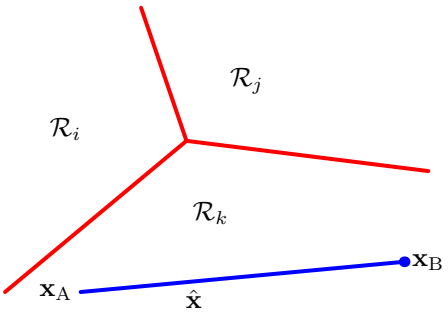


Figure 4.3 Illustration of the decision regions for a multiclass linear discriminant, with the decision boundaries shown in red. If two points $\mathbf{x}_\mathrm{A}$ and $\mathbf{x}_\mathrm{B}$ both lie inside the same decision region $\mathcal{R}_k$, then any point $\widehat{\mathbf{x}}$ that lies 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

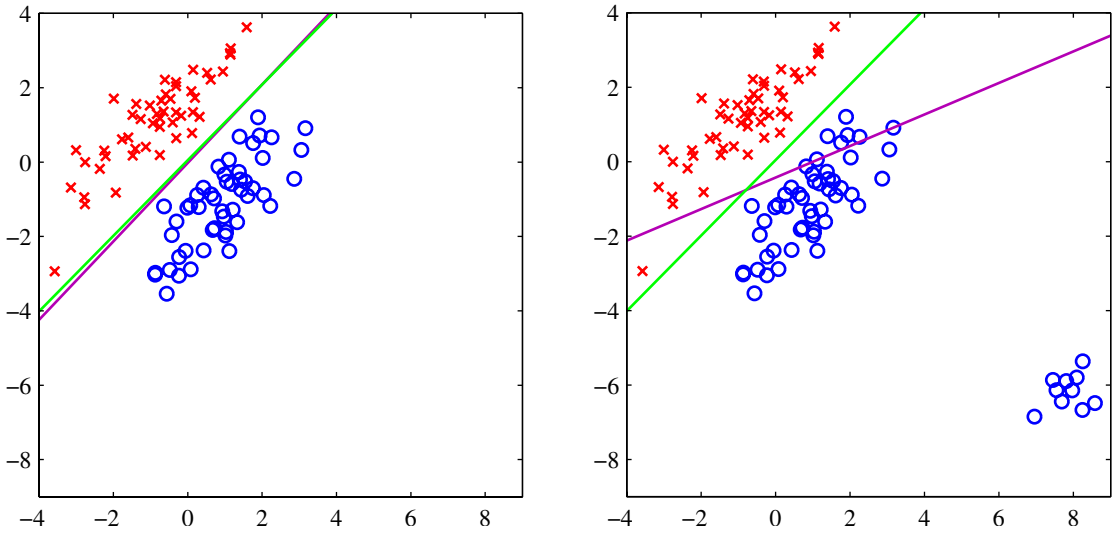


Figure 4.4 The left plot shows data from two classes, denoted by red crosses and blue circles, together with the decision boundary found by least squares (magenta curve) and also by the logistic regression model (green curve), which is discussed later in Section 4.3.2. The right-hand plot shows the corresponding 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-

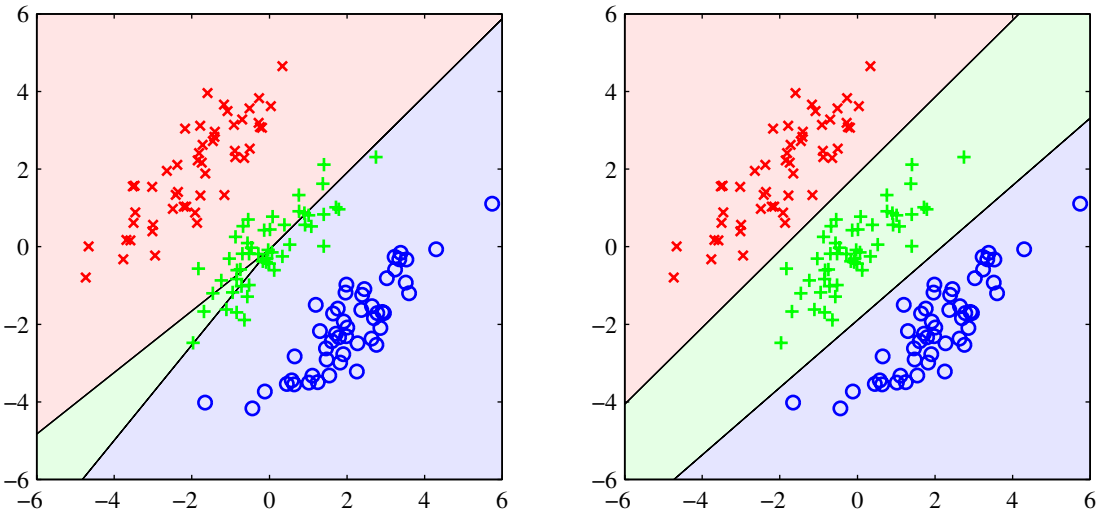


Figure 4.5 Example of a synthetic data set comprising three classes, with training data points denoted in red ($\times$), green ($+$), and blue ($\circ$). Lines denote the decision boundaries, and the background colours denote the respective classes of the decision regions. On the left is the result 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

$$m_k = \mathbf{w}^{\mathrm{T}}\mathbf{m}_k \tag{4.23}$$

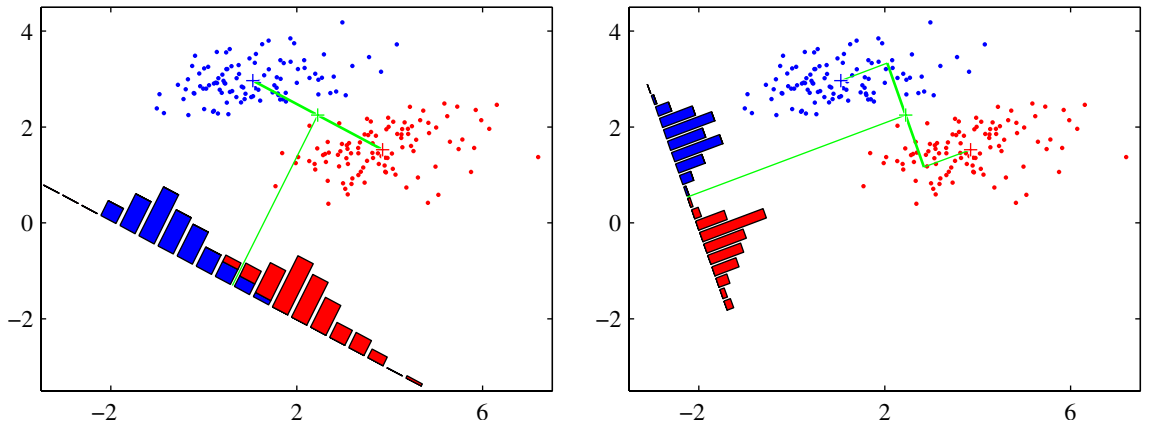


Figure 4.6 The left plot shows samples from two classes (depicted in red and blue) along with the histograms resulting from projection onto the line joining the class means. Note that there is considerable class overlap in the projected space. The right plot shows the corresponding projection based on the Fisher linear discriminant, showing the greatly improved class separation.

Appendix E
Exercise 4.4

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.

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.

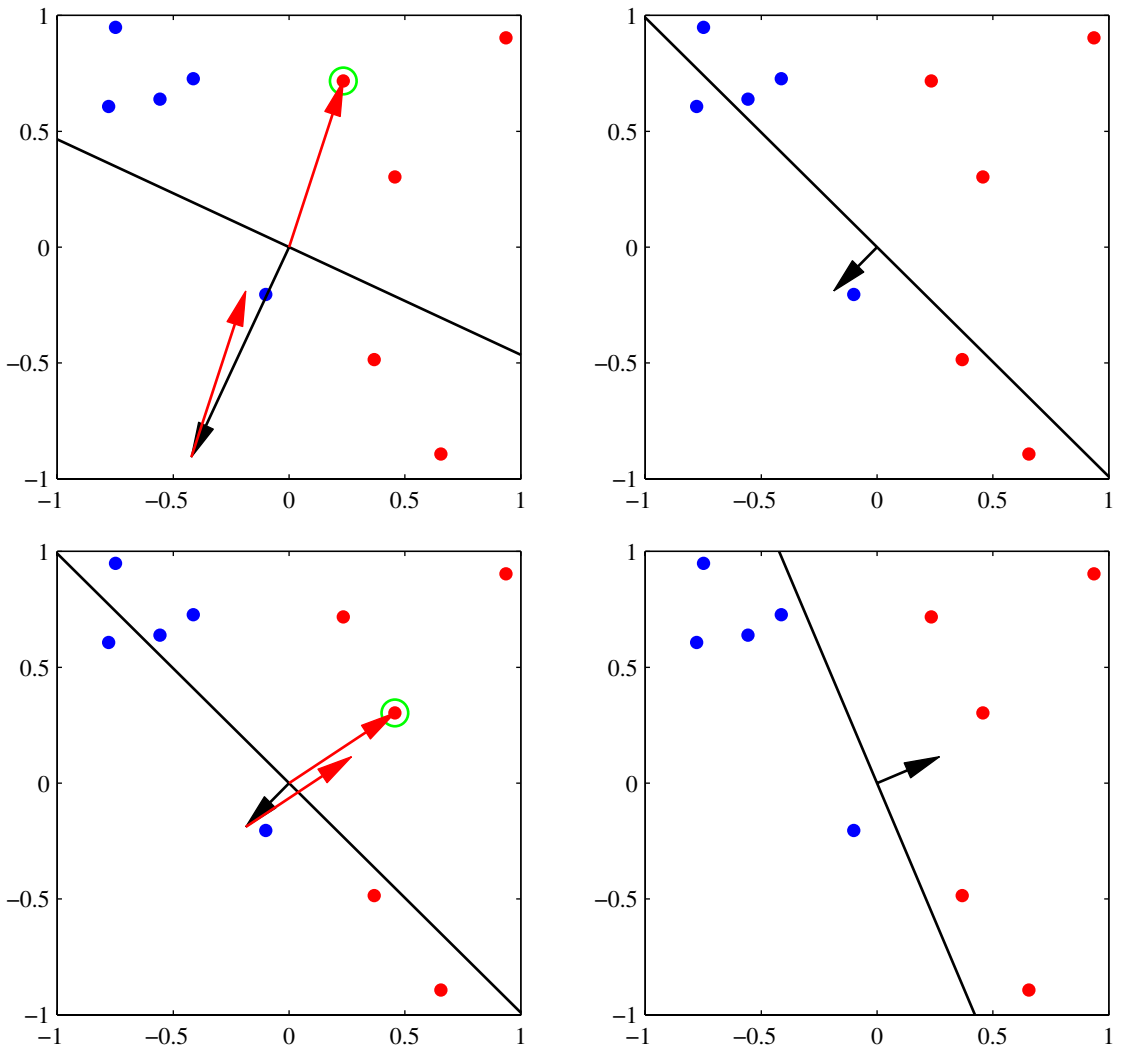


Figure 4.7 Illustration of the convergence of the perceptron learning algorithm, showing data points from two classes (red and blue) in a two-dimensional feature space (ϕ_1, ϕ_2). The top left plot shows the initial parameter vector $\mathbf{w}$ shown as a black arrow together with the corresponding 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

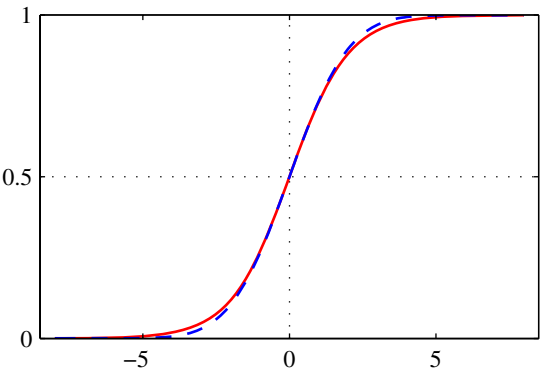


Figure 4.9 Plot of the logistic sigmoid function $\sigma(a)$ defined by (4.59), shown in red, together with the scaled probit function $\Phi(\lambda a)$, for $\lambda^2 = \pi/8$, shown in dashed blue, where $\Phi(a)$ is defined by (4.114). The scaling factor $\pi/8$ is chosen so that the derivatives 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

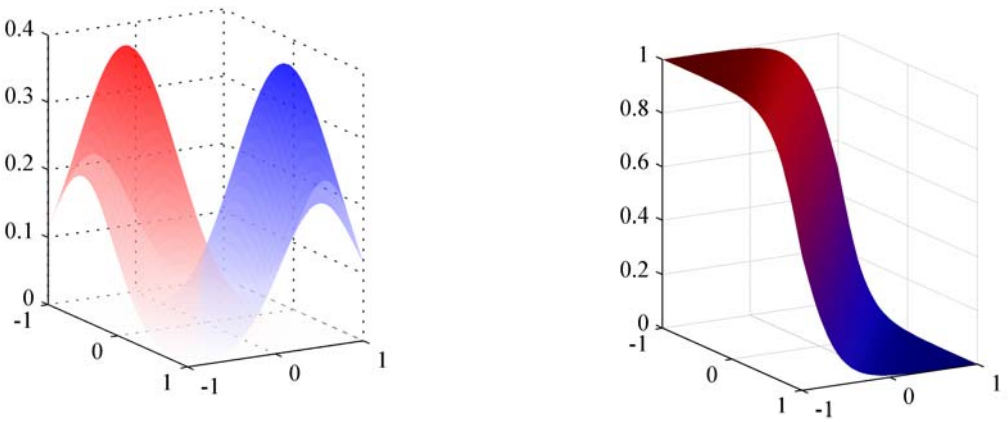


Figure 4.10 The left-hand plot shows the class-conditional densities for two classes, denoted red and blue. On the right is the corresponding posterior probability $p(\mathcal{C}_1|\mathbf{x})$, which is given by a logistic sigmoid of a linear function of $\mathbf{x}$. The surface in the right-hand plot 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.

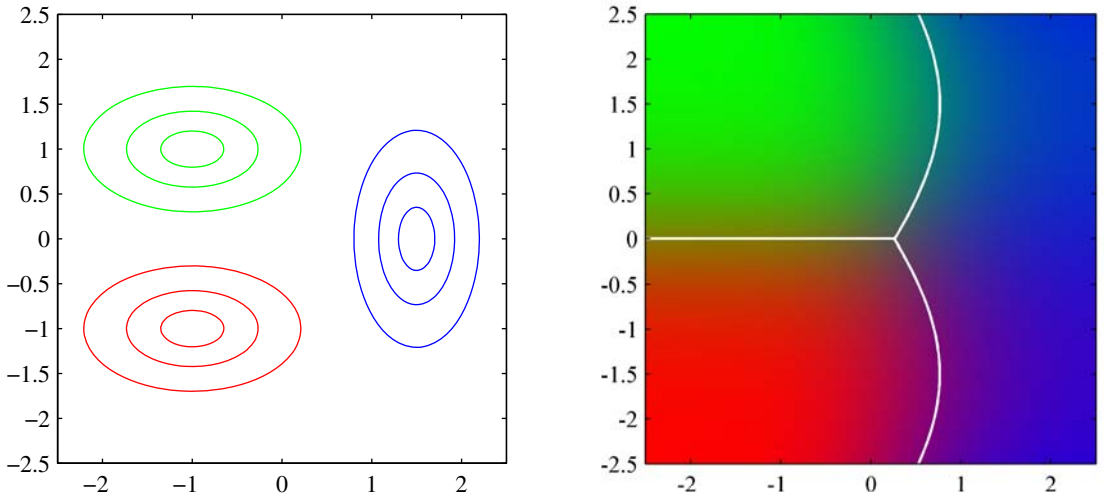


Figure 4.11 The left-hand plot shows the class-conditional densities for three classes each having a Gaussian distribution, coloured red, green, and blue, in which the red and green classes have the same covariance matrix. The right-hand plot shows the corresponding posterior probabilities, in which 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

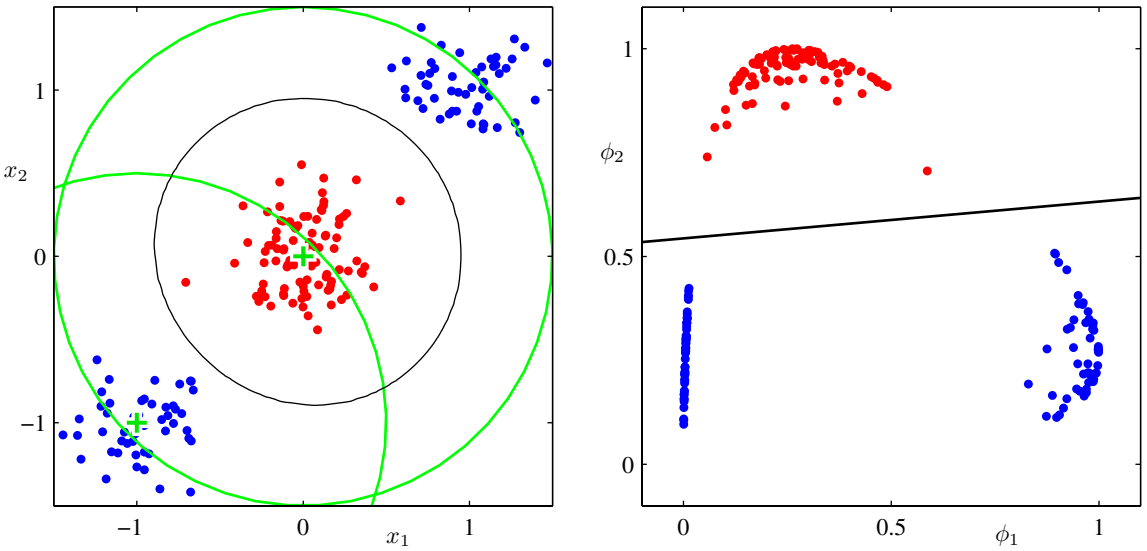


Figure 4.12 Illustration of the role of nonlinear basis functions in linear classification models. The left plot shows the original input space (x_1, x_2) together with data points from two classes labelled red and blue. Two 'Gaussian' basis functions $\phi_1(\mathbf{x})$ and $\phi_2(\mathbf{x})$ are 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

$$\begin{cases} t_n = 1 & \text{if } a_n \geqslant \theta \\ t_n = 0 & \text{otherwise.} \end{cases} \tag{4.112}$$

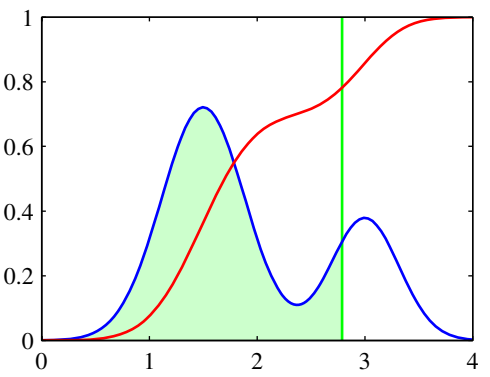


Figure 4.13 Schematic example of a probability density $p(\theta)$ shown by the blue curve, given in this example by a mixture of two Gaussians, along with its cumulative distribution function $f(a)$, shown by the red curve. Note that the value of the blue curve at any point, such as that indicated by 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.

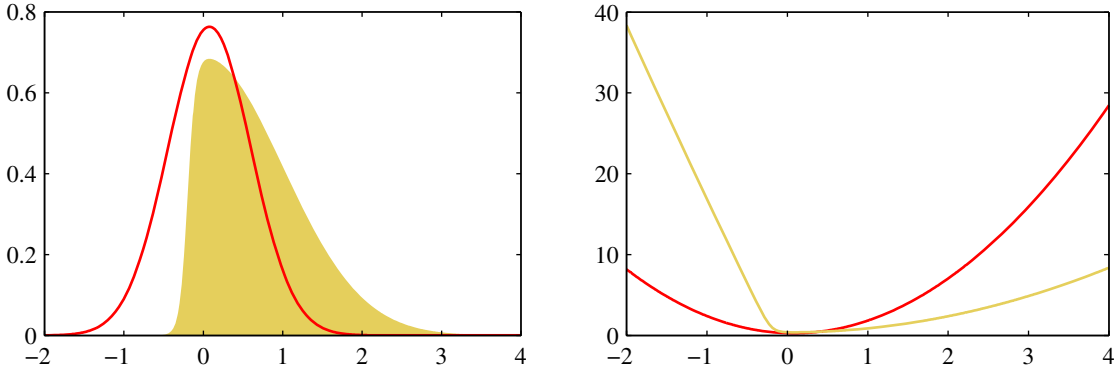


Figure 4.14 Illustration of the Laplace approximation applied to the distribution $p(z) \propto \exp(-z^2/2)\sigma(20z + 4)$ where $\sigma(z)$ is the logistic sigmoid function defined by $\sigma(z) = (1 + e^{-z})^{-1}$. The left plot shows the normalized distribution $p(z)$ in yellow, together with the 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.