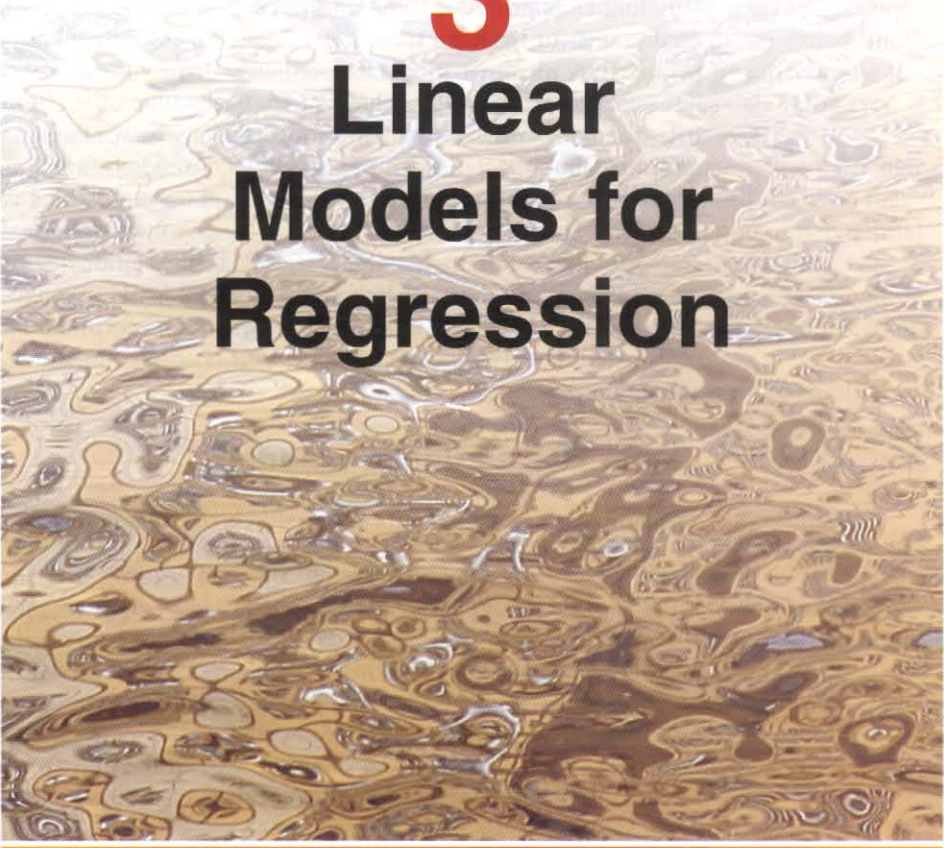

3

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 $n = 1, \dots, N$, 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

$$y(\mathbf{x}, \mathbf{w}) = w_0 + w_1x_1 + \dots + w_Dx_D \quad (3.1)$$

where $\mathbf{x} = (x_1, \dots, x_D)^T$. 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, \dots, 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

$$y(\mathbf{x}, \mathbf{w}) = w_0 + \sum_{j=1}^{M-1} w_j \phi_j(\mathbf{x}) \quad (3.2)$$

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

$$y(\mathbf{x}, \mathbf{w}) = \sum_{j=0}^{M-1} w_j \phi_j(\mathbf{x}) = \mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}) \quad (3.3)$$

where $\mathbf{w} = (w_0, \dots, w_{M-1})^T$ and $\boldsymbol{\phi} = (\phi_0, \dots, \phi_{M-1})^T$. 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

$$\phi_j(x) = \exp \left\{ -\frac{(x - \mu_j)^2}{2s^2} \right\} \quad (3.4)$$

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

$$\phi_j(x) = \sigma \left(\frac{x - \mu_j}{s} \right) \quad (3.5)$$

where $\sigma(a)$ is the logistic sigmoid function defined by

$$\sigma(a) = \frac{1}{1 + \exp(-a)}. \quad (3.6)$$

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

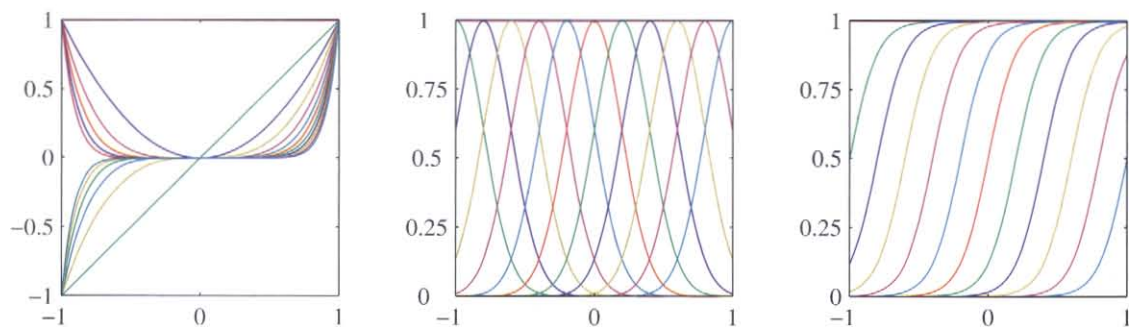


Figure 3.1 Examples of basis functions, showing polynomials on the left, Gaussians of the form (3.4) in the centre, and sigmoidal of the form (3.5) on the right.

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 $\phi(\mathbf{x})$ of basis functions is simply the identity $\phi(\mathbf{x}) = \mathbf{x}$. 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

$$t = y(\mathbf{x}, \mathbf{w}) + \epsilon \quad (3.7)$$

where ϵ is a zero mean Gaussian random variable with precision (inverse variance) β . Thus we can write

$$p(t|\mathbf{x}, \mathbf{w}, \beta) = \mathcal{N}(t|y(\mathbf{x}, \mathbf{w}), \beta^{-1}). \quad (3.8)$$

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

$$\mathbb{E}[t|\mathbf{x}] = \int t p(t|\mathbf{x}) dt = y(\mathbf{x}, \mathbf{w}). \quad (3.9)$$

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 $\mathbf{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$ with corresponding target values t_1, \dots, 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

$$p(\mathbf{t}|\mathbf{X}, \mathbf{w}, \beta) = \prod_{n=1}^N \mathcal{N}(t_n | \mathbf{w}^T \phi(\mathbf{x}_n), \beta^{-1}) \quad (3.10)$$

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

$$\begin{aligned} \ln p(\mathbf{t}|\mathbf{w}, \beta) &= \sum_{n=1}^N \ln \mathcal{N}(t_n | \mathbf{w}^T \phi(\mathbf{x}_n), \beta^{-1}) \\ &= \frac{N}{2} \ln \beta - \frac{N}{2} \ln(2\pi) - \beta E_D(\mathbf{w}) \end{aligned} \quad (3.11)$$

where the sum-of-squares error function is defined by

$$E_D(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^N \{t_n - \mathbf{w}^T \phi(\mathbf{x}_n)\}^2. \quad (3.12)$$

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

$$\nabla \ln p(\mathbf{t}|\mathbf{w}, \beta) = \sum_{n=1}^N \{t_n - \mathbf{w}^T \phi(\mathbf{x}_n)\} \phi(\mathbf{x}_n)^T. \quad (3.13)$$

Setting this gradient to zero gives

$$0 = \sum_{n=1}^N t_n \phi(\mathbf{x}_n)^T - \mathbf{w}^T \left(\sum_{n=1}^N \phi(\mathbf{x}_n) \phi(\mathbf{x}_n)^T \right). \quad (3.14)$$

Solving for \mathbf{w} we obtain

$$\mathbf{w}_{ML} = (\Phi^T \Phi)^{-1} \Phi^T \mathbf{t} \quad (3.15)$$

which are known as the *normal equations* for the least squares problem. Here Φ is an $N \times M$ matrix, called the *design matrix*, whose elements are given by $\Phi_{nj} = \phi_j(\mathbf{x}_n)$, so that

$$\Phi = \begin{pmatrix} \phi_0(\mathbf{x}_1) & \phi_1(\mathbf{x}_1) & \cdots & \phi_{M-1}(\mathbf{x}_1) \\ \phi_0(\mathbf{x}_2) & \phi_1(\mathbf{x}_2) & \cdots & \phi_{M-1}(\mathbf{x}_2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_0(\mathbf{x}_N) & \phi_1(\mathbf{x}_N) & \cdots & \phi_{M-1}(\mathbf{x}_N) \end{pmatrix}. \quad (3.16)$$

The quantity

$$\Phi^\dagger \equiv (\Phi^T \Phi)^{-1} \Phi^T \quad (3.17)$$

is known as the *Moore-Penrose pseudo-inverse* of the matrix Φ (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 Φ is square and invertible, then using the property $(\mathbf{A}\mathbf{B})^{-1} = \mathbf{B}^{-1}\mathbf{A}^{-1}$ we see that $\Phi^\dagger \equiv \Phi^{-1}$.

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

$$E_D(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^N \left\{ t_n - w_0 - \sum_{j=1}^{M-1} w_j \phi_j(\mathbf{x}_n) \right\}^2. \quad (3.18)$$

Setting the derivative with respect to w_0 equal to zero, and solving for w_0 , we obtain

$$w_0 = \bar{t} - \sum_{j=1}^{M-1} w_j \bar{\phi}_j \quad (3.19)$$

where we have defined

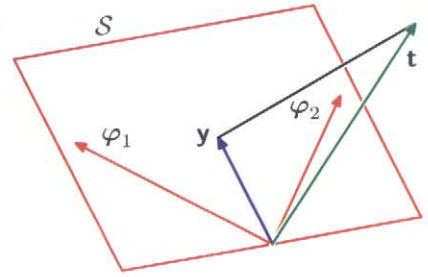
$$\bar{t} = \frac{1}{N} \sum_{n=1}^N t_n, \quad \bar{\phi}_j = \frac{1}{N} \sum_{n=1}^N \phi_j(\mathbf{x}_n). \quad (3.20)$$

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

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

Figure 3.2 Geometrical interpretation of the least-squares solution, in an N -dimensional space whose axes are the values of t_1, \dots, 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 φ_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 $\mathbf{t} = (t_1, \dots, t_N)^T$ 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 φ_j , as illustrated in Figure 3.2. Note that φ_j corresponds to the j^{th} column of Φ , whereas $\phi(\mathbf{x}_n)$ corresponds to the n^{th} row of Φ . 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 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 $n = 1, \dots, N$. Because \mathbf{y} is an arbitrary linear combination of the vectors φ_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 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 S . This is indeed the case, as can easily be verified by noting that the solution for \mathbf{y} is given by $\Phi \mathbf{w}_{\text{ML}}$, and then confirming that this takes the form of an orthogonal projection.

In practice, a direct solution of the normal equations can lead to numerical difficulties when $\Phi^T \Phi$ is close to singular. In particular, when two or more of the basis vectors φ_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

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} - \eta \nabla E_n \quad (3.22)$$

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

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} + \eta(t_n - \mathbf{w}^{(\tau)\top} \phi_n) \phi_n \quad (3.23)$$

where $\phi_n = \phi(\mathbf{x}_n)$. 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

$$E_D(\mathbf{w}) + \lambda E_W(\mathbf{w}) \quad (3.24)$$

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

$$E_W(\mathbf{w}) = \frac{1}{2} \mathbf{w}^\top \mathbf{w}. \quad (3.25)$$

If we also consider the sum-of-squares error function given by

$$E(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^N \{t_n - \mathbf{w}^\top \phi(\mathbf{x}_n)\}^2 \quad (3.26)$$

then the total error function becomes

$$\frac{1}{2} \sum_{n=1}^N \{t_n - \mathbf{w}^\top \phi(\mathbf{x}_n)\}^2 + \frac{\lambda}{2} \mathbf{w}^\top \mathbf{w}. \quad (3.27)$$

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

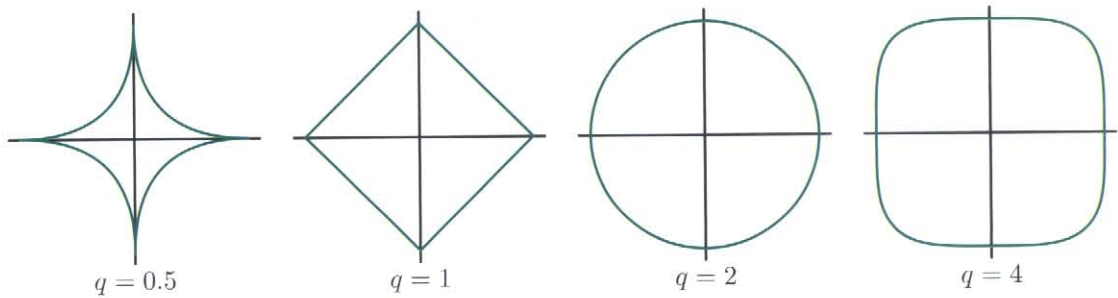


Figure 3.3 Contours of the regularization term in (3.29) for various values of the parameter q .

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

$$\mathbf{w} = (\lambda \mathbf{I} + \Phi^T \Phi)^{-1} \Phi^T \mathbf{t}. \quad (3.28)$$

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

$$\frac{1}{2} \sum_{n=1}^N \{t_n - \mathbf{w}^T \phi(\mathbf{x}_n)\}^2 + \frac{\lambda}{2} \sum_{j=1}^M |w_j|^q \quad (3.29)$$

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 known 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

$$\sum_{j=1}^M |w_j|^q \leq \eta \quad (3.30)$$

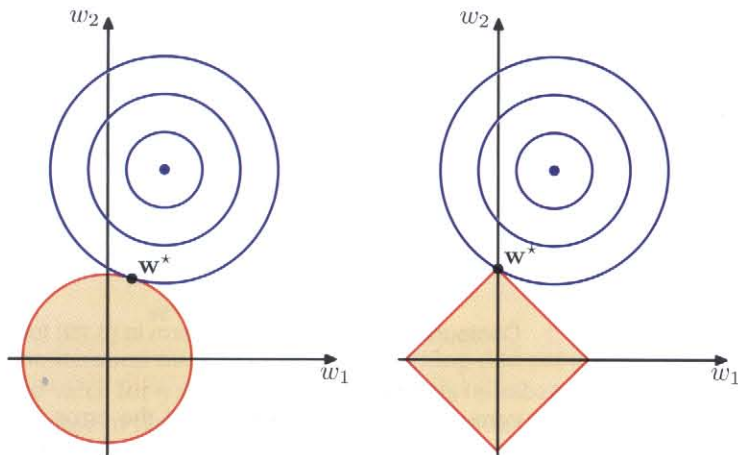
for an appropriate value of the parameter η , where the two approaches can be related using Lagrange multipliers. The origin of the sparsity can be seen from Figure 3.4, which shows that the minimum of the error function, subject to the constraint (3.30). As λ is increased, so an increasing number of parameters are driven to zero.

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.

Exercise 3.5

Appendix E

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}^* . The lasso gives a sparse solution in which $w_1^* = 0$.



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

$$\mathbf{y}(\mathbf{x}, \mathbf{w}) = \mathbf{W}^T \phi(\mathbf{x}) \quad (3.31)$$

where \mathbf{y} is a K -dimensional column vector, \mathbf{W} is an $M \times K$ matrix of parameters, and $\phi(\mathbf{x})$ 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

$$p(\mathbf{t}|\mathbf{x}, \mathbf{W}, \beta) = \mathcal{N}(\mathbf{t}|\mathbf{W}^T \phi(\mathbf{x}), \beta^{-1} \mathbf{I}). \quad (3.32)$$

If we have a set of observations $\mathbf{t}_1, \dots, \mathbf{t}_N$, 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^T . Similarly, we can combine the input vectors $\mathbf{x}_1, \dots, \mathbf{x}_N$ into a matrix \mathbf{X} . The log likelihood function is then given by

$$\begin{aligned} \ln p(\mathbf{T}|\mathbf{X}, \mathbf{W}, \beta) &= \sum_{n=1}^N \ln \mathcal{N}(\mathbf{t}_n | \mathbf{W}^T \phi(\mathbf{x}_n), \beta^{-1} \mathbf{I}) \\ &= \frac{NK}{2} \ln \left(\frac{\beta}{2\pi} \right) - \frac{\beta}{2} \sum_{n=1}^N \|\mathbf{t}_n - \mathbf{W}^T \phi(\mathbf{x}_n)\|^2. \end{aligned} \quad (3.33)$$

As before, we can maximize this function with respect to \mathbf{W} , giving

$$\mathbf{W}_{\text{ML}} = (\Phi^T \Phi)^{-1} \Phi^T \mathbf{T}. \quad (3.34)$$

If we examine this result for each target variable t_k , we have

$$\mathbf{w}_k = (\Phi^T \Phi)^{-1} \Phi^T \mathbf{t}_k = \Phi^\dagger \mathbf{t}_k \quad (3.35)$$

where \mathbf{t}_k is an N -dimensional column vector with components t_{nk} for $n = 1, \dots, N$. Thus the solution to the regression problem decouples between the different target variables, and we need only compute a single pseudo-inverse matrix Φ^\dagger , which is shared by all of the vectors \mathbf{w}_k .

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.

Exercise 3.6

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

$$h(\mathbf{x}) = \mathbb{E}[t|\mathbf{x}] = \int tp(t|\mathbf{x}) dt. \quad (3.36)$$

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

$$\mathbb{E}[L] = \int \{y(\mathbf{x}) - h(\mathbf{x})\}^2 p(\mathbf{x}) d\mathbf{x} + \int \{h(\mathbf{x}) - t\}^2 p(\mathbf{x}, t) d\mathbf{x} dt. \quad (3.37)$$

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

$$\{y(\mathbf{x}; \mathcal{D}) - h(\mathbf{x})\}^2. \quad (3.38)$$

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

$$\begin{aligned} & \{y(\mathbf{x}; \mathcal{D}) - \mathbb{E}_{\mathcal{D}}[y(\mathbf{x}; \mathcal{D})] + \mathbb{E}_{\mathcal{D}}[y(\mathbf{x}; \mathcal{D})] - h(\mathbf{x})\}^2 \\ &= \{y(\mathbf{x}; \mathcal{D}) - \mathbb{E}_{\mathcal{D}}[y(\mathbf{x}; \mathcal{D})]\}^2 + \{\mathbb{E}_{\mathcal{D}}[y(\mathbf{x}; \mathcal{D})] - h(\mathbf{x})\}^2 \\ & \quad + 2\{y(\mathbf{x}; \mathcal{D}) - \mathbb{E}_{\mathcal{D}}[y(\mathbf{x}; \mathcal{D})]\}\{\mathbb{E}_{\mathcal{D}}[y(\mathbf{x}; \mathcal{D})] - h(\mathbf{x})\}. \end{aligned} \quad (3.39)$$

We now take the expectation of this expression with respect to \mathcal{D} and note that the final term will vanish, giving

$$\begin{aligned} & \mathbb{E}_{\mathcal{D}} [\{y(\mathbf{x}; \mathcal{D}) - h(\mathbf{x})\}^2] \\ &= \underbrace{\{\mathbb{E}_{\mathcal{D}}[y(\mathbf{x}; \mathcal{D})] - h(\mathbf{x})\}^2}_{(\text{bias})^2} + \underbrace{\mathbb{E}_{\mathcal{D}} [\{y(\mathbf{x}; \mathcal{D}) - \mathbb{E}_{\mathcal{D}}[y(\mathbf{x}; \mathcal{D})]\}^2]}_{\text{variance}}. \end{aligned} \quad (3.40)$$

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

$$\text{expected loss} = (\text{bias})^2 + \text{variance} + \text{noise} \quad (3.41)$$

where

$$(\text{bias})^2 = \int \{\mathbb{E}_{\mathcal{D}}[y(\mathbf{x}; \mathcal{D})] - h(\mathbf{x})\}^2 p(\mathbf{x}) \, d\mathbf{x} \quad (3.42)$$

$$\text{variance} = \int \mathbb{E}_{\mathcal{D}} [\{y(\mathbf{x}; \mathcal{D}) - \mathbb{E}_{\mathcal{D}}[y(\mathbf{x}; \mathcal{D})]\}^2] p(\mathbf{x}) \, d\mathbf{x} \quad (3.43)$$

$$\text{noise} = \int \{h(\mathbf{x}) - t\}^2 p(\mathbf{x}, t) \, d\mathbf{x} \, dt \quad (3.44)$$

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 $l = 1, \dots, L$, where $L = 100$, and for each data set $\mathcal{D}^{(l)}$ we

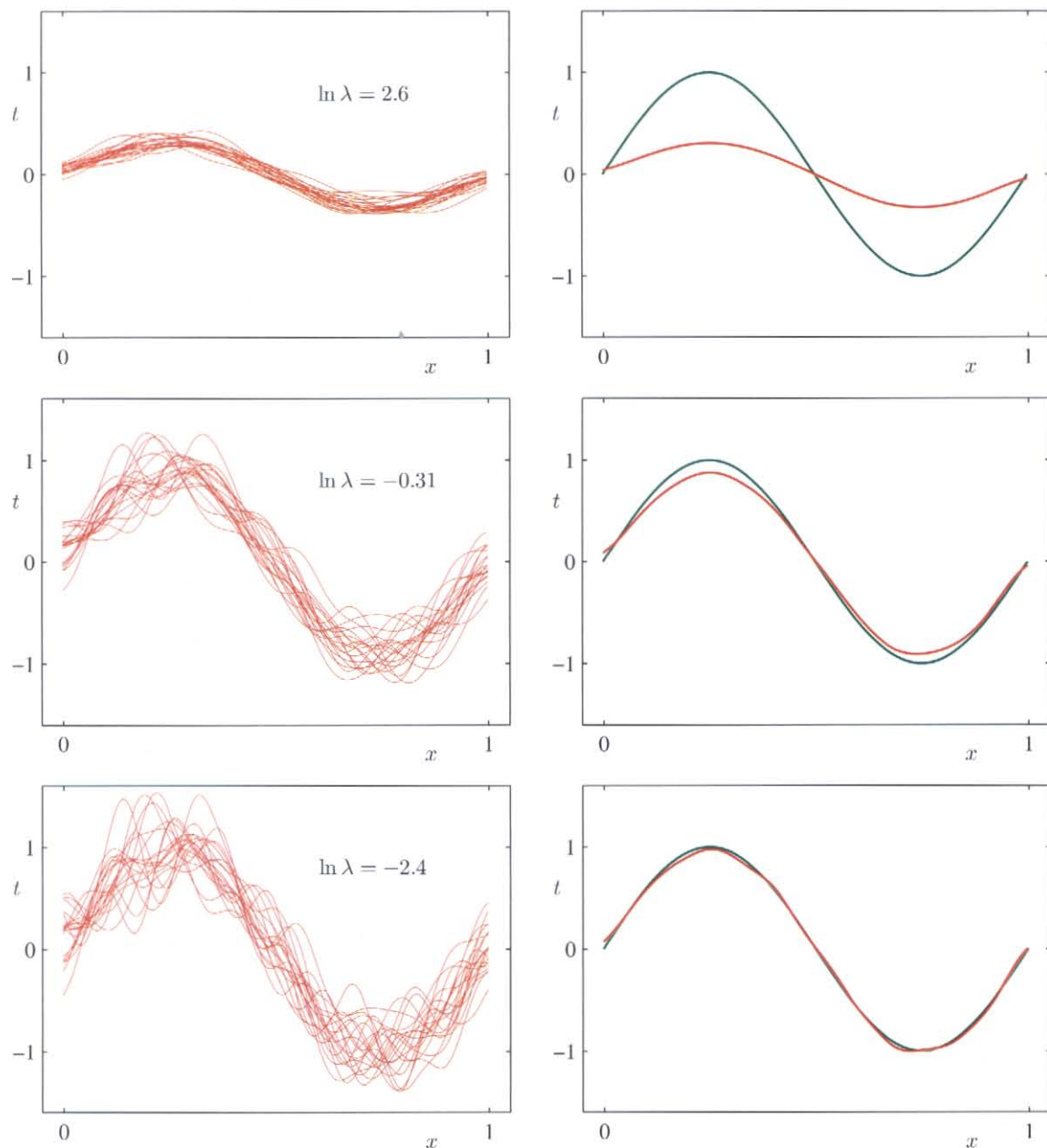
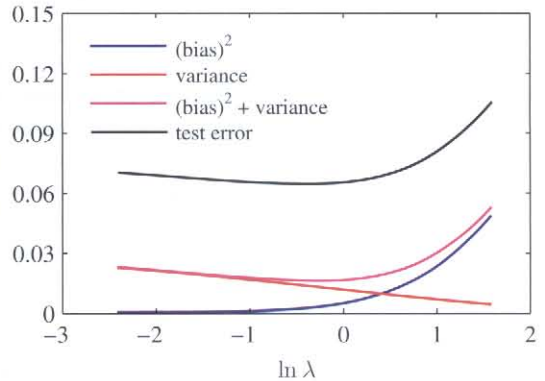


Figure 3.5 Illustration of the dependence of bias and variance on model complexity, governed by a regularization parameter λ , using the sinusoidal data set from Chapter 1. There are $L = 100$ data sets, each having $N = 25$ data points, and there are 24 Gaussian basis functions in the model so that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

$$\bar{y}(x) = \frac{1}{L} \sum_{l=1}^L y^{(l)}(x) \quad (3.45)$$

and the integrated squared bias and integrated variance are then given by

$$(\text{bias})^2 = \frac{1}{N} \sum_{n=1}^N \{\bar{y}(x_n) - h(x_n)\}^2 \quad (3.46)$$

$$\text{variance} = \frac{1}{N} \sum_{n=1}^N \frac{1}{L} \sum_{l=1}^L \{y^{(l)}(x_n) - \bar{y}(x_n)\}^2 \quad (3.47)$$

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

$$p(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\mathbf{m}_0, \mathbf{S}_0) \quad (3.48)$$

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

$$p(\mathbf{w}|\mathbf{t}) = \mathcal{N}(\mathbf{w}|\mathbf{m}_N, \mathbf{S}_N) \quad (3.49)$$

where

$$\mathbf{m}_N = \mathbf{S}_N (\mathbf{S}_0^{-1} \mathbf{m}_0 + \beta \Phi^T \mathbf{t}) \quad (3.50)$$

$$\mathbf{S}_N^{-1} = \mathbf{S}_0^{-1} + \beta \Phi^T \Phi. \quad (3.51)$$

Note that because the posterior distribution is Gaussian, its mode coincides with its mean. Thus the maximum posterior weight vector is simply given by $\mathbf{w}_{\text{MAP}} = \mathbf{m}_N$. If we consider an infinitely broad prior $\mathbf{S}_0 = \alpha^{-1} \mathbf{I}$ with $\alpha \rightarrow 0$, the mean \mathbf{m}_N of the posterior distribution reduces to the maximum likelihood value \mathbf{w}_{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).

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

$$p(\mathbf{w}|\alpha) = \mathcal{N}(\mathbf{w}|\mathbf{0}, \alpha^{-1} \mathbf{I}) \quad (3.52)$$

and the corresponding posterior distribution over \mathbf{w} is then given by (3.49) with

$$\mathbf{m}_N = \beta \mathbf{S}_N \Phi^T \mathbf{t} \quad (3.53)$$

$$\mathbf{S}_N^{-1} = \alpha \mathbf{I} + \beta \Phi^T \Phi. \quad (3.54)$$

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

$$\ln p(\mathbf{w}|\mathbf{t}) = -\frac{\beta}{2} \sum_{n=1}^N \{t_n - \mathbf{w}^T \phi(\mathbf{x}_n)\}^2 - \frac{\alpha}{2} \mathbf{w}^T \mathbf{w} + \text{const.} \quad (3.55)$$

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

Exercise 3.7

Exercise 3.8

a linear model of the form $y(x, \mathbf{w}) = w_0 + w_1x$. 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 $f(x, \mathbf{a}) = a_0 + a_1x$ with parameter values $a_0 = -0.3$ and $a_1 = 0.5$ by first choosing values of x_n from the uniform distribution $U(x|-1, 1)$, 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 20th 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

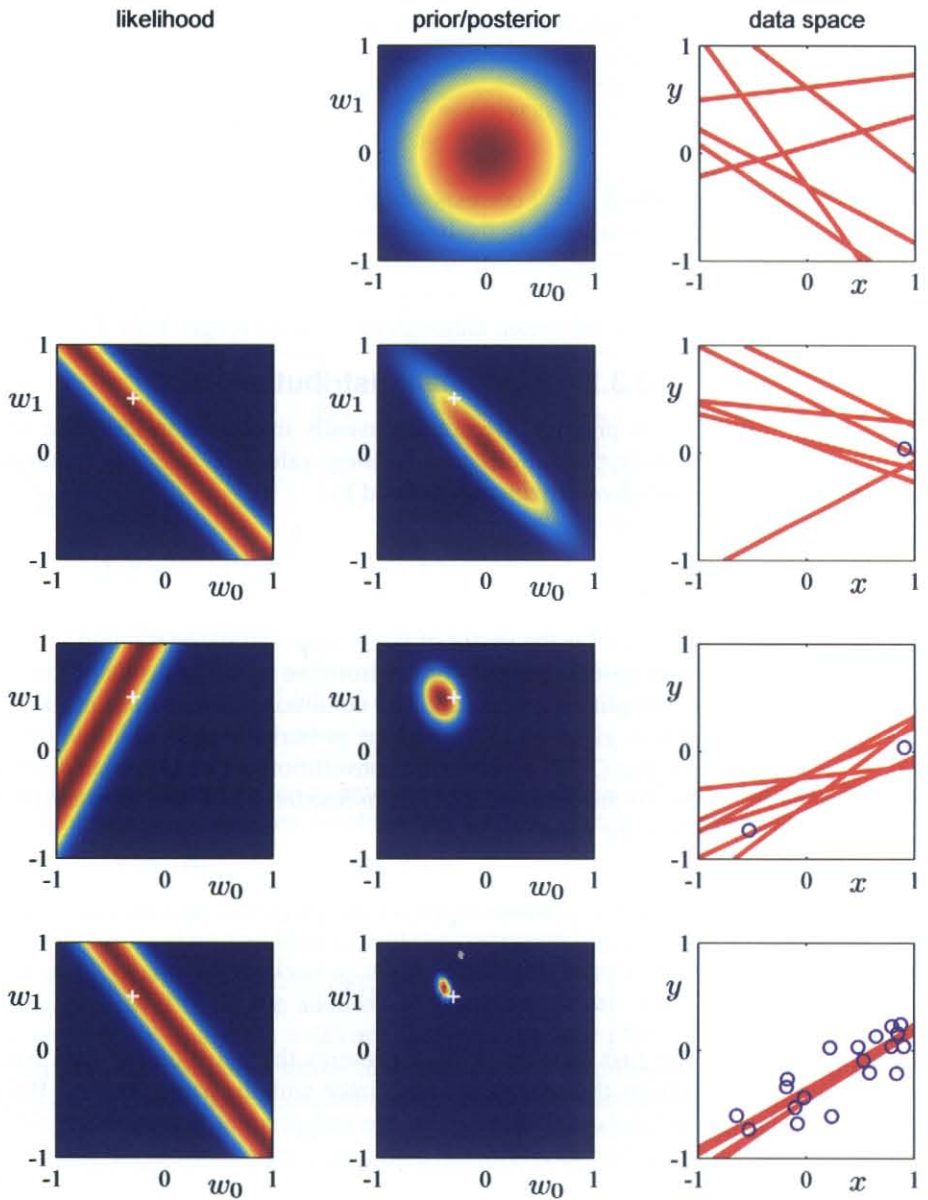


Figure 3.7 Illustration of sequential Bayesian learning for a simple linear model of the form $y(x, \mathbf{w}) = w_0 + w_1 x$. A detailed description of this figure is given in the text.

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

$$p(\mathbf{w}|\alpha) = \left[\frac{q}{2} \left(\frac{\alpha}{2} \right)^{1/q} \frac{1}{\Gamma(1/q)} \right]^M \exp \left(-\frac{\alpha}{2} \sum_{j=1}^M |w_j|^q \right) \quad (3.56)$$

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

$$p(t|\mathbf{t}, \alpha, \beta) = \int p(t|\mathbf{w}, \beta) p(\mathbf{w}|\mathbf{t}, \alpha, \beta) d\mathbf{w} \quad (3.57)$$

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

$$p(t|\mathbf{x}, \mathbf{t}, \alpha, \beta) = \mathcal{N}(t|\mathbf{m}_N^T \phi(\mathbf{x}), \sigma_N^2(\mathbf{x})) \quad (3.58)$$

where the variance $\sigma_N^2(\mathbf{x})$ of the predictive distribution is given by

$$\sigma_N^2(\mathbf{x}) = \frac{1}{\beta} + \phi(\mathbf{x})^T \mathbf{S}_N \phi(\mathbf{x}). \quad (3.59)$$

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 $\sigma_{N+1}^2(\mathbf{x}) \leq \sigma_N^2(\mathbf{x})$. In the limit $N \rightarrow \infty$, 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 .

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,

Exercise 3.10

Exercise 3.11

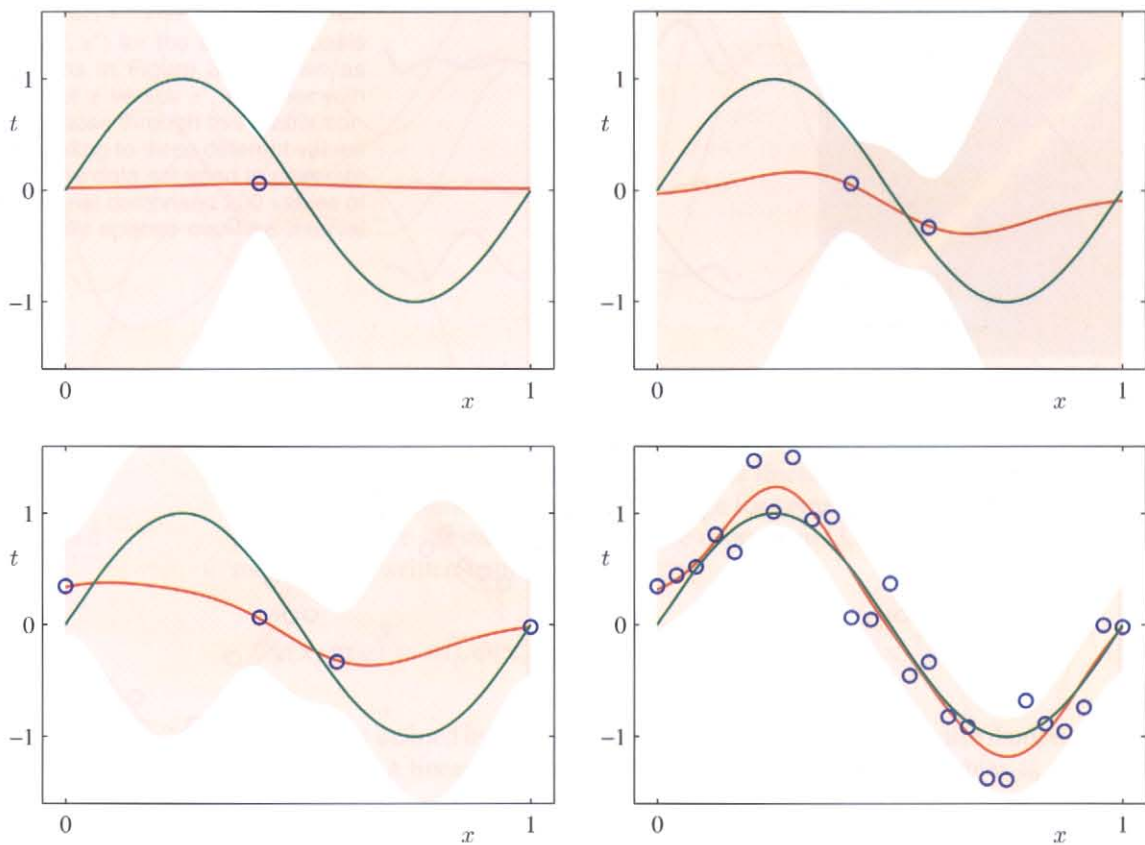


Figure 3.8 Examples of the predictive distribution (3.58) for a model consisting of 9 Gaussian basis functions of the form (3.4) using the synthetic sinusoidal data set of Section 1.1. See the text for a detailed discussion.

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.

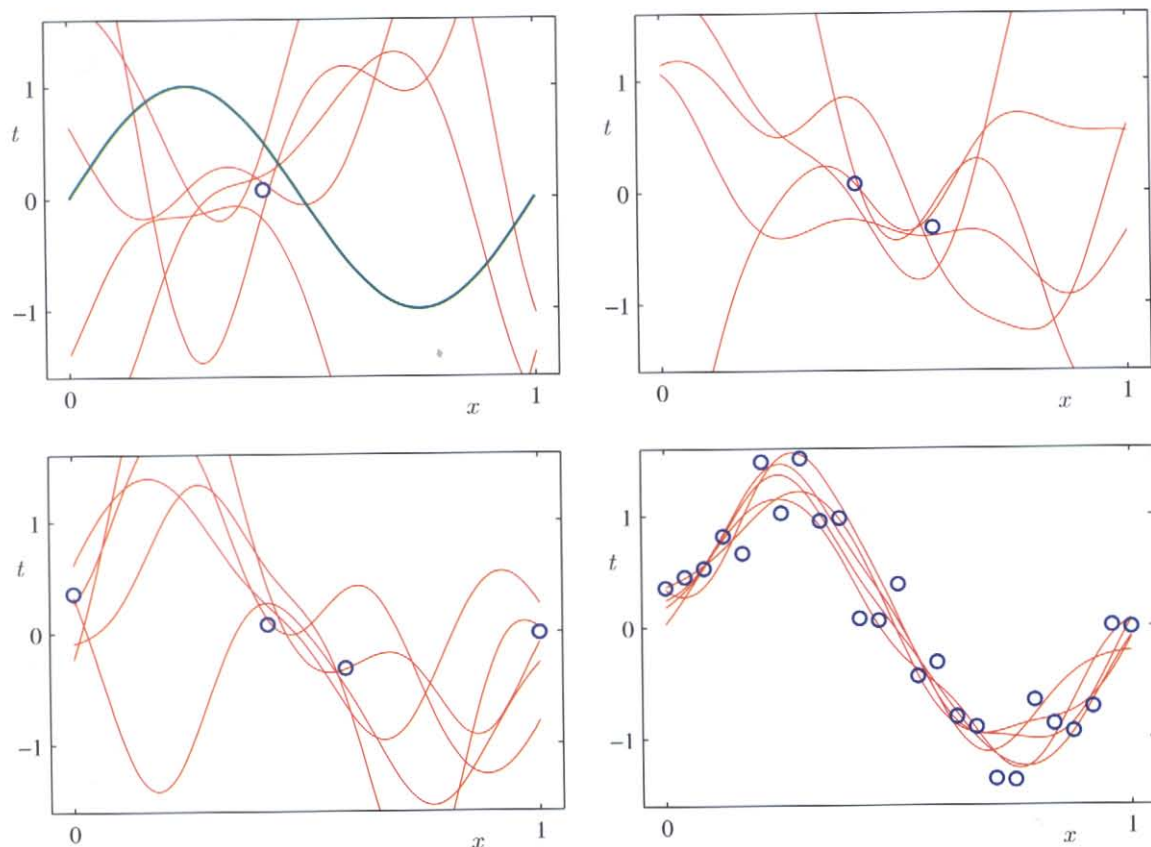


Figure 3.9 Plots of the function $y(x, \mathbf{w})$ using samples from the posterior distributions over \mathbf{w} corresponding to the plots in Figure 3.8.

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.

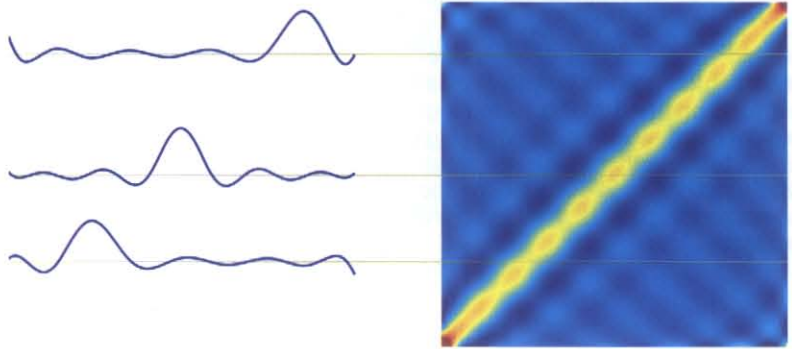
Note that, if both \mathbf{w} and β are treated as unknown, then we can introduce a conjugate prior distribution $p(\mathbf{w}, \beta)$ that, from the discussion in Section 2.3.6, will be given by a Gaussian-gamma distribution (Denison *et al.*, 2002). In this case, the predictive distribution is a Student's t -distribution.

Section 6.4

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

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

$$y(\mathbf{x}, \mathbf{m}_N) = \mathbf{m}_N^T \phi(\mathbf{x}) = \beta \phi(\mathbf{x})^T \mathbf{S}_N \Phi^T \mathbf{t} = \sum_{n=1}^N \beta \phi(\mathbf{x})^T \mathbf{S}_N \phi(\mathbf{x}_n) t_n \quad (3.60)$$

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

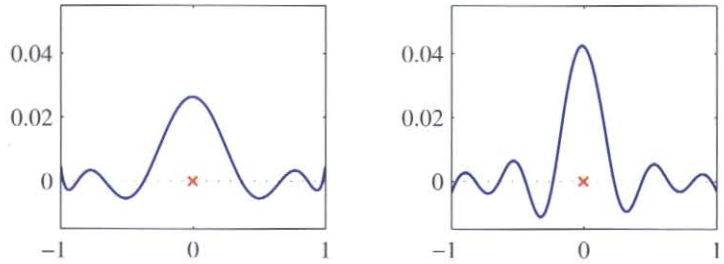
$$y(\mathbf{x}, \mathbf{m}_N) = \sum_{n=1}^N k(\mathbf{x}, \mathbf{x}_n) t_n \quad (3.61)$$

where the function

$$k(\mathbf{x}, \mathbf{x}') = \beta \phi(\mathbf{x})^T \mathbf{S}_N \phi(\mathbf{x}') \quad (3.62)$$

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

$$\begin{aligned} \text{cov}[y(\mathbf{x}), y(\mathbf{x}')] &= \text{cov}[\phi(\mathbf{x})^T \mathbf{w}, \mathbf{w}^T \phi(\mathbf{x}')] \\ &= \phi(\mathbf{x})^T \mathbf{S}_N \phi(\mathbf{x}') = \beta^{-1} k(\mathbf{x}, \mathbf{x}') \end{aligned} \quad (3.63)$$

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 point-wise 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

$$\sum_{n=1}^N k(\mathbf{x}, \mathbf{x}_n) = 1 \quad (3.64)$$

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 $\hat{y}(\mathbf{x})$ 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 $\hat{y}(\mathbf{x}) = 1$, 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 $\psi(\mathbf{x})$ of nonlinear functions, so that

$$k(\mathbf{x}, \mathbf{z}) = \psi(\mathbf{x})^T \psi(\mathbf{z}) \quad (3.65)$$

where $\psi(\mathbf{x}) = \beta^{1/2} \mathbf{S}_N^{1/2} \phi(\mathbf{x})$.

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 $i = 1, \dots, L$. 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

$$p(\mathcal{M}_i|\mathcal{D}) \propto p(\mathcal{M}_i)p(\mathcal{D}|\mathcal{M}_i). \quad (3.66)$$

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