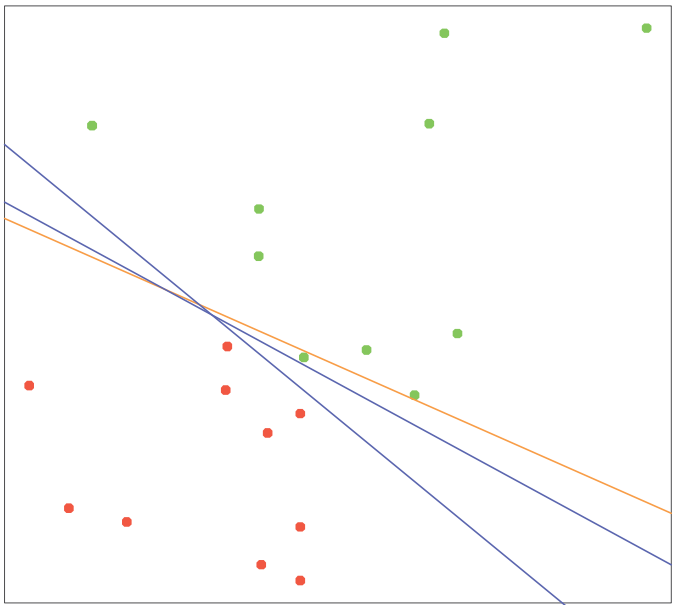

FIGURE 4.14. *A toy example with two classes separable by a hyperplane. The orange line is the least squares solution, which misclassifies one of the training points. Also shown are two blue separating hyperplanes found by the* perceptron learning algorithm *with different random starts.*

4.5 Separating Hyperplanes

We have seen that linear discriminant analysis and logistic regression both estimate linear decision boundaries in similar but slightly different ways. For the rest of this chapter we describe separating hyperplane classifiers. These procedures construct linear decision boundaries that explicitly try to separate the data into different classes as well as possible. They provide the basis for support vector classifiers, discussed in Chapter 12. The mathematical level of this section is somewhat higher than that of the previous sections.

Figure 4.14 shows 20 data points in two classes in $\mathbb{R}^2$. These data can be separated by a linear boundary. Included in the figure (blue lines) are two of the infinitely many possible *separating hyperplanes*. The orange line is the least squares solution to the problem, obtained by regressing the $-1/1$ response Y on X (with intercept); the line is given by

$$\{x : \hat{\beta}_0 + \hat{\beta}_1 x_1 + \hat{\beta}_2 x_2 = 0\}. \tag{4.39}$$

This least squares solution does not do a perfect job in separating the points, and makes one error. This is the same boundary found by LDA, in light of its equivalence with linear regression in the two-class case (Section 4.3 and Exercise 4.2).

Classifiers such as (4.39), that compute a linear combination of the input features and return the sign, were called *perceptrons* in the engineering liter-

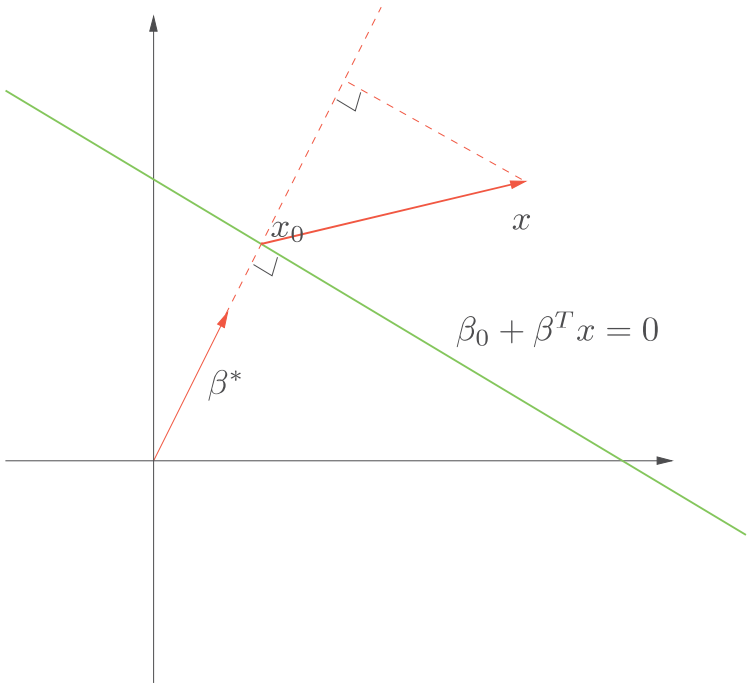

FIGURE 4.15. *The linear algebra of a hyperplane (affine set).*

ature in the late 1950s (Rosenblatt, 1958). Perceptrons set the foundations for the neural network models of the 1980s and 1990s.

Before we continue, let us digress slightly and review some vector algebra. Figure 4.15 depicts a hyperplane or *affine set* L defined by the equation $f(x) = \beta_0 + \beta^T x = 0$; since we are in $\mathbb{R}^2$ this is a line.

Here we list some properties:

1. For any two points x_1 and x_2 lying in L, $\beta^T(x_1 - x_2) = 0$, and hence $\beta^* = \beta/||\beta||$ is the vector normal to the surface of L.

2. For any point x_0 in L, $\beta^T x_0 = -\beta_0$.

3. The signed distance of any point x to L is given by

$$\begin{aligned} \beta^{*T}(x - x_0) &= \frac{1}{||\beta||}(\beta^T x + \beta_0) \\ &= \frac{1}{||f'(x)||} f(x). \end{aligned} \tag{4.40}$$

Hence $f(x)$ is proportional to the signed distance from x to the hyperplane defined by $f(x) = 0$.

4.5.1 Rosenblatt's Perceptron Learning Algorithm

The *perceptron learning algorithm* tries to find a separating hyperplane by minimizing the distance of misclassified points to the decision boundary. If

a response $y_i = 1$ is misclassified, then $x_i^T\beta + \beta_0 < 0$, and the opposite for a misclassified response with $y_i = -1$. The goal is to minimize

$$D(\beta, \beta_0) = -\sum_{i\in\mathcal{M}} y_i(x_i^T\beta + \beta_0), \tag{4.41}$$

where $\mathcal{M}$ indexes the set of misclassified points. The quantity is non-negative and proportional to the distance of the misclassified points to the decision boundary defined by $\beta^T x + \beta_0 = 0$. The gradient (assuming $\mathcal{M}$ is fixed) is given by

$$\partial\frac{D(\beta, \beta_0)}{\partial\beta} = -\sum_{i\in\mathcal{M}} y_i x_i, \tag{4.42}$$

$$\partial\frac{D(\beta, \beta_0)}{\partial\beta_0} = -\sum_{i\in\mathcal{M}} y_i. \tag{4.43}$$

The algorithm in fact uses *stochastic gradient descent* to minimize this piecewise linear criterion. This means that rather than computing the sum of the gradient contributions of each observation followed by a step in the negative gradient direction, a step is taken after each observation is visited. Hence the misclassified observations are visited in some sequence, and the parameters β are updated via

$$\begin{pmatrix}\beta\\ \beta_0\end{pmatrix} \leftarrow \begin{pmatrix}\beta\\ \beta_0\end{pmatrix} + \rho\begin{pmatrix}y_i x_i\\ y_i\end{pmatrix}. \tag{4.44}$$

Here ρ is the learning rate, which in this case can be taken to be 1 without loss in generality. If the classes are linearly separable, it can be shown that the algorithm converges to a separating hyperplane in a finite number of steps (Exercise 4.6). Figure 4.14 shows two solutions to a toy problem, each started at a different random guess.

There are a number of problems with this algorithm, summarized in Ripley (1996):

- When the data are separable, there are many solutions, and which one is found depends on the starting values.
- The "finite" number of steps can be very large. The smaller the gap, the longer the time to find it.
- When the data are not separable, the algorithm will not converge, and cycles develop. The cycles can be long and therefore hard to detect.

The second problem can often be eliminated by seeking a hyperplane not in the original space, but in a much enlarged space obtained by creating

many basis-function transformations of the original variables. This is analogous to driving the residuals in a polynomial regression problem down to zero by making the degree sufficiently large. Perfect separation cannot always be achieved: for example, if observations from two different classes share the same input. It may not be desirable either, since the resulting model is likely to be overfit and will not generalize well. We return to this point at the end of the next section.

A rather elegant solution to the first problem is to add additional constraints to the separating hyperplane.

4.5.2 Optimal Separating Hyperplanes

The *optimal separating hyperplane* separates the two classes and maximizes the distance to the closest point from either class (Vapnik, 1996). Not only does this provide a unique solution to the separating hyperplane problem, but by maximizing the margin between the two classes on the training data, this leads to better classification performance on test data.

We need to generalize criterion (4.41). Consider the optimization problem

$$\begin{aligned} &\max_{\beta,\beta_0,||\beta||=1} M \\ &\text{subject to } y_i(x_i^T\beta+\beta_0) \geq M,\ i=1,\ldots,N. \end{aligned} \tag{4.45}$$

The set of conditions ensure that all the points are at least a signed distance M from the decision boundary defined by β and β_0, and we seek the largest such M and associated parameters. We can get rid of the $||\beta|| = 1$ constraint by replacing the conditions with

$$\frac{1}{||\beta||}y_i(x_i^T\beta+\beta_0) \geq M, \tag{4.46}$$

(which redefines β_0) or equivalently

$$y_i(x_i^T\beta+\beta_0) \geq M||\beta||. \tag{4.47}$$

Since for any β and β_0 satisfying these inequalities, any positively scaled multiple satisfies them too, we can arbitrarily set $||\beta|| = 1/M$. Thus (4.45) is equivalent to

$$\begin{aligned} &\min_{\beta,\beta_0} \frac{1}{2}||\beta||^2 \\ &\text{subject to } y_i(x_i^T\beta+\beta_0) \geq 1,\ i=1,\ldots,N. \end{aligned} \tag{4.48}$$

In light of (4.40), the constraints define an empty slab or margin around the linear decision boundary of thickness $1/||\beta||$. Hence we choose β and β_0 to maximize its thickness. This is a convex optimization problem (quadratic

criterion with linear inequality constraints). The Lagrange (primal) function, to be minimized w.r.t. β and β_0, is

$$L_P = \frac{1}{2}||\beta||^2 - \sum_{i=1}^{N} \alpha_i[y_i(x_i^T\beta + \beta_0) - 1]. \tag{4.49}$$

Setting the derivatives to zero, we obtain:

$$\beta = \sum_{i=1}^{N} \alpha_i y_i x_i, \tag{4.50}$$

$$0 = \sum_{i=1}^{N} \alpha_i y_i, \tag{4.51}$$

and substituting these in (4.49) we obtain the so-called Wolfe dual

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{k=1}^{N} \alpha_i\alpha_k y_i y_k x_i^T x_k$$
$$\text{subject to } \alpha_i \geq 0 \text{ and } \sum_{i=1}^{N} \alpha_i y_i = 0. \tag{4.52}$$

The solution is obtained by maximizing L_D in the positive orthant, a simpler convex optimization problem, for which standard software can be used. In addition the solution must satisfy the Karush–Kuhn–Tucker conditions, which include (4.50), (4.51), (4.52) and

$$\alpha_i[y_i(x_i^T\beta + \beta_0) - 1] = 0 \; \forall i. \tag{4.53}$$

From these we can see that

- if $\alpha_i > 0$, then $y_i(x_i^T\beta + \beta_0) = 1$, or in other words, x_i is on the boundary of the slab;
- if $y_i(x_i^T\beta + \beta_0) > 1$, x_i is not on the boundary of the slab, and $\alpha_i = 0$.

From (4.50) we see that the solution vector β is defined in terms of a linear combination of the *support points* x_i—those points defined to be on the boundary of the slab via $\alpha_i > 0$. Figure 4.16 shows the optimal separating hyperplane for our toy example; there are three support points. Likewise, β_0 is obtained by solving (4.53) for any of the support points.

The optimal separating hyperplane produces a function $\hat{f}(x) = x^T\hat{\beta} + \hat{\beta}_0$ for classifying new observations:

$$\hat{G}(x) = \text{sign}\hat{f}(x). \tag{4.54}$$

Although none of the training observations fall in the margin (by construction), this will not necessarily be the case for test observations. The

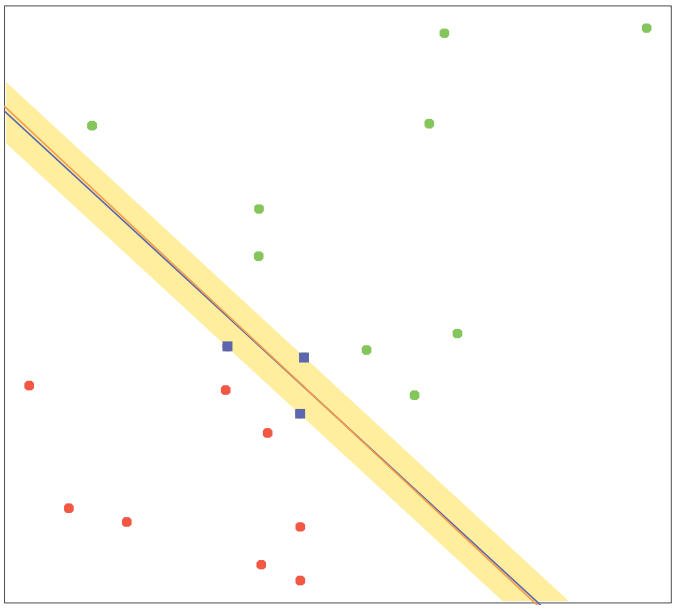

FIGURE 4.16. *The same data as in Figure 4.14. The shaded region delineates the maximum margin separating the two classes. There are three support points indicated, which lie on the boundary of the margin, and the optimal separating hyperplane (blue line) bisects the slab. Included in the figure is the boundary found using logistic regression (red line), which is very close to the optimal separating hyperplane (see Section 12.3.3).*

intuition is that a large margin on the training data will lead to good separation on the test data.

The description of the solution in terms of support points seems to suggest that the optimal hyperplane focuses more on the points that count, and is more robust to model misspecification. The LDA solution, on the other hand, depends on all of the data, even points far away from the decision boundary. Note, however, that the identification of these support points required the use of all the data. Of course, if the classes are really Gaussian, then LDA is optimal, and separating hyperplanes will pay a price for focusing on the (noisier) data at the boundaries of the classes.

Included in Figure 4.16 is the logistic regression solution to this problem, fit by maximum likelihood. Both solutions are similar in this case. When a separating hyperplane exists, logistic regression will always find it, since the log-likelihood can be driven to 0 in this case (Exercise 4.5). The logistic regression solution shares some other qualitative features with the separating hyperplane solution. The coefficient vector is defined by a weighted least squares fit of a zero-mean linearized response on the input features, and the weights are larger for points near the decision boundary than for those further away.

When the data are not separable, there will be no feasible solution to this problem, and an alternative formulation is needed. Again one can enlarge the space using basis transformations, but this can lead to artificial

separation through over-fitting. In Chapter 12 we discuss a more attractive alternative known as the *support vector machine*, which allows for overlap, but minimizes a measure of the extent of this overlap.

Bibliographic Notes

Good general texts on classification include Duda et al. (2000), Hand (1981), McLachlan (1992) and Ripley (1996). Mardia et al. (1979) have a concise discussion of linear discriminant analysis. Michie et al. (1994) compare a large number of popular classifiers on benchmark datasets. Linear separating hyperplanes are discussed in Vapnik (1996). Our account of the perceptron learning algorithm follows Ripley (1996).

Exercises

Ex. 4.1 Show how to solve the generalized eigenvalue problem $\max a^T \mathbf{B} a$ subject to $a^T \mathbf{W} a = 1$ by transforming to a standard eigenvalue problem.

Ex. 4.2 Suppose we have features $x \in \mathbb{R}^p$, a two-class response, with class sizes N_1, N_2, and the target coded as $-N/N_1, N/N_2$.

(a) Show that the LDA rule classifies to class 2 if

$$x^T \hat{\mathbf{\Sigma}}^{-1}(\hat{\mu}_2 - \hat{\mu}_1) > \frac{1}{2}(\hat{\mu}_2 + \hat{\mu}_1)^T \hat{\mathbf{\Sigma}}^{-1}(\hat{\mu}_2 - \hat{\mu}_1) - \log(N_2/N_1),$$

and class 1 otherwise.

(b) Consider minimization of the least squares criterion

$$\sum_{i=1}^{N}(y_i - \beta_0 - x_i^T \beta)^2. \tag{4.55}$$

Show that the solution $\hat{\beta}$ satisfies

$$\left[(N-2)\hat{\mathbf{\Sigma}} + N\hat{\mathbf{\Sigma}}_B\right]\beta = N(\hat{\mu}_2 - \hat{\mu}_1) \tag{4.56}$$

(after simplification), where $\hat{\mathbf{\Sigma}}_B = \frac{N_1 N_2}{N^2}(\hat{\mu}_2 - \hat{\mu}_1)(\hat{\mu}_2 - \hat{\mu}_1)^T$.

(c) Hence show that $\hat{\mathbf{\Sigma}}_B \beta$ is in the direction $(\hat{\mu}_2 - \hat{\mu}_1)$ and thus

$$\hat{\beta} \propto \hat{\mathbf{\Sigma}}^{-1}(\hat{\mu}_2 - \hat{\mu}_1). \tag{4.57}$$

Therefore the least-squares regression coefficient is identical to the LDA coefficient, up to a scalar multiple.