CHAPTER 3

LINEAR CLASSIFIERS

3.1 INTRODUCTION

Our major concern in Chapter 2 was to design classifiers based on probability density or probability functions. In some cases, we saw that the resulting classifiers were equivalent to a set of linear discriminant functions. In this chapter we will focus on the design of linear classifiers, *irrespective of the underlying distributions describing the training data*. The major advantage of linear classifiers is their simplicity and computational attractiveness. The chapter starts with the assumption that *all* feature vectors from the available classes can be classified correctly using a linear classifier, and we will develop techniques for the computation of the corresponding linear functions. In the sequel we will focus on a more general problem, in which a linear classifier cannot classify correctly all vectors, yet we will seek ways to design an *optimal linear* classifier by adopting an appropriate optimality criterion.

3.2 LINEAR DISCRIMINANT FUNCTIONS AND DECISION HYPERPLANES

Let us once more focus on the two-class case and consider linear discriminant functions. Then the respective decision hypersurface in the l-dimensional feature space is a hyperplane, that is

$$g(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{x} + w_0 = 0 \tag{3.1}$$

where $\boldsymbol{w} = [w_1, w_2, \ldots, w_l]^T$ is known as the *weight vector* and w_0 as the *threshold*. If $\boldsymbol{x}_1, \boldsymbol{x}_2$ are two points on the decision hyperplane, then the following is valid

$$\begin{aligned} 0 = \boldsymbol{w}^T \boldsymbol{x}_1 + w_0 = \boldsymbol{w}^T \boldsymbol{x}_2 + w_0 \quad \Rightarrow \\ \boldsymbol{w}^T (\boldsymbol{x}_1 - \boldsymbol{x}_2) = 0 \end{aligned} \tag{3.2}$$

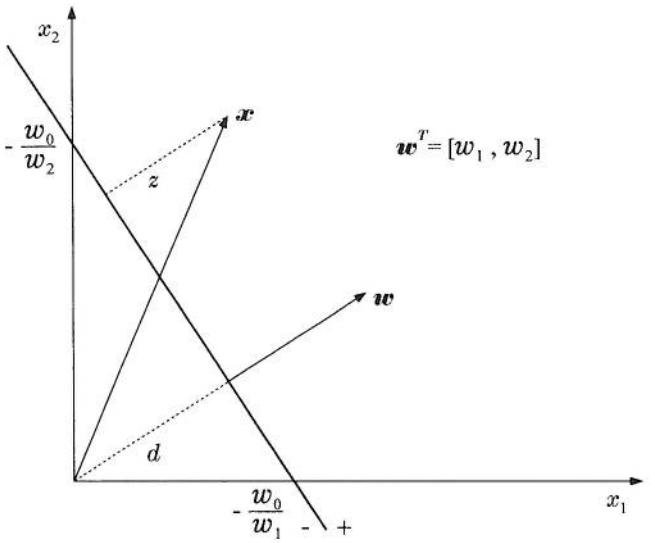

FIGURE 3.1: Geometry for the decision line. On one side of the line it is $g(\boldsymbol{x}) > 0(+)$ and on the other $g(\boldsymbol{x}) < 0\ (-)$.

Since the difference vector $\boldsymbol{x}_1 - \boldsymbol{x}_2$ obviously lies on the decision hyperplane (for any $\boldsymbol{x}_1, \boldsymbol{x}_2$), it is apparent from Eq. (3.2) that the vector $\boldsymbol{w}$ is *orthogonal* to the decision hyperplane.

Figure 3.1 shows the corresponding geometry (for $w_1 > 0, w_2 > 0, w_0 < 0$). Recalling our high school math, it is easy to see that the quantities entering in the figure are given by

$$d = \frac{|w_0|}{\sqrt{w_1^2 + w_2^2}} \tag{3.3}$$

and

$$z = \frac{|g(\boldsymbol{x})|}{\sqrt{w_1^2 + w_2^2}} \tag{3.4}$$

In other words, $|g(\boldsymbol{x})|$ is a measure of the Euclidean distance of the point $\boldsymbol{x}$ from the decision hyperplane. On one side of the plane $g(\boldsymbol{x})$ takes positive values and on the other negative. In the special case that $w_0 = 0$ the hyperplane passes through the origin.

the marginal probability densities ($p(\boldsymbol{x}_k)$) play their own part, since they enter implicitly into the game. However, in the case of logistic discrimination marginal densities contribute to C and do not affect the solution. Thus, if the Gaussian assumption is a reasonable one for the problem at hand LDA is the natural approach since it exploits all available information. On the other hand, if this is not a good assumption then logistic discrimination seems to be a better candidate, since it relies on fewer assumptions. However, in practice it has been reported [Hast 01] that there is little difference between the results obtained by the two methods. Generalizations of the logistic discrimination method to include nonlinear models have also been suggested. See, for example, [Yee 96, Hast 01].

3.7 SUPPORT VECTOR MACHINES

3.7.1 Separable Classes

In this section an alternative rationale for designing linear classifiers will be adopted. We will start with the two-class linearly separable task and then we will extend the method to more general cases where data are not separable.

Let $\boldsymbol{x}_i, i = 1, 2, \ldots, N$, be the feature vectors of the training set, X. These belong to either of two classes, ω_1, ω_2, which are assumed to be linearly separable. The goal, once more, is to design a hyperplane

$$g(\boldsymbol{x}) = \boldsymbol{w}^T\boldsymbol{x} + w_0 = 0 \tag{3.69}$$

that classifies correctly all the training vectors. As we have already discussed in Section 3.3, such a hyperplane is not unique. The perceptron algorithm may converge to any one of the possible solutions. Having gained in experience, this time we will be more demanding. Figure 3.7 illustrates the classification task with two possible hyperplane[1] solutions. Both hyperplanes do the job for the training set. However, which one of the two would any sensible engineer choose as the classifier for operation in practice, where data outside the training set will be fed to it? No doubt the answer is: the full-line one. The reason is that this hyperplane leaves more "room" on either side, so that data in both classes can move a bit more freely, with less risk of causing an error. Thus such a hyperplane can be trusted more, when it is faced with the challenge of operating with unknown data. Here we have touched a very important issue in the classifier design stage. It is known as the *generalization performance of the classifier*. This refers to the capability of the classifier, designed using the training data set, to operate satisfactorily with data outside this set. We will come to this issue over and over again.

[1] We will refer to lines as hyperplanes to cover the general case.

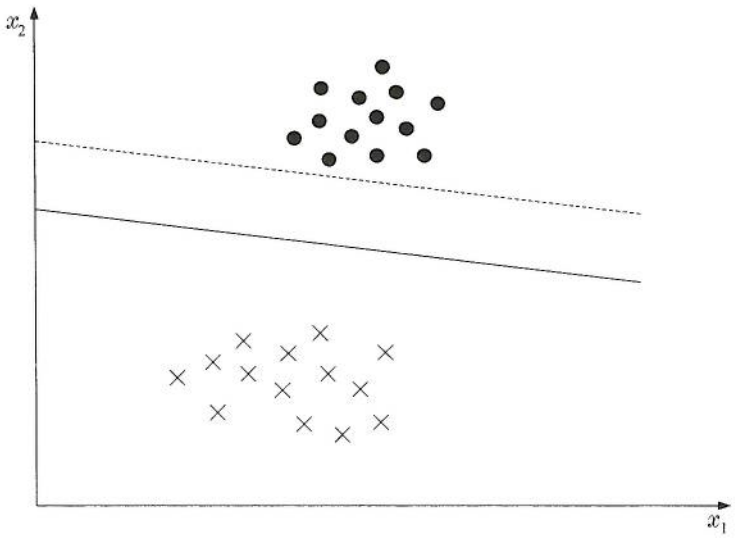

FIGURE 3.7: An example of a linearly separable two class problem with two possible linear classifiers.

After the above brief discussion, we are ready to accept that a very sensible choice for the hyperplane classifier would be the one that leaves the maximum margin from both classes. Later on, at the end of Chapter 5, we will see that this sensible choice has a deeper justification, springing from the elegant mathematical formulation that Vapnik and Chernovenkis have offered to us.

Let us now quantify the term "margin" that a hyperplane leaves from both classes. Every hyperplane is characterized by its direction (determined by $\boldsymbol{w}$) and its exact position in space (determined by w_0). Since we want to give no preference to either of the classes, then it is reasonable for each direction to select that hyperplane which has the same distance from the respective nearest points in ω_1 and ω_2. This is illustrated in Figure 3.8. The hyperplanes shown with dark lines are the selected ones from the infinite set in the respective direction. The margin for direction "1" is $2z_1$ and the margin for direction "2" is $2z_2$. *Our goal is to search for the direction that gives the maximum possible margin*. However, each hyperplane is determined within a scaling factor. We will free ourselves from it, by appropriate scaling of all the candidate hyperplanes. Recall from Section 3.2 that the distance of a point from a hyperplane is given by

$$z = \frac{|g(\boldsymbol{x})|}{\|\boldsymbol{w}\|}$$

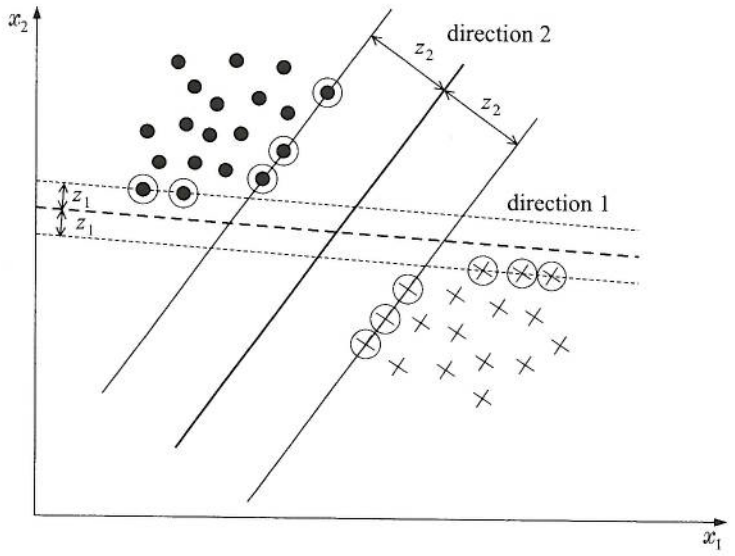

FIGURE 3.8: The margin for direction 2 is larger than the margin for direction 1.

We can now scale $\boldsymbol{w}$, w_0 so that the value of $g(\boldsymbol{x})$, at the nearest points in ω_1, ω_2 (circled in Figure 3.8), is equal to 1 for ω_1 and, thus, equal to -1 for ω_2. This is equivalent with

1. Having a margin of $\frac{1}{\|\boldsymbol{w}\|} + \frac{1}{\|\boldsymbol{w}\|} = \frac{2}{\|\boldsymbol{w}\|}$
2. Requiring that

$$\boldsymbol{w}^T\boldsymbol{x} + w_0 \geq 1, \quad \forall \boldsymbol{x} \in \omega_1$$
$$\boldsymbol{w}^T\boldsymbol{x} + w_0 \leq -1, \quad \forall \boldsymbol{x} \in \omega_2$$

We have now reached the point where mathematics will take over. For each $\boldsymbol{x}_i$, we denote the corresponding class indicator by y_i ($+1$ for ω_1, -1 for ω_2.) Our task can now be summarized as: Compute the parameters $\boldsymbol{w}$, w_0 of the hyperplane so that to:

$$\text{minimize} \quad J(\boldsymbol{w}) \equiv \frac{1}{2}\|\boldsymbol{w}\|^2 \tag{3.70}$$

$$\text{subject to} \quad y_i(\boldsymbol{w}^T\boldsymbol{x}_i + w_0) \geq 1, \quad i = 1, 2, \ldots, N \tag{3.71}$$

Obviously, minimizing the norm makes the margin maximum. This is a nonlinear (quadratic) optimization task subject to a set of linear inequality constraints.

The Karush–Kuhn–Tucker (KKT) conditions (Appendix C) that the minimizer of (3.70), (3.71) has to satisfy are

$$\frac{\partial}{\partial \boldsymbol{w}}\mathcal{L}(\boldsymbol{w}, w_0, \boldsymbol{\lambda}) = \mathbf{0} \tag{3.72}$$

$$\frac{\partial}{\partial w_0}\mathcal{L}(\boldsymbol{w}, w_0, \boldsymbol{\lambda}) = 0 \tag{3.73}$$

$$\lambda_i \geq 0, \quad i = 1, 2, \ldots, N \tag{3.74}$$

$$\lambda_i[y_i(\boldsymbol{w}^T\boldsymbol{x}_i + w_0) - 1] = 0, \quad i = 1, 2, \ldots, N \tag{3.75}$$

where $\boldsymbol{\lambda}$ is the vector of the Lagrange multipliers, λ_i, and $\mathcal{L}(\boldsymbol{w}, w_0, \boldsymbol{\lambda})$ is the Lagrangian function defined as

$$\mathcal{L}(\boldsymbol{w}, w_0, \boldsymbol{\lambda}) = \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} - \sum_{i=1}^{N}\lambda_i[y_i(\boldsymbol{w}^T\boldsymbol{x}_i + w_0) - 1] \tag{3.76}$$

Combining (3.76) with (3.72) and (3.73) results in

$$\boldsymbol{w} = \sum_{i=1}^{N}\lambda_i y_i \boldsymbol{x}_i \tag{3.77}$$

$$\sum_{i=1}^{N}\lambda_i y_i = 0 \tag{3.78}$$

Remarks

- The Lagrange multipliers can be either zero or positive (Appendix C.) Thus, the vector parameter $\boldsymbol{w}$ of the optimal solution is a linear combination of $N_s \leq N$ feature vectors which are associated with $\lambda_i \neq 0$. That is,

$$\boldsymbol{w} = \sum_{i=1}^{N_s}\lambda_i y_i \boldsymbol{x}_i \tag{3.79}$$

  These are known as *support vectors* and the optimum hyperplane classifier as a *support vector machine* (SVM). As it is pointed out in Appendix C, a nonzero Lagrange multiplier corresponds to a so called active constraint. Hence, as the set of constraints in (3.75) suggest for $\lambda_i \neq 0$, *the support vectors lie on either of the two hyperplanes*, i.e.,

$$\boldsymbol{w}^T\boldsymbol{x} + w_0 = \pm 1 \tag{3.80}$$

that is, they are the training vectors that are closest to the linear classifier, and they constitute the *critical elements of the training set*. Feature vectors corresponding to $\lambda_i = 0$ can either lie outside the "class separation band," defined as the region between the two hyperplanes given in (3.80), or they can also lie on one of these hyperplanes (degenerate case, Appendix C). The resulting hyperplane classifier is insensitive to the number and position of such feature vectors, provided they do not cross the class separation band.

- Although $\boldsymbol{w}$ is explicitly given, w_0 can be implicitly obtained by any of the (*complementary slackness*) conditions (3.75), satisfying strict complementarity (i.e., $\lambda_i \neq 0$, Appendix C). In practice, w_0 is computed as an average value obtained using all conditions of this type.
- The cost function in (3.70) is a strict convex one (Appendix C), a property that is guaranteed by the fact that the corresponding Hessian matrix is positive definite [Flet 87]. Furthermore, the inequality constraints consist of linear functions. As discussed in Appendix C, these two conditions guarantee that any local minimum is also global and unique. This is most welcome. *The optimal hyperplane classifier of a support vector machine is unique.*

Having stated all these very interesting properties of the optimal hyperplane of a support vector machine, the next step is the computation of the involved parameters. From a computational point of view this is not always an easy task and a number of algorithms exist, e.g., [Baza 79]. We will move to a path, which is suggested to us by the special nature of our optimization task, given in (3.70) and (3.71). It belongs to the *convex programming* family of problems, since the cost function is convex and the set of constraints are linear and define a convex set of feasible solutions. As we discuss in Appendix C, such problems can be solved by considering the so called *Lagrangian duality* and the problem can be stated equivalently by its Wolfe dual representation form, i.e.,

$$\text{maximize} \quad \mathcal{L}(\boldsymbol{w}, w_0, \boldsymbol{\lambda}) \tag{3.81}$$

$$\text{subject to} \quad \boldsymbol{w} = \sum_{i=1}^{N} \lambda_i y_i \boldsymbol{x}_i \tag{3.82}$$

$$\sum_{i=1}^{N} \lambda_i y_i = 0 \tag{3.83}$$

$$\boldsymbol{\lambda} \geq \mathbf{0} \tag{3.84}$$

The two equality constraints are the result of equating to zero the gradient of the Lagrangian, with respect to $\boldsymbol{w}$, w_0. We have already gained something. The training

feature vectors enter into the problem via equality constraints and not inequality ones, which can be easier to handle. Substituting (3.82) and (3.83) into (3.81) and after a bit of algebra we end up with the equivalent optimization task

$$\max_{\boldsymbol{\lambda}} \left(\sum_{i=1}^{N} \lambda_i - \frac{1}{2} \sum_{i,j} \lambda_i \lambda_j y_i y_j \boldsymbol{x}_i^T \boldsymbol{x}_j \right) \tag{3.85}$$

$$\text{subject to} \quad \sum_{i=1}^{N} \lambda_i y_i = 0 \tag{3.86}$$

$$\boldsymbol{\lambda} \geq \mathbf{0} \tag{3.87}$$

Once the optimal Lagrange multipliers have been computed, by maximizing (3.85), the optimal hyperplane is obtained via (3.82), and w_0 via the complementary slackness conditions, as before.

Remarks

- Besides the more attractive setting of the involved constraints in (3.85), (3.86), there is another important reason that makes this formulation popular. The training vectors enter into the game in pairs, in the form of inner products. This is most interesting. *The cost function does not depend explicitly on the dimensionality of the input space*! This property allows for efficient generalizations in the case of nonlinearly separable classes. We will return to this at the end of Chapter 4.
- Although the resulting optimal hyperplane is unique, there is no guarantee about the uniqueness of the associated Lagrange multipliers λ_i. In words, the expansion of $\boldsymbol{w}$ in terms of support vectors in (3.82) may not be unique, although the final result is unique (Example 3.4).

3.7.2 Nonseparable Classes

In the case where the classes are not separable, the above setup is not valid any more. Figure 3.9 illustrates the case. The two classes are not separable. Any attempt to draw a hyperplane will never end up with a class separation band with no data points inside it, as was the case in the linearly separable task. Recall that the margin is defined as the distance between the pair of parallel hyperplanes described by

$$\boldsymbol{w}^T \boldsymbol{x} + w_0 = \pm 1$$

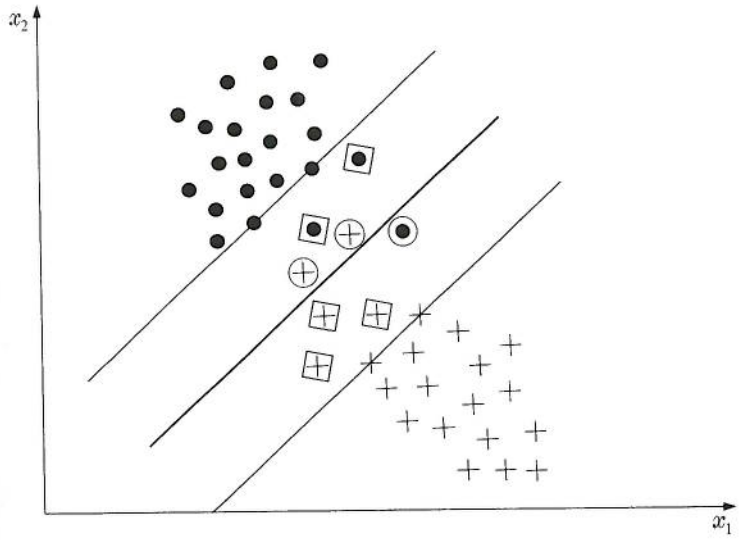

FIGURE 3.9: In the nonseparable class case, points fall inside the class separation band.

The training feature vectors now belong to one of the following three categories:

- Vectors that fall outside the band and are correctly classified. These vectors comply with the constraints in (3.71).
- Vectors falling inside the band and which are correctly classified. These are the points placed in squares in Figure 3.9 and they satisfy the inequality

$$0 \leq y_i(\boldsymbol{w}^T\boldsymbol{x} + w_0) < 1$$

- Vectors that are misclassified. They are enclosed by circles and obey the inequality

$$y_i(\boldsymbol{w}^T\boldsymbol{x} + w_0) < 0$$

All three cases can be treated under a single type of constraints by introducing a new set of variables, namely

$$y_i[\boldsymbol{w}^T\boldsymbol{x} + w_0] \geq 1 - \xi_i \tag{3.88}$$

The first category of data correspond to $\xi_i = 0$, the second to $0 < \xi_i \leq 1$, and the third to $\xi_i > 1$. The variables ξ_i are known as *slack variables*. The optimizing task becomes more involved, yet it falls under the same rationale as before. The goal now is to make the margin as large as possible but at the same time to keep the number of points with $\xi > 0$ as small as possible. In mathematical terms, this is equivalent to adopting to minimize the cost function

$$J(\boldsymbol{w}, w_0, \boldsymbol{\xi}) = \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{N} I(\xi_i) \tag{3.89}$$

where $\boldsymbol{\xi}$ is the vector of the parameters ξ_i and

$$I(\xi_i) = \begin{cases} 1 & \xi_i > 0 \\ 0 & \xi_i = 0 \end{cases} \tag{3.90}$$

The parameter C is a positive constant that controls the relative influence of the two competing terms. However, optimization of the above is difficult since it involves a discontinuous function $I(\cdot)$. As it is common in such cases, we choose to optimize a closely related cost function, and the goal becomes

$$\text{minimize} \quad J(\boldsymbol{w}, w_0, \boldsymbol{\xi}) = \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{N} \xi_i \tag{3.91}$$

$$\text{subject to} \quad y_i[\boldsymbol{w}^T\boldsymbol{x}_i + w_0] \geq 1 - \xi_i, \quad i = 1, 2, \ldots, N \tag{3.92}$$

$$\xi_i \geq 0, \quad i = 1, 2, \ldots, N \tag{3.93}$$

The problem is again a convex programming one, and the corresponding Lagrangian is given by

$$\begin{aligned} \mathcal{L}(\boldsymbol{w}, w_0, \boldsymbol{\xi}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{N} \xi_i - \sum_{i=1}^{N} \mu_i \xi_i \\ - \sum_{i=1}^{N} \lambda_i [y_i(\boldsymbol{w}^T\boldsymbol{x}_i + w_0) - 1 + \xi_i] \end{aligned} \tag{3.94}$$

The corresponding Karush–Kuhn–Tucker conditions are

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{w}} = \mathbf{0} \quad \text{or} \quad \boldsymbol{w} = \sum_{i=1}^{N} \lambda_i y_i \boldsymbol{x}_i \tag{3.95}$$

$$\frac{\partial \mathcal{L}}{\partial w_0} = 0 \quad \text{or} \quad \sum_{i=1}^{N} \lambda_i y_i = 0 \tag{3.96}$$

$$\frac{\partial \mathcal{L}}{\partial \xi_i} = 0 \quad \text{or} \quad C - \mu_i - \lambda_i = 0, \quad i = 1, 2, \ldots, N \tag{3.97}$$

$$\lambda_i [y_i (\boldsymbol{w}^T \boldsymbol{x}_i + w_0) - 1 + \xi_i] = 0, \quad i = 1, 2, \ldots, N \tag{3.98}$$

$$\mu_i \xi_i = 0, \quad i = 1, 2, \ldots, N \tag{3.99}$$

$$\mu_i \geq 0, \quad \lambda_i \geq 0, \quad i = 1, 2, \ldots, N \tag{3.100}$$

The associated Wolfe dual representation now becomes

$$\begin{aligned} \text{maximize} \quad & \mathcal{L}(\boldsymbol{w}, w_0, \boldsymbol{\lambda}, \boldsymbol{\xi}, \boldsymbol{\mu}) \\ \text{subject to} \quad & \boldsymbol{w} = \sum_{i=1}^{N} \lambda_i y_i \boldsymbol{x}_i \\ & \sum_{i=1}^{N} \lambda_i y_i = 0 \\ & C - \mu_i - \lambda_i = 0, \quad i = 1, 2, \ldots, N \\ & \lambda_i \geq 0, \ \mu_i \geq 0, \quad i = 1, 2, \ldots, N \end{aligned}$$

Substituting the above equality constraints into the Lagrangian we end up with

$$\max_{\boldsymbol{\lambda}} \left(\sum_{i=1}^{N} \lambda_i - \frac{1}{2} \sum_{i,j} \lambda_i \lambda_j y_i y_j \boldsymbol{x}_i^T \boldsymbol{x}_j \right) \tag{3.101}$$

$$\text{subject to} \quad 0 \leq \lambda_i \leq C, \quad i = 1, 2, \ldots, N \tag{3.102}$$

$$\sum_{i=1}^{N} \lambda_i y_i = 0 \tag{3.103}$$

Note that the Lagrange multipliers corresponding to the points residing either within the margin or on the wrong side of the classifier, i.e., $\xi_i > 0$, are all equal to the maximum allowable value C. Indeed, at the solution, for $\xi_i \neq 0$ the KKT conditions give $\mu_i = 0$ leading to $\lambda_i = C$. In other words, these points have the largest possible "share" in the final solution $\boldsymbol{w}$.

Remarks

- The only difference with the previously considered case of linearly separable classes is in the first of the two constraints, where the Lagrange multipliers need to be bounded above by C. The linearly separable case corresponds to $C \to \infty$. The slack variables, ξ_i, and their associated Lagrange multipliers, μ_i, do not enter into the problem explicitly. Their presence is indirectly reflected through C.
- A major limitation of support vector machines is the high computational burden required, both during training and in the test phase. For problems with a relatively small number of training data, any general purpose optimization algorithm can be used. However, for a large number of training points (of the order of a few thousands), a special treatment is required. Training of SVM is usually performed in batch mode. For large problems this sets high demands on computer memory requirements. To attack such problems a number of procedures have been devised. Their philosophy relies on the decomposition, in one way or another, of the optimization problem into a sequence of smaller ones, e.g., [Bose 92, Osun 97, Chan 00]. The main rationale behind such algorithms is to start with an arbitrary data subset (chunk of data, working set) which can fit in the computer memory. Optimization is, then, performed on this subset via a general optimizer. Support vectors remain in the working set while others are replaced by new ones, outside the current working set, that violate severely the KKT conditions. It can be shown that this iterative procedure guarantees that the cost function is decreasing at each iteration step. In [Plat 99, Matt 99] the idea of decomposition is pushed to its extreme and each working set consists of only two points. The great advantage of it is that the optimization can now be performed analytically. In [Keer 01], a set of heuristics is used for the choice of the pair of points that constitute the working set. To this end, it is suggested that the use of two thresholded parameters can lead to considerable speed ups. In [Joac 98] the working set is the result of a search for the steepest feasible direction. More recently, [Dong 05] suggested a technique to quickly remove most of the nonsupport vectors, using a parallel optimization step and the original problem can be split into many subproblems which can be solved more efficiently. Another sequential algorithm has been proposed in [Navi 01], where an iterative reweighted least squares procedure is employed and alternates weight optimization with constraint forcing. An advantage of the latter technique is that it naturally leads to online and adaptive implementations.

  For large problems, the test phase can also be quite demanding, if the number of support vectors is excessively high. Methods that speed up computations have also been suggested, e.g., [Burg 97].

- In all our discussions, so far, we have been involved with the two-class classification task. In an M-class problem, a straightforward extension can be to look at it as a set of M two-class problems. For each one of the classes, we seek to design an optimal discriminant function, $g_i(\boldsymbol{x})$, $i = 1, 2, \ldots, M$, so that $g_i(\boldsymbol{x}) > g_j(\boldsymbol{x})$, $\forall j \neq i$, if $\boldsymbol{x} \in \omega_i$. Adopting the SVM methodology, we can design the discriminant functions so that $g_i(\boldsymbol{x}) = 0$ to be the optimal hyperplane separating class ω_i from all the others, assuming of course that this is possible. Thus, the resulting linear function will give $g_i(\boldsymbol{x}) > 0$ for $\boldsymbol{x} \in \omega_i$ and $g_i(\boldsymbol{x}) < 0$ otherwise. Classification is then achieved according to the following rule:

$$\text{assign } \boldsymbol{x} \text{ in } \omega_i \text{ if } i = arg \max_k \{g_k(\boldsymbol{x})\}$$

  This technique, however, may lead to indeterminate regions, where more than one $g_i(\boldsymbol{x})$ is positive (Problem 3.15).
  Another approach is to extent the two class SVM mathematical formulation to the M-class problem, see, for example, [Vapn 98]. Comparative studies of the various methods for multiclass SVM classification can be found in [Rifk 04, Hsu 02].

Example 3.4. Consider the two-class classification task that consists of the following points:

$$\omega_1\text{: } [1, 1]^T, [1, -1]^T$$
$$\omega_2\text{: } [-1, 1]^T, [-1, -1]$$

Using the SVM approach, we will demonstrate that the optimal separating hyperplane (line) is $x_1 = 0$ and that this is obtained via different sets of Lagrange multipliers.

The points lie on the corners of a square, as shown in Figure 3.10. The simple geometry of the problem allows for a straightforward computation of the SVM linear classifier. Indeed, a careful observation of Figure 3.10 suggests that the optimal line

$$g(\boldsymbol{x}) = w_1 x_1 + w_2 x_2 + w_0 = 0$$

is obtained for $w_2 = w_0 = 0$ and $w_1 = 1$, i.e.,

$$g(\boldsymbol{x}) = x_1 = 0$$

Hence for this case, all four points become support vectors and the margin of the separating line from both classes is equal to 1. For any other direction, e.g., $g_1(\boldsymbol{x}) = 0$, the margin is smaller. It must be pointed out that, the same solution is obtained if one solves the associated KKT conditions (Problem 3.16.)

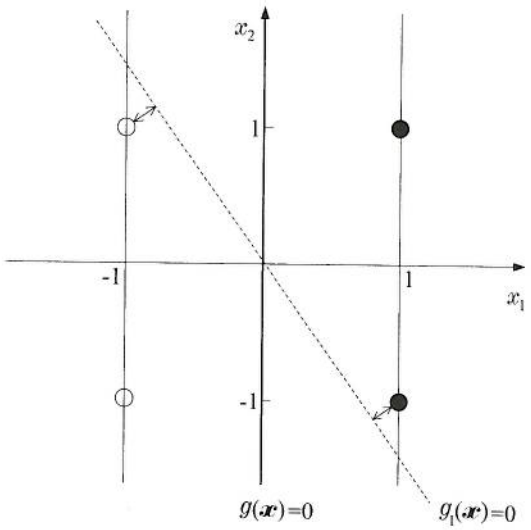

FIGURE 3.10: In this example all four points are support vectors. The margin associated with $g_1(\boldsymbol{x}) = 0$ is smaller compared to the margin defined by the optimal $g(\boldsymbol{x}) = 0$.

Let us now consider the mathematical formulation of our problem. The linear inequality constraints are

$$\begin{aligned} w_1 + w_2 + w_0 - 1 &\geq 0 \\ w_1 - w_2 + w_0 - 1 &\geq 0 \\ w_1 - w_2 - w_0 - 1 &\geq 0 \\ w_1 + w_2 - w_0 - 1 &\geq 0 \end{aligned}$$

and the associated Lagrangian function becomes

$$\begin{aligned} \mathcal{L}(w_2, w_1, w_0, \boldsymbol{\lambda}) = \frac{w_1^2 + w_2^2}{2} &- \lambda_1(w_1 + w_2 + w_0 - 1) \\ &- \lambda_2(w_1 - w_2 + w_0 - 1) \\ &- \lambda_3(w_1 - w_2 - w_0 - 1) \\ &- \lambda_4(w_1 + w_2 - w_0 - 1) \end{aligned}$$

The KKT conditions are given by

$$\frac{\partial \mathcal{L}}{\partial w_1} = 0 \Rightarrow w_1 = \lambda_1 + \lambda_2 + \lambda_3 + \lambda_4 \tag{3.104}$$

$$\frac{\partial \mathcal{L}}{\partial w_2} = 0 \Rightarrow w_2 = \lambda_1 + \lambda_4 - \lambda_2 - \lambda_3 \tag{3.105}$$

$$\frac{\partial \mathcal{L}}{\partial w_0} = 0 \Rightarrow \lambda_1 + \lambda_2 - \lambda_3 - \lambda_4 = 0 \tag{3.106}$$

$$\lambda_1(w_1 + w_2 + w_0 - 1) = 0 \tag{3.107}$$

$$\lambda_2(w_1 - w_2 + w_0 - 1) = 0 \tag{3.108}$$

$$\lambda_3(w_1 - w_2 - w_0 - 1) = 0 \tag{3.109}$$

$$\lambda_4(w_1 + w_2 - w_0 - 1) = 0 \tag{3.110}$$

$$\lambda_1, \lambda_2, \lambda_3, \lambda_4 \geq 0 \tag{3.111}$$

Since we know that the solution for $\boldsymbol{w}, w_0$ is unique, we can substitute the solution $w_1 = 1, w_2 = w_0 = 0$ into the above equations. Then we are left with a linear system of three equations with four unknowns, i.e.,

$$\lambda_1 + \lambda_2 + \lambda_3 + \lambda_4 = 1 \tag{3.112}$$

$$\lambda_1 + \lambda_4 - \lambda_2 - \lambda_3 = 0 \tag{3.113}$$

$$\lambda_1 + \lambda_2 - \lambda_3 - \lambda_4 = 0 \tag{3.114}$$

which has, obviously, more than one solution. However, all of them lead to the unique optimal separating line.

Example 3.5. Figure 3.11 shows a set of training data points residing in the two-dimensional space and divided into two nonseparable classes. The full line in Figure 3.11a is the resulting hyperplane using Platt's algorithm and corresponds to the value $C = 0.2$. Dotted lines meet the conditions given in (3.80) and define the margin that separates the two classes, for those points with $\xi_i = 0$. The setting in Figure 3.11b corresponds to $C = 1000$ and has been obtained with the same algorithm and the same set of trimming parameters (e.g., stopping criteria).

It is readily observed that the margin associated with the classifier corresponding to the larger value of C is smaller. This is because the second term in (3.89) has now more influence in the cost, and the optimization process tries to satisfy this demand by reducing the margin and consequently the number of points with $\xi_i > 0$. In other words, the width of the margin does not depend entirely on the data distribution, as it was the case with the separable class case, but is heavily affected by the choice of C. This is the reason SVM classifiers, designed via (3.89), are also known as *soft margin classifiers*.

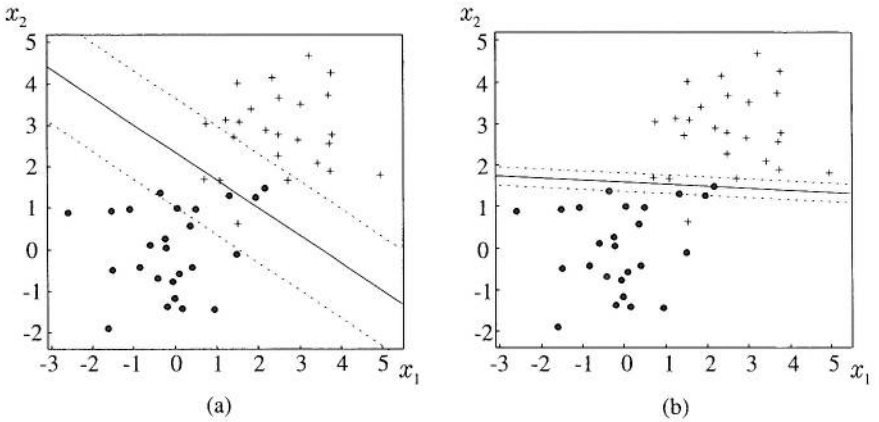

FIGURE 3.11: An example of two nonseparable classes and the resulting SVM linear classifier (full line) with the associated margin (dotted lines) for the values (a) $C = 0.2$ and (b) $C = 1000$.

3.7.3 ν-SVM

Example 3.5 demonstrated the close relation that exists between the parameter C and the width of the margin obtained as a result of the optimization process. However, since the margin is such an important entity in the design of SVM (after all, the essence of the SVM methodology is to maximize it) a natural question that arises is why not involve it in a more direct way in the cost function, instead of leaving its control to a parameter (i.e., C) whose relation with the margin, although strong, is not transparent to us. To this end, in [Scho 00] a variant of the soft margin SVM was introduced. The margin is defined by the pair of hyperplanes

$$\boldsymbol{w}^T\boldsymbol{x} + w_0 = \pm\rho \tag{3.115}$$

and $\rho \geq 0$ is left as a free variable to be optimized. Under this new setting, the primal problem given in (3.91)–(3.93) can now be cast as

$$\text{Minimize} \quad J(\boldsymbol{w}, w_0, \boldsymbol{\xi}, \rho) = \frac{1}{2}\|\boldsymbol{w}\|^2 - \nu\rho + \frac{1}{N}\sum_{i=1}^{N}\xi_i \tag{3.116}$$

$$\text{subject to} \quad y_i[\boldsymbol{w}^T\boldsymbol{x}_i + w_0] \geq \rho - \xi_i, \quad i = 1, 2, \ldots, N \tag{3.117}$$

$$\xi_i \geq 0, \quad i = 1, 2, \ldots, N \tag{3.118}$$

$$\rho \geq 0 \tag{3.119}$$

To understand the role of ρ, note that for $\xi_i = 0$ the constraints in (3.117) state that the margin separating the two classes is equal to $\frac{2\rho}{\|\boldsymbol{w}\|}$. In the previous formulation, also known as ν-SVM, we simply count and average the number of points with $\xi_i > 0$, whose number is now controlled by the margin variable ρ. The larger the ρ the wider the margin and the higher the number of points within the margin, for a specific direction $\boldsymbol{w}$. The parameter ν controls the influence of the second term in the cost function, and its value lies in the range $[0, 1]$ (we will come to this issue later on).

The Lagrangian function associated with the task (3.116)–(3.119) is given by

$$\mathcal{L}(\boldsymbol{w}, w_0, \boldsymbol{\lambda}, \boldsymbol{\xi}, \boldsymbol{\mu}, \delta) = \frac{1}{2}\|\boldsymbol{w}\|^2 - \nu\rho + \frac{1}{N}\sum_{i=1}^{N}\xi_i - \sum_{i=1}^{N}\mu_i\xi_i$$
$$- \sum_{i=1}^{N}\lambda_i\left[y_i(\boldsymbol{w}^T\boldsymbol{x}_i + w_0) - \rho + \xi_i\right] - \delta\rho \qquad (3.120)$$

Adopting similar steps as in Section 3.7.2, the following KKT conditions result:

$$\boldsymbol{w} = \sum_{i=1}^{N}\lambda_i y_i \boldsymbol{x}_i \qquad (3.121)$$

$$\sum_{i=1}^{N}\lambda_i y_i = 0 \qquad (3.122)$$

$$\mu_i + \lambda_i = \frac{1}{N}, \quad i = 1, 2, \ldots, N \qquad (3.123)$$

$$\sum_{i=1}^{N}\lambda_i - \delta = \nu \qquad (3.124)$$

$$\lambda_i\left[y_i(\boldsymbol{w}^T\boldsymbol{x}_i + w_0) - \rho + \xi_i\right] = 0, \quad i = 1, 2, \ldots, N \qquad (3.125)$$

$$\mu_i\xi_i = 0, \quad i = 1, 2, \ldots, N \qquad (3.126)$$

$$\delta\rho = 0 \qquad (3.127)$$

$$\mu_i \geq 0, \quad \lambda_i \geq 0, \quad \delta \geq 0, \quad i = 1, 2, \ldots, N \tag{3.128}$$

The associated Wolfe dual representation is easily shown to be

$$\text{Maximize} \quad \mathcal{L}(\boldsymbol{w}, w_0, \boldsymbol{\lambda}, \boldsymbol{\xi}, \boldsymbol{\mu}, \delta) \tag{3.129}$$

$$\text{subject to} \quad \boldsymbol{w} = \sum_{i=1}^{N} \lambda_i y_i \boldsymbol{x}_i \tag{3.130}$$

$$\sum_{i=1}^{N} \lambda_i y_i = 0 \tag{3.131}$$

$$\mu_i + \lambda_i = \frac{1}{N}, \quad i = 1, 2, \ldots, N \tag{3.132}$$

$$\sum_{i=1}^{N} \lambda_i - \delta = \nu \tag{3.133}$$

$$\lambda_i \geq 0, \ \mu_i \geq 0, \ \delta \geq 0, \quad i = 1, 2, \ldots, N \tag{3.134}$$

If we substitute the equality constraints (3.130)–(3.133) in the Lagrangian, the dual problem becomes equivalent to (Problem 3.17)

$$\max_{\boldsymbol{\lambda}} \left(-\frac{1}{2} \sum_{i,j} \lambda_i \lambda_j y_i y_j \boldsymbol{x}_i^T \boldsymbol{x}_j \right) \tag{3.135}$$

$$\text{subject to} \quad 0 \leq \lambda_i \leq \frac{1}{N}, \quad i = 1, 2, \ldots, N \tag{3.136}$$

$$\sum_{i=1}^{N} \lambda_i y_i = 0 \tag{3.137}$$

$$\sum_{i=1}^{N} \lambda_i \geq \nu \tag{3.138}$$

Once more, only the Lagrange multipliers $\boldsymbol{\lambda}$ enter into the problem explicitly and the presence of ρ and of the slack variables, ξ_i, make their presence felt through the bounds appearing in the constraints. Observe that in contrast to (3.101) the cost function is now quadratically homogeneous and the linear term $\sum_{i=1}^{N} \lambda_i$ is not present. Also, the new formulation has an extra constraint.

Remarks

- In [Chan 01] it is shown that the ν-SVM and the more standard SVM formulation [(3.101)–(3.103)], sometimes referred to as C-SVM, lead to the same solution for appropriate values of C and ν. Also, it is shown that in order for the optimization problem to be feasible the constant ν must lie in a range $0 \leq \nu_{\min} \leq \nu \leq \nu_{\max} \leq 1$.
- Although both SVM formulations result in the same solution, for appropriate choices of ν and C the ν-SVM offers certain advantages to the designer. As we will see in the next section, it leads to a geometric interpretation of the SVM task for nonseparable classes. Furthermore, the constant ν, controlled by the designer, offers itself to serve two important bounds concerning (a) the error rate and (b) the number of the resulting support vectors.
  At the solution, the points lying either within the margin or outside it but on the wrong side of the separating hyperplane correspond to $\xi_i > 0$ and hence to $\mu_i = 0$ [Eq. (3.126)], forcing the respective Lagrange multipliers to be $\lambda_i = \frac{1}{N}$ [Eq. (3.123)]. Also, since at the solution, for $\rho > 0$, $\delta = 0$ [Eq. (3.127)], it turns out that $\sum_{i=1}^{N} \lambda_i = \nu$ [Eq. (3.124)]. Combining these and taking into account that all points that lie in the wrong side of the classifier correspond to $\xi_i > 0$, the total number of errors can, at most, be equal to $N\nu$. Thus, the error rate, P_e, on the training set is upper bounded as

$$P_e \leq \nu. \tag{3.139}$$

Also, at the solution, from the constraints (3.124) and (3.123) we have that

$$\nu = \sum_{i=1}^{N} \lambda_i = \sum_{i=1}^{N_s} \lambda_i \leq \sum_{i=1}^{N_s} \frac{1}{N} \tag{3.140}$$

or

$$N\nu \leq N_s \tag{3.141}$$

Thus, the designer, by controlling the value of ν, may have a feeling for both the error rate on the training set and the number of the support vectors to result from the optimization process. The number of the support vectors, N_s, is very important for the performance of the classifier in practice. First, as we have already commented, it directly affects the computational load, since large N_s means a large number of inner products to be computed for classifying an unknown pattern. Second, as we will see at the end of Section 5.9, a large number of support vectors can limit the error performance of the SVM classifier when it is fed with data outside the training set (this is

also known as the generalization performance of the classifier). For more on the ν-SVM, the interested reader can consult [Scho 00, Chan 01, Chen 03], where implementation issues are also discussed.

3.7.4 Support Vector Machines: A Geometric Viewpoint

In this section, we will close the circle around the SVM design task via a path that is very close to what we call common sense. Figure 3.12a illustrates the case of two separable data classes together with their respective convex hulls. The convex hull of a data set X is denoted as $\text{conv}\{X\}$ and is defined as the intersection of all convex sets (see Appendix C.4) containing X. It can be shown (e.g., [Luen 69]) that $\text{conv}\{X\}$ consists of all the convex combinations of the N elements of X. That is,

$$\text{conv}\{X\} = \left\{ \boldsymbol{y} : \ \boldsymbol{y} = \sum_{i=1}^{N} \lambda_i \boldsymbol{x}_i : \ \boldsymbol{x}_i \in X, \right.$$
$$\left. \sum_{i=1}^{N} \lambda_i = 1, \ 0 \leq \lambda_i \leq 1, \ i = 1, 2, \ldots, N \right\} \tag{3.142}$$

It turns out that solving the dual optimization problem in (3.85)–(3.87) for the linearly separable task results in the hyperplane that bisects the linear segment joining two *nearest points between the convex hulls* of the data classes [Figure 3.12b]. In other words, *searching for the maximum margin hyperplane is equivalent to searching for two nearest points between the corresponding convex hulls!* Let us investigate this a bit further.

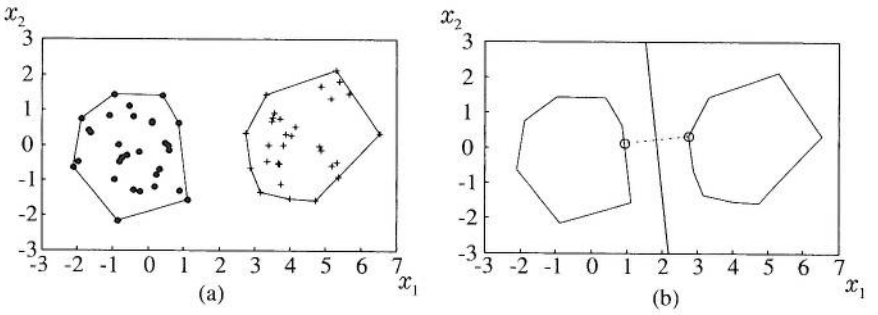

FIGURE 3.12: (a) A data set for two separable classes with the respective convex hulls. (b) The SVM optimal hyperplane bisects the segment joining the two nearest points between the convex hulls.

Denote the convex hull of the vectors in class ω_1 as $\text{conv}\{X^+\}$ and the convex hull corresponding to class ω_2 as $\text{conv}\{X^-\}$. Following our familiar notation, any point in $\text{conv}\{X^+\}$, being a convex combination of all the points in ω_1, can be written as $\sum_{i:y_i=1} \lambda_i \boldsymbol{x}_i$ and any point in $\text{conv}\{X^-\}$ as $\sum_{i:y_i=-1} \lambda_i \boldsymbol{x}_i$, provided that λ_i fulfill the convexity constraints in (3.142). Searching for the closest points, it suffices to find the specific values of λ_i, $i = 1, 2, \ldots N$, such that

$$\min_{\boldsymbol{\lambda}} \| \sum_{i:y_i=1} \lambda_i \boldsymbol{x}_i - \sum_{i:y_i=-1} \lambda_i \boldsymbol{x}_i \|^2 \tag{3.143}$$

$$\text{subject to} \quad \sum_{i:y_i=1} \lambda_i = 1, \quad \sum_{i:y_i=-1} \lambda_i = 1 \tag{3.144}$$

$$\lambda_i \geq 0, \; i = 1, 2, \ldots N \tag{3.145}$$

Elaborating the norm in (3.143) and reshaping the constraints in (3.144), we end up with the following equivalent formulation.

$$\text{Minimize} \quad \sum_{i,j} y_i y_j \lambda_i \lambda_j \boldsymbol{x}_i^T \boldsymbol{x}_j \tag{3.146}$$

$$\text{subject to} \quad \sum_{i=1}^{N} y_i \lambda_i = 0, \quad \sum_{i=1}^{N} \lambda_i = 2 \tag{3.147}$$

$$\lambda_i \geq 0, \; i = 1, 2, \ldots N \tag{3.148}$$

It takes a few lines of algebra to show that the optimization task in (3.85)–(3.87) results in the same solution as the task given in (3.146)–(3.148) ([Keer 00] and Problem 3.18). Having established the geometric interpretation of the SVM optimization task, any algorithm that has been developed to search for nearest points between convex hulls (e.g., [Gilb 66, Mitc 74, Fran 03]) can now, in principle, be mobilized to compute the maximum margin linear classifier.

It is now the turn of the nonseparable class problem to enter into the game, which, now, becomes more exciting. Let us return to the ν-SVM formulation and reparameterize the primal problem in (3.116)–(3.119) by dividing the cost function by $\frac{\nu^2}{2}$ and the set of constraints by ν ([Crisp 99]). Obviously, this has no effect on

the solution. The optimization task now becomes

$$\text{minimize} \quad J(\boldsymbol{w}, w_0, \boldsymbol{\xi}, \rho) = \|\boldsymbol{w}\|^2 - 2\rho + \mu \sum_{i=1}^{N} \xi_i \tag{3.149}$$

$$\text{subject to} \quad y_i[\boldsymbol{w}^T \boldsymbol{x}_i + w_0] \geq \rho - \xi_i, \quad i = 1, 2, \ldots, N \tag{3.150}$$

$$\xi_i \geq 0, \quad i = 1, 2, \ldots, N \tag{3.151}$$

$$\rho \geq 0 \tag{3.152}$$

where $\mu = \frac{2}{\nu N}$ and we have kept, for economy, the same notation, although the parameters in (3.149)–(3.152) are scaled versions of those in (3.116)–(3.119). That is, $\boldsymbol{w} \rightarrow \frac{\boldsymbol{w}}{\nu}$, $w_0 \rightarrow \frac{w_0}{\nu}$, $\rho \rightarrow \frac{\rho}{\nu}$, $\xi_i \rightarrow \frac{\xi_i}{\nu}$. Hence, the solution obtained via (3.149)–(3.152) is a scaled version of the solution resulting via (3.116)–(3.119). The Wolfe dual representation of the primal problem in (3.149)–(3.152) is easily shown to be equivalent to

$$\text{minimize} \quad \sum_{i,j} y_i y_j \lambda_i \lambda_j \boldsymbol{x}_i^T \boldsymbol{x}_j \tag{3.153}$$

$$\text{subject to} \quad \sum_i y_i \lambda_i = 0, \quad \sum_i \lambda_i = 2 \tag{3.154}$$

$$0 \leq \lambda_i \leq \mu, \; i = 1, 2, \ldots N \tag{3.155}$$

This set of relations is almost the same with those defining the nearest points between the convex hulls in the separable class case, (3.146)–(3.148), with a small, yet significant, difference. The Lagrange multipliers are bounded by μ and for $\mu < 1$ they are not permitted to span their entire allowable range (i.e., [0, 1]).

3.7.5 Reduced Convex Hulls

The *reduced convex hull* (RCH) of a vector space, X, is denoted as $R(X, \mu)$ and is defined as the convex set

$$R(X, \mu) = \left\{ \boldsymbol{y} : \boldsymbol{y} = \sum_{i=1}^{N} \lambda_i \boldsymbol{x}_i : \; \boldsymbol{x}_i \in X, \right.$$
$$\left. \sum_{i=1}^{N} \lambda_i = 1, \; 0 \leq \lambda_i \leq \mu, \; i = 1, 2, \ldots, N \right\} \tag{3.156}$$

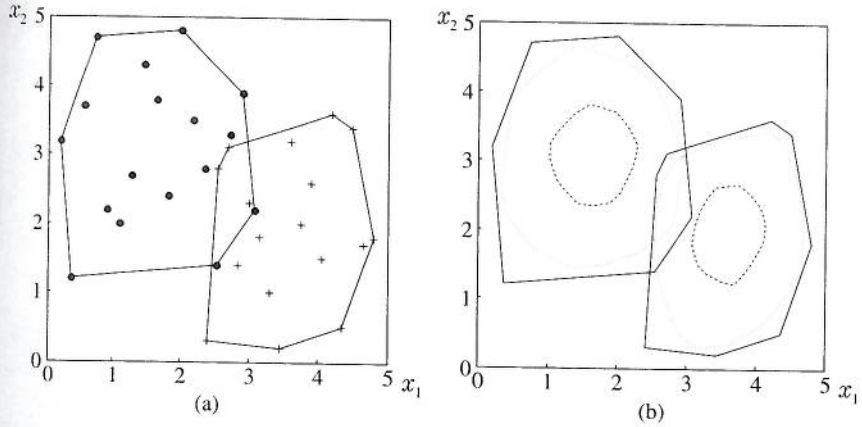

FIGURE 3.13: (a) Example of a data set with two intersecting classes and their respective convex hulls. (b) The convex hulls (indicated by full lines) and the resulting reduced convex hulls (indicated by dotted lines) corresponding to $\mu = 0.4$ and $\mu = 0.1$, respectively, for each class. The smaller the value of μ the smaller the RCH size.

It is apparent from the previous definition that $R(X, 1) \equiv \text{conv}\{X\}$ and that

$$R(X, \mu) \subseteq \text{conv}\{X\} \tag{3.157}$$

Figure 3.13a shows the respective convex hulls for the case of two intersecting data classes. In Figure 3.13b, full lines indicate the convex hulls, $\text{conv}\{X^+\}$ and $\text{conv}\{X^-\}$, and the dotted lines the reduced convex hulls $R(X^+, \mu)$, $R(X^-, \mu)$, for two different values of $\mu = 0.4$ and $\mu = 0.1$, respectively. It is readily apparent that the smaller the value of μ the smaller the size of the reduced convex hull. For small enough values of μ, one can make $R(X^+, \mu)$ and $R(X^-, \mu)$ nonintersecting. Adopting a procedure similar to the one that led to (3.146)–(3.148), it is not difficult to see that finding two nearest points between $R(X^+, \mu)$ and $R(X^-, \mu)$ results in the ν-SVM dual optimization task given in (3.153)–(3.155). Observe that the only difference between the latter and the task for the separable case, defined in (3.146)–(3.148), lies in the range in which the Lagrange multipliers are allowed to be. In the separable class case, the constraints (3.147) and (3.148) imply that $0 \leq \lambda_i \leq 1$, which in its geometric interpretation means that the full convex hulls are searched for the nearest points. In contrast, in the nonseparable class case a lower upper bound (i.e., $\mu \leq 1$) is imposed for the Lagrange multipliers. From the geometry point of view, this means that the search for the nearest points is limited within the respective reduced convex hulls.

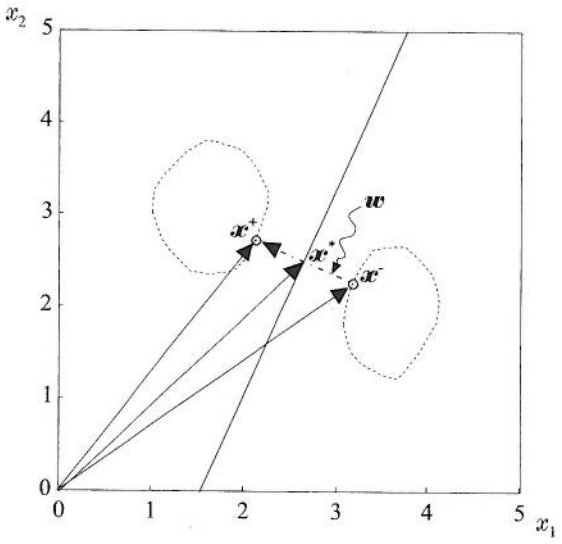

FIGURE 3.14: The optimal linear classifier resulting as the bisector of the segment joining the two closest points between the reduced convex hulls of the classes, for the case of the data set shown in Figure 3.13 and for $\mu = 0.1$.

Having established the geometric interpretation of the ν-SVM dual representation form, let us follow pure geometric arguments to draw the separating hyperplane. It is natural to choose it as the one bisecting the line segment joining two nearest points between the reduced convex hulls. Let $\boldsymbol{x}^+$ and $\boldsymbol{x}^-$ be two nearest points, with $x^+ \in R(X^+, \mu)$ and $x^- \in R(X^-, \mu)$. Let also λ_i, $i = 1, 2, \ldots, N$, be the optimal set of multipliers resulting from the optimization task. Then, as can be deduced from Figure 3.14,

$$\boldsymbol{w} = \boldsymbol{x}^+ - \boldsymbol{x}^- = \sum_{i:y_i=1} \lambda_i \boldsymbol{x}_i - \sum_{i:y_i=-1} \lambda_i \boldsymbol{x}_i \tag{3.158}$$

$$= \sum_{i=1}^{N} \lambda_i y_i \boldsymbol{x}_i \tag{3.159}$$

This is the same (within a scaling factor) as the $\boldsymbol{w}$ obtained from the KKT conditions associated with the ν-SVM task [Eq. (3.121)]. Thus, both approaches result in a separating hyperplane pointing in the same direction (recall from Section 3.2 that $\boldsymbol{w}$ defines the direction of the hyperplane). However, it is early to say that the two solutions are exactly the same. The hyperplane bisecting the line segment

joining the nearest points crosses the middle of this segment; that is, the point $\boldsymbol{x}^* = \frac{1}{2}(\boldsymbol{x}^+ + \boldsymbol{x}^-)$. Thus,

$$\boldsymbol{w}^T \boldsymbol{x}^* + w_0 = 0 \tag{3.160}$$

from which we get

$$w_0 = -\frac{1}{2}\boldsymbol{w}^T \left(\sum_{i:y_i=1} \lambda_i \boldsymbol{x}_i + \sum_{i:y_i=-1} \lambda_i \boldsymbol{x}_i \right) \tag{3.161}$$

This value for w_0 is, in general, different from the value resulting from the KKT conditions in (3.125). In conclusion, the geometric approach in the case of the nonseparable problem is equivalent to the ν-SVM formulation only to the extent that both approaches result in hyperplanes pointing in the same direction. However, note that the value in Eq. (3.125) can be obtained from that given in Eq. (3.161) in a trivial way [Crisp 99].

Remarks

- The choice of μ and consequently of $\nu = \frac{2}{\mu N}$ must guarantee that the feasible region is nonempty (i.e., a solution exists, Appendix C) and also that the solution is a nontrivial one (i.e., $\boldsymbol{w} \neq \boldsymbol{0}$). Let N^+ be the number of points in X^+ and N^- the number of points in X^-, where $N^+ + N^- = N$. Let $N_{\min} = \min\{N^+, N^-\}$. Then it is readily seen from the crucial constraint $0 \leq \lambda_i \leq \mu$ and the fact that $\sum_i \lambda_i = 1$, in the definition of the reduced convex hull, that $\mu \geq \mu_{\min} = \frac{1}{N_{\min}}$. This readily suggests that ν cannot take any value but must be upper bounded as

$$\nu \leq \nu_{\max} = 2\frac{N_{\min}}{N} \leq 1$$

  Also, if the respective reduced convex hulls intersect then the distance between the closest points is zero, leading to the trivial solution (Problem 3.19). Thus, nonintersection is guaranteed for some value $\mu_{\max}$ such that $\mu \leq \mu_{\max} \leq 1$, which leads to

$$\nu \geq \nu_{\min} = \frac{2}{\mu_{\max} N}$$

  From the previous discussion it is easily deduced that for the feasible region to be nonempty it is required that

$$R(X^+, \mu_{\min}) \cap R(X^-, \mu_{\min}) = \emptyset$$

If $N^+ = N^- = \frac{N}{2}$, it is easily checked out that in this case each of the reduced convex hulls is shrunk to a point, which is the centroid of the respective class (e.g., $\frac{2}{N}\sum_{i:y_i=1} \boldsymbol{x}_i$). In other words, a solution is feasible if the centroids of the two classes do not coincide. Most natural!

- Some theorems concerning properties of the reduced convex hulls, which are important for the development of efficient geometric algorithms, are derived in [Mavr 06].

Problems

3.1 Explain why the perceptron cost function is a *continuous* piecewise linear function.

3.2 Show that if $\rho_k = \rho$ in the perceptron algorithm, the algorithm converges after $k_0 = \frac{\|\boldsymbol{w}(0)-\alpha\boldsymbol{w}^*\|}{\beta^2\rho(2-\rho)}$ steps, where $\alpha = \frac{\beta^2}{|\gamma|}$ and $\rho < 2$.

3.3 Show that the reward and punishment form of the perceptron algorithm converges in a finite number of iteration steps.

3.4 Consider a case in which class ω_1 consists of the two feature vectors $[0, 0]^T$ and $[0, 1]^T$ and class ω_2 of $[1, 0]^T$ and $[1, 1]^T$. Use the perceptron algorithm in its reward and punishment form, with $\rho = 1$ and $\boldsymbol{w}(0) = [0, 0]^T$, to design the line separating the two classes.

3.5 Consider the two-class task of Problem 2.12 of the previous chapter with

$$\boldsymbol{\mu}_1^T = [1, 1], \quad \boldsymbol{\mu}_2^T = [0,0], \quad \sigma_1^2 = \sigma_2^2 = 0.2$$

Produce 50 vectors from each class. To guarantee linear separability of the classes, disregard vectors with $x_1 + x_2 < 1$ for the $[1, 1]$ class and vectors with $x_1 + x_2 > 1$ for the $[0, 0]$ class. In the sequel use these vectors to design a linear classifier using the perceptron algorithm of (3.21). After convergence, draw the corresponding decision line.

3.6 Consider once more the classification task of Problem 2.12. Produce 100 samples for each of the classes. Use these data to design a linear classifier via the LMS algorithm. Once all samples have been presented to the algorithm, draw the corresponding hyperplane to which the algorithm has converged. Use $\rho_k = \rho = 0.01$.

3.7 Show, using Kesler's construction, that the tth iteration step of the reward and punishment form of the perceptron algorithm (3.21), for an $\boldsymbol{x}_{(t)} \in \omega_i$, becomes

$$\begin{aligned}
&\boldsymbol{w}_i(t+1) = \boldsymbol{w}_i(t) + \rho\boldsymbol{x}_{(t)} \quad \text{if } \boldsymbol{w}_i^T(t)\boldsymbol{x}_{(t)} \le \boldsymbol{w}_j^T(t)\boldsymbol{x}_{(t)},\ j \neq i \\
&\boldsymbol{w}_j(t+1) = \boldsymbol{w}_j(t) - \rho\boldsymbol{x}_{(t)} \quad \text{if } \boldsymbol{w}_i^T(t)\boldsymbol{x}_{(t)} \le \boldsymbol{w}_j^T(t)\boldsymbol{x}_{(t)},\ j \neq i \\
&\boldsymbol{w}_k(t+1) = \boldsymbol{w}_k(t), \quad \forall k \neq j \text{ and } k \neq i
\end{aligned}$$

3.8 Show that the sum of error squares optimal weight vector tends asymptotically to the MSE solution.

3.9 Repeat Problem 3.6 and design the classifier using the sum of error squares criterion.

3.10 Show that the design of an M class linear, sum of error squares optimal, classifier reduces to M equivalent ones, with scalar desired responses.

3.11 Show that, if x, y are jointly Gaussian, the regression of y on x is given by

$$E[y|x] = \frac{\alpha\sigma_y x}{\sigma_x} + \mu_y - \frac{\alpha\sigma_y\mu_x}{\sigma_x}, \quad \text{where } \Sigma = \begin{bmatrix} \sigma_x^2 & \alpha\sigma_x\sigma_y \\ \alpha\sigma_x\sigma_y & \sigma_y^2 \end{bmatrix} \tag{3.162}$$

3.12 Let an M class classifier be given in the form of parameterized functions $g(\boldsymbol{x}; \boldsymbol{w}_k)$. The goal is to estimate the parameters $\boldsymbol{w}_k$ so that the outputs of the classifier give desired response values, depending on the class of $\boldsymbol{x}$. Assume that as $\boldsymbol{x}$ varies randomly in each class, the classifier outputs vary around the corresponding desired response values, according to a Gaussian distribution of known variance, assumed to be the same for all outputs. Show that in this case the sum of error squares criterion and the ML estimation result in identical estimates.

Hint: Take N training data samples of known class labels. For each of them form $y_i = g(\boldsymbol{x}_i; \boldsymbol{w}_k) - d_k^i$, where d_k^i is the desired response for the kth class of the ith sample. The y_i's are normally distributed with zero mean and variance σ^2. Form the likelihood function using the y_i's.

3.13 In a two-class problem the Bayes optimal decision surface is given by $g(\boldsymbol{x}) = P(\omega_1|\boldsymbol{x}) - P(\omega_2|\boldsymbol{x}) = 0$. Show that if we train a decision surface $f(\boldsymbol{x}; \boldsymbol{w})$ in the MSE so as to give $+1(-1)$ for the two classes, respectively, this is equivalent to approximating $g(\cdot)$ in terms of $f(\cdot; \boldsymbol{w})$, in the MSE optimal sense.

3.14 Consider a two-class classification task with jointly Gaussian distributed feature vectors and with the same variance Σ in both classes. Design the linear MSE classifier and show that in this case the Bayesian classifier (Problem 2.11) and the resulting MSE one differ only in the threshold value. For simplicity, consider equiprobable classes.

Hint: To compute the MSE hyperplane $\boldsymbol{w}^T\boldsymbol{x} + w_0 = 0$, increase the dimension of $\boldsymbol{x}$ by one and show that the solution is provided by

$$\begin{bmatrix} R & E[\boldsymbol{x}] \\ E[\boldsymbol{x}]^T & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{w} \\ w_0 \end{bmatrix} = \begin{bmatrix} \frac{1}{2}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2) \\ 0 \end{bmatrix}$$

Then relate R with Σ and show that the MSE classifier takes the form

$$(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^T \Sigma^{-1}\Big(\boldsymbol{x} - \frac{1}{2}(\boldsymbol{\mu}_1 + \boldsymbol{\mu}_2)\Big) \gtrless 0$$

3.15 In an M class classification task the classes can be linearly separated. Design M hyperplanes, so that hyperplane $g_i(\boldsymbol{x}) = 0$ leaves class ω_i on its positive side and the rest of the classes on its negative side. Demonstrate via an example, e.g., $M = 3$, that the partition of the space using this rule creates indeterminate regions (where no training data exist) for which more than one $g_i(\boldsymbol{x})$ is positive or all of them are negative.

3.16 Obtain the optimal line for the task of Example 3.4, via the KKT conditions. Restrict the search for the optimum among the lines crossing the origin.

3.17 Show that if the equality constraints (3.130)–(3.133) are substituted in the Lagrangian (3.120), the dual problem is described by the set of relations in (3.135)–(3.138).

3.18 Show that for the case of two linearly separable classes the hyperplane obtained as the SVM solution is the same with that bisecting the segment joining two closest points between the convex hulls of the classes.

3.19 Show that if ν in the ν-SVM is chosen smaller than $\nu_{\min}$ it leads to the trivial zero solution.

3.20 Show that if the soft margin SVM cost function is chosen to be

$$\frac{1}{2}||\boldsymbol{w}||^2 + \frac{C}{2}\sum_{i=1}^{N}\xi_i^2$$

the task can be transformed into an instance of the class-separable case problem [Frie 98].

References

[Ande 82] Anderson J.A. "Logistic discrimination," in *Handbook of Statistics* (Krishnaiah R.P., Kanal L.N., eds.), North Holland, 1982.

[Baza 79] Bazaraa M.S., Shetty C.M. *Nonlinear Programming*, John Wiley & Sons, 1979.

[Bish 95] Bishop C. *Neural Networks for Pattern Recognition*, Oxford University Press, 1995.

[Bose 92] Bose B.E., Guyon I.M., Vapnik, V.N. "A training algorithm for optimal margin classifiers," *Proceedings of the 5th Annual Workshop on Computational Learning Theory*, pp. 144–152, Morgan Kaufman, San Mateo, CA, 1992.

[Burg 97] Burges C.J.C., Schölkoff B. "Improving the accuracy and speed of support vectors learning machines," in *Advances in Neural Information Processing Systems 9* (Mozer M, Jordan M., Petsche T., eds.), pp. 375–381, MIT Press, Cambridge, MA, 1997.

[Chan 00] Chang C.C., Hsu C.W., Lin C.J. "The analysis of decomposition methods for SVM," *IEEE Transactions on Neural Networks*, Vol. 11(4), pp. 1003–1008, 2000.

[Chan 01] Chang C.C., Lin C.J. "Training ν-support vector classifiers: Theory and algorithms," *Neural Computation* Vol. 13(9), pp. 2119–2147, 2001.

[Chen 03] Chen P-H., Lin C.J., Schölkopf B. "A tutorial on ν-support vector machines," *Applied Stochastic Models in Business and Industry*, Vol. 21, pp. 111–136, 2005.

[Cid 99] Cid-Sueiro J., Arribas J.I., Urban-Munoz S., Figueras-Vidal A.R. "Cost functions to estimate a-posteriori probabilities in multiclass problems," *IEEE Transactions on Neural Networks*, Vol. 10(3), pp. 645–656, 1999.

[Crisp 99] Crisp D.J., Burges C.J.C. "A geometric interpretation of ν-SVM classifiers," *Proceedings of Neural Information Processing*, Vol. 12, MIT Press, Cambridge, MA, 1999.

[Dong 05] Dong J.X., Krzyzak A., Suen C.Y. "Fast SVM training algorithm with decomposition on very large data sets," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 27(4), pp. 603–618, 2005.

[Flet 87] Fletcher R. *Practical Methods of Optimization*, 2nd ed., John Wiley & Sons, 1987.

[Fran 03] Franc V., Hlaváč V. "An iterative algorithm learning the maximal margin classifier," *Pattern Recognition*, Vol. 36, pp. 1985–1996, 2003.