Support Vector Machines

INTRODUCTION.

The main idea behind the machine may be summed up as follows:

Given a training sample, the support vector machine constructs a hyperplane as the decision surface in such a way that the margin of separation between positive and negative examples is maximized.

This basic idea is extended in a principled way to deal with the more difficult case of nonlinearly separable patterns.

A notion that is central to the development of the support vector learning algorithm is the inner-product kernel between a "support vector" $\mathbf{x}_i$ and a vector $\mathbf{x}$ drawn from the input data space. Most importantly, the support vectors consist of a small subset of data points extracted by the learning algorithm from the training sample itself. Indeed, it is because of this central property that the learning algorithm, involved in the construction of a support vector machine, is also referred to as a kernel method. However, unlike the suboptimal kernel method described in Chapter 5, the kernel method basic to the design of a support vector machine is optimal, with the optimality being rooted in convex optimization. However, this highly desirable feature of the machine is achieved at the cost of increased computational complexity.

The support vector machine can be used to solve both pattern-classification and nonlinear-regression problems. However, it is in solving difficult pattern-classification problems where support vector machines have made their most significant impact.

Case 1) OPTIMAL HYPERPLANE FOR LINEARLY SEPARABLE PATTERNS

Consider the training sample $\{(\mathbf{x}_i, \mathbf{d}_i)\}^{N}_{i=1}$, where xi is the input pattern for the ith example and $\mathbf{d}_i$ is the corresponding desired response (target output). To begin with, we assume that the pattern (class) represented by the subset $d_i = +1$ and the pattern represented by the subset $d_i = -1$ are "linearly separable." The equation of a decision surface in the form of a hyperplane that does the separation is

$$\mathbf{w}^T\mathbf{x} + b = 0 \tag{6.1}$$

where $\mathbf{x}$ is an input vector, w is an adjustable weight vector, and b is a bias. We may thus write

$$\begin{aligned} \mathbf{w}^T\mathbf{x}_i + b &\geq 0 \qquad \text{for } d_i = +1 \\ \mathbf{w}^T\mathbf{x}_i + b &< 0 \qquad \text{for } d_i = -1 \end{aligned} \tag{6.2}$$

The assumption of linearly separable patterns is made here to explain the basic idea behind a support vector machine in a rather simple setting; this assumption will be relaxed in Section 6.3.

For a given weight vector **w** and bias **b**, the separation between the hyperplane defined in Eq. (6.1) and the closest data point is called the margin of separation, denoted by **ρ**. The goal of a support vector machine is to find the particular hyperplane for which the margin of separation **ρ**, is maximized. Under this condition, the decision surface is referred to as the optimal hyperplane. Figure 6.1 illustrates the geometric construction of an optimal hyperplane for a two-dimensional input space.

Let $\mathbf{w_o}$ and $\mathbf{b_o}$ denote the optimum values of the weight vector and bias, respectively. Correspondingly, the optimal hyperplane, representing a multidimensional linear decision surface in the input space, is defined by

$$\mathbf{w}_o^T\mathbf{x} + b_o = 0 \tag{6.3}$$

which is a rewrite of Eq. (6.1).

$$g(\mathbf{x}) = \mathbf{w}_o^T\mathbf{x} + b_o \tag{6.4}$$

The discriminant function gives an algebraic measure of the distance from x to the optimal hyperplane (Duda and Hart, 1973). Perhaps the easiest way to see this is to express x as:

$$\mathbf{x} = \mathbf{x}_p + r\frac{\mathbf{w}_o}{\|\mathbf{w}_o\|}$$

where x_p is the normal projection of x onto the optimal hyperplane and r is the desired algebraic distance;r is positive if x is on the positive side of the optimal hyperplane and negative if x is on the negative side. Since, by definition, $g(x_p) = 0$, it follows that

$$g(\mathbf{x}) = \mathbf{w}_o^T\mathbf{x} + b_o = r\|\mathbf{w}_o\|$$

or, equivalently,

$$r = \frac{g(\mathbf{x})}{\|\mathbf{w}_o\|} \tag{6.5}$$

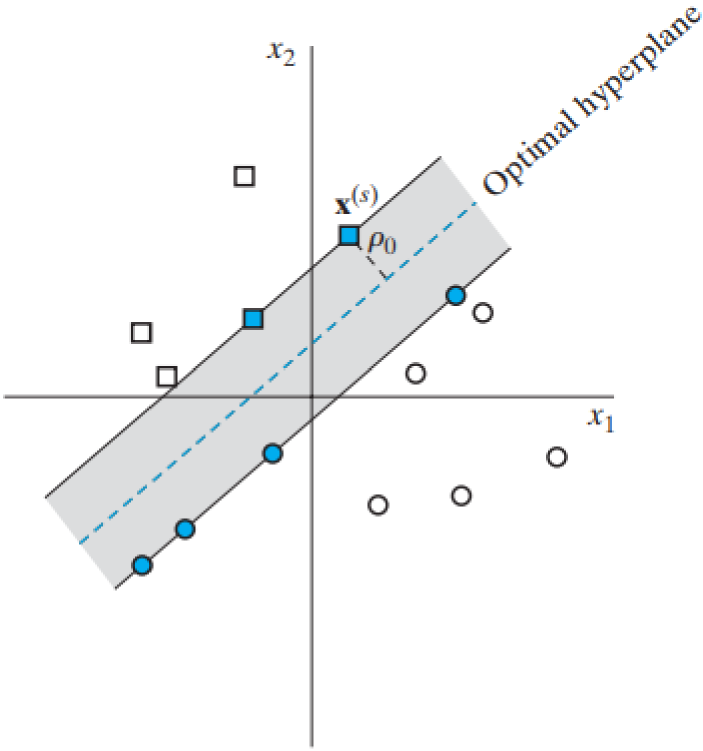


Illustration of the idea of an optimal hyperplane for linearly separable patterns: The data points shaded in red are support vectors.

In particular, the distance from the origin (i.e., x=0) to the optimal hyperplane is given by $\mathbf{b_0/\|w_0\|}$. If $\mathbf{b_o > 0}$, the origin is on the positive side of the optimal hyperplane; if $\mathbf{b_0 < 0}$, it is on the negative side. If $\mathbf{b_o = 0}$, the optimal hyperplane passes through the origin. A geometric interpretation of these algebraic results is given in Fig. 6.2.

The issue at hand is to find the parameters $\mathbf{w_o}$ and $\mathbf{b_o}$ for the optimal hyperplane, given the training set $\mathbf{T = \{(x_i, d_i)\}}$. In light of the results portrayed in Fig. 6.2, we see that the pair (w_0, b_0) must satisfy the following constraint:

$$\begin{aligned} \mathbf{w}_o^T\mathbf{x}_i + b_o &\geq 1 \qquad \text{for } d_i = +1 \\ \mathbf{w}_o^T\mathbf{x}_i + b_o &\leq -1 \qquad \text{for } d_i = -1 \end{aligned} \tag{6.6}$$

Note that if Eq. (6.2) holds—that is, if the patterns are linearly separable—we can always rescale $\mathbf{w_o}$ and $\mathbf{b_o}$ such that Eq. (6.6) holds; this scaling operation leaves Eq. (6.3) unaffected.

The particular data points **(xi, di)** for which the first or second line of Eq. (6.6) is satisfied with the equality sign are called support vectors—hence the name “support vector machine”. All the remaining examples in the training sample are completely irrelevant. Because of their distinct property, the support vectors play a prominent role in the operation of this class of learning machines. In conceptual terms, the support vectors are those data points that lie closest to the optimal hyperplane and are therefore the most difficult to classify. As such, they have a direct bearing on the optimum location of the decision surface.

Consider a support vector $\mathbf{x}^{(s)}$ for which $\mathbf{d}^{(s)} = +1$. Then, by definition, we have From Eq. (6.5), the algebraic distance from the support vector $x^{(s)}$ to the optimal hyperplane is

$$g(\mathbf{x}^{(s)}) = \mathbf{w}_o^T\mathbf{x}^{(s)} + b_o = \mp 1 \qquad \text{for } d^{(s)} = \mp 1 \tag{6.7}$$

$$\begin{aligned} r &= \frac{g(\mathbf{x}^{(s)})}{\|\mathbf{w}_o\|} \\ &= \begin{cases} \dfrac{1}{\|\mathbf{w}_o\|} & \text{if } d^{(s)} = +1 \\ -\dfrac{1}{\|\mathbf{w}_o\|} & \text{if } d^{(s)} = -1 \end{cases} \end{aligned} \tag{6.8}$$

where the plus sign indicates that $x^{(s)}$ lies on the positive side of the optimal hyperplane and the minus sign indicates that $x^{(s)}$ lies on the negative side of the optimal hyperplane. Let & denote the optimum value of the margin of separation between the two classes that constitute the training sample T Then, from Eq. (6.8), it follows that

$$\begin{aligned} \rho &= 2r \\ &= \frac{2}{\|\mathbf{w}_o\|} \end{aligned} \tag{6.9}$$

Equation (6.9) states the following:

Maximizing the margin of separation between binary classes is equivalent to minimizing the Euclidean norm of the weight vector w.

In summary, the optimal hyperplane defined by Eq. (6.3) is unique in the sense that the optimum weight vector wo provides the maximum possible separation between positive and negative examples. This optimum condition is attained by minimizing the Euclidean norm of the weight vector w.

Quadratic Optimization for Finding the Optimal Hyperplane

The support vector machine is cleverly formulated under the umbrella of convex optimization, hence the well-defined optimality of the machine. In basic terms, the formulation proceeds along four major steps:

1. The problem of finding the optimal hyperplane starts with a statement of the problem in the primal weight space as a constrained-optimization problem.
2. The Lagrange function of the problem is constructed.
3. The conditions for optimality of the machine are derived.
4. The stage is finally set for solving the optimization problem in the dual space of Lagrange multipliers.

To proceed then, we first note that the training sample is embodied in the two-line constraint of Eq. (6.6). It is instructive to combine the two lines of this equation into the single line as:

$$\mathcal{T} = \{\mathbf{x}_i, d_i\}_{i=1}^{N}$$

$$d_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 \qquad \text{for } i = 1, 2, \ldots, N \tag{6.10}$$

With this form of the constraint at hand, we are now ready to formally state the constrained-optimization problem as follows: Given the training sample $\{(\mathbf{x}_i\ \mathbf{d}_i)\}_{i=1}^{N}$,find the optimum values of the weight vector **w** and bias **b** such that they satisfy the constraints and the weight vector w minimizes the cost function:

$$d_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 \qquad \text{for } i = 1, 2, \ldots, N$$

$$\Phi(\mathbf{w}) = \frac{1}{2}\mathbf{w}^T\mathbf{w}$$

The scaling factor ½ is included here for convenience of presentation. This constrained-optimization problem is called the primal problem. It is basically characterized as follows:

- The cost function **Φ(w)** is a convex function of w.
- The constraints are linear in w.

Accordingly, we may solve the constrained-optimization problem by using the method of Lagrange multipliers (Bertsekas, 1995). First, we construct the function:

$$J(\mathbf{w}, b, \alpha) = \frac{1}{2}\mathbf{w}^T\mathbf{w} - \sum_{i=1}^{N}\alpha_i[d_i(\mathbf{w}^T\mathbf{x}_i + b) - 1] \tag{6.11}$$

where the auxiliary nonnegative variables $\boldsymbol{\alpha}_i$ are called Lagrange multipliers. The solution to the constrained-optimization problem is determined by the saddle point of the Lagrangian function **J(w, b, α).** A saddle point of a Lagrangian is a point where the roots are real, but of opposite signs; such a singularity is always unstable. The saddle point has to be minimized with respect to w and b; it also has to be maximized with respect to α. Thus, differentiating **J(w, b, α)** with respect to w and b and setting the results equal to zero, we get the following two conditions of optimality:

$$\text{Condition 1:} \qquad \frac{\partial J(\mathbf{w}, b, \alpha)}{\partial \mathbf{w}} = \mathbf{0}$$

$$\text{Condition 2:} \qquad \frac{\partial J(\mathbf{w}, b, \alpha)}{\partial b} = 0$$

Application of optimality condition 1 to the Lagrangian function of Eq. (6.11) yields the following (after the rearrangement of terms):

$$\mathbf{w} = \sum_{i=1}^{N} \alpha_i d_i \mathbf{x}_i \tag{6.12}$$

Application of optimality condition 2 to the Lagrangian function of Eq. (6.11) yields

$$\sum_{i=1}^{N} \alpha_i d_i = 0 \tag{6.13}$$

The solution vector w is defined in terms of an expansion that involves the N training examples. Note, however, that although this solution is unique by virtue of the convexity of the Lagrangian, the same cannot be said about the Lagrange multipliers α_i . It is also important to note that for all the constraints that are not satisfied as equalities, the corresponding multiplier α_i must be zero. In other words, only those multipliers that exactly satisfy the condition:

$$\alpha_i[d_i(\mathbf{w}^T\mathbf{x}_i + b) - 1] = 0 \tag{6.14}$$

can assume nonzero values. This property is a statement of the Karush–Kuhn–Tucker conditions (Fletcher, 1987; Bertsekas, 1995).

As noted earlier, the primal problem deals with a convex cost function and linear constraints. Given such a constrained-optimization problem, it is possible to construct another problem called the dual problem. This second problem has the same optimal value as the primal problem, but with the Lagrange multipliers providing the optimal solution.

In particular, we may state the following duality theorem (Bertsekas, 1995):

(a) If the primal problem has an optimal solution, the dual problem also has an optimal solution, and the corresponding optimal values are equal.
(b) In order for w_0 to be an optimal primal solution and α_0 to be an optimal dual solution, it is necessary and sufficient that w_0 is feasible for the primal problem, and

$$\Phi(\mathbf{w}_o) = J(\mathbf{w}_o, b_o, \alpha_o) = \min_{\mathbf{w}} J(\mathbf{w}, b, \alpha)$$

To postulate the dual problem for our primal problem, we first expand Eq. (6.11), term by term, obtaining

$$J(\mathbf{w}, b, \alpha) = \frac{1}{2}\mathbf{w}^T\mathbf{w} - \sum_{i=1}^{N} \alpha_i d_i \mathbf{w}^T \mathbf{x}_i - b\sum_{i=1}^{N} \alpha_i d_i + \sum_{i=1}^{N} \alpha_i \tag{6.15}$$

The third term on the right-hand side of Eq. (6.15) is zero by virtue of the optimality condition of Eq. (6.13). Furthermore, from Eq. (6.12), we have

Accordingly, setting the objective function J(w, b, α) = Q(α), we may reformulate Eq. (6.15) as

$$Q(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j d_i d_j \mathbf{x}_i^T \mathbf{x}_j$$

where the α_i are all nonnegative. Note that we have changed the notation from J(w, b, α) to Q(α) so as to reflect the transformation from the primal optimization problem to its dual.

We may now state the dual problem as follows: Given the training sample $T = \{x_i, d_i\}_{i=1}^{N}$, find the Lagrange multipliers $\{\alpha_i\}_{i=1}^{N}$ that maximize the objective function

$$Q(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j d_i d_j \mathbf{x}_i^T \mathbf{x}_j$$

subject to the constraints

$$(1) \quad \sum_{i=1}^{N} \alpha_i d_i = 0$$

$$(2) \quad \alpha_i \geq 0 \qquad \text{for } i = 1, 2, \ldots, N$$

Unlike the primal optimization problem based on the Lagrangian of Eq. (6.11), the dual problem defined in Eq. (6.16) is cast entirely in terms of the training data. Moreover, the function Q(α) to be maximized depends only on the input patterns in the form of a set of dot products

$$\{\mathbf{x}_i^T \mathbf{x}_j\}_{i,j=1}^N$$

Typically, the support vectors constitute a subset of the training sample, which means that the solution vector is sparse That is to say, constraint (2) of the dual problem is satisfied with the inequality sign for all the support vectors for which the α's are nonzero, and with the equality sign for all the other data points in the training sample, for which the α's are all zero. Accordingly, having determined the optimum Lagrange multipliers, denoted by $\alpha_{o,i}$, we may compute the optimum weight vector w_o by using Eq. (6.12) as

$$\mathbf{w}_o = \sum_{i=1}^{N_s} \alpha_{o,i} d_i \mathbf{x}_i \tag{6.17}$$

where N_s is the number of support vectors for which the Lagrange multipliers $\alpha_{o,i}$ are all nonzero. To compute the optimum bias b_o, we may use the w_o thus obtained and take advantage of Eq. (6.7), which pertains to a positive support vector:

$$\begin{aligned} b_o &= 1 - \mathbf{w}_o^T \mathbf{x}^{(s)} \qquad \text{for } d^{(s)} = 1 \\ &= 1 - \sum_{i=1}^{N_s} \alpha_{o,i} d_i \mathbf{x}_i^T \mathbf{x}^{(s)} \end{aligned} \tag{6.18}$$

Recall that the support vector $\mathbf{x}^{(s)}$ corresponds to any point $(\mathbf{x}_i, \mathbf{d}_i)$ in the training sample for which the Lagrange multiplier $\boldsymbol{\alpha}_{o,i}$ is nonzero. From a numerical (practical) perspective, it is better to average Eq. (6.18) over all the support vectors—that is, over all the nonzero Lagrange multipliers.

Statistical Properties of the Optimal Hyperplane

In a support vector machine, a structure is imposed on the set of separating hyperplanes by constraining the Euclidean norm of the weight vector **w**. Specifically, we may state the following theorem (Vapnik, 1995, 1998):

Let D denote the diameter of the smallest ball containing all the input **vectors $\mathbf{x}_i$, $\mathbf{x}_2..\mathbf{x}_N$.** The set of optimal hyperplanes described by the equation

$$\mathbf{w}_o^T \mathbf{x} + b_o = 0$$

has a VC dimension, h, bounded from above as

$$h \le \min\left\{\left\lceil \frac{D^2}{\rho^2} \right\rceil, m_0\right\} + 1 \tag{6.19}$$

where the ceiling sign means the smallest integer greater than or equal to the number enclosed within the sign, & ρ is the margin of separation equal to $2/\|w_o\|$ and m_o is the dimensionality of the input space.

As mentioned previously in Chapter 4, the VC dimension, short for Vapnik–

Chervonenkis dimension, provides a measure of the complexity of a space of functions. The theorem just stated tells us that we may exercise control over the VC dimension (i.e., complexity) of the optimal hyperplane, independently of the dimensionality m0 of the input space, by properly choosing the margin of separation ρ.

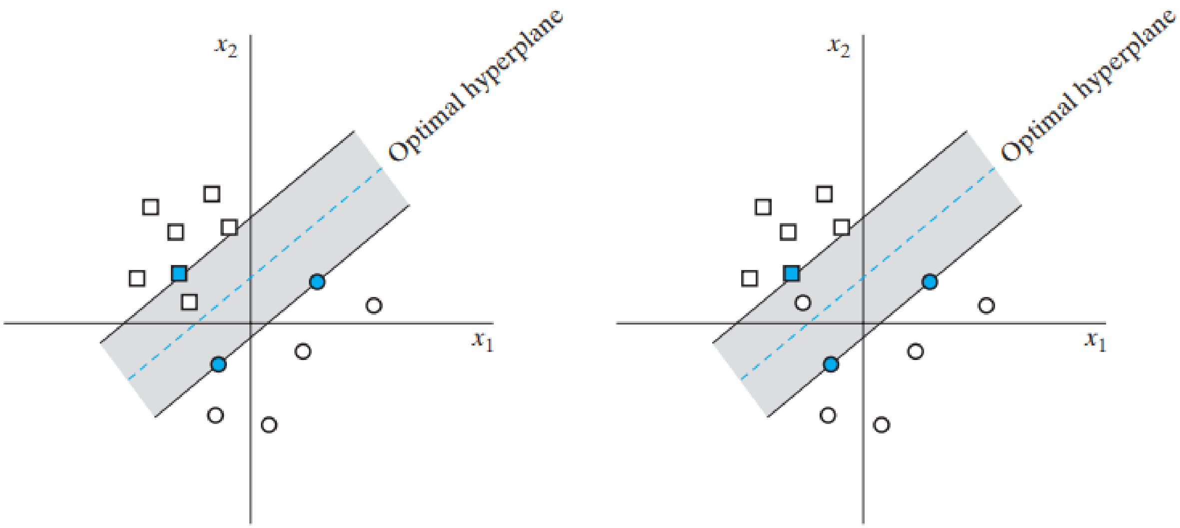
Case 2) OPTIMAL HYPERPLANE FOR NONSEPARABLE PATTERNS

In this section, we consider the more difficult case of nonseparable patterns. Given such a sample of training data, it is not possible to construct a separating hyperplane without encountering classification errors. Nevertheless, we would like to find an optimal hyperplane that minimizes the probability of classification error, averaged over the training sample. The margin of separation between classes is said to be soft if a data point (xi , di) violates the following condition (see Eq. (6.10)):

$$d_i(\mathbf{w}^T\mathbf{x}_i + b) \geq +1, \qquad i = 1, 2, \ldots, N$$

This violation can arise in one of two ways:

• The data point (xi , di) falls inside the region of separation, but on the correct side of the decision surface, as illustrated in Fig. 6.3a.
• The data point (xi , di) falls on the wrong side of the decision surface, as illustrated in Fig. 6.3b.



Soft margin hyperplane (a) Data point $\mathbf{x}_i$ (belonging to class C_1 , represented by a small square) falls inside the region of separation, but on the correct side of the decision surface. (b) Data point $\mathbf{x}_i$ (belonging to class C_2 , represented by a small circle) falls on the wrong side of the decision surface.

Note that we have correct classification in the first case, but misclassification in the second. To set the stage for a formal treatment of nonseparable data points, we introduce a new set of nonnegative scalar variables $\{ \boldsymbol{\varepsilon}_i \}^{N}_{i=1}$, into the definition of the separating hyperplane (i.e., decision surface), as shown here:

$$d_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i, \qquad i = 1, 2, \ldots, N$$
(6.22)

The ξ_i are called slack variables; they measure the deviation of a data point from the ideal condition of pattern separability. For **[$0 < \xi_i \leq 1$],** the data point falls inside the region of separation, but on the correct side of the decision surface, as illustrated in Fig. 6.3a. For **[$\xi_i > 1$],** it falls on the wrong side of the separating hyperplane, as illustrated in Fig. 6.3b. The support vectors are those particular data points that satisfy Eq. (6.22) precisely even if **[$\xi_i > 0$].** Moreover, there can be support vectors satisfying the condition **[$\xi_i = 0$].** Note that if an example with [$\xi_i > 0$] is left out of the training sample, the decision surface will change. The support vectors are thus defined in exactly the same way for both linearly separable and non-separable cases.

Our goal is to find a separating hyperplane for which the misclassification error, averaged over the training sample, is minimized. We may do this by minimizing the functional

$$\Phi(\xi) = \sum_{i=1}^{N} I(\xi_i - 1)$$

with respect to the weight vector w, subject to the constraint described in Eq. (6.22) and the constraint on $\|w\|^2$. The function $I(\xi)$ is an indicator function, defined by

$$I(\xi) = \begin{cases} 0 & \text{if } \xi \leq 0 \\ 1 & \text{if } \xi > 0 \end{cases}$$

Unfortunately, minimization of $\boldsymbol{\Phi(\xi)}$ with respect to **w** is a nonconvex optimization problem that is NP complete. To make the optimization problem mathematically tractable, we approximate the functional $\Phi(\xi)$ by writing

$$\Phi(\xi) = \sum_{i=1}^{N} \xi_i$$

Moreover, we simplify the computation by formulating the functional to be minimized with respect to the weight vector w as follows:

$$\Phi(\mathbf{w}, \xi) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{N} \xi_i$$
(6.23)

As before, minimizing the first term in Eq. (6.23) is related to the support vector machine. As for the second term $\sum_i \xi_i$ it is an upper bound on the number of test errors. The parameter C controls the tradeoff between complexity of the machine and the number of nonseparable points; it may therefore be viewed as the reciprocal of a parameter commonly referred to as the "regularization" parameter. When the parameter C is assigned a large value, the implication is that the designer of the support vector machine has high confidence in the quality of the training sample T. Conversely, when C is assigned a small value, the training sample T is considered to be noisy, and less emphasis should therefore be placed on it.
In any event, the parameter C has to be selected by the user. It may be determined

experimentally via the standard use of a training (validation) sample, which is a crude form of resampling; the use of cross-validation for optimum selection of regularization parameter (i.e., 1/C).

In any event, the functional $\Phi(\mathbf{w}, \xi)$ is optimized with respect to **w** and $\{\xi_i\}^{N}_{i=1}$, subject to the constraint described in Eq. (6.22), and $\xi_i \geq 0$. In so doing, the squared norm of w is treated as a quantity to be jointly minimized with respect to the nonseparable points rather than as a constraint imposed on the minimization of the number of nonseparable points. The optimization problem for nonseparable patterns just stated includes the optimization problem for linearly separable patterns as a special case. Specifically, setting $\xi_i = 0$ for all i in both Eqs. (6.22) and (6.23) reduces them to the corresponding forms for the linearly separable case.

We may now formally state the primal problem for the nonseparable case as follows:

$$d_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i \qquad \text{for } i = 1, 2, ..., N \tag{6.24}$$

$$\xi_i \geq 0 \qquad \text{for all } i \tag{6.25}$$

and such that the weight vector w and the slack variables i minimize the cost functional

$$\Phi(\mathbf{w}, \xi) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{N}\xi_i \tag{6.26}$$

where C is a user-specified positive parameter

Using the method of Lagrange multipliers and proceeding in a manner similar to that described in Section 6.2, we may formulate the dual problem for nonseparable patterns as follows (see Problem 6.3):

Given the training sample $\{(\mathbf{x}_i, d_i)\}_{i=1}^{N}$, find the Lagrange multipliers $\{\alpha_i\}_{i=1}^{N}$ that maximize the objective function

$$Q(\alpha) = \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j d_i d_j \mathbf{x}_i^T\mathbf{x}_j \tag{6.27}$$

subject to the constraints

$$(1) \quad \sum_{i=1}^{N}\alpha_i d_i = 0$$

$$(2) \quad 0 \leq \alpha_i \leq C \qquad \text{for } i = 1, 2, ..., N$$

where C is a user-specified positive parameter.

Note that neither the slack variables $\boldsymbol{\xi}_i$ nor their own Lagrange multipliers appear in the dual problem. The dual problem for the case of nonseparable patterns is thus similar to that for the simple case of linearly separable patterns, except for a minor, but important, difference.The objective function **Q(α)** to be maximized is the same in both cases. The nonseparable case differs from the separable case in that the constraint $\boldsymbol{\alpha_i \geq 0}$ is replaced with the more stringent **constraint** $\boldsymbol{0 \leq \alpha_i \leq C}$. Except for this modification, the constrained optimization for the nonseparable case and computations of the optimum values of the weight vector w and bias b proceed in the same way as in the linearly separable case. Note also that the support vectors are defined in exactly the same way as before.

Case 3) THE SUPPORT VECTOR MACHINE VIEWED AS A KERNEL MACHINE

Inner-Product Kernel

Let x denote a vector drawn from the input space of dimension m_0. Let $\{\varphi_j(\mathbf{x})\}_{j=1}^{\infty}$ denote a set of nonlinear functions that, between them, transform the input space of dimension m0 to a feature space of infinite dimensionality. Given this transformation, we may define a hyperplane acting as the decision surface in accordance with the formula

$$\sum_{j=1}^{\infty} w_j \varphi_j(\mathbf{x}) = 0 \tag{6.28}$$

where $\{w_j\}_{j=1}^{\infty}$ denotes an infinitely large set of weights that transforms the feature space to the output space. It is in the output space where the decision is made on whether the input vector x belongs to one of two possible classes, positive or negative. Using matrix notation, we may rewrite this equation in the compact form

$$\mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}) = 0 \tag{6.29}$$

where $\boldsymbol{\phi}(\mathbf{x})$ is the feature vector and **w** is the corresponding weight vector.
As in Section 6.3,we seek “linear separability of the transformed patterns”in the feature space.With this objective in mind,we may adapt Eq.(6.17) to our present situation by expressing the weight vector as

$$\mathbf{w} = \sum_{i=1}^{N_s} \alpha_i d_i \boldsymbol{\phi}(\mathbf{x}_i) \tag{6.30}$$

where the feature vector is expressed as

$$\boldsymbol{\phi}(\mathbf{x}_i) = [\varphi_1(\mathbf{x}_i), \varphi_2(\mathbf{x}_i), \ldots]^T \tag{6.31}$$

and N_s is the number of support vectors. Hence, substituting Eq.(6.29) into Eq.(6.30), we may express the decision surface in the output space as

$$\sum_{i=1}^{N_s} \alpha_i d_i \boldsymbol{\phi}^T(\mathbf{x}_i) \boldsymbol{\phi}(\mathbf{x}) = 0 \tag{6.32}$$

We now immediately see that the scalar term $\boldsymbol{\phi}^T(\mathbf{x}_i)\boldsymbol{\phi}(\mathbf{x})$ in Eq. (6.32) represents an inner product. Accordingly, let this inner-product term be denoted as the scalar

$$k(\mathbf{x}, \mathbf{x}_i) = \boldsymbol{\phi}^T(\mathbf{x}_i)\boldsymbol{\phi}(\mathbf{x})$$

$$= \sum_{j=1}^{\infty} \varphi_j(\mathbf{x}_i)\varphi_j(\mathbf{x}), \qquad i = 1, 2, ..., N_s \tag{6.33}$$

Correspondingly, we may express the optimal decision surface (hyperplane) in the output space as

$$\sum_{i=1}^{N_s} \alpha_i d_i k(\mathbf{x}, \mathbf{x}_i) = 0 \tag{6.34}$$

The function $k(\mathbf{x}, \mathbf{x}_i)$ is called the inner-product kernel, or simply the kernel which is formally defined as follows:

The kernel $k(\mathbf{x}, \mathbf{x}_i)$ is a function that computes the inner product of the images produced in the feature space under the embedding Φ of two data points in the input space.

The kernel $\mathbf{k(x, x_i)}$ is a function that has two basic properties –

Property 1. *The function $k(\mathbf{x}, \mathbf{x}_i)$ is symmetric about the center point $\mathbf{x}_i$, that is,*

$$k(\mathbf{x}, \mathbf{x}_i) = k(\mathbf{x}_i, \mathbf{x}) \qquad \text{for all } \mathbf{x}_i$$

and it attains its maximum value at the point $\mathbf{x} = \mathbf{x}_i$.

Property 2. *The total volume under the surface of the function $k(\mathbf{x}, \mathbf{x}_i)$ is a constant.*

The Kernel Trick

Examining Eq.(6.34),we may now make two important observations:

1. Insofar as pattern classification in the output space is concerned, specifying the kernel $\mathbf{k(x, x_i)}$ is sufficient; in other words,we need never explicitly compute the weight vector $\mathbf{w_o}$; it is for this reason that the application of Eq. (6.33) is commonly referred to as the *kernel trick*.

2. Even though we assumed that the feature space could be of infinite dimensionality, the linear equation of Eq.(6.34),defining the optimal hyperplane, consists of a finite number of terms that is equal to the number of training patterns used in the classifier.

It is in light of observation 1 that the support vector machine is also referred to as a kernel machine. For pattern classification, the machine is parameterized by an N-dimensional vector whose ith term is defined by the product $\alpha_i d_i$ for i = 1, 2, ..., N.

We may view $k(\mathbf{x}_i, \mathbf{x}_j)$ as the ij-th element of the symmetric N-by-N matrix

$$\mathbf{K} = \{k(\mathbf{x}_i, \mathbf{x}_j)\}_{i,j=1}^{N} \tag{6.35}$$

The matrix **K** is a nonnegative definite matrix called the kernel matrix; it is also referred to simply as the *Gram*. It is nonnegative definite or positive semidefinite in that it satisfies the condition

$$\mathbf{a}^T\mathbf{K}\mathbf{a} \geq 0$$

for any real-valued vector a whose dimension is compatible with that of **K**.

Mercer's Theorem

Let $k(\mathbf{x}, \mathbf{x}')$ be a continuous symmetric kernel that is defined in the closed interval $\mathbf{a} \le \mathbf{x} \le \mathbf{b}$, and likewise for $\mathbf{x}'$. The kernel $k(\mathbf{x}, \mathbf{x}')$ can be expanded in the series

$$k(\mathbf{x}, \mathbf{x}') = \sum_{i=1}^{\infty} \lambda_i \varphi_i(\mathbf{x}) \varphi_i(\mathbf{x}') \tag{6.36}$$

with positive coefficients $\lambda_i > 0$ for all i. For this expansion to be valid and for it to converge absolutely and uniformly, it is necessary and sufficient that the condition

$$\int_{\mathbf{b}}^{\mathbf{a}} \int_{\mathbf{b}}^{\mathbf{a}} k(\mathbf{x}, \mathbf{x}') \psi(\mathbf{x}) \psi(\mathbf{x}') d\mathbf{x} d\mathbf{x}' \ge 0 \tag{6.37}$$

holds for all $\psi(\cdot)$, for which we have

$$\int_{\mathbf{b}}^{\mathbf{a}} \psi^2(\mathbf{x}) d\mathbf{x} < \infty \tag{6.38}$$

where $\mathbf{a}$ and $\mathbf{b}$ are the constants of integretion.

The features $\varphi_i(\mathbf{x})$ are called eigenfunctions of the expansion, and the numbers λ_i are called eigenvalues. The fact that all of the eigenvalues are positive means that the kernel **k(x, x')** is positive definite. This property, in turn, means that we have a complex problem that can be solved efficiently for the weight vector **w**, as discussed next.
Note, however, that Mercer's theorem tells us only whether a candidate kernel is actually an inner-product kernel in some space and therefore admissible for use in a support vector machine. It says nothing about how to construct the functions $\varphi_i(\mathbf{x})$; we have to do that ourselves. Nevertheless, Mercer's theorem is important because it places a limit on the number of admissible kernels. Note also that the expansion of Eq.(6.33) is a special case of Mercer's theorem, since all the eigenvalues of this expansion are unity. It is for this reason that an inner-product kernel is also referred to as a Mercer kernel.

The expansion of the kernel **k(x, x$_i$)** in Eq.(6.33) permits us to construct a decision surface that is nonlinear in the input space, but whose image in the feature space is linear. With this expansion at hand, we may now state the dual form for the constrained optimization of a support vector machine as follows:

Given the training sample $\{(\mathbf{x}_i, d_i)\}_{i=1}^{N}$, find the Lagrange multipliers $\{\alpha_i\}_{i=1}^{N}$ that maximize the objective function

$$Q(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j d_i d_j k(\mathbf{x}_i, \mathbf{x}_j) \tag{6.39}$$

subject to the constraints

$$(1) \quad \sum_{i=1}^{N} \alpha_i d_i = 0$$

$$(2) \quad 0 \le \alpha_i \le C \qquad \text{for } i = 1, 2, \ldots, N$$

where C is a user-specified positive parameter.

Constraint (1) arises from optimization of the Lagrangian $Q(\alpha)$ with respect to the bias **b**.

The requirement on the kernel **k(x, x$_i$)** is to satisfy Mercer's theorem. Within this requirement, there is some freedom in how the kernel is chosen. Summary of three common types of support vector machines are given below-

Type of support vector machine	Mercer kernel $k(\mathbf{x}, \mathbf{x}_i)$, $i = 1, 2, \ldots, N$	Comments
Polynomial learning machine	$(\mathbf{x}^T\mathbf{x}_i + 1)^p$	Power p is specified *a priori* by the user
Radial-basis-function network	$\exp\left(-\frac{1}{2\sigma^2}\|\mathbf{x} - \mathbf{x}_i\|^2\right)$	The width σ^2, common to all the kernels, is specified *a priori* by the user
Two-layer perceptron	$\tanh(\beta_0\mathbf{x}^T\mathbf{x}_i + \beta_1)$	Mercer's theorem is satisfied only for some values of β_0 and β_1

The following points are noteworthy:

1. The Mercer kernels for polynomial and radial-basis-function types of support vector machines always satisfy Mercer's theorem. In contrast, the Mercer kernel for a two-layer perceptron type of support vector machine is somewhat restricted, as indicated in the last row. This latter entry is a testament to the fact that the determination of whether a given kernel satisfies Mercer's theorem can indeed be a difficult matter.
2. For all three machine types, the dimensionality of the feature space is determined by the number of support vectors extracted from the training data by the solution to the constrained-optimization problem.
3. The underlying theory of a support vector machine avoids the need for heuristics often used in the design of conventional radial-basis-function networks and multilayer perceptron.
4. In the radial-basis-function type of a support vector machine, the number of radial basis functions and their centers are determined automatically by the number of support vectors and their values, respectively.