CSE 5526: Introduction to Neural Networks

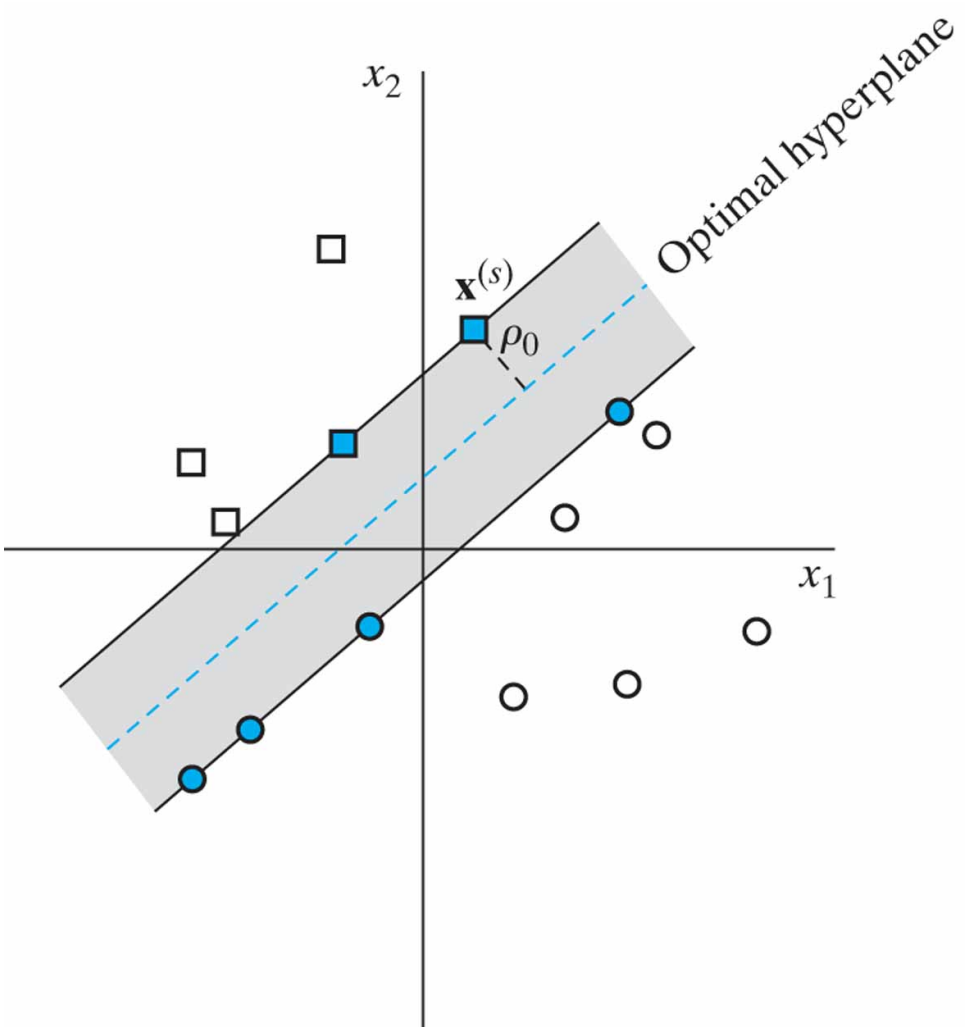
Support Vector Machines (SVM)

Motivation

- For a linearly separable classification task, there are generally infinitely many separating hyperplanes. Perceptron learning, however, stops as soon as one of them is reached

- To improve generalization, we want to place a decision boundary as far away from training classes as possible. In other words, place the boundary at equal distances from class boundaries

Optimal hyperplane

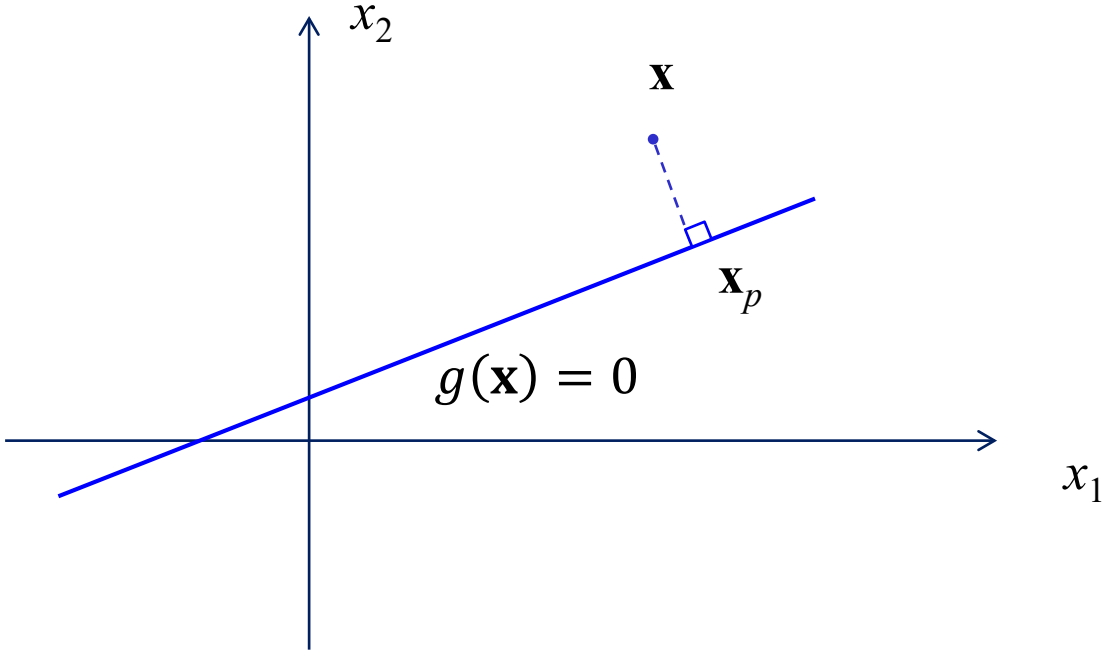


Decision boundary

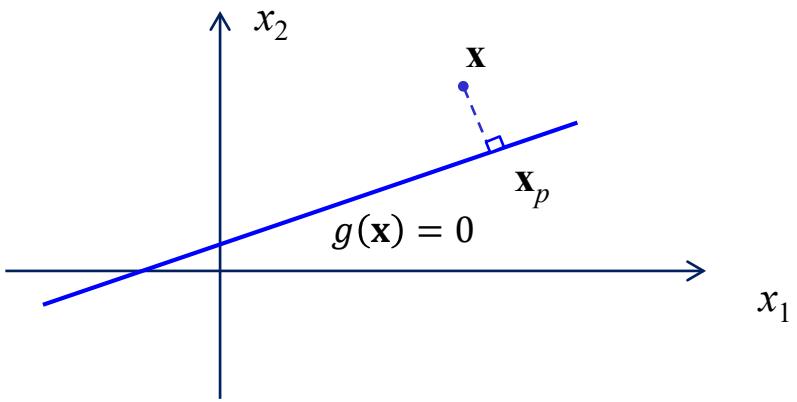
- Given a linear discriminant function

$$g(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + b = 0$$

- To find its distance to a given pattern **x**, project **x** onto the decision boundary:



Decision boundary (cont.)



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

where $\mathbf{x}_p$ is $\mathbf{x}$'s projection and the second term arises from the fact that the weight vector is perpendicular to the decision boundary. The algebraic distance r is positive if $\mathbf{x}$ is on the positive side of the boundary and negative if $\mathbf{x}$ is on the negative side

Decision boundary (cont.)

Then

$$
\begin{aligned}
g(\mathbf{x}) &= g\left(\mathbf{x}_p + r\frac{\mathbf{w}}{||\mathbf{w}||}\right) \\
&= \mathbf{w}^T\left(\mathbf{x}_p + r\frac{\mathbf{w}}{||\mathbf{w}||}\right) + b \\
&= \mathbf{w}^T\mathbf{x}_p + b + r||\mathbf{w}|| \\
&= r||\mathbf{w}||
\end{aligned}
$$

Decision boundary (cont.)

Thus

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

- As a special case, for the origin, $r = \frac{b}{||\mathbf{w}||}$, as discussed before

Margin of separation

- The margin of separation is the smallest distance of the hyperplane to a data set. Equivalently, the margin is the distance to the nearest data points
- The training patterns closest to the optimal hyperplane are called *support vectors*

Finding optimal hyperplane for linearly separable problems

- Question: Given N pairs of input and desired output $< \mathbf{x}_i, d_i >$, how to find $\mathbf{w}_o$ and b_o for the optimal hyperplane?
- Without loss of generality, $\mathbf{w}_o$ and b_o must satisfy

$$\mathbf{w}_o{}^T \mathbf{x}_i + b_o \geq 1 \quad \text{for } d_i = 1$$
$$\mathbf{w}_o{}^T \mathbf{x}_i + b_o \leq -1 \quad \text{for } d_i = -1$$

or $$d_i(\mathbf{w}_o{}^T \mathbf{x}_i + b_o) \geq 1$$

where the equality holds for support vectors only

Optimal hyperplane

- For a support vector $\mathbf{x}^{(s)}$, its algebraic distance to the optimal hyperplane:

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

- Thus the margin of separation:

$$|r| = \frac{1}{||\mathbf{w}_o||}$$

- In other words, maximizing the margin of separation is equivalent to minimizing $||\mathbf{w}_o||$

Primal problem

- Therefore $\mathbf{w}_o$ and b_o satisfy

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

$$\text{and } \Phi(\mathbf{w}) = \frac{1}{2}\mathbf{w}^T\mathbf{w} \text{ is minimized}$$

  - The above constrained minimization problem is called the *primal* problem

Lagrangian formulation

- Using Lagrangian formulation, we construct the Lagrangian function:

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

where nonnegative variables, α_i's, are called Lagrange multipliers

Lagrangian formulation (cont.)

- The solution is a saddle point, minimized with respect to $\mathbf{w}$ and b, but maximized with respect to α

Condition 1:

$$\frac{\partial J(\mathbf{w}, b, \alpha)}{\partial \mathbf{w}} = \mathbf{0}$$

Condition 2:

$$\frac{\partial J(\mathbf{w}, b, \alpha)}{\partial b} = 0$$

Lagrangian formulation (cont.)

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

- From condition 1:

$$\mathbf{w} - \sum_i \alpha_i d_i \mathbf{x}_i = \mathbf{0}$$

or $\mathbf{w} = \sum_i \alpha_i d_i \mathbf{x}_i$

- From condition 2:

$$\sum_i \alpha_i d_i = 0$$

Karush-Kuhn-Tucker conditions

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

- **Remark**: The above constrained optimization problem satisfies:

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

called Karush-Kuhn-Tucker conditions

- In other words, $\alpha_i = 0$ when $d_i(\mathbf{w}^T\mathbf{x}_i + b) - 1 > 0$
- α_i can be greater than 0 only when $d_i(\mathbf{w}^T\mathbf{x}_i + b) - 1 = 0$, that is, for support vectors

How to find α?

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

- The primal problem has a corresponding dual problem in terms of α. From the Lagrangian

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

How to find α (cont.)

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

- Because of the two conditions, the third term is zero and

$$\mathbf{w}^T\mathbf{w} = \sum_i \alpha_i d_i \mathbf{w}^T\mathbf{x}_i$$

Hence

$$Q(\alpha) = \sum_i \alpha_i - \frac{1}{2}\sum_i\sum_j \alpha_i\alpha_j d_i d_j \mathbf{x}_i{}^T\mathbf{x}_j$$

Dual problem

- The *dual* problem is stated as follows:

  The Lagrange multipliers maximize

$$Q(\alpha) = \sum_i \alpha_i - \frac{1}{2}\sum_i \sum_j \alpha_i \alpha_j d_i d_j \mathbf{x}_i{}^T \mathbf{x}_j$$

  subject to

$$(1)\ \textstyle\sum_i \alpha_i d_i = 0$$

$$(2)\ \alpha_i \geq 0$$

  - The dual problem can be solved as a quadratic optimization problem. Note that $\alpha_i > 0$ only for support vectors

Solution

- Having found optimal multipliers, $\alpha_{o,i}$

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

where N_s is the number of support vectors

Solution (cont.)

- For any support vector $\mathbf{x}^{(s)}$, we have

$$d^{(s)}\left(\mathbf{w}_o^T\mathbf{x}^{(s)} + b_o\right) = 1$$

$$b_o = \frac{1}{d^{(s)}} - \mathbf{w}_o^T\mathbf{x}^{(s)} = \frac{1}{d^{(s)}} - \sum_{i=1}^{N_s} \alpha_{o,i} d_i \mathbf{x}_i^T\mathbf{x}^{(s)}$$

  - Note that for robustness, one should average over all support vectors to compute b_o

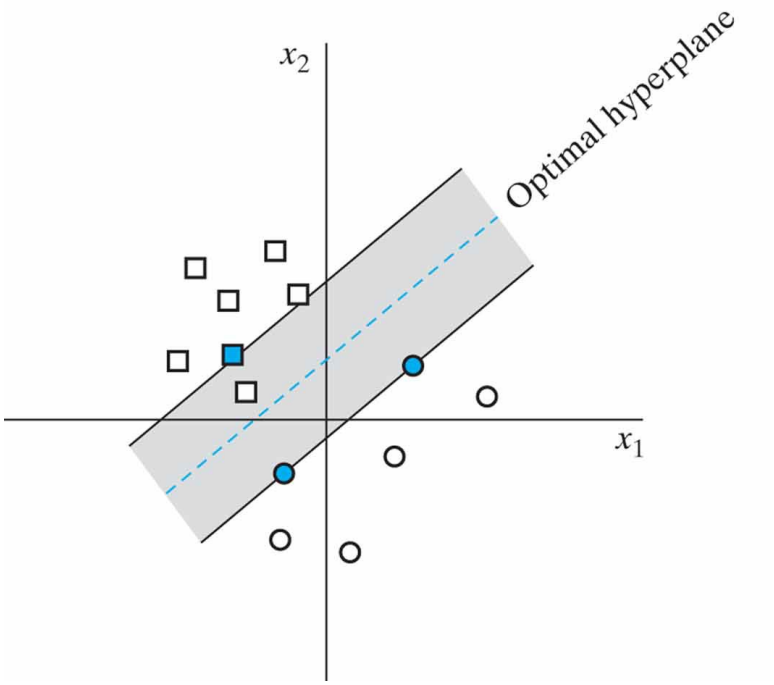
Linearly inseparable problems

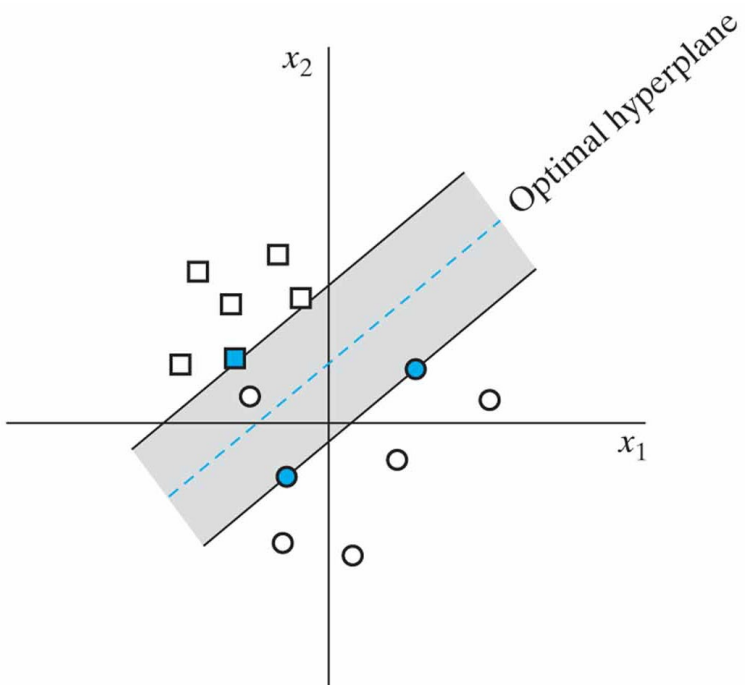
- How to find optimal hyperplanes for linearly inseparable cases?
- For such problems, the condition

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

is violated. In such cases, the margin of separation is called soft.

Linearly inseparable cases





Slack variables

- To extend the constrained optimization framework, we introduce a set of nonnegative variables, $\xi_i (i = 1, \ldots, N)$, called slack variables, into the condition:

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

  - For $0 < \xi_i \leq 1$, $\mathbf{x}_i$ falls into the region of separation, but on the correct side of the decision boundary
  - For $\xi_i > 1$, $\mathbf{x}_i$ falls on the wrong side
  - The equality holds for support vectors, no matter whether $\xi_i > 0$ or $\xi_i = 0$. Thus, linearly separable problems can be treated as a special case

Classification error

- In general, the goal is to minimize the classification error:

$$\Phi(\xi) = \sum_i I(\xi_i - 1)$$

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

Classification error

- To turn the above problem into a convex optimization problem with respect to **w** and b, we minimize instead

$$\Phi(\xi) = \sum_i \xi_i$$

Classification error (cont.)

- Adding the term to the minimization of $\|\mathbf{w}\|$, we have

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

  - The parameter C controls the tradeoff between minimizing the classification error and maximizing the margin of separation. C has to be chosen by the user, reflecting the confidence on the training sample

Primal problem for linearly inseparable case

- The primal problem becomes:

  Find optimal $\mathbf{w}_o$ and b_o so that

$$d_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i \quad \text{for } i = 1, \ldots, N$$
$$\xi_i \geq 0$$

  and

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

  is minimized

Lagrangian formulation

- Again using Lagrangian formulation,

$$J(\mathbf{w}, b, \xi, \alpha, \mu)$$
$$= \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_i \xi_i - \sum_i \alpha_i\left[d_i(\mathbf{w}^T\mathbf{x}_i + b) - 1 + \xi_i\right]$$
$$- \sum_i \mu_i\xi_i$$

with nonnegative Lagrange multipliers α_i's and μ_i's

Dual problem for linearly inseparable case

- By a similar derivation, the dual problem is to find α_i's to maximize

$$Q(\alpha) = \sum_i \alpha_i - \frac{1}{2}\sum_i \sum_j \alpha_i \alpha_j d_i d_j \mathbf{x}_i{}^T \mathbf{x}_j$$

subject to

$$(1)\ \textstyle\sum_i \alpha_i d_i = 0$$

$$(2)\ 0 \le \alpha_i \le C$$

Solution

- The dual problem contains neither ξ_i nor μ_i, and is the same as for the linearly separable case, except for the more stringent constraint $0 \leq \alpha_i \leq C$. So it can be solved as a quadratic optimization problem

Solution (cont.)

- With optimal multipliers found,

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

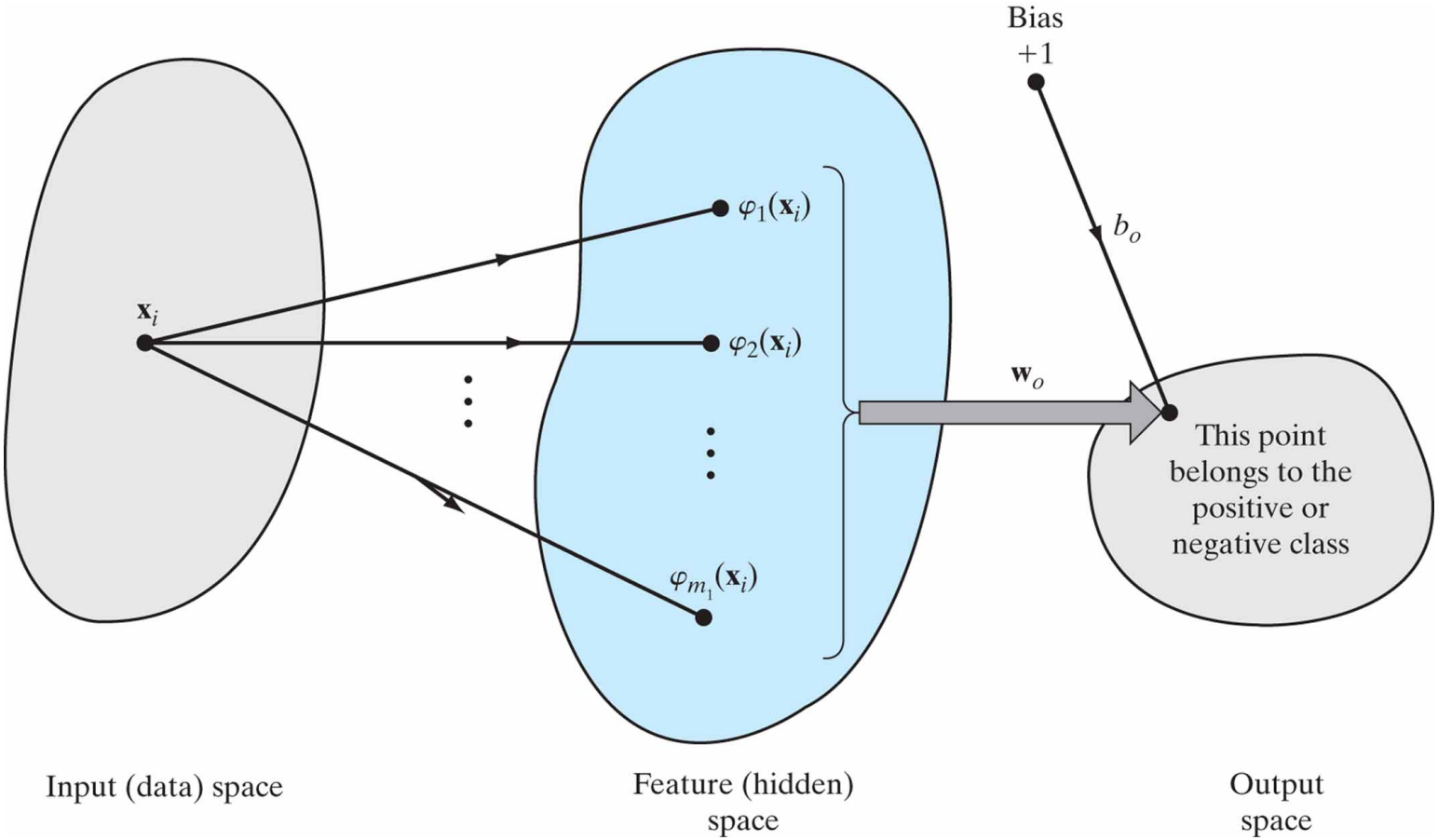
- Due to the Karush-Kuhn-Tucker conditions, for any $0 < \alpha_i < C$, ξ_i must be zero, corresponding to a support vector. Hence b_o can be computed in the same way as for linearly separable cases for such α_i

SVM as a kernel machine

- **Cover's theorem**: A complex classification problem, cast in a high-dimensional space nonlinearly, is more likely to be linearly separable than in the low-dimensional input space

- SVM for pattern classification
  1. Nonlinear mapping of the input space onto a high-dimensional feature space
  2. Constructing the optimal hyperplane for the feature space

Kernel machine illustration



Inner product kernel

- Let $\varphi_j(\mathbf{x}), j = 1, \ldots, \infty,$ denote a set of nonlinear mapping functions onto the feature space
- Without loss of generality, set $b = 0$. For a given weight vector $\mathbf{w}^T$, the discriminant function is

$$\sum_{j=1}^{\infty} w_j \varphi_j(\mathbf{x}) = \mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}) = 0$$

Inner product kernel (cont.)

- Treating the feature space as input to an SVM, we have

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

- Then the optimal hyperplane becomes

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

Optimal hyperplane

- Denote the inner product of the mapping functions as

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

$$= \sum_{j=1}^{\infty} \varphi_j(\mathbf{x}_i)\varphi_j(\mathbf{x}), \quad i = 1, \ldots, N$$

- Then we have

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

Kernel trick

- Function $k(\mathbf{x}, \mathbf{x_i})$ is called an inner-product kernel, satisfying the condition $k(\mathbf{x}, \mathbf{x}_i) = k(\mathbf{x}_i, \mathbf{x})$

- Kernel trick: For pattern classification in the output space, specifying the kernel $k(\mathbf{x}, \mathbf{x}_i)$ is sufficient. That is, no need to train $\mathbf{w}$

Kernel matrix

- The matrix

$$\mathbf{K} = \begin{bmatrix} k(\mathbf{x}_1, \mathbf{x}_1) & \cdots & k(\mathbf{x}_1, \mathbf{x}_N) \\ \vdots & & \\ \ldots & k(\mathbf{x}_i, \mathbf{x}_j) & \ldots \\ \vdots & & \\ k(\mathbf{x}_N, \mathbf{x}_1) & \ldots & k(\mathbf{x}_N, \mathbf{x}_N) \end{bmatrix}$$

is called the kernel matrix, or the Gram matrix. **K** is positive, semidefinite

Remarks

- Even though the feature space could be of infinite dimensionality, the optimal hyperplane for classification has a finite number of terms that is equal to the number of support vectors in the feature space
- Mercer's theorem (see textbook) specifies the conditions for a candidate kernel to be an inner-product kernel, admissible for SVM

SVM design

- Given N pairs of input and desired output $< \mathbf{x}_i, d_i >$, find the Lagrange multipliers, $\alpha_1, \alpha_2, \ldots, \alpha_N$, by maximizing the objective function:

$$Q(\alpha) = \sum_i \alpha_i - \frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j d_i d_j k(\mathbf{x}_i, \mathbf{x}_j)$$

subject to

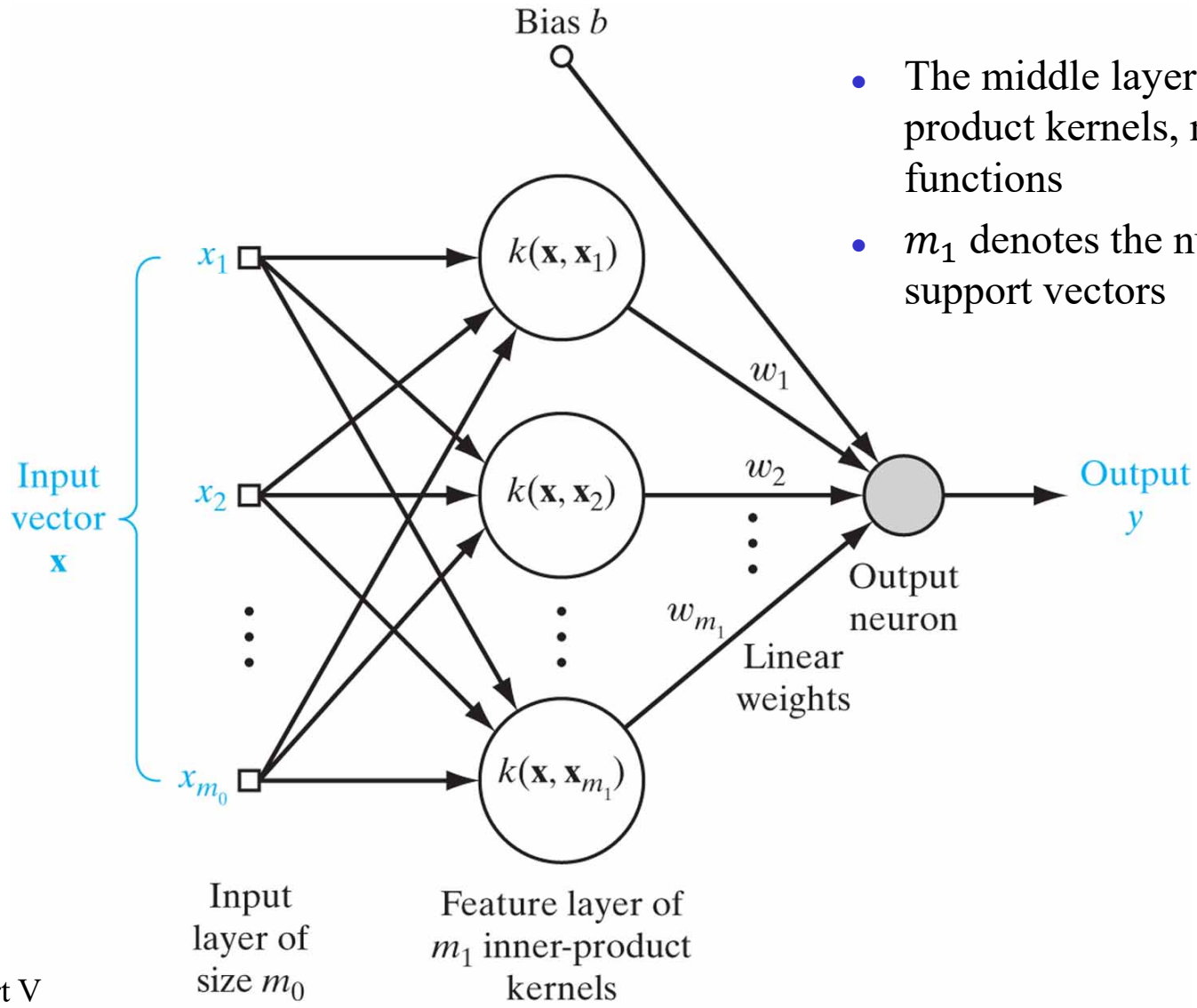
$$(1)\ \textstyle\sum_i \alpha_i d_i = 0$$

$$(2)\ 0 \leq \alpha_i \leq C$$

SVM design (cont.)

- The above dual problem has the same form as for linearly inseparable problems except for the substitution of $k(\mathbf{x}_i, \mathbf{x}_j)$ for $\mathbf{x}_i{}^T \mathbf{x}_j$

Optimal hyperplane



- The middle layer depicts inner-product kernels, not mapping functions
- m_1 denotes the number of support vectors

Typical kernels

1. Polynomial kernel:

$$(\mathbf{x}^T\mathbf{x}_i + 1)^p$$

- p is a parameter

2. RBF kernels:

$$\exp(-\frac{1}{2\sigma^2}||\mathbf{x} - \mathbf{x}_i||^2)$$

- σ is a parameter common to all kernels

Typical kernels (cont.)

3. Hyperbolic tangent kernel (two-layer perceptron):

$$\tanh(\beta_0 \mathbf{x}^T \mathbf{x}_i + \beta_1)$$

- only certain values of β_0 and β_1 satisfy Mercer's theorem

- Note that SVM theory avoids the need for heuristics in RBF and MLP design, and guarantees a measure of optimality

Example: XOR problem again

- See blackboard

TABLE 6.2 XOR Problem

Input vector $\mathbf{x}$	Desired response d
$(-1, -1)$	-1
$(-1, +1)$	$+1$
$(+1, -1)$	$+1$
$(+1, +1)$	-1

SVM summary

- SVM builds on strong theoretical foundations, eliminating the need for much user design
- SVM produces very good results for classification, and is the kernel method of choice
- Computational complexity (both time & memory) increases quickly with the size of the training sample