CHAPTER 6
Support Vector Machines

Problem 6.1

From Eqs. (6.2) in the text we recall that the optimum weight vector $\mathbf{w}_o$ and optimum bias b_o satisfy the following pair of conditions:

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

where $i = 1, 2, ..., N$. Equivalently, we may write

$$\min_{i\ =\ 1, 2, \ldots, N} \left|\mathbf{w}^T\mathbf{x}_i + b\right| = 1$$

as the defining condition for the pair ($\mathbf{w}_o$, b_o).

Problem 6.2

In the context of a support vector machine, we note the following:

1. Misclassification of patterns can only arise if the patterns are nonseparable.
2. If the patterns are nonseparable, it is possible for a pattern to lie inside the margin of separation and yet be on the correct side of the decision boundary. Hence, nonseparability does not necessarily mean misclassification.

Problem 6.3

We start with the primel problem formulated as follows (see Eq. (6.15)) of the text

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

Recall from (6.12) in the text that

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

Premultiplying $\mathbf{w}$ by $\mathbf{w}^T$:

$$\mathbf{w}^T\mathbf{w} = \sum_{i=1}^{N} \alpha_i d_i \mathbf{w}^T \mathbf{x}_i \tag{2}$$

We may also write

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

Accordingly, we may redefine the inner product $\mathbf{w}^T\mathbf{w}$ as the double summation:

$$\mathbf{w}^T\mathbf{w} = \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i d_i \alpha_j d_j \mathbf{x}_j^T \mathbf{x}_i \tag{3}$$

Thus substituting (2) and (3) into (1) yields

$$Q(\alpha) = -\frac{1}{2}\sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i d_i \alpha_j d_j \mathbf{x}_j^T \mathbf{x}_i + \sum_{i=1}^{N} \alpha_i \tag{4}$$

subject to the constraint

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

Recognizing that $\alpha_i > 0$ for all i, we see that (4) is the formulation of the dual problem.

Problem 6.4

Consider a support vector machine designed for nonseparable patterns. Assuming the use of the "leave-one-out-method" for training the machine, the following situations may arise when the example left out is used as a test example:

1. The example is a support vector.
   Result: Correct classification.
2. The example lies inside the margin of separation but on the correct side of the decision boundary.
   Result: Correct classification.
3. The example lies inside the margin of separation but on the wrong side of the decision boundary.
   Result: Incorrect classification.

Problem 6.5

By definition, a support vector machine is designed to maximize the margin of separation between the examples drawn from different classes. This definition applies to all sources of data, be they noisy or otherwise. It follows therefore that by the very nature of it, the support vector machine is robust to the presence of additive noise in the data used for training and testing, provided that all the data are drawn from the same population.

Problem 6.6

Since theGram $\mathbf{K} = \{K(\mathbf{x}_i, \mathbf{x}_j)\}$ is a square matrix, it can be diagonalized using the similarity transformation:

$$\mathbf{K} = \mathbf{Q}\Lambda\mathbf{Q}^T$$

where Λ is a diagonal matrix consisting of the eigenvalues of $\mathbf{K}$ and $\mathbf{Q}$ is an orthogonal matrix whose columns are the associated eigenvectors. With $\mathbf{K}$ being a positive matrix, Λ has nonnegative entries. The inner-product (i.e., Mercer) kernel $k(\mathbf{x}_i, \mathbf{x}_j)$ is the ijth element of matrix $\mathbf{K}$. Hence,

$$\begin{aligned} k(\mathbf{x}_i, \mathbf{x}_j) &= (\mathbf{Q}\Lambda\mathbf{Q}^T)_{ij} \\ &= \sum_{l=1}^{m_1} (\mathbf{Q})_{il}(\Lambda)_{ll}(\mathbf{Q}^T)_{lj} \\ &= \sum_{l=1}^{m_1} (\mathbf{Q})_{il}(\Lambda)_{ll}(\mathbf{Q})_{lj} \qquad (1) \end{aligned}$$

Let $\mathbf{u}_i$ denote the ith row of matrix $\mathbf{Q}$. (Note that $\mathbf{u}_i$ is *not* an eigenvector.) We may then rewrite (1) as the inner product

$$\begin{aligned} k(\mathbf{x}_i, \mathbf{x}_j) &= \mathbf{u}_i^T \Lambda \mathbf{u}_j \\ &= (\Lambda^{1/2}\mathbf{u}_i)^T(\Lambda^{1/2}\mathbf{u}_j) \qquad (2) \end{aligned}$$

where $\Lambda^{1/2}$ is the square root of Λ.

By definition, we have

$$k(\mathbf{x}_i, \mathbf{x}_j) = \phi^T(\mathbf{x}_i)\varphi(\mathbf{x}_j) \qquad (3)$$

Comparing (2) and (3), we deduce that the mapping from the input space to the hidden (feature) space of a support vector machine is described by

$$\varphi:\ \mathbf{x}_i \rightarrow \Lambda^{1/2}\mathbf{u}_i$$

Problem 6.7

(a) From the solution to Problem 6.6, we have

$$\phi:\ \mathbf{x}_i \rightarrow \Lambda^{1/2}\mathbf{u}_i$$

Suppose the input vector $\mathbf{x}_i$ is multiplied by the orthogonal (unitary) matrix $\mathbf{Q}$. We then have a new mapping ϕ' described by

$$\phi':\ \mathbf{Q}\mathbf{x}_i \rightarrow \mathbf{Q}\Lambda^{1/2}\mathbf{u}_i$$

Correspondingly, we may write

$$\begin{aligned} k(\mathbf{Q}\mathbf{x}_i, \mathbf{Q}\mathbf{x}_j) &= (\mathbf{Q}\Lambda^{1/2}\mathbf{u}_i)^T(\mathbf{Q}\Lambda^{1/2}\mathbf{u}_j) \\ &= (\Lambda^{1/2}\mathbf{u}_i)^T\mathbf{Q}^T\mathbf{Q}(\Lambda^{1/2}\mathbf{u}_j) \qquad (1) \end{aligned}$$

where $\mathbf{u}_i$ is the ith row of $\mathbf{Q}$. From the definition of an orthogonal (unitary) matrix:

$$\mathbf{Q}^{-1} = \mathbf{Q}^T$$

or equivalently

$$\mathbf{Q}^T\mathbf{Q} = \mathbf{I}$$

where $\mathbf{I}$ is the identity matrix. Hence, (1) reduces to

$$\begin{aligned} k(\mathbf{Q}\mathbf{x}_i, \mathbf{Q}\mathbf{x}_j) &= (\Lambda^{1/2}\mathbf{u}_i)^T(\Lambda^{1/2}\mathbf{u}_j) \\ &= k(\mathbf{x}_i, \mathbf{x}_j) \end{aligned}$$

In words, the Mercer kernel exhibits the *unitary* invariance property.

(b) Consider first the polynomial machine described by

$$\begin{aligned} k(\mathbf{Qx}_i, \mathbf{Qx}_j) &= ((\mathbf{Qx}_i)^T(\mathbf{Qx}_j) + 1)^p \\ &= (\mathbf{x}_i^T\mathbf{Q}^T\mathbf{Qx}_j + 1)^p \\ &= (\mathbf{x}_i^T\mathbf{x}_j + 1)^p \\ &= k(\mathbf{x}_i, \mathbf{x}_J) \end{aligned}$$

Consider next the RBF network described by the Mercer kernel:

$$\begin{aligned} k(\mathbf{Qx}_i, \mathbf{Qx}_j) &= \exp\left(-\frac{1}{2\sigma^2}\|\mathbf{Qx}_i - \mathbf{Qx}_j\|^2\right) \\ &= \exp\left(-\frac{1}{2\sigma^2}(\mathbf{Qx}_i - \mathbf{Qx}_j)^T(\mathbf{Qx}_i - \mathbf{Qx}_j)\right) \\ &= \exp\left(-\frac{1}{2\sigma^2}(\mathbf{x}_i - \mathbf{x}_j)^T\mathbf{Q}^T\mathbf{Q}(\mathbf{x}_i - \mathbf{x}_j)\right) \\ &= \exp\left(-\frac{1}{2\sigma^2}(\mathbf{x}_i - \mathbf{x}_j)^T(\mathbf{x}_i - \mathbf{x}_j)\right), \qquad \mathbf{Q}^T\mathbf{Q} = \mathbf{I} \\ &= k(\mathbf{x}_i, \mathbf{x}_J) \end{aligned}$$

Finally, consider the multilayer perceptron described by

$$\begin{aligned} k(\mathbf{Qx}_i, \mathbf{Qx}_j) &= \tanh(\beta_0(\mathbf{Qx}_i)^T(\mathbf{Qx}_j) + \beta_1) \\ &= \tanh(\beta_0\mathbf{x}_i^T\mathbf{Q}^T\mathbf{Qx}_j + \beta_1) \\ &= \tanh(\beta_0\mathbf{x}_i^T\mathbf{x}_j + \beta_1) \\ &= k(\mathbf{x}_i, \mathbf{x}_J) \end{aligned}$$

Thus all three types of the support vector machine, namely, the polynomial machine, RBF network, and MLP, satisfy the unitary invariance property in their own individual ways.

Problem 6.17

The truth table for the XOR function, operating on a three-dimensional pattern x, is as follows:

Table 1

Inputs			Desired response
x_1	x_2	x_3	y
+1	+1	+1	+1
+1	-1	+1	-1
-1	+1	+1	-1
+1	+1	-1	-1
+1	-1	-1	+1
-1	+1	-1	+1
-1	-1	-1	-1
-1	-1	+1	+1

To proceed with the support vector machine for solving this multidimensional XOR problem, let the Mercer kernel

$$k(\mathbf{x}, \mathbf{x}_j) = (1 + \mathbf{x}^T \mathbf{x}_i)^p$$

The minimum value of power p (denoting a positive integer) needed for this problem is $p = 3$. For $p = 2$, we end up with a zero weight vector, which is clearly unacceptable.

Setting $p = 3$, we thus have

$$\begin{aligned} k(\mathbf{x}, \mathbf{x}_i) &= (1 + \mathbf{x}^T \mathbf{x}_i)^3 \\ &= 1 + 3\mathbf{x}^T \mathbf{x}_i + 3(\mathbf{x}^T \mathbf{x}_i)^2 + (\mathbf{x}^T \mathbf{x}_i)^3 \end{aligned}$$

where

$$\mathbf{x} = [x_1, x_2, x_3]^T$$

and likewise for $\mathbf{x}_i$. Then, proceeding in a manner similar but much more cumbersome than that described for the two-dimensional XOR problem in Section 6.6, we end up with a polynomial machine defined by

$$y = x_1, x_2, x_3$$

This machine satisfies the entries of Table 1.