MANIPAL UNIVERSITY
JAIPUR

INSPIRED BY LIFE

DATA SCIENCE & MACHINE LEARNING

CS-3203

LECTURE-19-20: SUPPORT VECTOR MACHINE

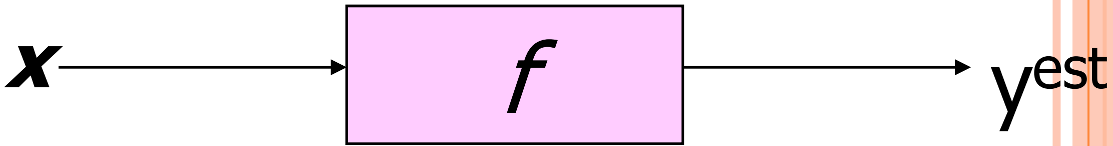
(METHODS & EXAMPLES)

Spring 2023

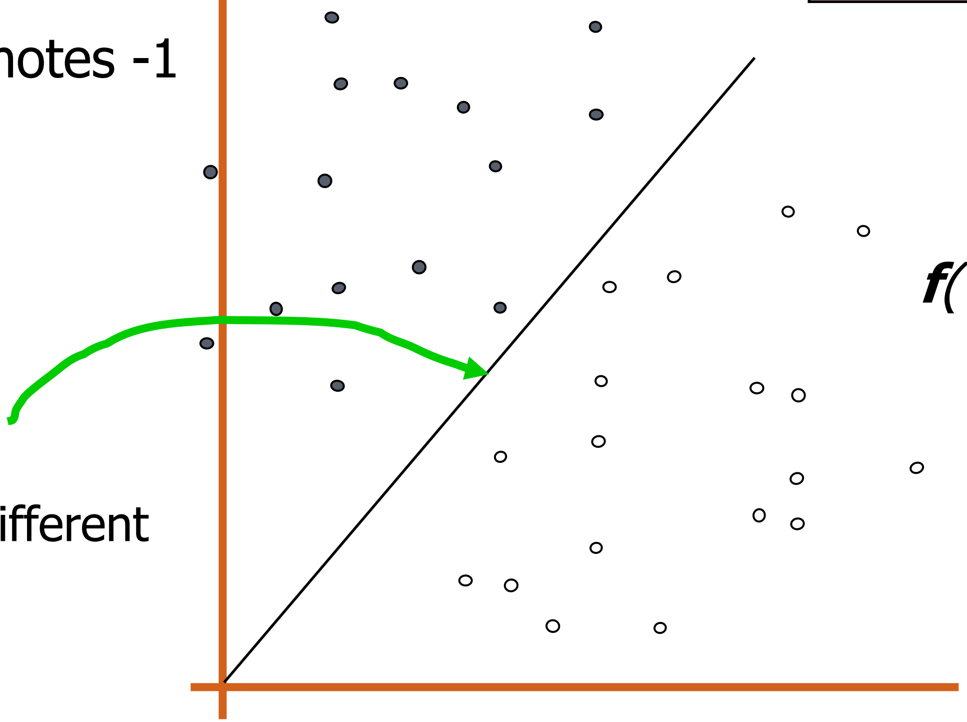
Introduction: Support Vector Machine

- SVM is related to statistical learning theory
- SVM was first introduced in 1992.
- SVM becomes popular because of its success in handwritten digit recognition
  - 1.1% test error rate for SVM. This is the same as the error rates of a carefully constructed neural network (NN).
- SVM is now regarded as an important example of "kernel methods", one of the key area in machine learning.

Support Vector Machine: Linear Classifiers



- ● denotes +1
- ○ denotes -1



Plane

Separating different classes

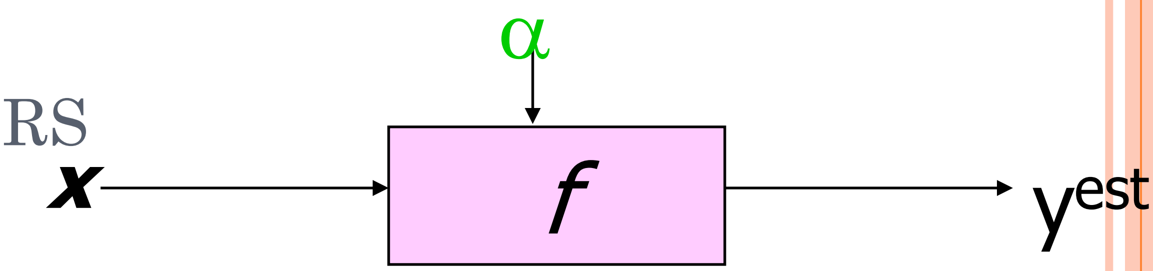
Estimation:

$$\boldsymbol{f}(\mathbf{w},b) = sign(\mathbf{w}.\ \mathbf{x} + b)$$

w: weight vector
x: data vector

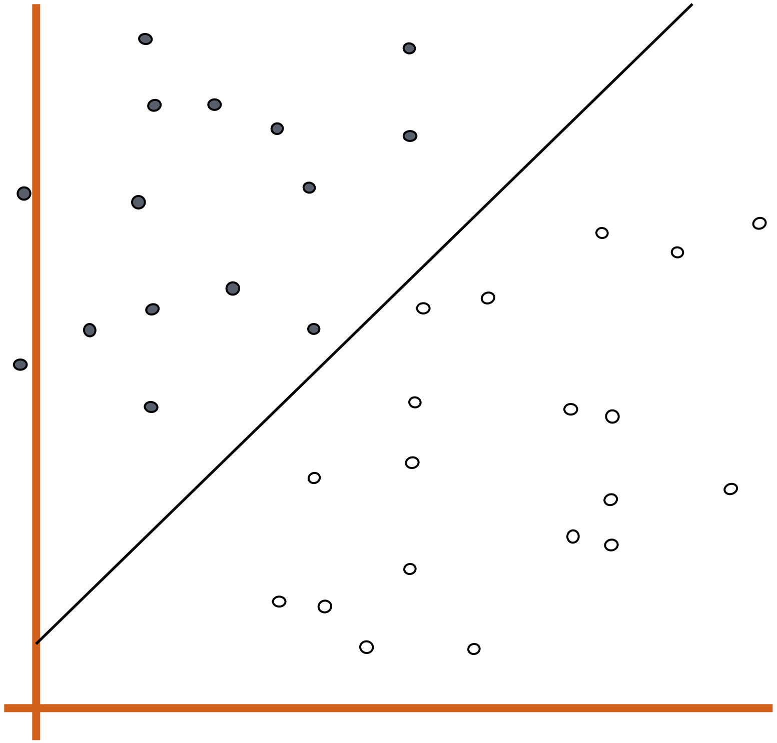
How would you classify this data?

Linear Classifiers



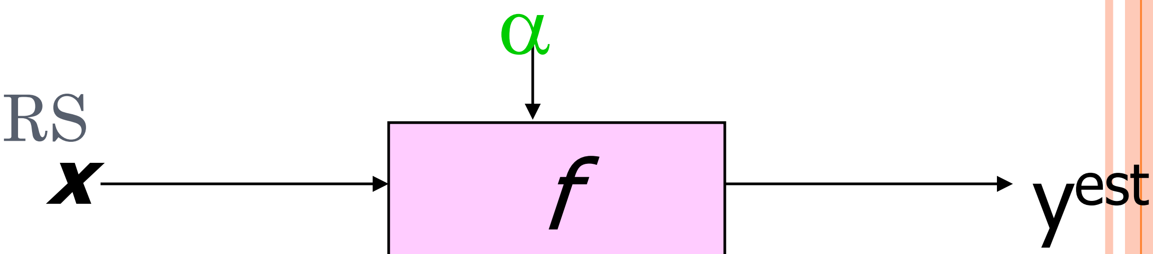
$$f(\mathbf{w},b) = sign(\mathbf{w}.\ \mathbf{x} + b)$$

- ● denotes +1
- ○ denotes -1



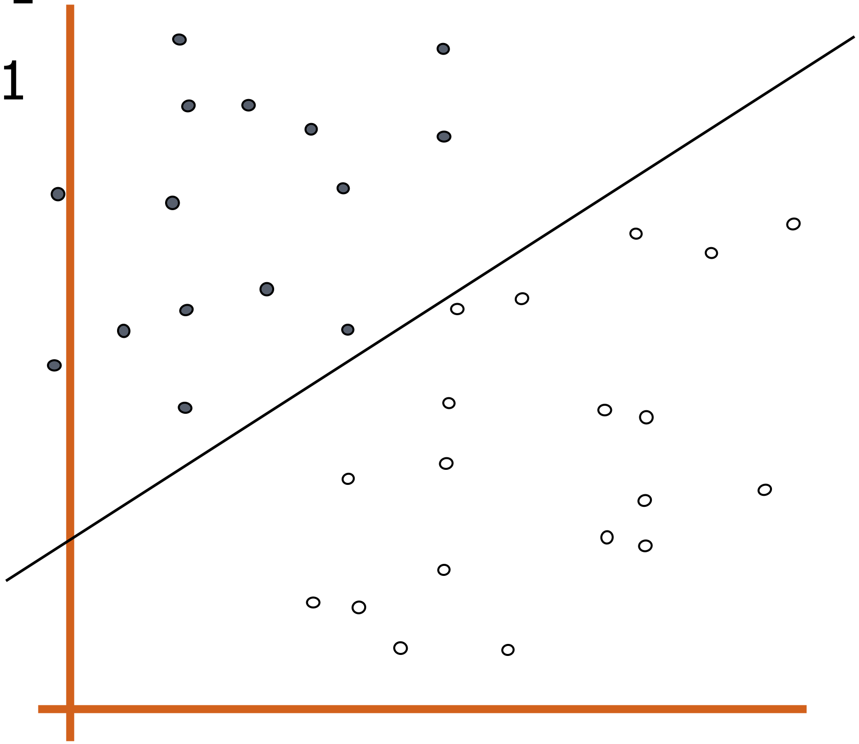
How would you classify this data?

Linear Classifiers



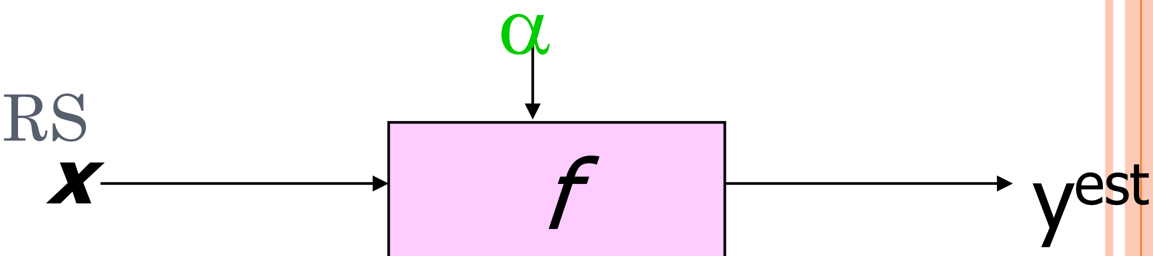
$$f(\mathbf{w},b) = sign(\mathbf{w}.\ \mathbf{x} + b)$$

- ● denotes +1
- ○ denotes -1



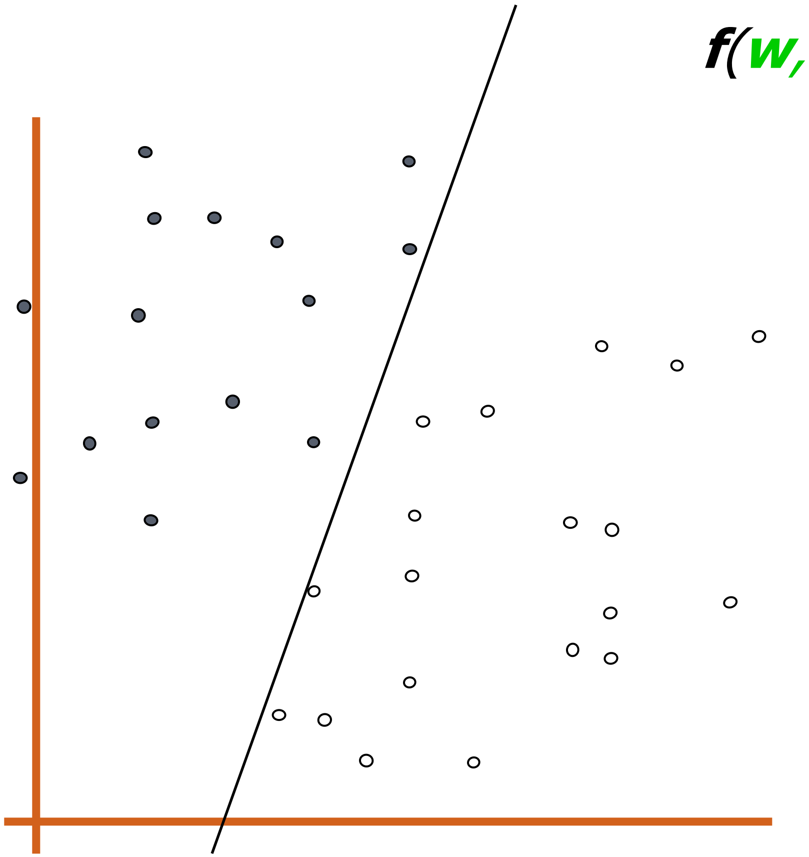
How would you classify this data?

Linear Classifiers



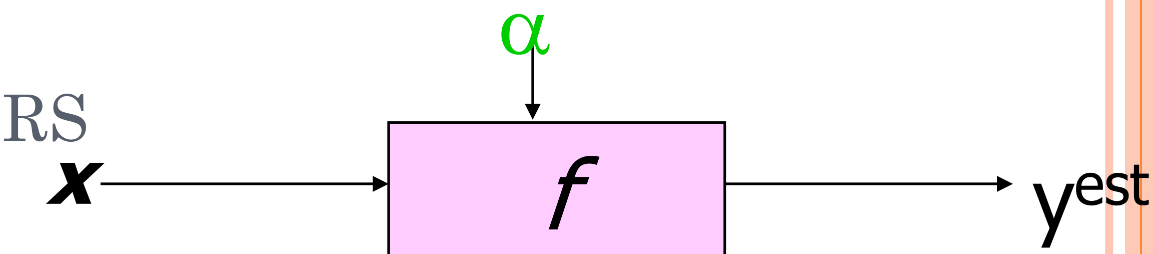
$f(\mathbf{w},b) = sign(\mathbf{w}.\ \mathbf{x} + b)$

- ● denotes +1
- ○ denotes -1



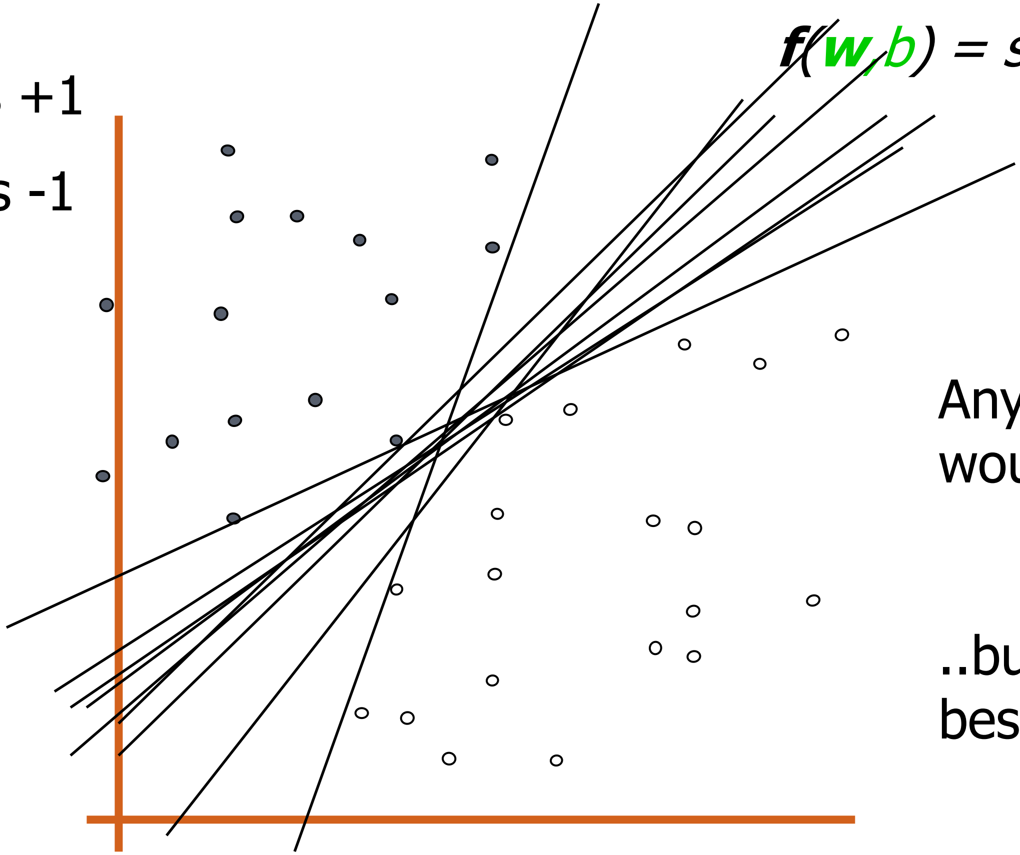
How would you classify this data?

Linear Classifiers



$f(\mathbf{w},b) = sign(\mathbf{w}.\ \mathbf{x} + b)$

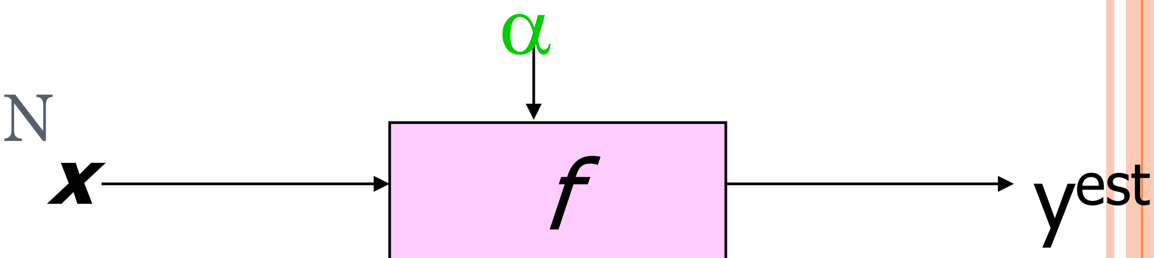
- ● denotes +1
- ○ denotes -1



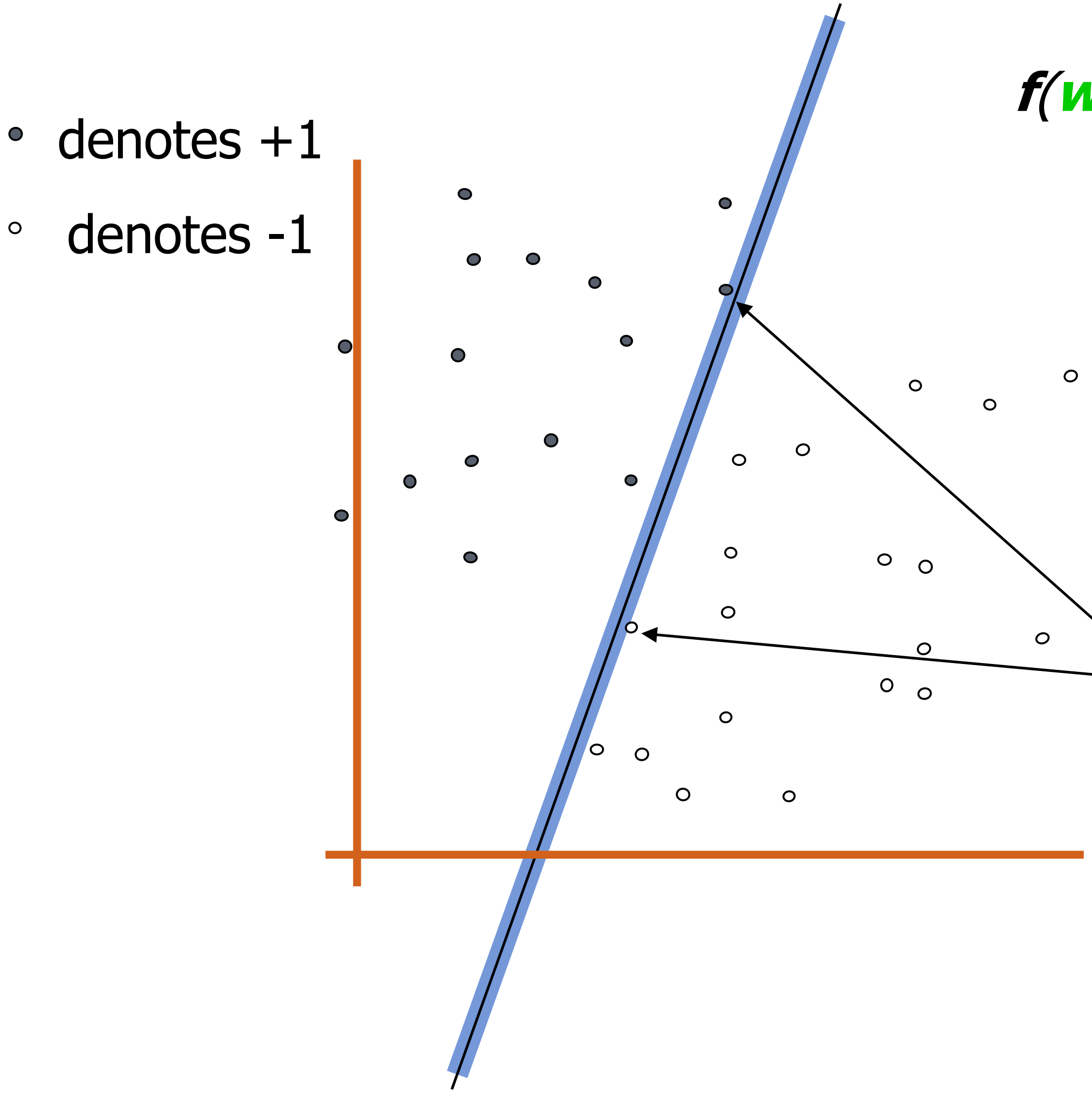
Any of these would be fine..

..but which is best?

Classifier Margin

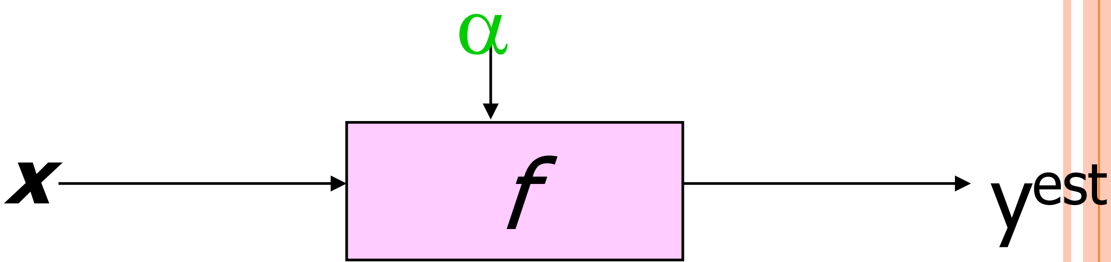


$f(\mathbf{w},b) = sign(\mathbf{w}.\,\mathbf{x} + b)$



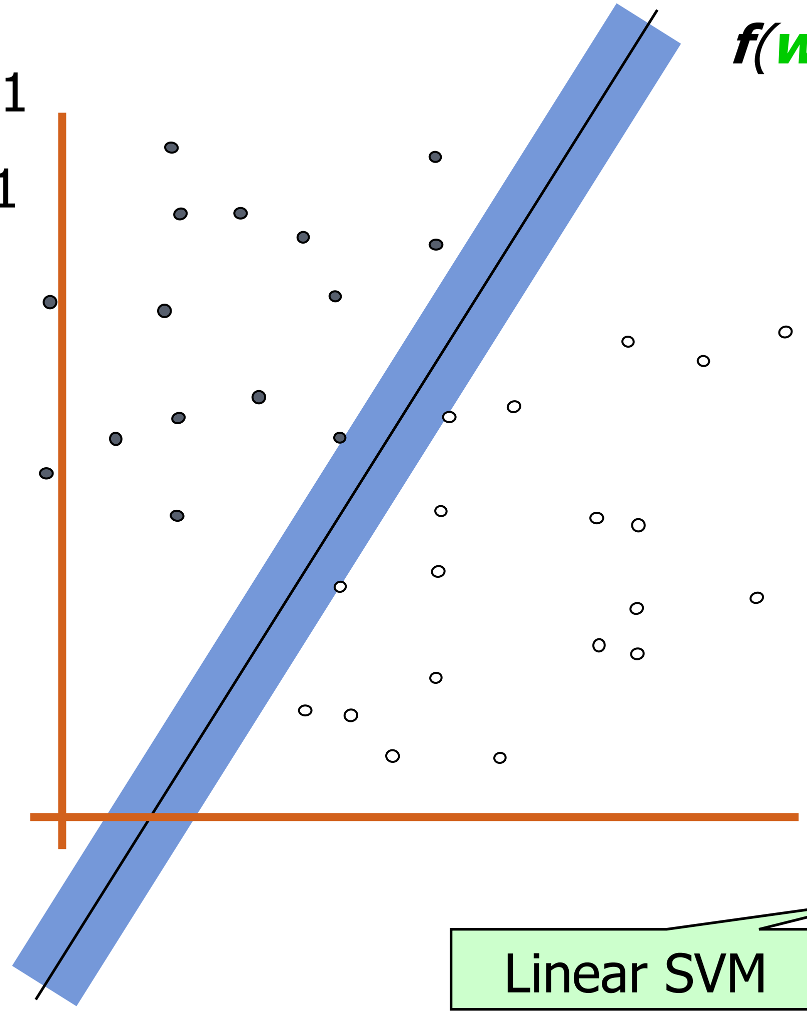
Define the margin of a linear classifier as the width that the boundary could be increased by before hitting a datapoint.

Maximum Margin



$f(\mathbf{w},b) = sign(\mathbf{w}.\ \mathbf{x} - b)$

- ● denotes +1
- ○ denotes -1

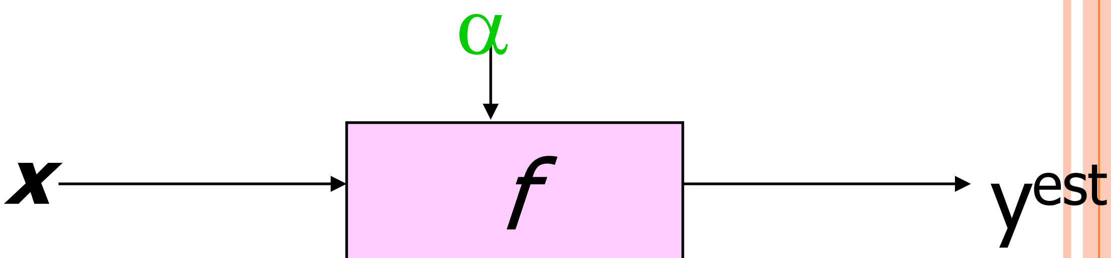


The maximum margin linear classifier is the linear classifier with the, um, maximum margin.

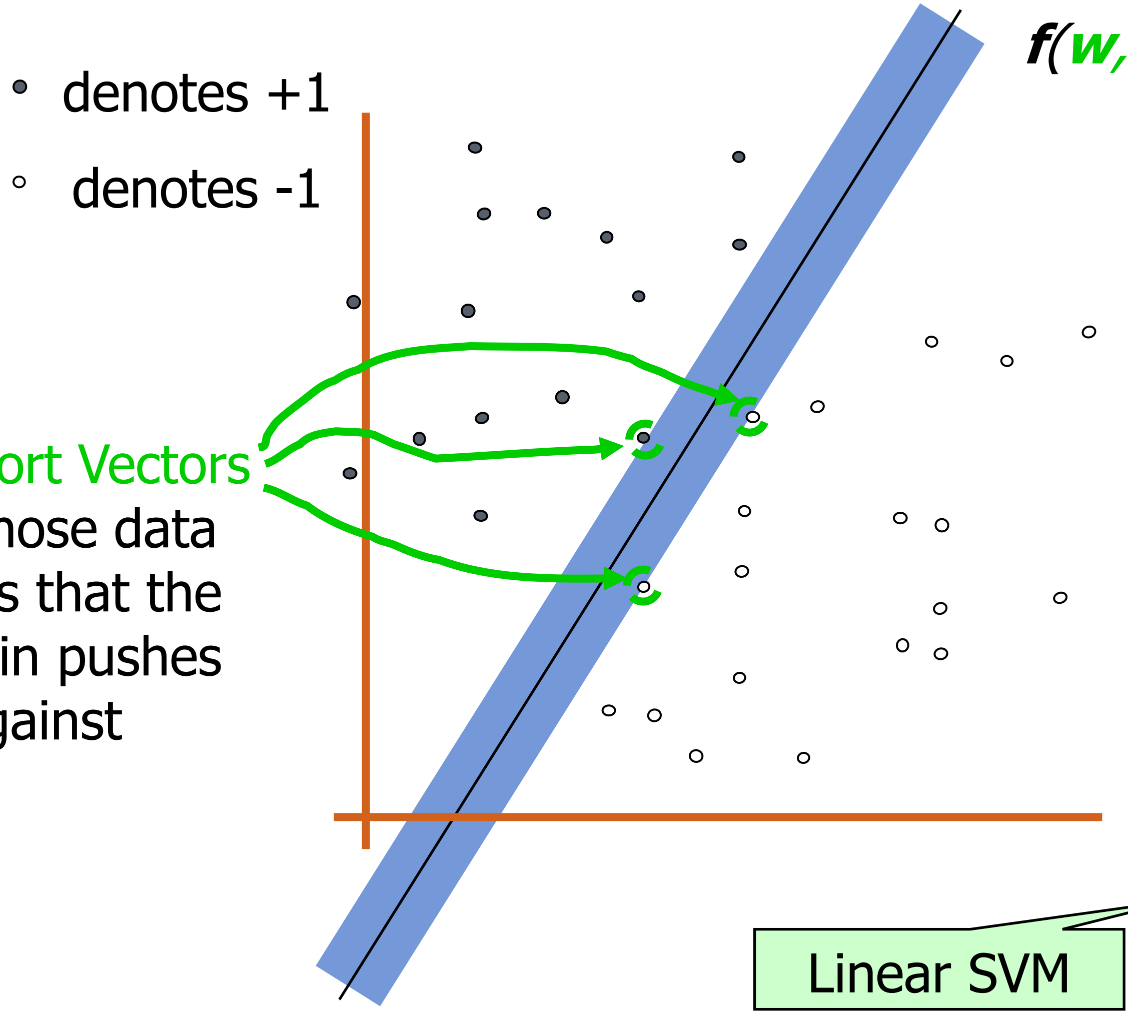
This is the simplest kind of SVM (Called an LSVM)

Linear SVM

Maximum Margin



$f(\mathbf{w},b) = sign(\mathbf{w}.\ \mathbf{x} + b)$



Support Vectors are those data points that the margin pushes up against

Linear SVM

The maximum margin linear classifier is the linear classifier with the, um, maximum margin.

This is the simplest kind of SVM (Called an LSVM)

Hyperplane : Numerical

The idea behind SVMs is to make use of a (nonlinear) mapping function Φ that transforms data in input space to data in feature space in such a way as to render a problem linearly separable.

The SVM then automatically discovers the optimal separating hyperplane (which, when mapped back into input space via φ^{-1}, can be a complex decision surface).

Hyperplane : Numerical-1

Suppose we are given the following positively labeled data points in $\Re^2$:

$$\left\{ \begin{pmatrix} 3 \\ 1 \end{pmatrix}, \begin{pmatrix} 3 \\ -1 \end{pmatrix}, \begin{pmatrix} 6 \\ 1 \end{pmatrix}, \begin{pmatrix} 6 \\ -1 \end{pmatrix} \right\}$$

and the following negatively labeled data points in $\Re^2$ (see Figure 1):

$$\left\{ \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \begin{pmatrix} 0 \\ -1 \end{pmatrix}, \begin{pmatrix} -1 \\ 0 \end{pmatrix} \right\}$$

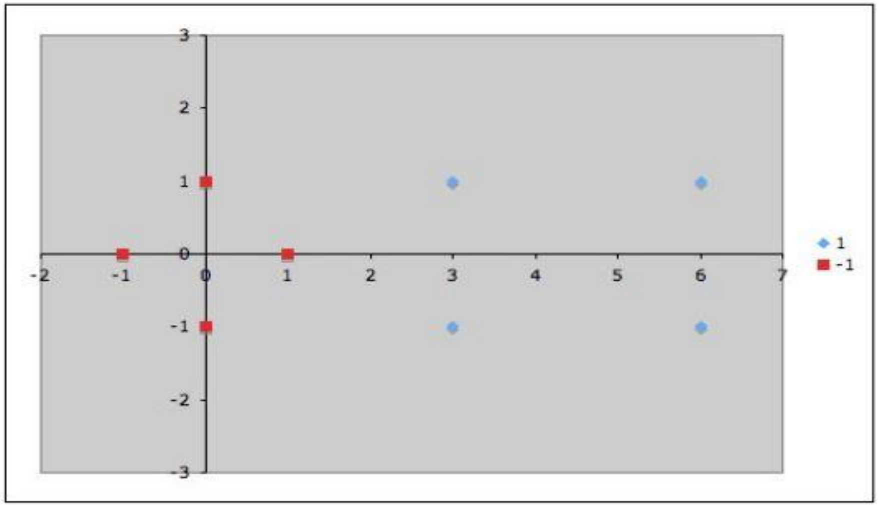


Figure 1: Sample data points in $\Re^2$. Blue diamonds are positive examples and red squares are negative examples.

Hyperplane : Numerical

We would like to discover a simple SVM that accurately discriminates the two classes. Since the data is linearly separable, we can use a linear SVM (that is, one whose mapping function **Φ**() is the identity function). By inspection, it should be obvious that there are three support vectors (see Figure 2):

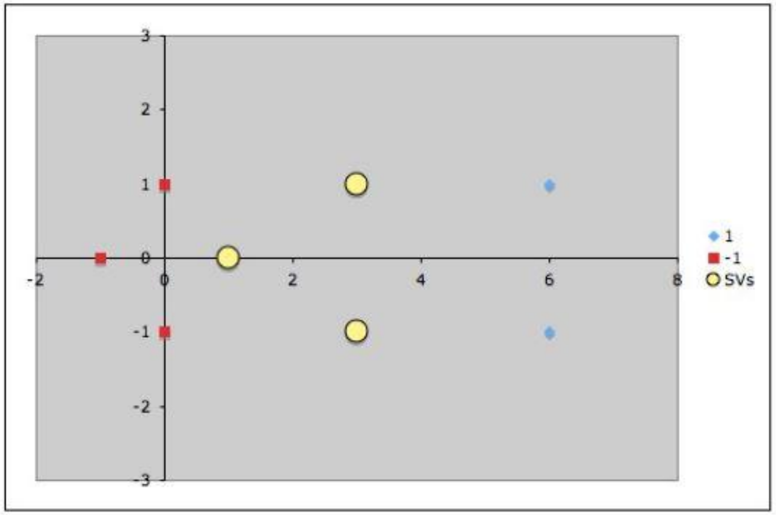
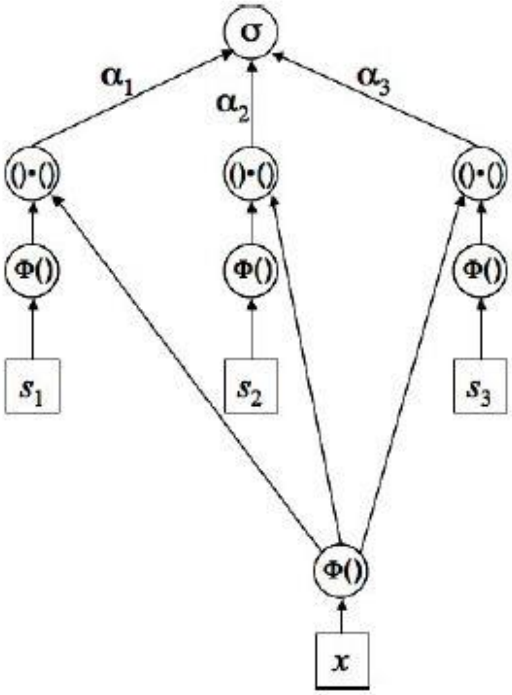


Figure 2: The three support vectors are marked as yellow circles.



Hyperplane : Numerical-1

$$\left\{ s_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, s_2 = \begin{pmatrix} 3 \\ 1 \end{pmatrix}, s_3 = \begin{pmatrix} 3 \\ -1 \end{pmatrix} \right\}$$

In what follows we will use vectors augmented with a 1 as a bias input, and for clarity we will differentiate these with an over-tilde. So, if $s_1 = (10)$, then $\tilde{s_1} = (101)$. Figure 3 shows the SVM architecture, and our task is to find values for the α_i such that

$$\begin{aligned}
\alpha_1 \Phi(s_1) \cdot \Phi(s_1) + \alpha_2 \Phi(s_2) \cdot \Phi(s_1) + \alpha_3 \Phi(s_3) \cdot \Phi(s_1) &= -1 \\
\alpha_1 \Phi(s_1) \cdot \Phi(s_2) + \alpha_2 \Phi(s_2) \cdot \Phi(s_2) + \alpha_3 \Phi(s_3) \cdot \Phi(s_2) &= +1 \\
\alpha_1 \Phi(s_1) \cdot \Phi(s_3) + \alpha_2 \Phi(s_2) \cdot \Phi(s_3) + \alpha_3 \Phi(s_3) \cdot \Phi(s_3) &= +1
\end{aligned}$$

Hyperplane : Numerical-1

Since for now we have let $\Phi() = I$, this reduces to

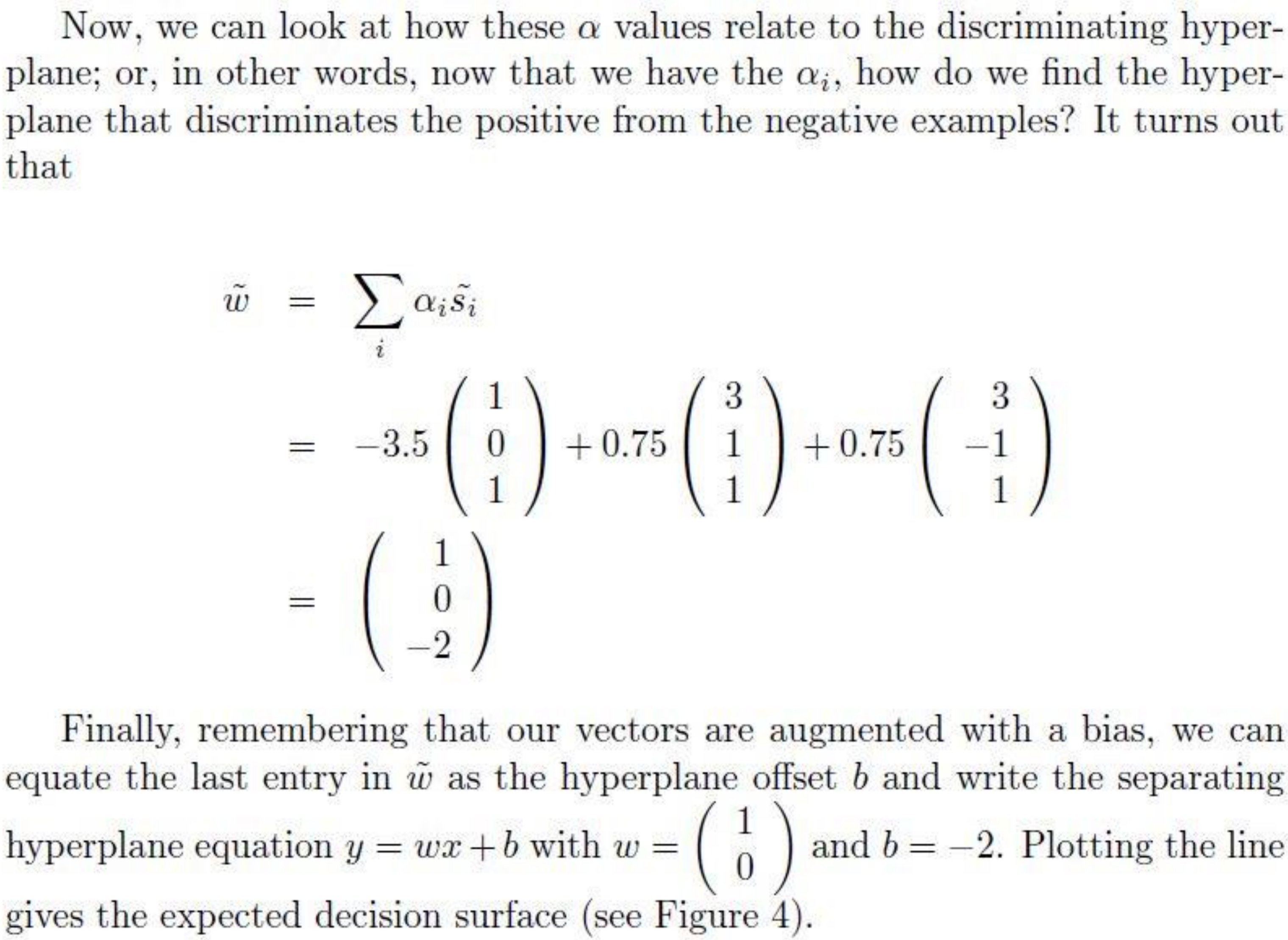
$$\begin{aligned}
\alpha_1 \tilde{s_1} \cdot \tilde{s_1} + \alpha_2 \tilde{s_2} \cdot \tilde{s_1} + \alpha_3 \tilde{s_3} \cdot \tilde{s_1} &= -1 \\
\alpha_1 \tilde{s_1} \cdot \tilde{s_2} + \alpha_2 \tilde{s_2} \cdot \tilde{s_2} + \alpha_3 \tilde{s_3} \cdot \tilde{s_2} &= +1 \\
\alpha_1 \tilde{s_1} \cdot \tilde{s_3} + \alpha_2 \tilde{s_2} \cdot \tilde{s_3} + \alpha_3 \tilde{s_3} \cdot \tilde{s_3} &= +1
\end{aligned}$$

Now, computing the dot products results in

$$\begin{aligned}
2\alpha_1 + 4\alpha_2 + 4\alpha_3 &= -1 \\
4\alpha_1 + 11\alpha_2 + 9\alpha_3 &= +1 \\
4\alpha_1 + 9\alpha_2 + 11\alpha_3 &= +1
\end{aligned}$$

A little algebra reveals that the solution to this system of equations is $\alpha_1 = -3.5$, $\alpha_2 = 0.75$ and $\alpha_3 = 0.75$.

HYPERPLANE : NUMERICAL-1



Now, we can look at how these α values relate to the discriminating hyperplane; or, in other words, now that we have the α_i, how do we find the hyperplane that discriminates the positive from the negative examples? It turns out that

$$\begin{aligned}
\tilde{w} &= \sum_i \alpha_i \tilde{s}_i \\
&= -3.5 \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} + 0.75 \begin{pmatrix} 3 \\ 1 \\ 1 \end{pmatrix} + 0.75 \begin{pmatrix} 3 \\ -1 \\ 1 \end{pmatrix} \\
&= \begin{pmatrix} 1 \\ 0 \\ -2 \end{pmatrix}
\end{aligned}$$

Finally, remembering that our vectors are augmented with a bias, we can equate the last entry in $\tilde{w}$ as the hyperplane offset b and write the separating hyperplane equation $y = wx + b$ with $w = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ and $b = -2$. Plotting the line gives the expected decision surface (see Figure 4).

HYPERPLANE : NUMERICAL-1

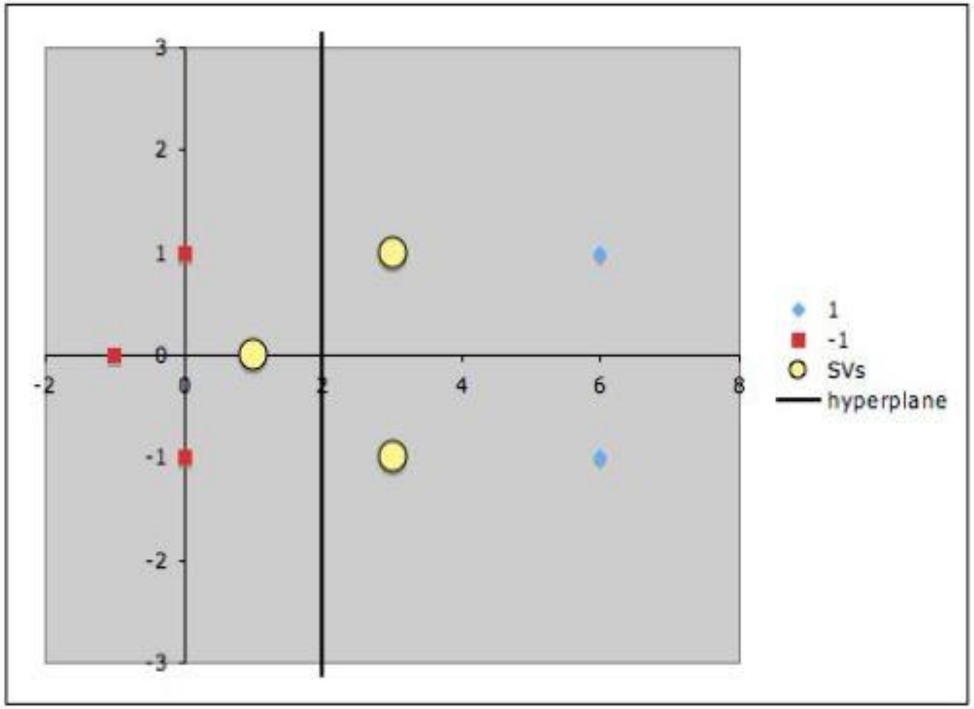


Figure 4: The discriminating hyperplane corresponding to the values $\alpha_1 = -3.5, \alpha_2 = 0.75$ and $\alpha_3 = 0.75$.

Hyperplane : Numerical-2

Now suppose instead that we are given the following positively labeled data points in $\Re^2$:

$$\left\{ \begin{pmatrix} 2 \\ 2 \end{pmatrix}, \begin{pmatrix} 2 \\ -2 \end{pmatrix}, \begin{pmatrix} -2 \\ -2 \end{pmatrix}, \begin{pmatrix} -2 \\ 2 \end{pmatrix} \right\}$$

and the following negatively labeled data points in $\Re^2$ (see Figure 5):

$$\left\{ \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ -1 \end{pmatrix}, \begin{pmatrix} -1 \\ -1 \end{pmatrix}, \begin{pmatrix} -1 \\ 1 \end{pmatrix} \right\}$$

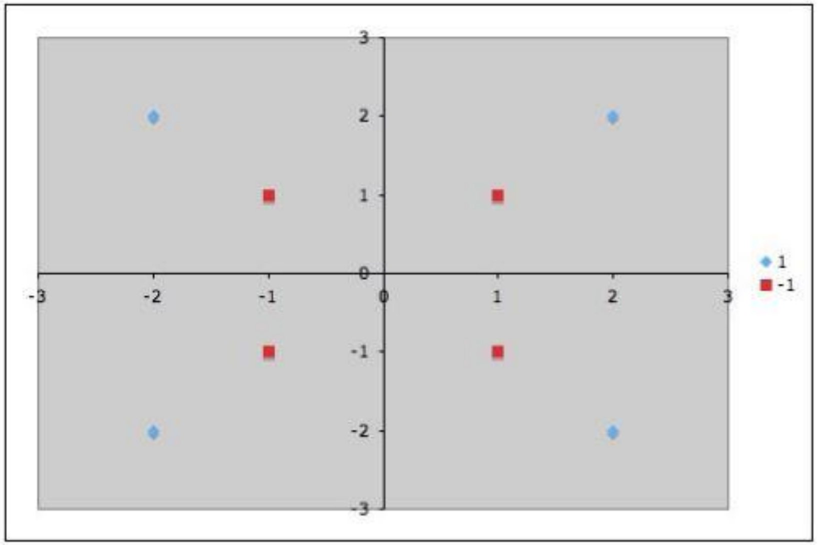


Figure 5: Nonlinearly separable sample data points in $\Re^2$. Blue diamonds are positive examples and red squares are negative examples.

HYPERPLANE : NUMERICAL-2

Our goal, again, is to discover a separating hyperplane that accurately discriminates the two classes. Of course, it is obvious that no such hyperplane exists in the input space (that is, in the space in which the original input data live). Therefore, we must use a nonlinear SVM (that is, one whose mapping function is a nonlinear mapping from input space into some feature space).

Define

$$\Phi_1\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{cases} \begin{pmatrix} 4 - x_2 + |x_1 - x_2| \\ 4 - x_1 + |x_1 - x_2| \end{pmatrix} & \text{if } \sqrt{x_1^2 + x_2^2} > 2 \\ \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} & \text{otherwise} \end{cases}$$

Hyperplane : Numerical-2

$$\left\{ \begin{pmatrix} 2 \\ 2 \end{pmatrix}, \begin{pmatrix} 6 \\ 2 \end{pmatrix}, \begin{pmatrix} 6 \\ 6 \end{pmatrix}, \begin{pmatrix} 2 \\ 6 \end{pmatrix} \right\}$$

for the positive examples and

$$\left\{ \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ -1 \end{pmatrix}, \begin{pmatrix} -1 \\ -1 \end{pmatrix}, \begin{pmatrix} -1 \\ 1 \end{pmatrix} \right\}$$

for the negative examples (see Figure 6). Now we can once again easily identify the support vectors (see Figure 7):

$$\left\{ s_1 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, s_2 = \begin{pmatrix} 2 \\ 2 \end{pmatrix} \right\}$$

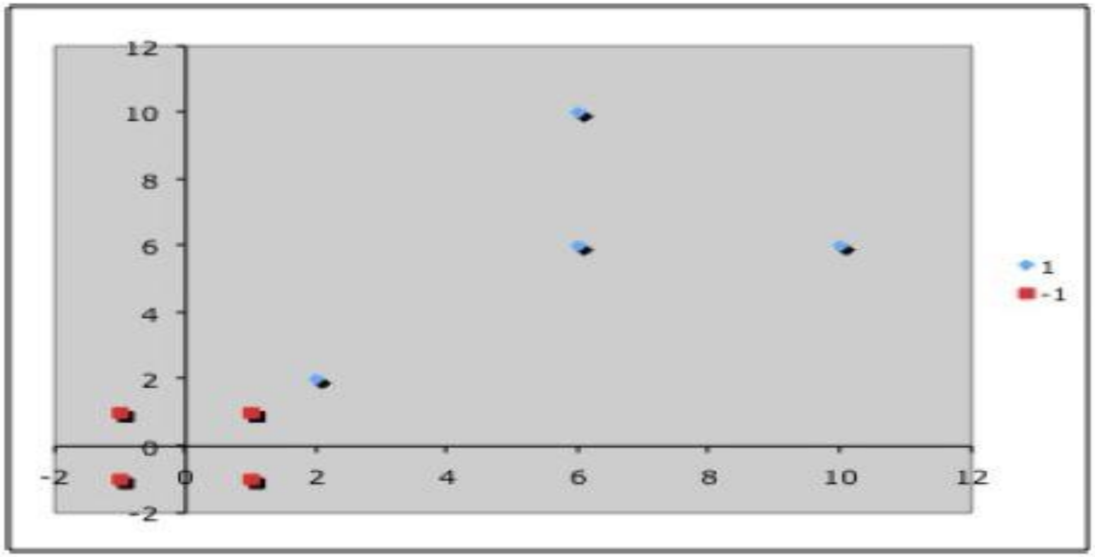


Figure 6: The data represented in feature space.

Hyperplane : Numerical-2

We again use vectors augmented with a 1 as a bias input and will differentiate them as before. Now given the [augmented] support vectors, we must again and values for the α^i

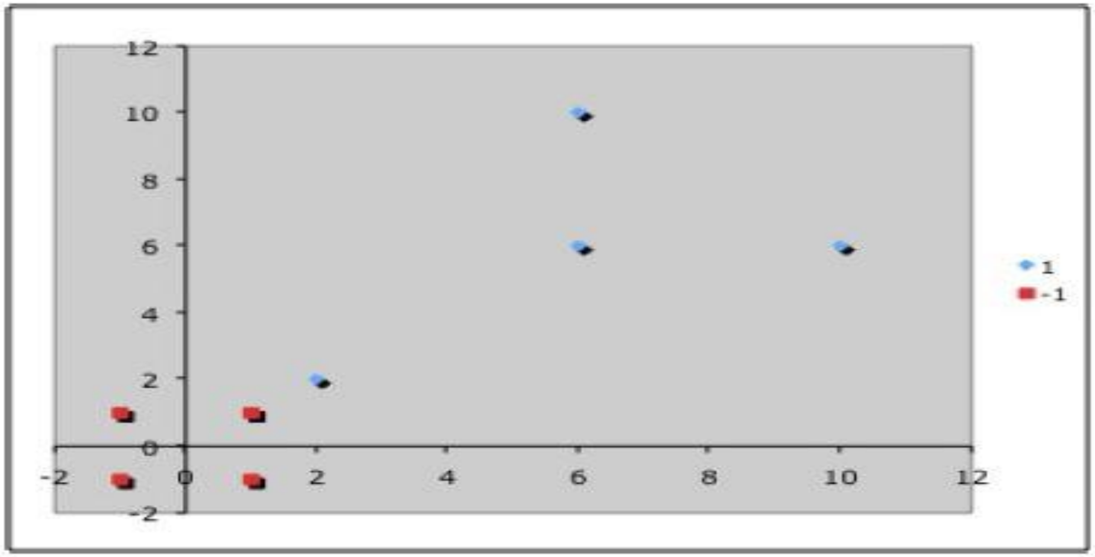


Figure 6: The data represented in feature space.

HYPERPLANE : NUMERICAL-2

$$\begin{aligned}\alpha_1\Phi_1(s_1)\cdot\Phi_1(s_1)+\alpha_2\Phi_1(s_2)\cdot\Phi_1(s_1) &= -1\\ \alpha_1\Phi_1(s_1)\cdot\Phi_1(s_2)+\alpha_2\Phi_1(s_2)\cdot\Phi_1(s_2) &= +1\end{aligned}$$

Given Eq. 1, this reduces to

$$\begin{aligned}\alpha_1\tilde{s_1}\cdot\tilde{s_1}+\alpha_2\tilde{s_2}\cdot\tilde{s_1} &= -1\\ \alpha_1\tilde{s_1}\cdot\tilde{s_2}+\alpha_2\tilde{s_2}\cdot\tilde{s_2} &= +1\end{aligned}$$

(Note that even though Φ_1 is a nontrivial function, both s_1 and s_2 map to themselves under Φ_1. This will not be the case for other inputs as we will see later.)

Now, computing the dot products results in

$$\begin{aligned}3\alpha_1+5\alpha_2 &= -1\\ 5\alpha_1+9\alpha_2 &= +1\end{aligned}$$

And the solution to this system of equations is $\alpha_1=-7$ and $\alpha_2=4$.

Hyperplane : Numerical-2

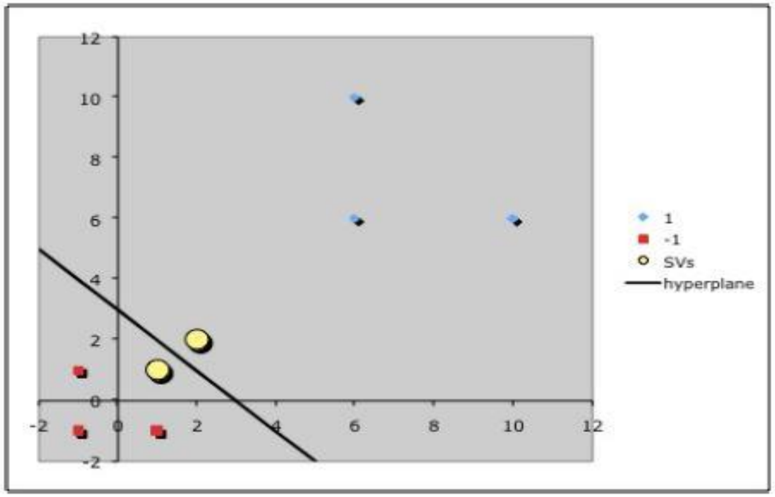


Figure 8: The discriminating hyperplane corresponding to the values $\alpha_1 = -7$ and $\alpha_2 = 4$

$$= \quad -7 \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} + 4 \begin{pmatrix} 2 \\ 2 \\ 1 \end{pmatrix}$$

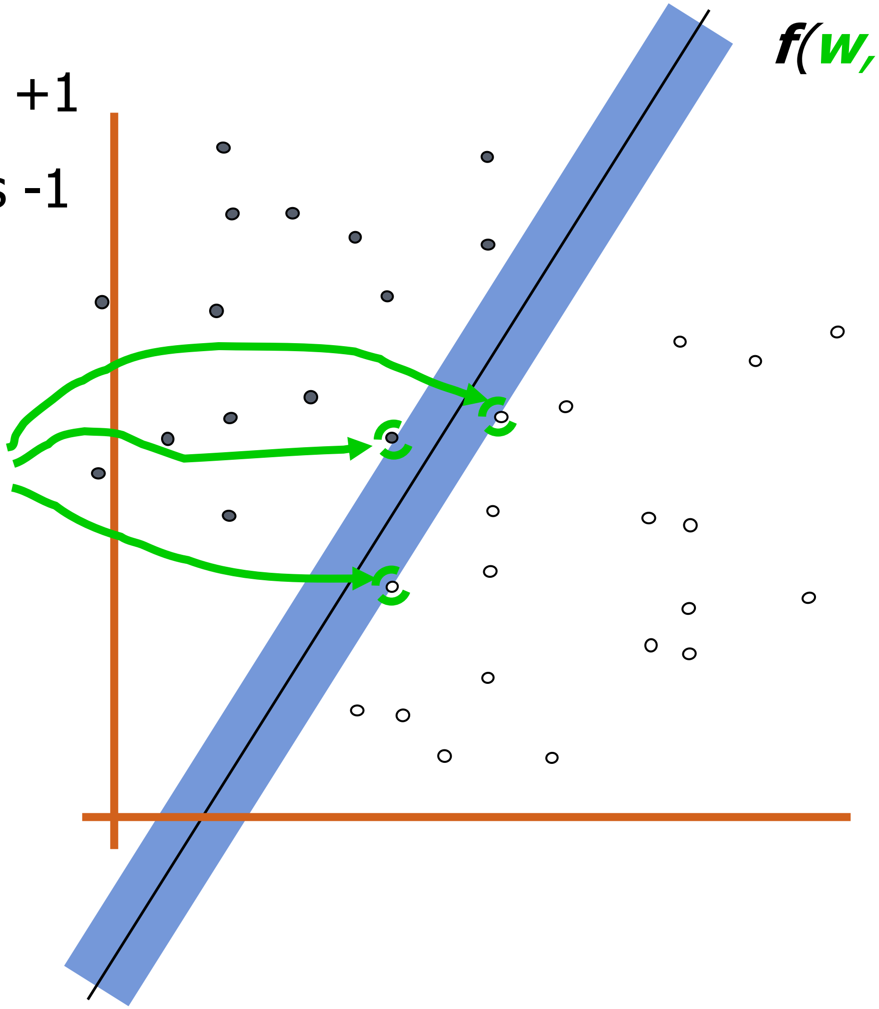
$$= \quad \begin{pmatrix} 1 \\ 1 \\ -3 \end{pmatrix}$$

giving us the separating hyperplane equation $y = wx + b$ with $w = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$ and $b = -3$. Plotting the line gives the expected decision surface (see Figure 8).

Why Maximum Margin?

● denotes +1

○ denotes -1



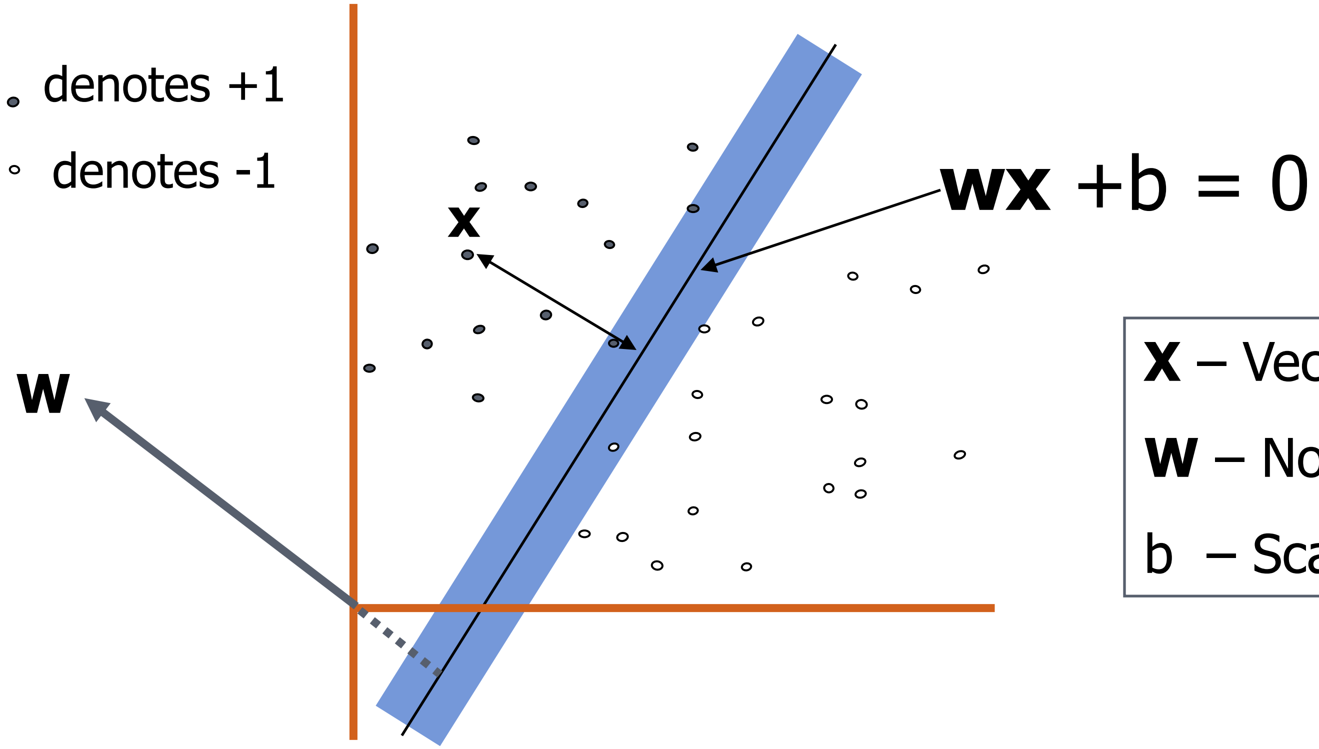
$f(\mathbf{w},b) = sign(\mathbf{w}.\,\mathbf{x} + b)$

Support Vectors are those datapoints that the margin pushes up against

The maximum margin linear classifier is the linear classifier with the, maximum margin.

This is the simplest kind of SVM (Called an LSVM)

How to calculate the distance from a point to a line?



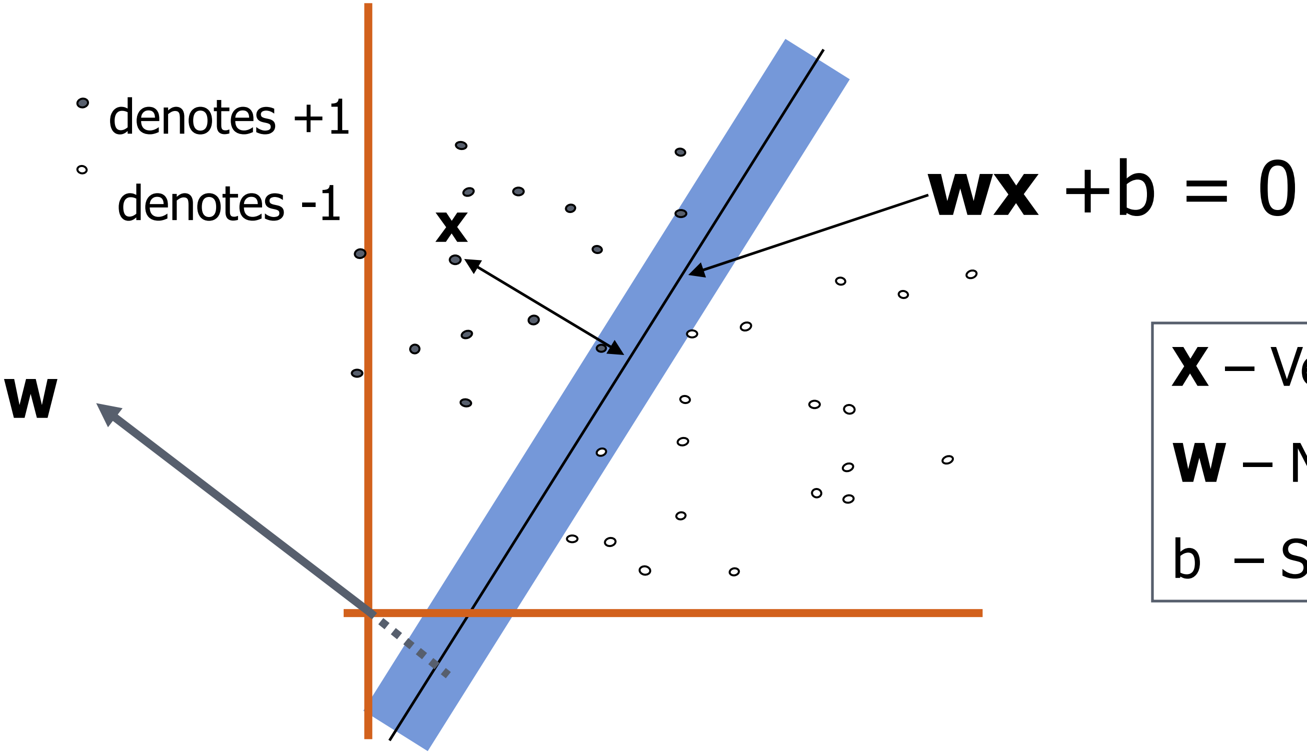
X – Vector

W – Normal Vector

b – Scale Value (bias)

- In our case, $w_1*x_1+w_2*x_2+b=0$,
- thus, $\mathbf{w}=(w_1,w_2)$, $\mathbf{x}=(x_1,x_2)$

Estimate the Margin



x – Vector

w – Normal Vector

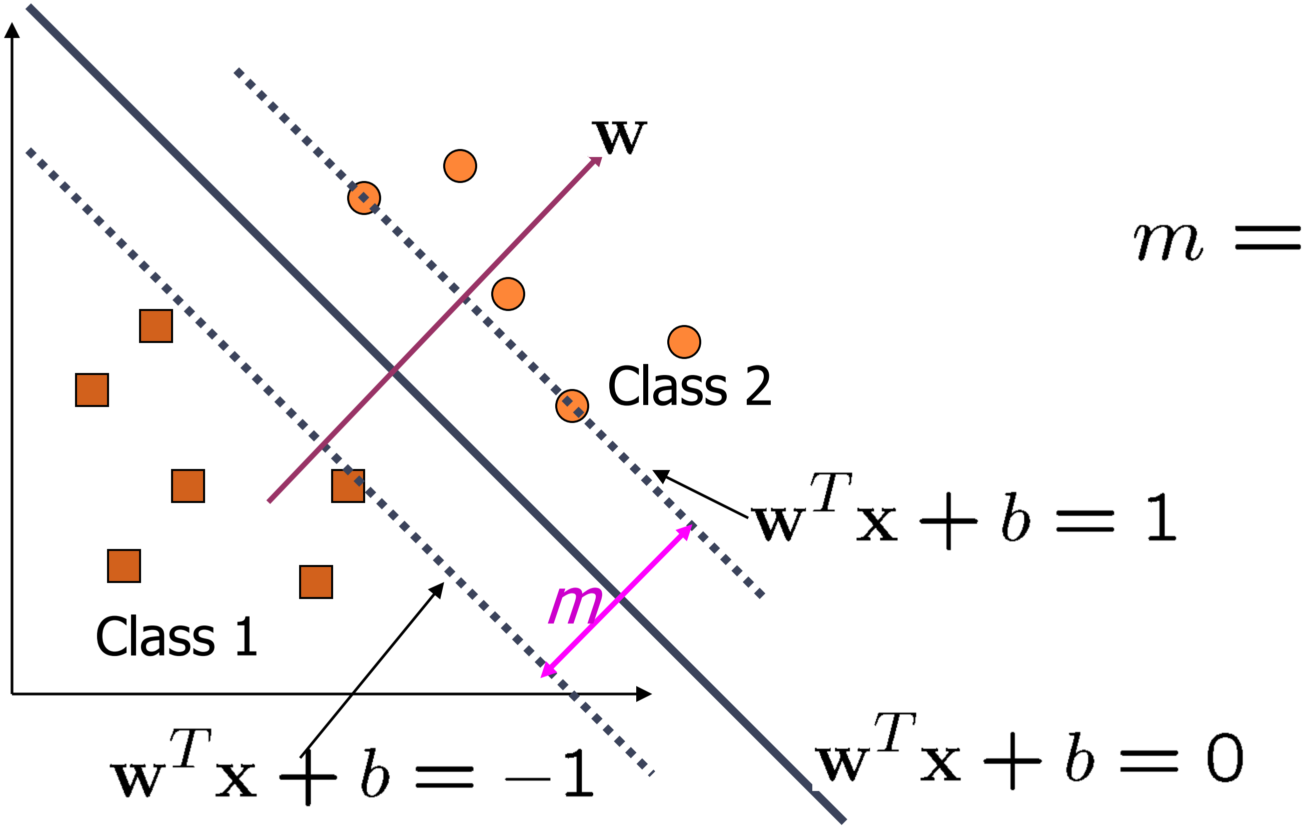
b – Scale Value

- What is the distance expression for a point **x** to a line **wx**+b= 0?

$$d(\mathbf{x}) = \frac{|\mathbf{x}\cdot\mathbf{w}+b|}{\sqrt{\|\mathbf{w}\|_2^2}} = \frac{|\mathbf{x}\cdot\mathbf{w}+b|}{\sqrt{\sum_{i=1}^{d} w_i^2}}$$

Large-margin Decision Boundary

- The decision boundary should be as far away from the data of both classes as possible
  - We should maximize the margin, *m*
  - Distance between the origin and the line $\mathbf{w}^T\mathbf{x}$=-b is b/||**w**||



$\mathbf{w}$
Class 2
$\mathbf{w}^T\mathbf{x} + b = 1$
m
Class 1
$\mathbf{w}^T\mathbf{x} + b = -1$
$\mathbf{w}^T\mathbf{x} + b = 0$

$$m = \frac{2}{||\mathbf{w}||}$$

Finding the Decision Boundary

- Let $\{x_1, ..., x_n\}$ be our data set and let $y_i \in \{1,-1\}$ be the class label of x_i
- The decision boundary should classify all points correctly $\Rightarrow$ $y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1, \quad \forall i$
- To see this:

  when y= -1, we wish (wx+b)<1,

  when y =1, we wish (wx+b)>1.

  For support vectors, we wish y(wx+b)=1.
- The decision boundary can be found by solving the following constrained optimization problem

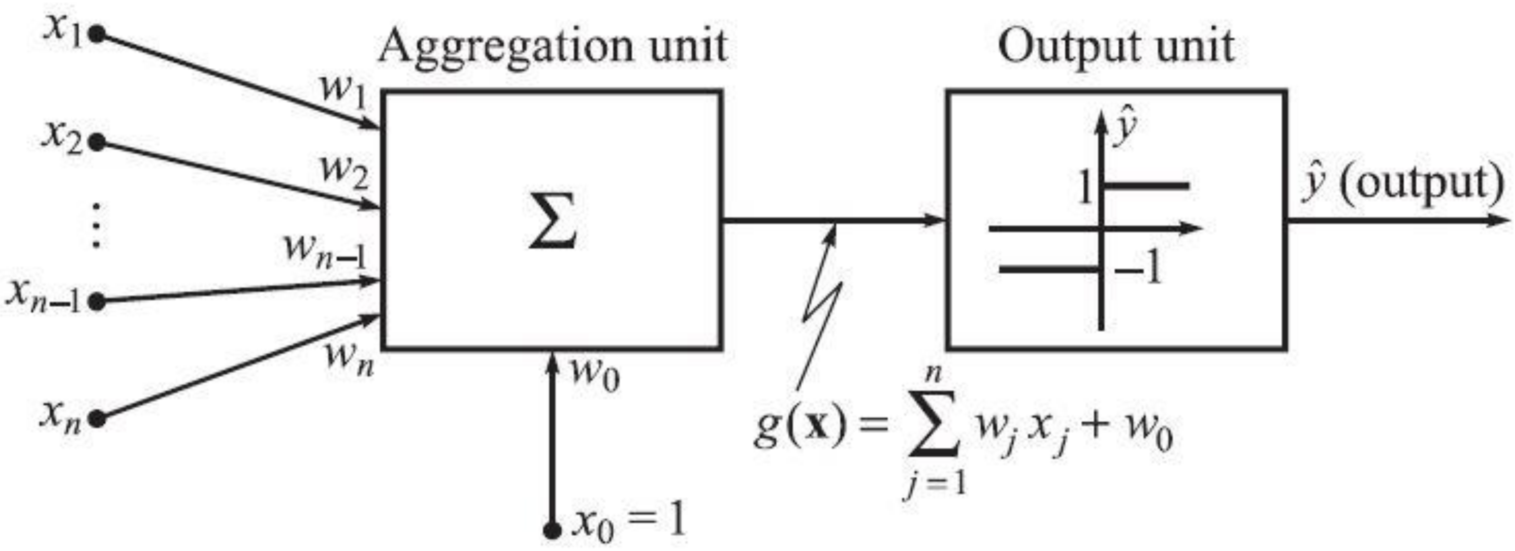
$$\text{Minimize } \frac{1}{2}||\mathbf{w}||^2$$

$$\text{subject to } y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 \qquad \forall i$$

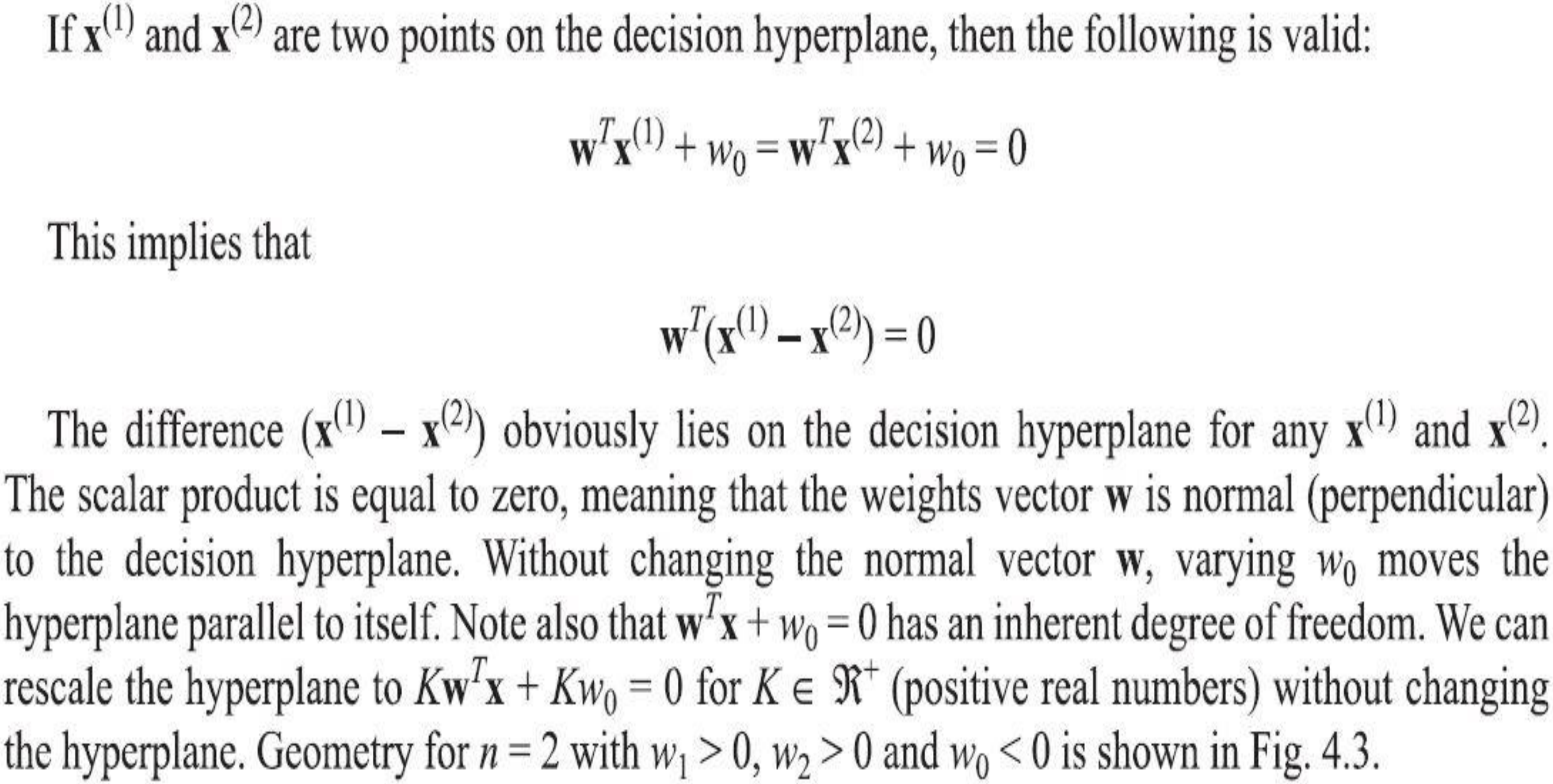
NEXT STEP… OPTIONAL

- Converting SVM to a form we can solve
  - Dual form
- Allowing a few errors
  - Soft margin
- Allowing nonlinear boundary
  - Kernel functions

DERIVATION


x_1
x_2
x_{n-1}
x_n
w_1
w_2
w_{n-1}
w_n
Aggregation unit
Σ
w_0
$x_0 = 1$
Output unit
$\hat{y}$
1
−1
$\hat{y}$ (output)
$g(\mathbf{x}) = \sum_{j=1}^{n} w_j x_j + w_0$

DERIVATION



If $\mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$ are two points on the decision hyperplane, then the following is valid:

$$\mathbf{w}^T\mathbf{x}^{(1)} + w_0 = \mathbf{w}^T\mathbf{x}^{(2)} + w_0 = 0$$

This implies that

$$\mathbf{w}^T(\mathbf{x}^{(1)} - \mathbf{x}^{(2)}) = 0$$

The difference $(\mathbf{x}^{(1)} - \mathbf{x}^{(2)})$ obviously lies on the decision hyperplane for any $\mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$. The scalar product is equal to zero, meaning that the weights vector **w** is normal (perpendicular) to the decision hyperplane. Without changing the normal vector **w**, varying w_0 moves the hyperplane parallel to itself. Note also that $\mathbf{w}^T\mathbf{x} + w_0 = 0$ has an inherent degree of freedom. We can rescale the hyperplane to $K\mathbf{w}^T\mathbf{x} + Kw_0 = 0$ for $K \in \Re^+$ (positive real numbers) without changing the hyperplane. Geometry for $n = 2$ with $w_1 > 0$, $w_2 > 0$ and $w_0 < 0$ is shown in Fig. 4.3.

DERIVATION

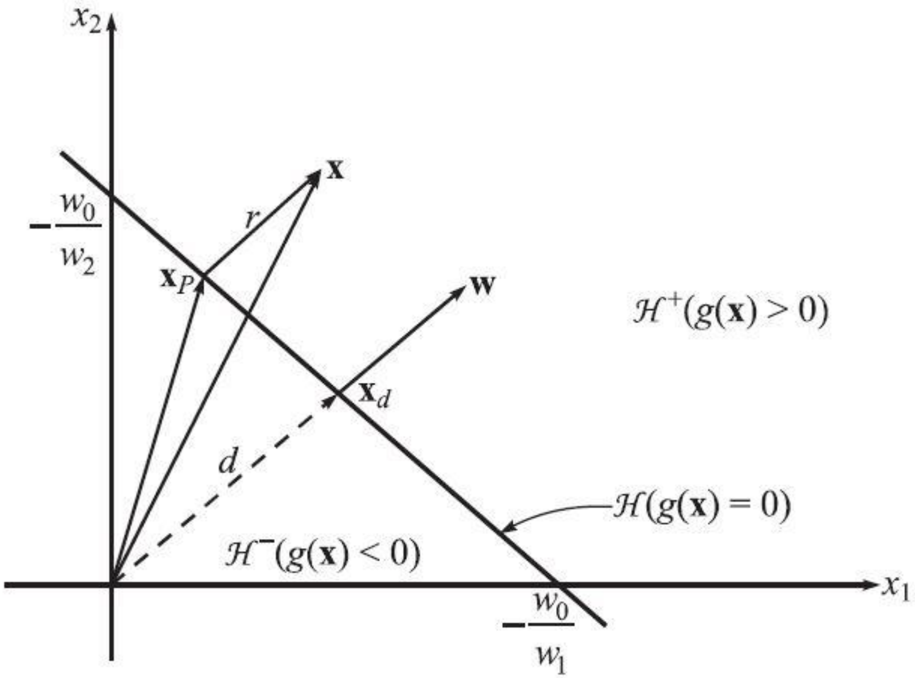


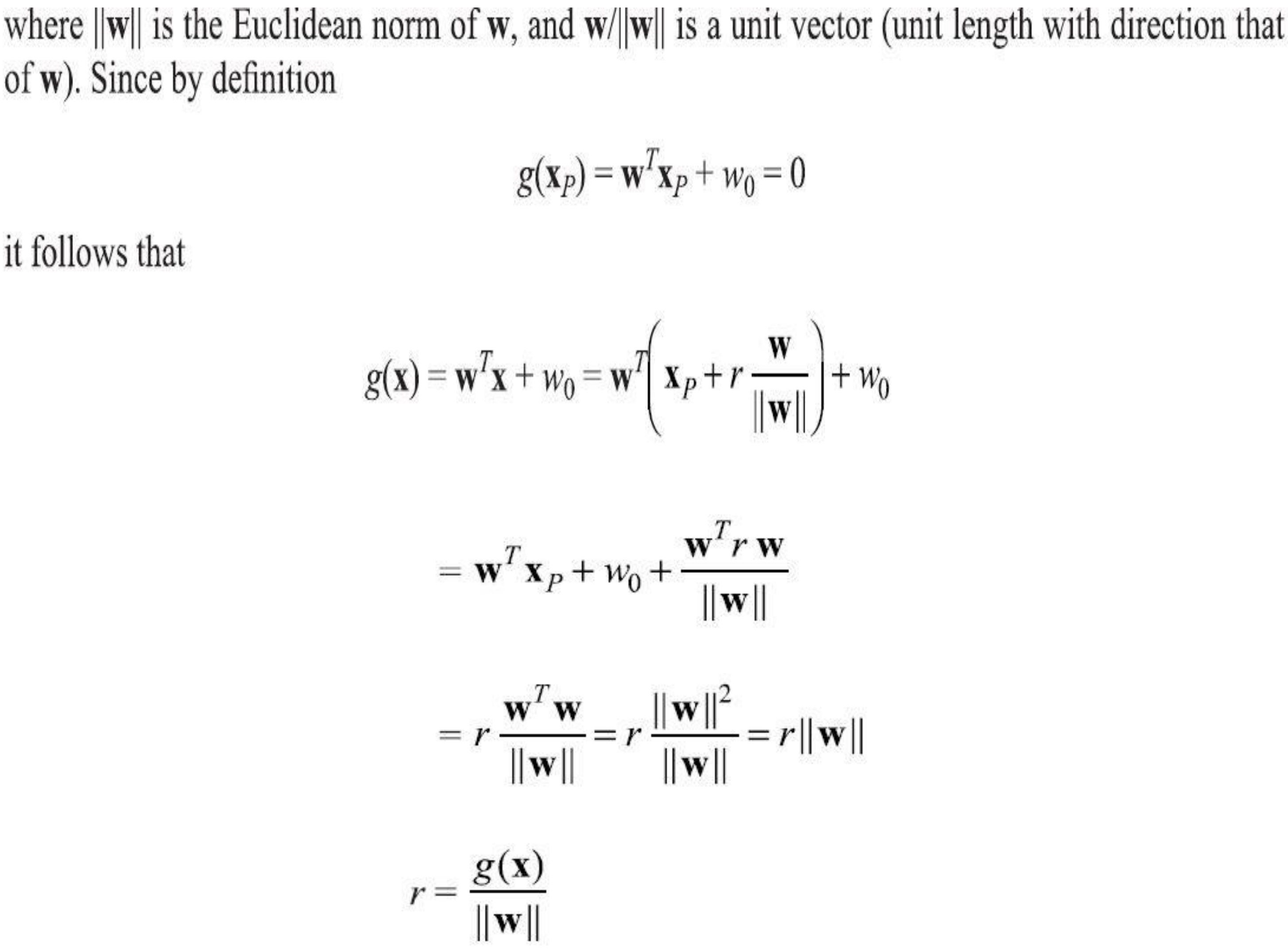
Figure 4.3 Linear decision boundary between two classes

The location of any point **x** may be considered relative to the hyperplane $\mathcal{H}$. Defining $\mathbf{x}_P$ as the normal projection of **x** onto $\mathcal{H}$ (shown in Fig. 4.3), we may decompose **x** as,

$$\mathbf{x} = \mathbf{x}_P + r\frac{\mathbf{w}}{\|\mathbf{w}\|} \tag{4.6}$$

where $\|\mathbf{w}\|$ is the Euclidean norm of **w**, and $\mathbf{w}/\|\mathbf{w}\|$ is a unit vector (unit length with direction that of **w**). Since by definition

Derivation



where $||\mathbf{w}||$ is the Euclidean norm of $\mathbf{w}$, and $\mathbf{w}/||\mathbf{w}||$ is a unit vector (unit length with direction that of $\mathbf{w}$). Since by definition

$$g(\mathbf{x}_P) = \mathbf{w}^T\mathbf{x}_P + w_0 = 0$$

it follows that

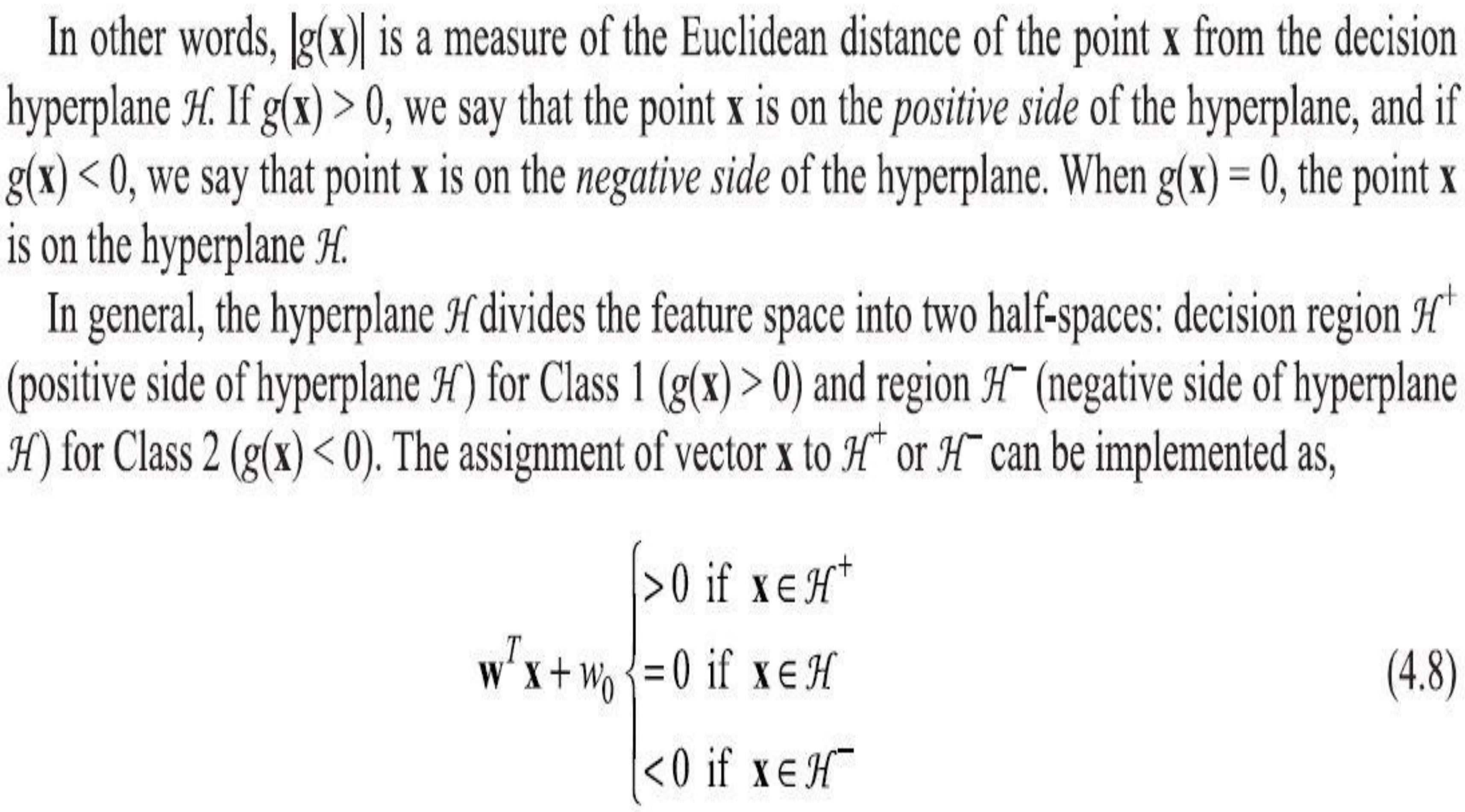
$$g(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + w_0 = \mathbf{w}^T\left(\mathbf{x}_P + r\frac{\mathbf{w}}{||\mathbf{w}||}\right) + w_0$$

$$= \mathbf{w}^T\mathbf{x}_P + w_0 + \frac{\mathbf{w}^T r\,\mathbf{w}}{||\mathbf{w}||}$$

$$= r\frac{\mathbf{w}^T\mathbf{w}}{||\mathbf{w}||} = r\frac{||\mathbf{w}||^2}{||\mathbf{w}||} = r||\mathbf{w}||$$

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

Derivation



In other words, $|g(\mathbf{x})|$ is a measure of the Euclidean distance of the point $\mathbf{x}$ from the decision hyperplane $\mathcal{H}$. If $g(\mathbf{x}) > 0$, we say that the point $\mathbf{x}$ is on the *positive side* of the hyperplane, and if $g(\mathbf{x}) < 0$, we say that point $\mathbf{x}$ is on the *negative side* of the hyperplane. When $g(\mathbf{x}) = 0$, the point $\mathbf{x}$ is on the hyperplane $\mathcal{H}$.

In general, the hyperplane $\mathcal{H}$ divides the feature space into two half-spaces: decision region $\mathcal{H}^+$ (positive side of hyperplane $\mathcal{H}$) for Class 1 ($g(\mathbf{x}) > 0$) and region $\mathcal{H}^-$ (negative side of hyperplane $\mathcal{H}$) for Class 2 ($g(\mathbf{x}) < 0$). The assignment of vector $\mathbf{x}$ to $\mathcal{H}^+$ or $\mathcal{H}^-$ can be implemented as,

$$\mathbf{w}^T\mathbf{x} + w_0 \begin{cases} > 0 \text{ if } \mathbf{x} \in \mathcal{H}^+ \\ = 0 \text{ if } \mathbf{x} \in \mathcal{H} \\ < 0 \text{ if } \mathbf{x} \in \mathcal{H}^- \end{cases} \tag{4.8}$$

DERIVATION

The perpendicular distance d from the coordinates origin to the hyperplane $\mathcal{H}$ is given by $w_0/\|\mathbf{w}\|$, as is seen below.

$$g(\mathbf{x}_d) = \mathbf{w}^T \mathbf{x}_d + w_0 = 0; \mathbf{x}_d = d \frac{\mathbf{w}}{\|\mathbf{w}\|}$$

Therefore,

$$d \frac{\mathbf{w}^T \mathbf{w}}{\|\mathbf{w}\|} + w_0 = 0; d \frac{\|\mathbf{w}\|^2}{\|\mathbf{w}\|} = -w_0; d = \frac{-w_0}{\|\mathbf{w}\|} \qquad (4.9)$$

The origin is on the negative side of $\mathcal{H}$ if $w_0 < 0$, and if $w_0 > 0$, the origin is on the positive side of $\mathcal{H}$. If $w_0 = 0$, the hyperplane passes through the origin.

Geometry for $n = 3$ is shown in Fig. 4.4.

DERIVATION

Let the set of training (data) examples $\mathcal{D}$ be

$$\mathcal{D} = \{\mathbf{x}^{(1)}, y^{(1)}), (\mathbf{x}^{(2)}, y^{(2)}), \ldots, (\mathbf{x}^{(N)}, y^{(N)})\} \tag{4.16}$$

where $\mathbf{x} = [x_1\ x_2\ \ldots\ x_n]^T$ is an n-dimensional *input vector* (pattern with n-features) for the ith example in a real-valued space $\mathbf{X} \subseteq \Re^n$; y is its *class label* (output value), and $y \in \{+1, -1\}$. +1 denotes Class 1 and –1 denotes Class 2.

To build a classifier, SVM finds a linear function of the form

$$g(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + w_0 \tag{4.17}$$

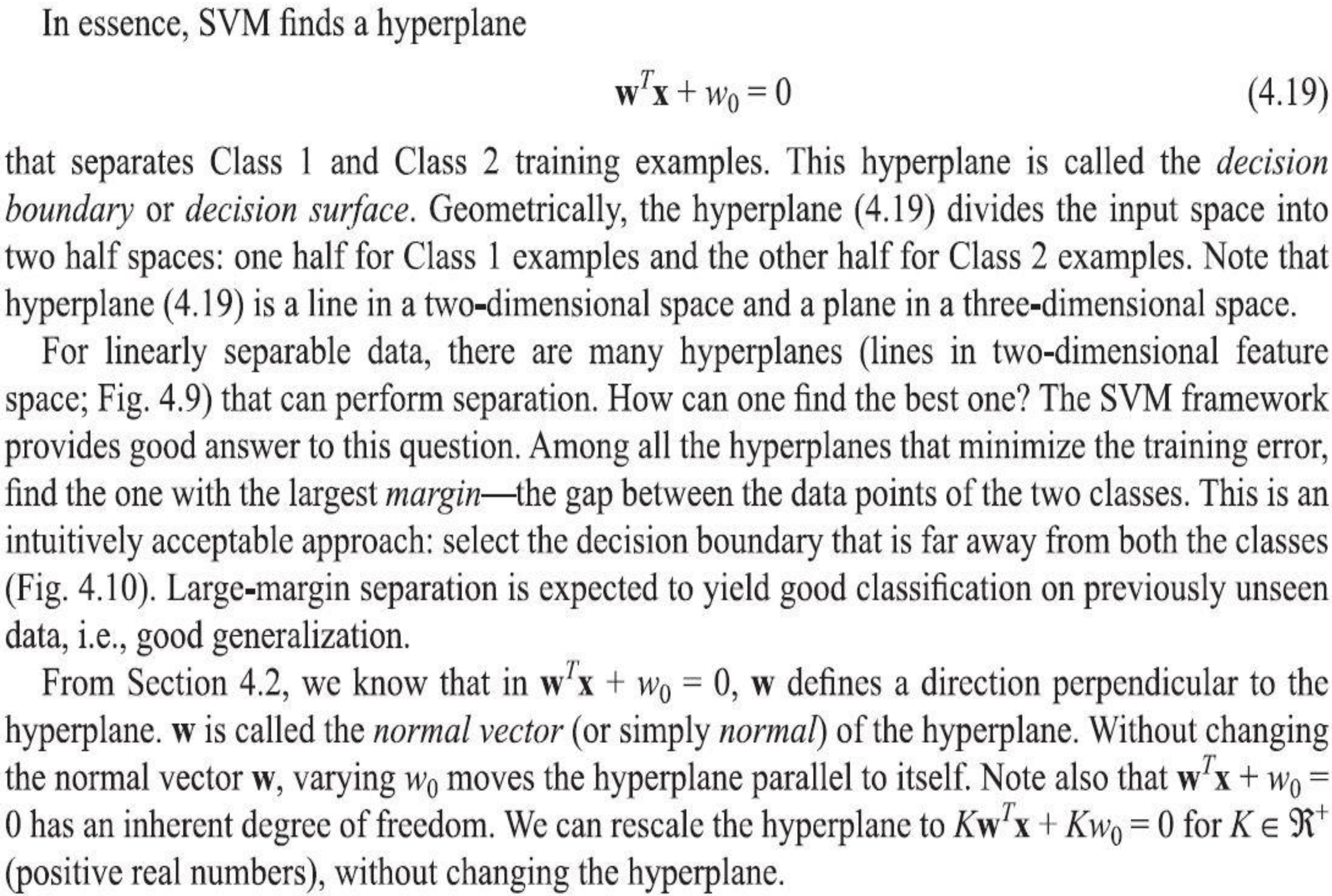
so that the input vector $\mathbf{x}^{(i)}$ is assigned to Class 1 if $g(\mathbf{x}^{(i)}) > 0$, and to Class 2 if $g(\mathbf{x}^{(i)}) < 0$, i.e.,

$$y^{(i)} = \begin{cases} +1 \text{ if } \mathbf{w}^T\mathbf{x}^{(i)} + w_0 > 0 \\ -1 \text{ if } \mathbf{w}^T\mathbf{x}^{(i)} + w_0 < 0 \end{cases} \tag{4.18}$$

Hence, $g(\mathbf{x})$ is a real-valued function; $g: \mathbf{X} \subseteq \Re^n \to \Re$.

$\mathbf{w} = [w_1\ w_2\ \ldots\ w_n]^T \in \Re^n$ is called the *weight vector* and $w_0 \in \Re$ is called the *bias*.

DERIVATION



In essence, SVM finds a hyperplane

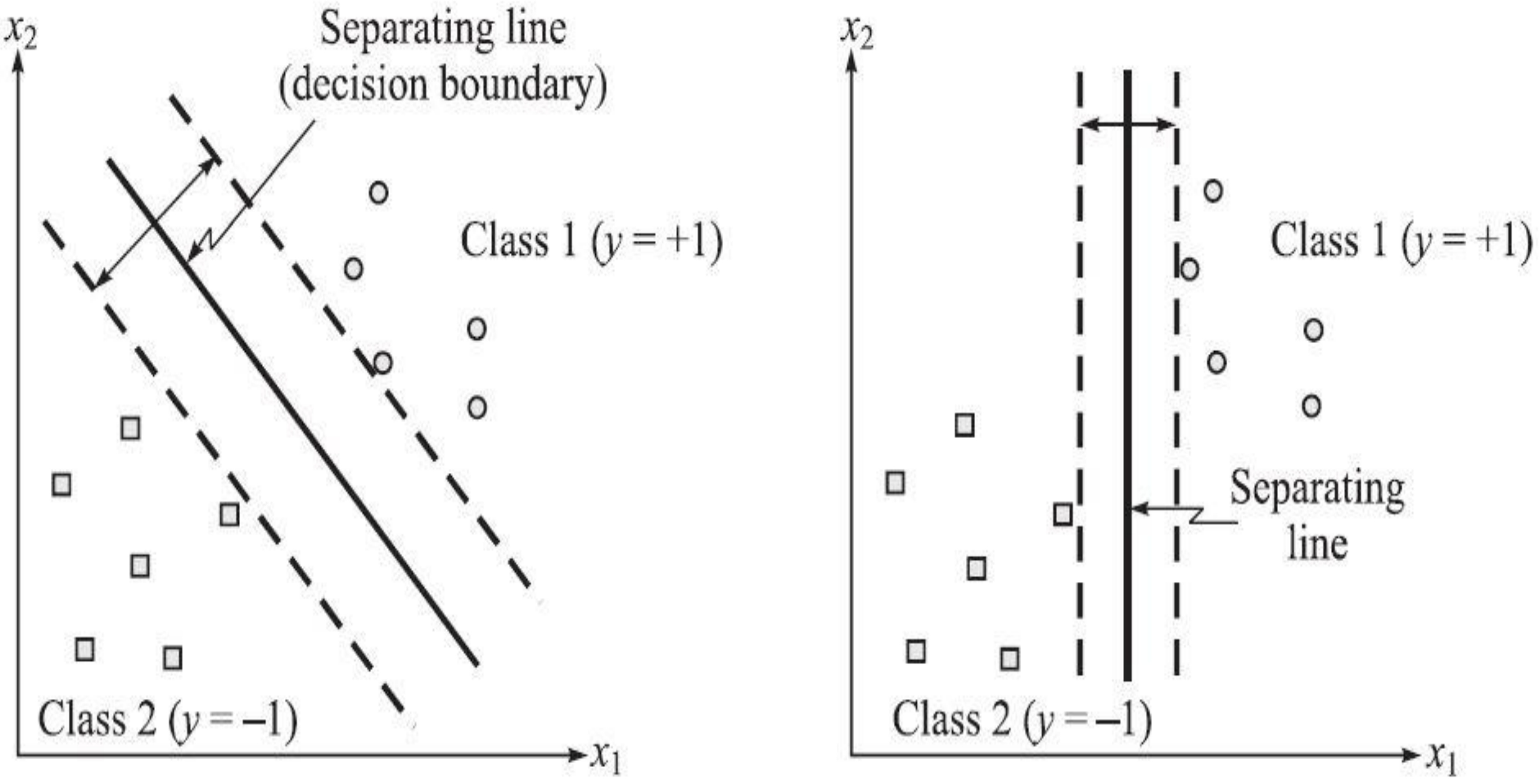
$$\mathbf{w}^T\mathbf{x} + w_0 = 0 \tag{4.19}$$

that separates Class 1 and Class 2 training examples. This hyperplane is called the *decision boundary* or *decision surface*. Geometrically, the hyperplane (4.19) divides the input space into two half spaces: one half for Class 1 examples and the other half for Class 2 examples. Note that hyperplane (4.19) is a line in a two-dimensional space and a plane in a three-dimensional space.

For linearly separable data, there are many hyperplanes (lines in two-dimensional feature space; Fig. 4.9) that can perform separation. How can one find the best one? The SVM framework provides good answer to this question. Among all the hyperplanes that minimize the training error, find the one with the largest *margin*—the gap between the data points of the two classes. This is an intuitively acceptable approach: select the decision boundary that is far away from both the classes (Fig. 4.10). Large-margin separation is expected to yield good classification on previously unseen data, i.e., good generalization.

From Section 4.2, we know that in $\mathbf{w}^T\mathbf{x} + w_0 = 0$, $\mathbf{w}$ defines a direction perpendicular to the hyperplane. $\mathbf{w}$ is called the *normal vector* (or simply *normal*) of the hyperplane. Without changing the normal vector $\mathbf{w}$, varying w_0 moves the hyperplane parallel to itself. Note also that $\mathbf{w}^T\mathbf{x} + w_0 = 0$ has an inherent degree of freedom. We can rescale the hyperplane to $K\mathbf{w}^T\mathbf{x} + Kw_0 = 0$ for $K \in \Re^+$ (positive real numbers), without changing the hyperplane.

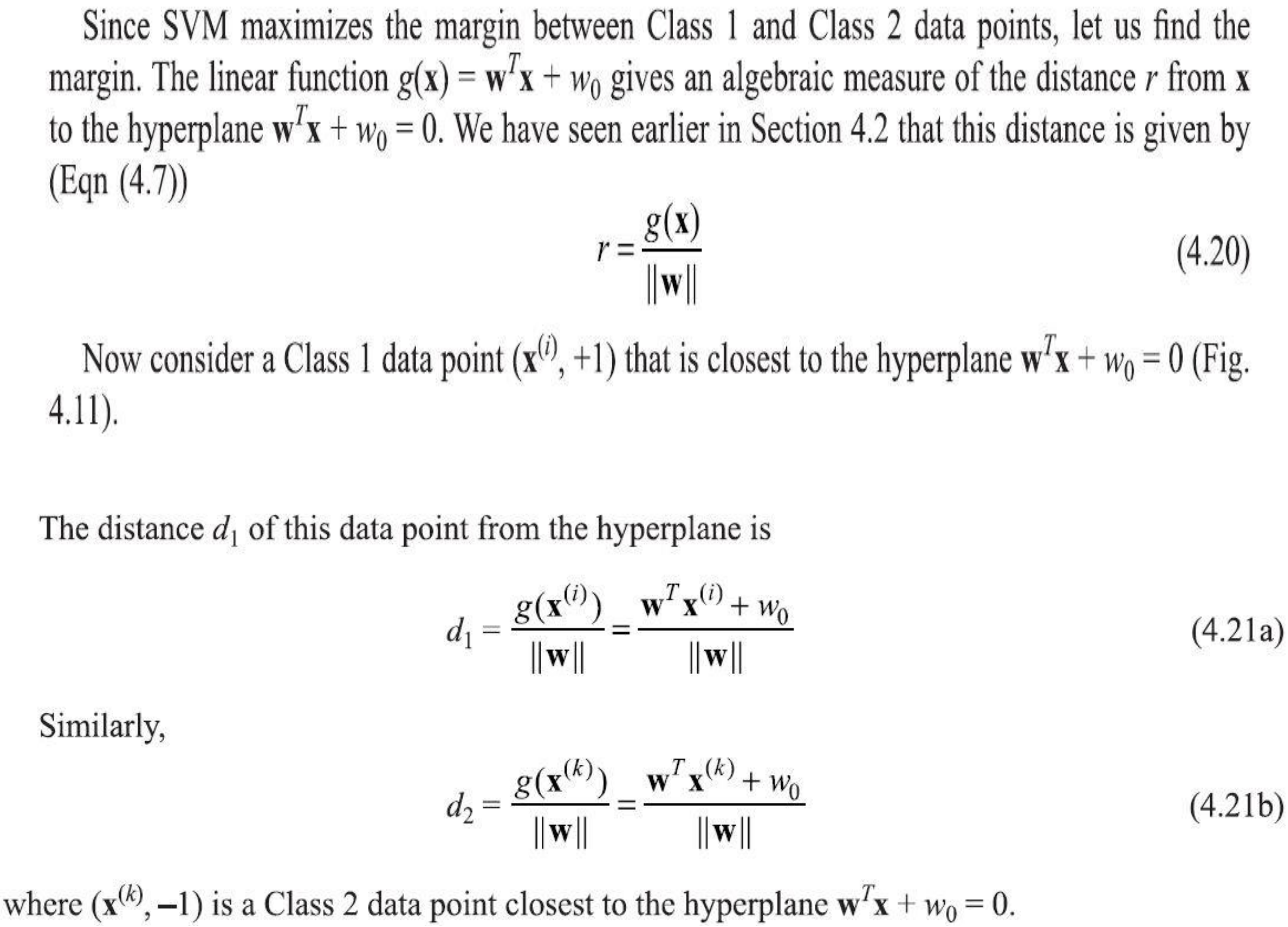
Derivation



(a) Large margin separation

(b) Small margin separation

DERIVATION



Since SVM maximizes the margin between Class 1 and Class 2 data points, let us find the margin. The linear function $g(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + w_0$ gives an algebraic measure of the distance r from $\mathbf{x}$ to the hyperplane $\mathbf{w}^T\mathbf{x} + w_0 = 0$. We have seen earlier in Section 4.2 that this distance is given by (Eqn (4.7))

$$r = \frac{g(\mathbf{x})}{\|\mathbf{w}\|} \tag{4.20}$$

Now consider a Class 1 data point $(\mathbf{x}^{(i)}, +1)$ that is closest to the hyperplane $\mathbf{w}^T\mathbf{x} + w_0 = 0$ (Fig. 4.11).

The distance d_1 of this data point from the hyperplane is

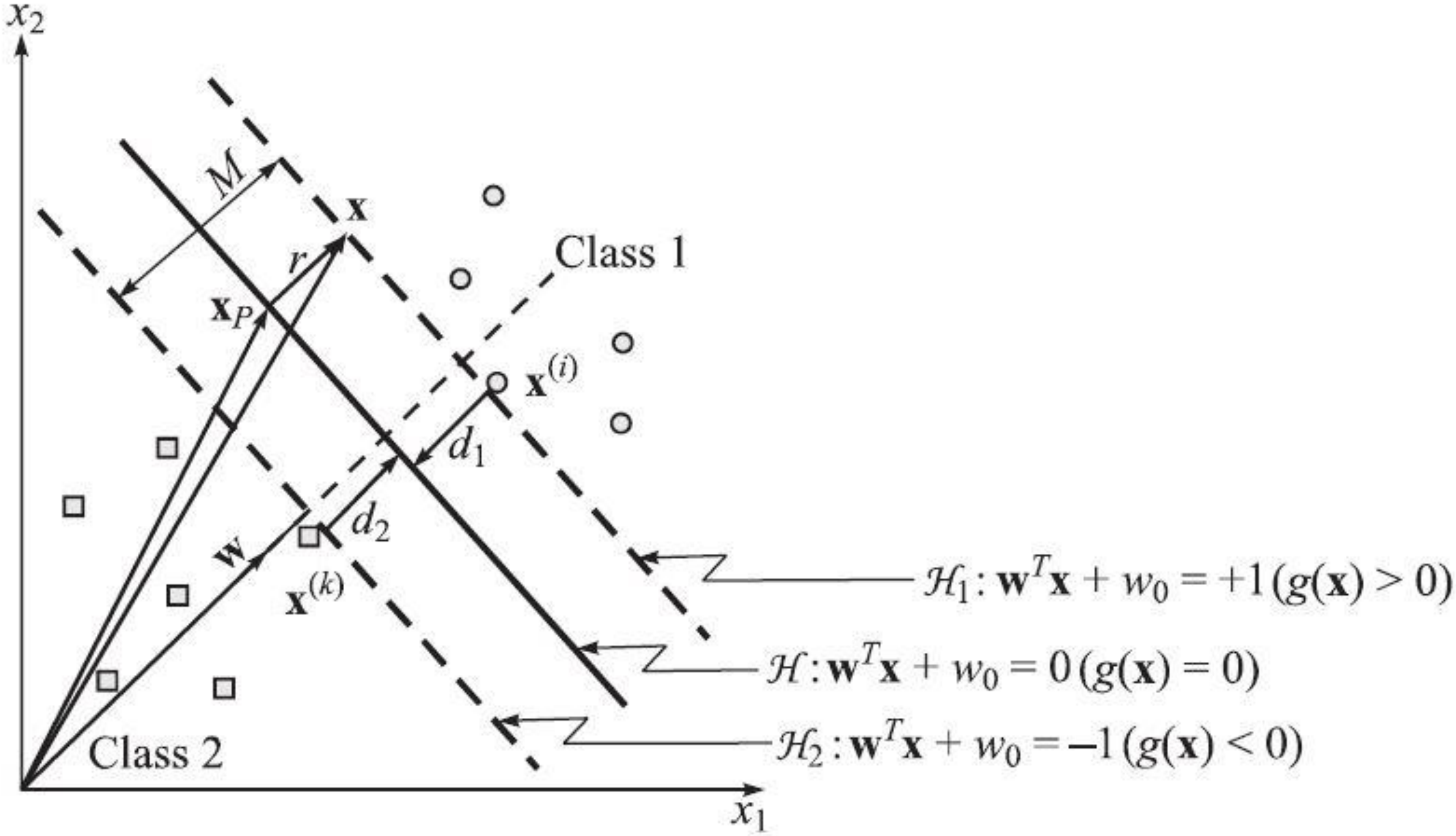
$$d_1 = \frac{g(\mathbf{x}^{(i)})}{\|\mathbf{w}\|} = \frac{\mathbf{w}^T\mathbf{x}^{(i)} + w_0}{\|\mathbf{w}\|} \tag{4.21a}$$

Similarly,

$$d_2 = \frac{g(\mathbf{x}^{(k)})}{\|\mathbf{w}\|} = \frac{\mathbf{w}^T\mathbf{x}^{(k)} + w_0}{\|\mathbf{w}\|} \tag{4.21b}$$

where $(\mathbf{x}^{(k)}, -1)$ is a Class 2 data point closest to the hyperplane $\mathbf{w}^T\mathbf{x} + w_0 = 0$.

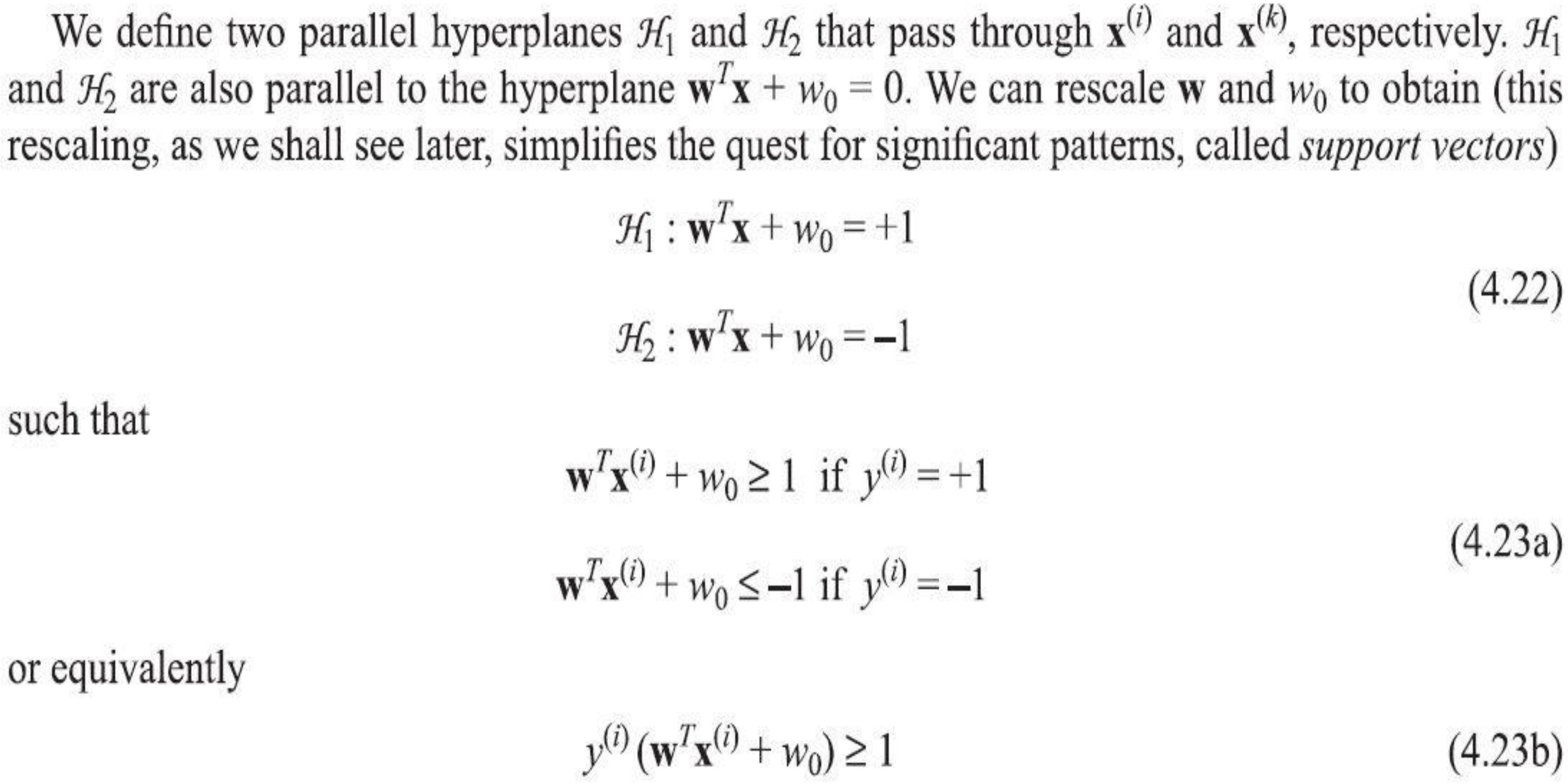
DERIVATION



x_2
M
$\mathbf{x}$
r
Class 1
$\mathbf{x}_P$
$\mathbf{x}^{(i)}$
d_1
d_2
$\mathbf{w}$
$\mathbf{x}^{(k)}$
$\mathcal{H}_1 : \mathbf{w}^T\mathbf{x} + w_0 = +1\,(g(\mathbf{x}) > 0)$
$\mathcal{H} : \mathbf{w}^T\mathbf{x} + w_0 = 0\,(g(\mathbf{x}) = 0)$
$\mathcal{H}_2 : \mathbf{w}^T\mathbf{x} + w_0 = -1\,(g(\mathbf{x}) < 0)$
Class 2
x_1

Figure 4.11 Geometric interpretation of algebraic distances of points to a hyperplane for two-dimenstional case

Derivation



We define two parallel hyperplanes $\mathcal{H}_1$ and $\mathcal{H}_2$ that pass through $\mathbf{x}^{(i)}$ and $\mathbf{x}^{(k)}$, respectively. $\mathcal{H}_1$ and $\mathcal{H}_2$ are also parallel to the hyperplane $\mathbf{w}^T\mathbf{x} + w_0 = 0$. We can rescale $\mathbf{w}$ and w_0 to obtain (this rescaling, as we shall see later, simplifies the quest for significant patterns, called *support vectors*)

$$\begin{aligned} \mathcal{H}_1 &: \mathbf{w}^T\mathbf{x} + w_0 = +1 \\ \mathcal{H}_2 &: \mathbf{w}^T\mathbf{x} + w_0 = -1 \end{aligned} \tag{4.22}$$

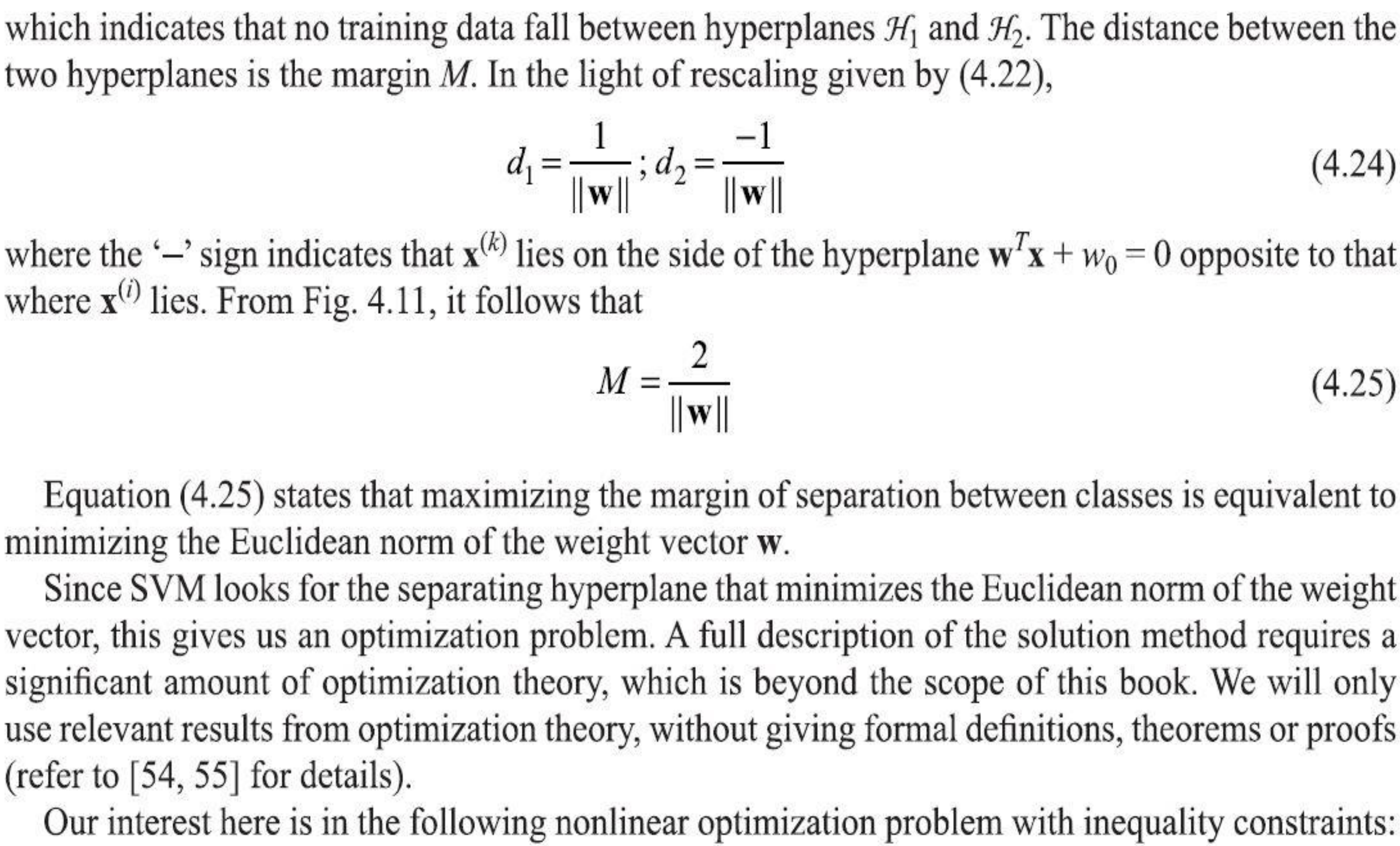
such that

$$\begin{aligned} \mathbf{w}^T\mathbf{x}^{(i)} + w_0 &\geq 1 \text{ if } y^{(i)} = +1 \\ \mathbf{w}^T\mathbf{x}^{(i)} + w_0 &\leq -1 \text{ if } y^{(i)} = -1 \end{aligned} \tag{4.23a}$$

or equivalently

$$y^{(i)}(\mathbf{w}^T\mathbf{x}^{(i)} + w_0) \geq 1 \tag{4.23b}$$

DERIVATION



which indicates that no training data fall between hyperplanes $\mathcal{H}_1$ and $\mathcal{H}_2$. The distance between the two hyperplanes is the margin M. In the light of rescaling given by (4.22),

$$d_1 = \frac{1}{\|\mathbf{w}\|}\,;\, d_2 = \frac{-1}{\|\mathbf{w}\|} \tag{4.24}$$

where the '–' sign indicates that $\mathbf{x}^{(k)}$ lies on the side of the hyperplane $\mathbf{w}^T\mathbf{x} + w_0 = 0$ opposite to that where $\mathbf{x}^{(i)}$ lies. From Fig. 4.11, it follows that

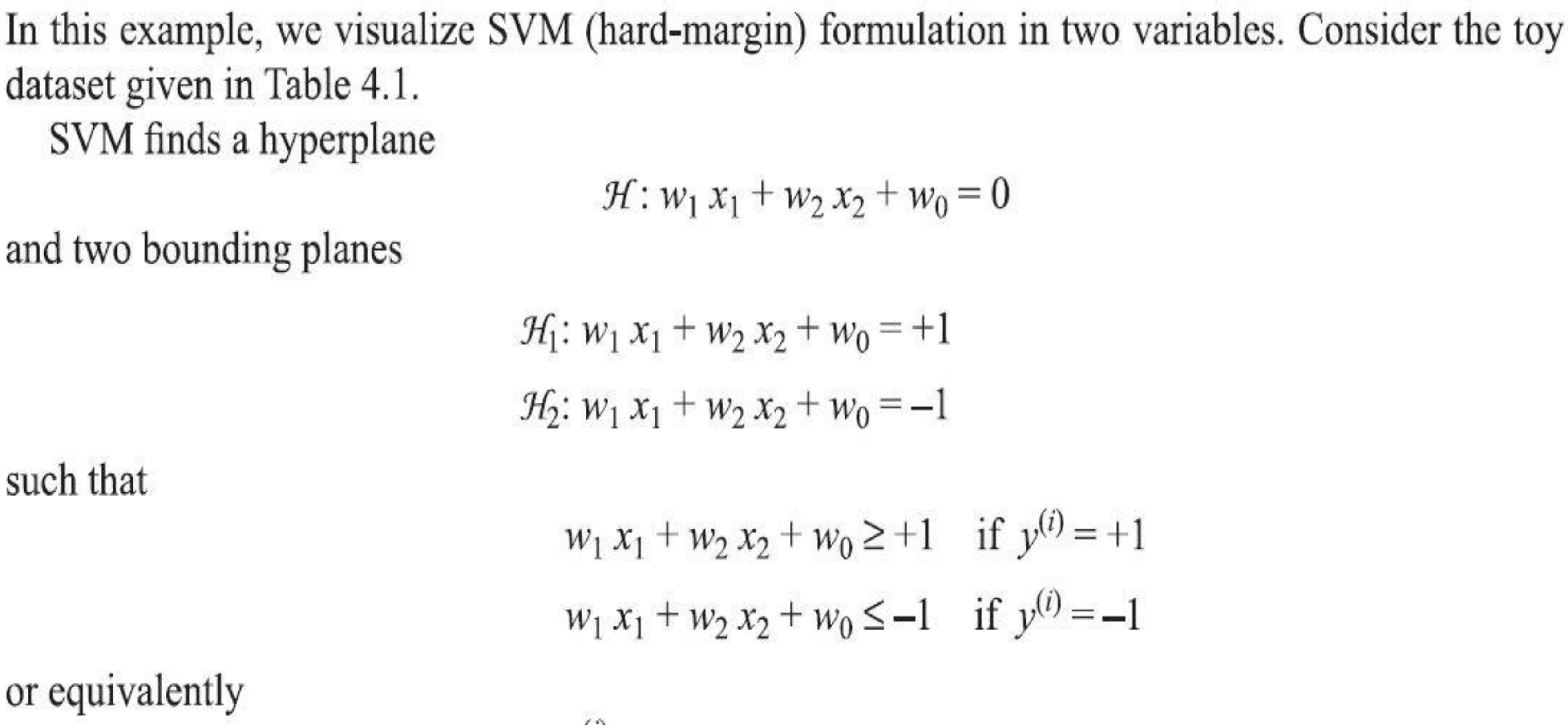
$$M = \frac{2}{\|\mathbf{w}\|} \tag{4.25}$$

Equation (4.25) states that maximizing the margin of separation between classes is equivalent to minimizing the Euclidean norm of the weight vector $\mathbf{w}$.

Since SVM looks for the separating hyperplane that minimizes the Euclidean norm of the weight vector, this gives us an optimization problem. A full description of the solution method requires a significant amount of optimization theory, which is beyond the scope of this book. We will only use relevant results from optimization theory, without giving formal definitions, theorems or proofs (refer to [54, 55] for details).

Our interest here is in the following nonlinear optimization problem with inequality constraints:

SVM : Example



In this example, we visualize SVM (hard-margin) formulation in two variables. Consider the toy dataset given in Table 4.1.

SVM finds a hyperplane

$$\mathcal{H}: w_1 x_1 + w_2 x_2 + w_0 = 0$$

and two bounding planes

$$\mathcal{H}_1: w_1 x_1 + w_2 x_2 + w_0 = +1$$

$$\mathcal{H}_2: w_1 x_1 + w_2 x_2 + w_0 = -1$$

such that

$$w_1 x_1 + w_2 x_2 + w_0 \geq +1 \quad \text{if } y^{(i)} = +1$$

$$w_1 x_1 + w_2 x_2 + w_0 \leq -1 \quad \text{if } y^{(i)} = -1$$

or equivalently

SVM : Example

We write these constraints explicitly as (refer to Table 4.1),

$(-1)\,[w_1 + w_2 + w_0] \geq 1$

$(-1)\,[2w_1 + w_2 + w_0] \geq 1$

$(-1)\,[w_1 + 2w_2 + w_0] \geq 1$

$(-1)\,[2w_1 + 2w_2 + w_0] \geq 1$

$(+1)\,[4w_1 + 4w_2 + w_0] \geq 1$

$(+1)\,[4w_1 + 5w_2 + w_0] \geq 1$

$(+1)\,[5w_1 + 4w_2 + w_0] \geq 1$

$(+1)\,[5w_1 + 5w_2 + w_0] \geq 1$

SVM : Example

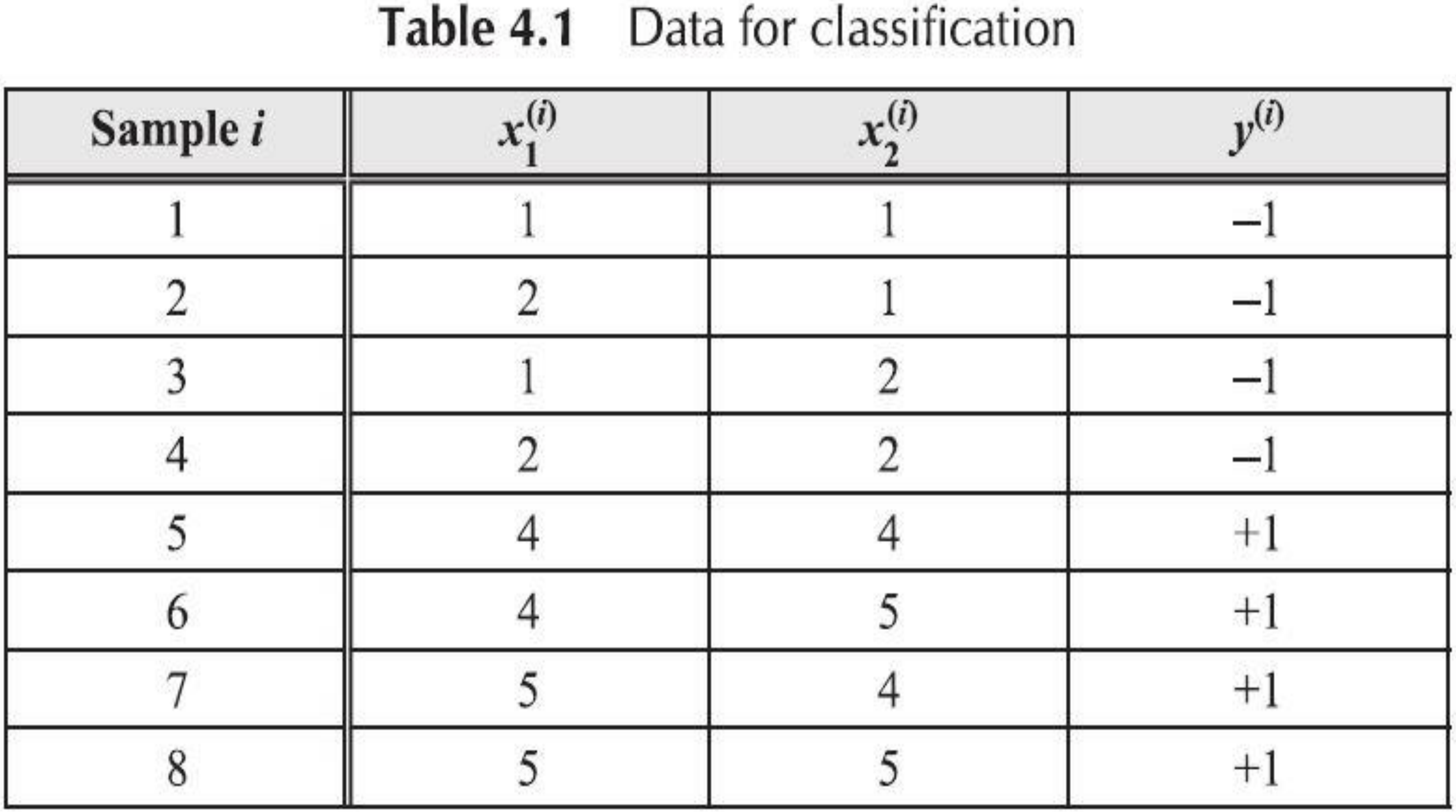


Table 4.1 Data for classification

Sample *i*	$x_1^{(i)}$	$x_2^{(i)}$	$y^{(i)}$
1	1	1	–1
2	2	1	–1
3	1	2	–1
4	2	2	–1
5	4	4	+1
6	4	5	+1
7	5	4	+1
8	5	5	+1

SVM : Example

THE DUAL PROBLEM

- The new objective function is in terms of α_i only
- It is known as the dual problem: if we know **w**, we know all $\boldsymbol{\alpha}_i$; if we know all $\boldsymbol{\alpha}_i$, we know **w**
- The original problem is known as the primal problem
- The objective function of the dual problem needs to be maximized!
- The dual problem is therefore:

$$\text{max. } W(\boldsymbol{\alpha}) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1,j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$$

$$\text{subject to } \alpha_i \geq 0, \qquad \sum_{i=1}^{n} \alpha_i y_i = 0$$

Properties of α_i when we introduce the Lagrange multipliers

The result when we differentiate the original Lagrangian w.r.t. b

The Dual Problem

$$\text{max.} \quad W(\boldsymbol{\alpha}) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1,j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$$

$$\text{subject to } \alpha_i \geq 0, \sum_{i=1}^{n} \alpha_i y_i = 0$$

- This is a quadratic programming (QP) problem
  - A global maximum of α_i can always be found

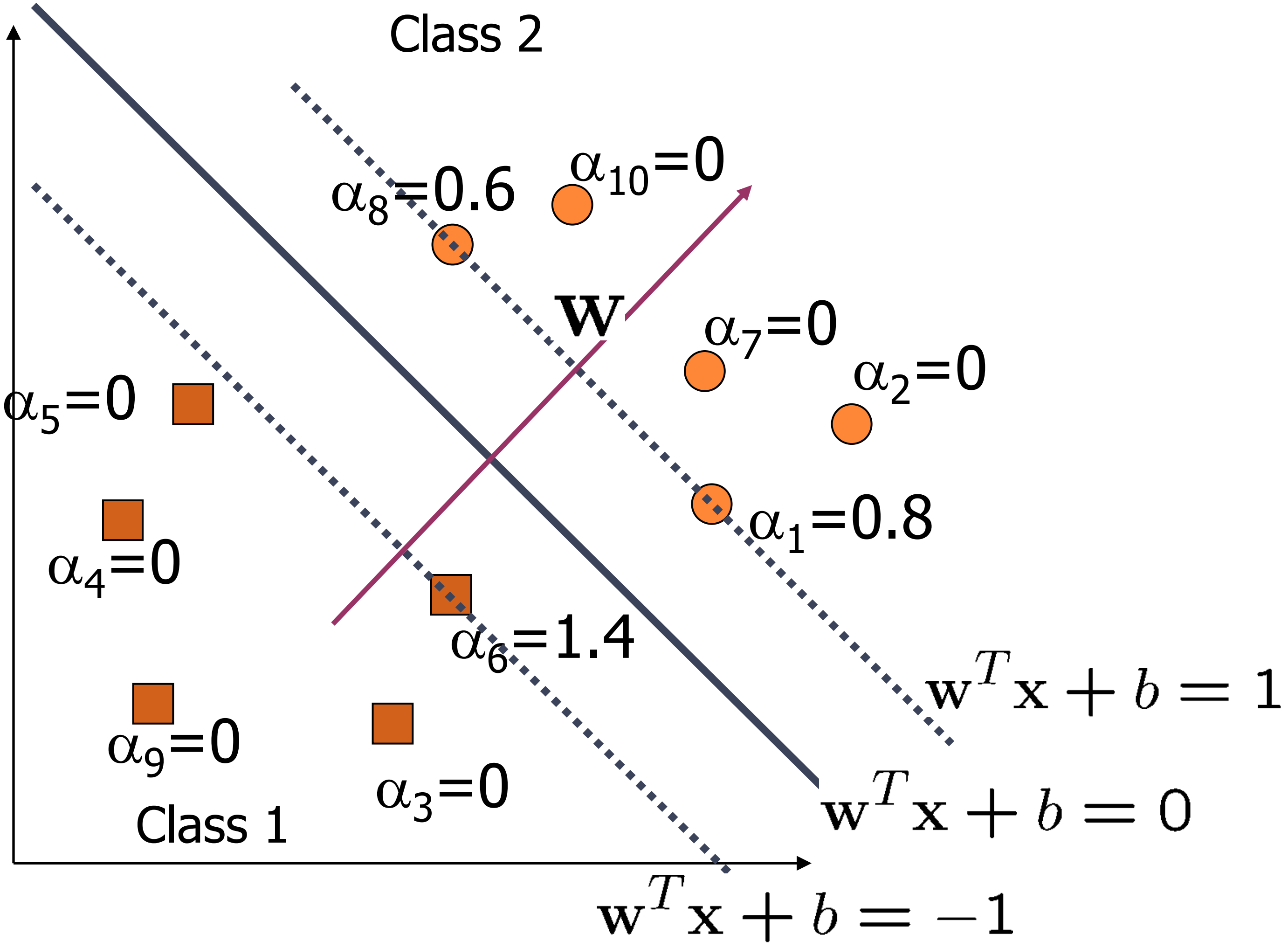
- **w** can be recovered by

$$\mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i$$

Characteristics of the Solution

- Many of the α_i are zero (see next page for example)
  - **w** is a linear combination of a small number of data points
  - This "sparse" representation can be viewed as data compression as in the construction of KNN classifier
- $\mathbf{x}_i$ with non-zero α_i are called support vectors (SV)
  - The decision boundary is determined only by the SV
  - Let t_j (j=1, ..., s) be the indices of the s support vectors. We can write $\mathbf{w} = \sum_{j=1}^{s} \alpha_{t_j} y_{t_j} \mathbf{x}_{t_j}$
- For testing with a new data **z**
  - Compute $\mathbf{w}^T\mathbf{z} + b = \sum_{j=1}^{s} \alpha_{t_j} y_{t_j} (\mathbf{x}_{t_j}^T \mathbf{z}) + b$ and classify **z** as class 1 if the sum is positive, and class 2 otherwise
  - Note: **w** need not be formed explicitly

Support Vector: Geometrical Interpretation



Linear SVM: Example

Suppose we are given the following positively labeled data points in $\Re^2$:

$$\left\{ \begin{pmatrix} 3 \\ 1 \end{pmatrix}, \begin{pmatrix} 3 \\ -1 \end{pmatrix}, \begin{pmatrix} 6 \\ 1 \end{pmatrix}, \begin{pmatrix} 6 \\ -1 \end{pmatrix} \right\}$$

and the following negatively labeled data points in $\Re^2$ (see Figure 1):

$$\left\{ \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \begin{pmatrix} 0 \\ -1 \end{pmatrix}, \begin{pmatrix} -1 \\ 0 \end{pmatrix} \right\}$$

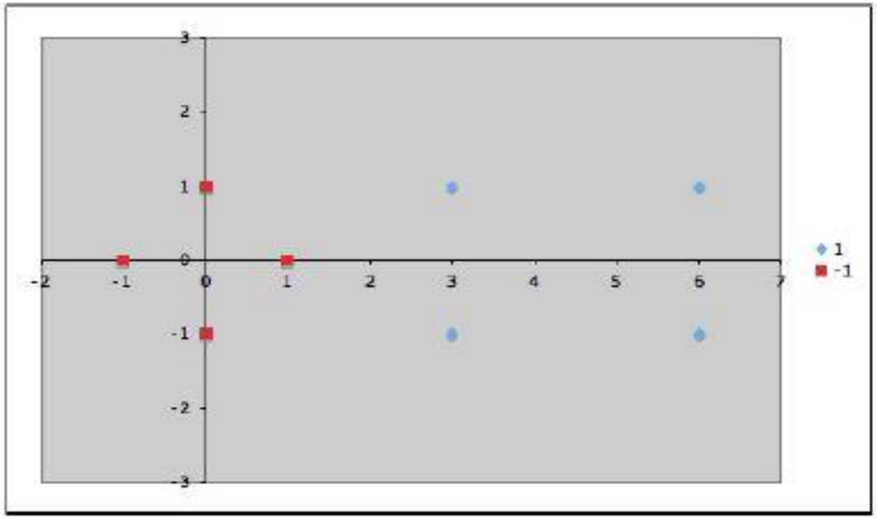


Figure 1: Sample data points in $\Re^2$. Blue diamonds are positive examples and red squares are negative examples.

Linear SVM: Example

- We would like to discover a simple SVM that accurately discriminates the two classes. Since the data is linearly separable, we can use a linear SVM.
- By inspection, it is obvious that there are three support vectors.

$$\left\{ s_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, s_2 = \begin{pmatrix} 3 \\ 1 \end{pmatrix}, s_3 = \begin{pmatrix} 3 \\ -1 \end{pmatrix} \right\}$$

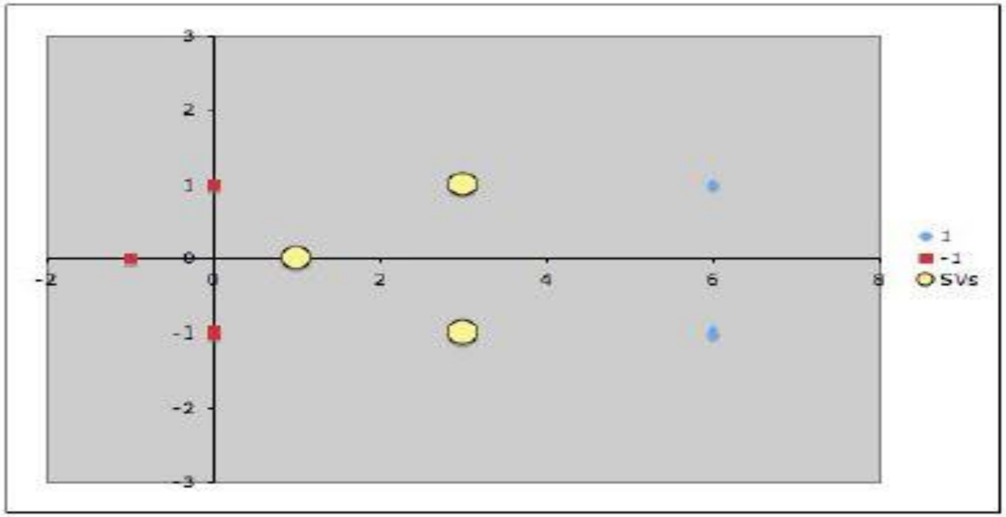


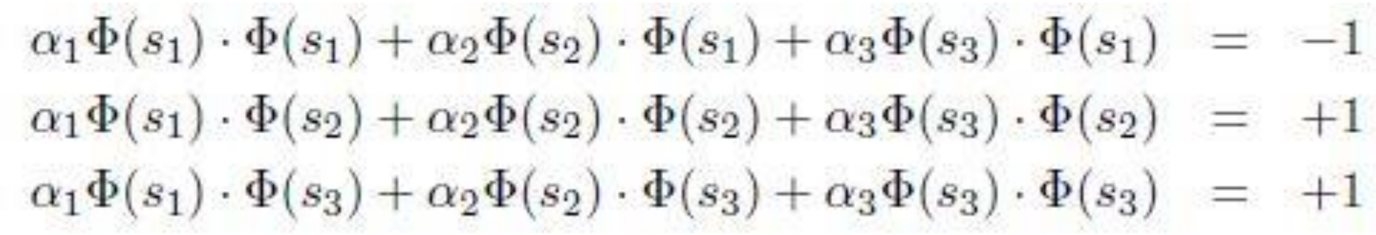
Figure 2: The three support vectors are marked as yellow circles.

LINEAR SVM: EXAMPLE

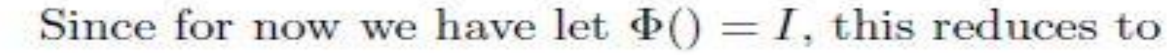
- In what follows we will use vectors augmented with a 1 as a bias input, and for clarity we will differentiate these with an over-tilde.

  So, if s1 = (10), then ~ s1 = (101).

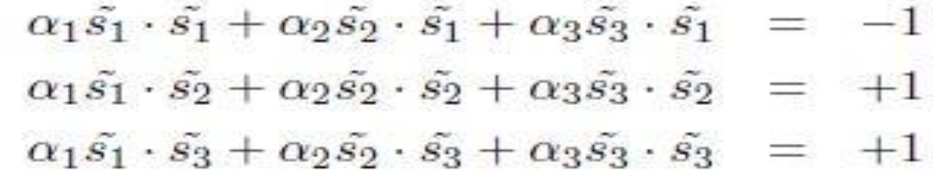
- task is to find values for the i such that (based on SVM architecture)



$$\begin{aligned}
\alpha_1\Phi(s_1)\cdot\Phi(s_1)+\alpha_2\Phi(s_2)\cdot\Phi(s_1)+\alpha_3\Phi(s_3)\cdot\Phi(s_1) &= -1\\
\alpha_1\Phi(s_1)\cdot\Phi(s_2)+\alpha_2\Phi(s_2)\cdot\Phi(s_2)+\alpha_3\Phi(s_3)\cdot\Phi(s_2) &= +1\\
\alpha_1\Phi(s_1)\cdot\Phi(s_3)+\alpha_2\Phi(s_2)\cdot\Phi(s_3)+\alpha_3\Phi(s_3)\cdot\Phi(s_3) &= +1
\end{aligned}$$

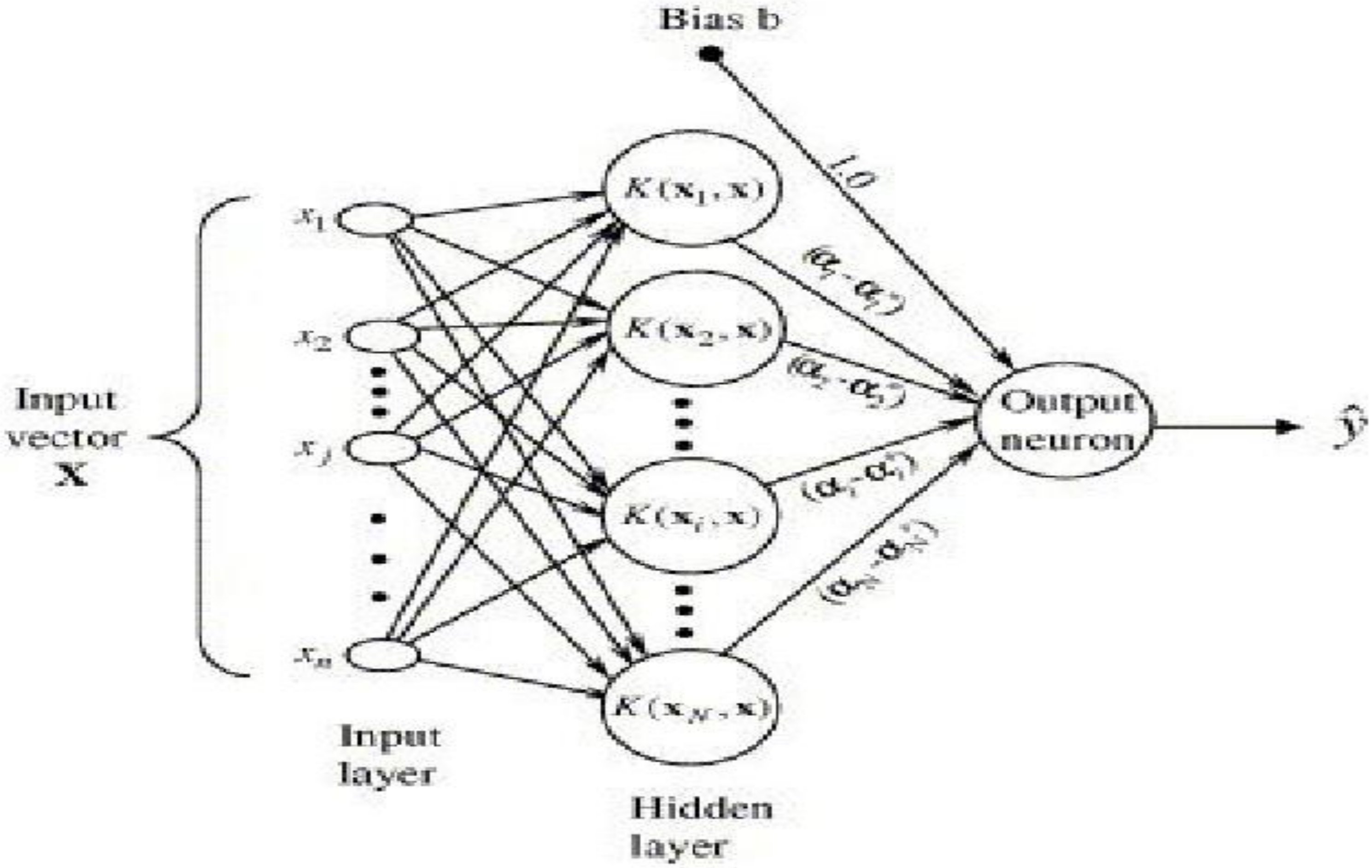


Since for now we have let $\Phi() = I$, this reduces to

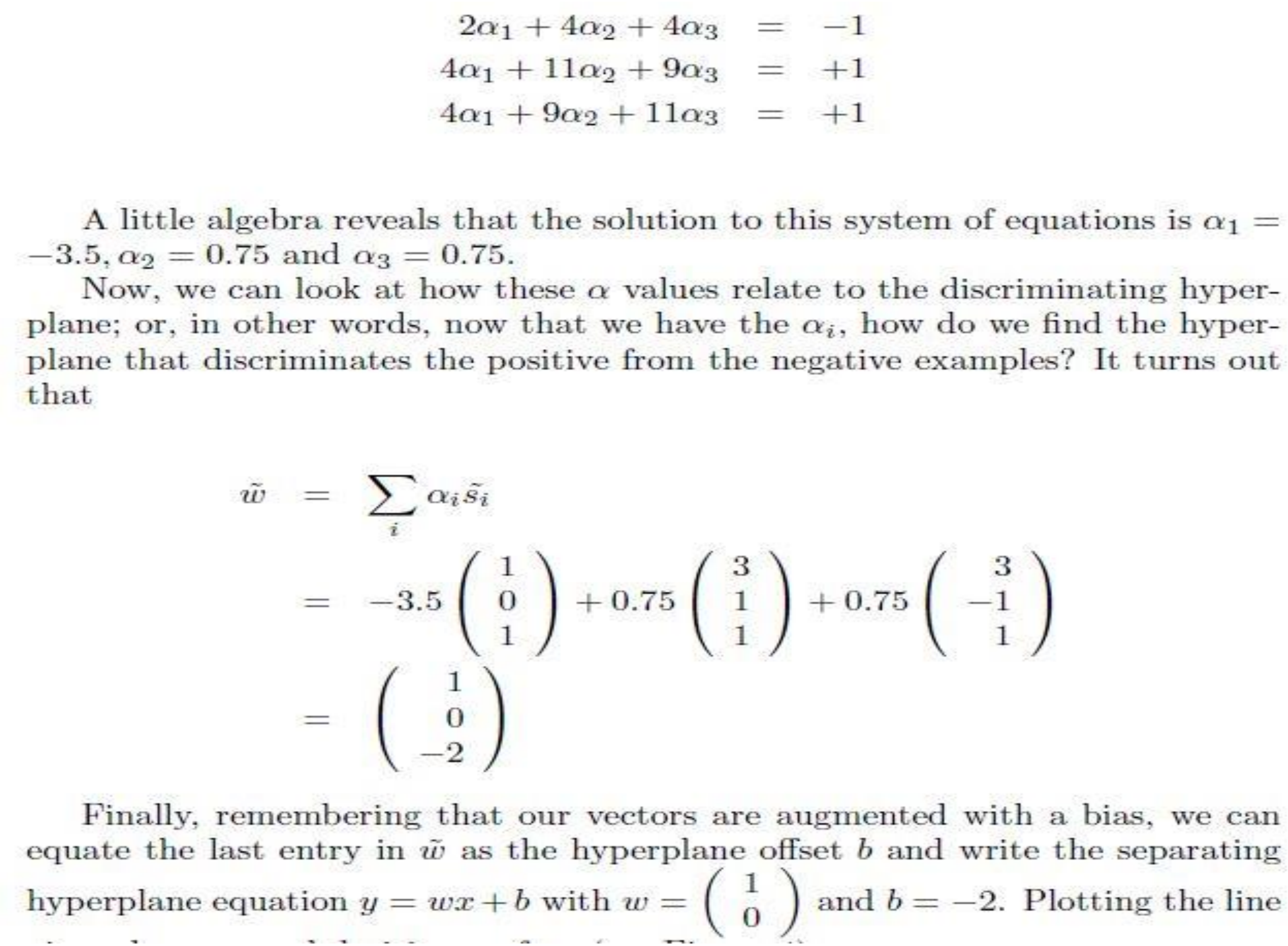


$$\begin{aligned}
\alpha_1\tilde{s_1}\cdot\tilde{s_1}+\alpha_2\tilde{s_2}\cdot\tilde{s_1}+\alpha_3\tilde{s_3}\cdot\tilde{s_1} &= -1\\
\alpha_1\tilde{s_1}\cdot\tilde{s_2}+\alpha_2\tilde{s_2}\cdot\tilde{s_2}+\alpha_3\tilde{s_3}\cdot\tilde{s_2} &= +1\\
\alpha_1\tilde{s_1}\cdot\tilde{s_3}+\alpha_2\tilde{s_2}\cdot\tilde{s_3}+\alpha_3\tilde{s_3}\cdot\tilde{s_3} &= +1
\end{aligned}$$

Support Vector Architecture


Bias b
1.0
Input vector
X
x_1
x_2
x_j
x_n
Input layer
$K(\mathbf{x}_1, \mathbf{x})$
$K(\mathbf{x}_2, \mathbf{x})$
$K(\mathbf{x}_i, \mathbf{x})$
$K(\mathbf{x}_N, \mathbf{x})$
Hidden layer
$(\alpha_1 - \alpha_1^*)$
$(\alpha_2 - \alpha_2^*)$
$(\alpha_i - \alpha_i^*)$
$(\alpha_N - \alpha_N^*)$
Output neuron
$\hat{y}$

Example Continues…



$$\begin{aligned} 2\alpha_1 + 4\alpha_2 + 4\alpha_3 &= -1 \\ 4\alpha_1 + 11\alpha_2 + 9\alpha_3 &= +1 \\ 4\alpha_1 + 9\alpha_2 + 11\alpha_3 &= +1 \end{aligned}$$

A little algebra reveals that the solution to this system of equations is $\alpha_1 = -3.5$, $\alpha_2 = 0.75$ and $\alpha_3 = 0.75$.

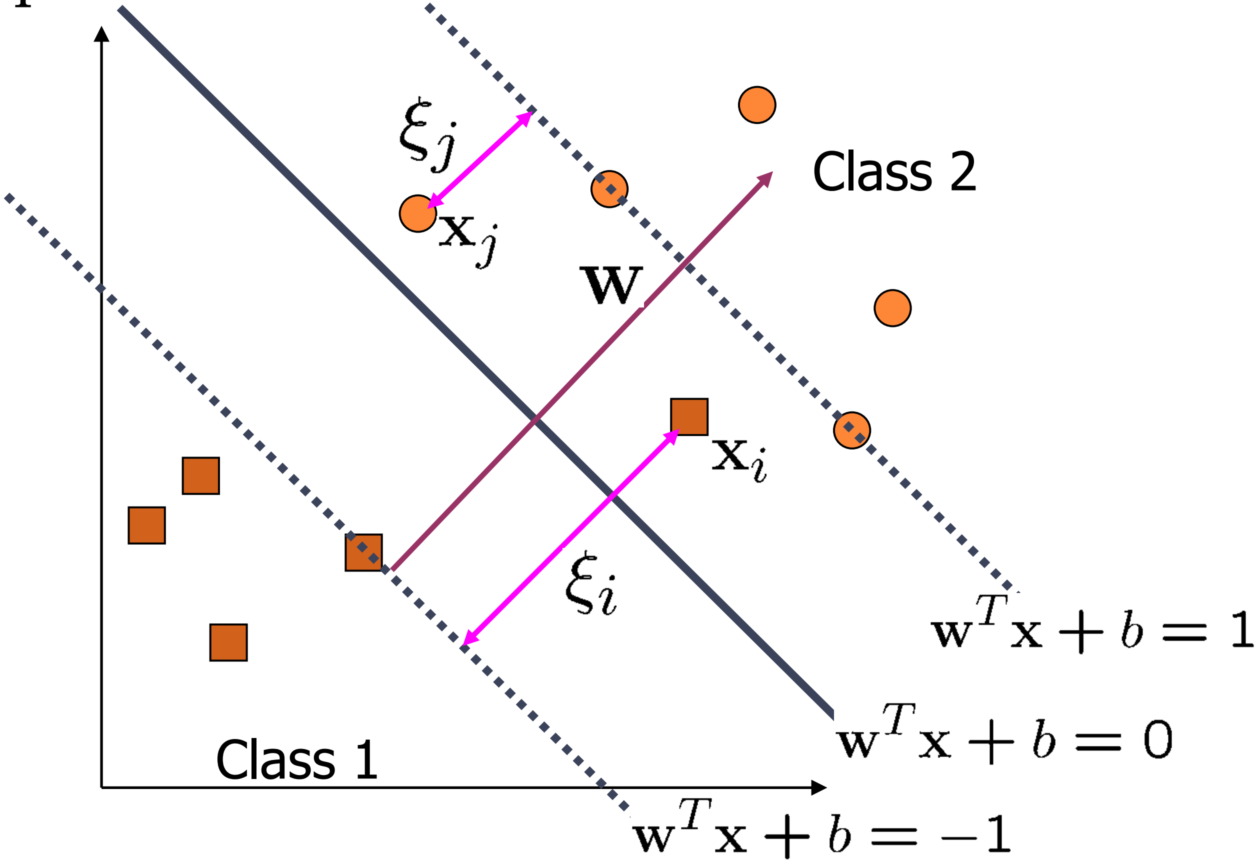
Now, we can look at how these α values relate to the discriminating hyperplane; or, in other words, now that we have the α_i, how do we find the hyperplane that discriminates the positive from the negative examples? It turns out that

$$\begin{aligned} \tilde{w} &= \sum_i \alpha_i \tilde{s}_i \\ &= -3.5 \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} + 0.75 \begin{pmatrix} 3 \\ 1 \\ 1 \end{pmatrix} + 0.75 \begin{pmatrix} 3 \\ -1 \\ 1 \end{pmatrix} \\ &= \begin{pmatrix} 1 \\ 0 \\ -2 \end{pmatrix} \end{aligned}$$

Finally, remembering that our vectors are augmented with a bias, we can equate the last entry in $\tilde{w}$ as the hyperplane offset b and write the separating hyperplane equation $y = wx + b$ with $w = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ and $b = -2$. Plotting the line

Allowing errors in our solutions

- We allow "error" ξ_i in classification; it is based on the output of the discriminant function $\mathbf{w}^T\mathbf{x}$+b
- ξ_i approximates the number of misclassified samples



ξ_j
$\mathbf{x}_j$
Class 2
$\mathbf{w}$
$\mathbf{x}_i$
ξ_i
$\mathbf{w}^T\mathbf{x} + b = 1$
$\mathbf{w}^T\mathbf{x} + b = 0$
Class 1
$\mathbf{w}^T\mathbf{x} + b = -1$

Soft Margin Hyperplane

- If we minimize $\sum_i \xi_i$, ξ_i can be computed by

$$\begin{cases} \mathbf{w}^T\mathbf{x}_i + b \geq 1 - \xi_i & y_i = 1 \\ \mathbf{w}^T\mathbf{x}_i + b \leq -1 + \xi_i & y_i = -1 \\ \xi_i \geq 0 & \forall i \end{cases}$$

  - ξ_i are "slack variables" in optimization
  - Note that ξ_i=0 if there is no error for $\mathbf{x}_i$
  - ξ_i is an upper bound of t he number of errors

- We want to minimize $\frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{n}\xi_i$
  - C : tradeoff parameter between error and margin
- The optimization problem becomes

$$\begin{aligned}&\text{Minimize } \tfrac{1}{2}||\mathbf{w}||^2 + C\textstyle\sum_{i=1}^{n}\xi_i \\ &\text{subject to } y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0\end{aligned}$$

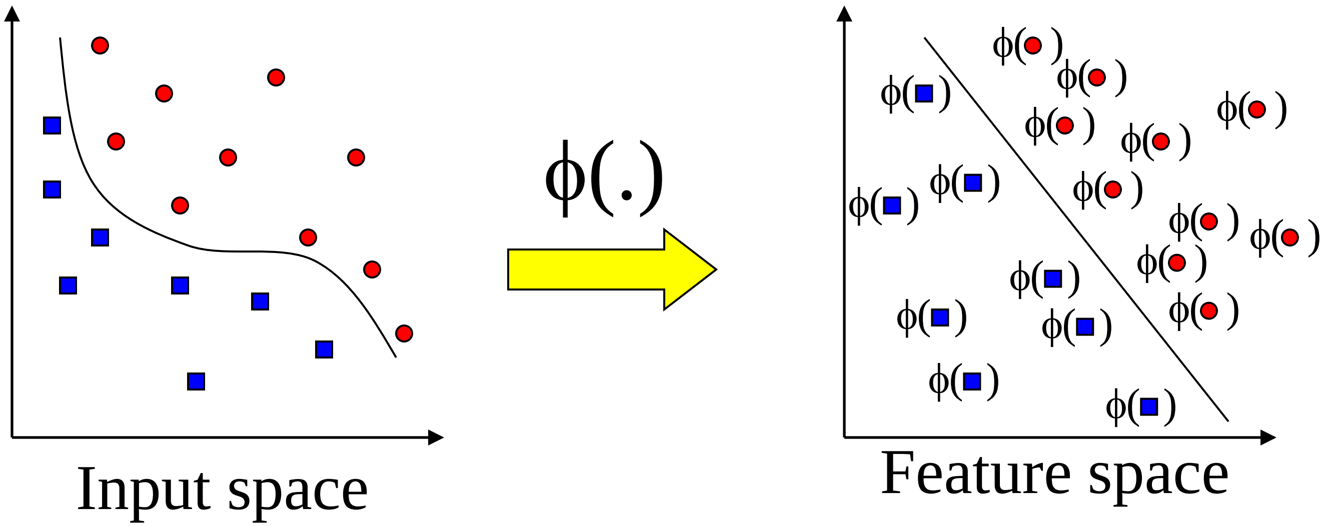
Extension to Non-linear Decision Boundary

- So far, we have only considered large-margin classifier with a linear decision boundary

- How to generalize it to become nonlinear?

- Key idea: transform $\mathbf{x}_i$ to a higher dimensional space to "make life easier"
  - Input space: the space the point $\mathbf{x}_i$ are located
  - Feature space: the space of $\phi(\mathbf{x}_i)$ after transformation

TRANSFORMING THE DATA



Note: feature space is of higher dimension than the input space in practice

- Computation in the feature space can be costly because it is high dimensional
  - The feature space is typically infinite-dimensional!
- The kernel trick comes to rescue

The Kernel Trick

- Recall the SVM optimization problem

$$\text{max. } W(\boldsymbol{\alpha}) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1,j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$$

$$\text{subject to } C \geq \alpha_i \geq 0, \sum_{i=1}^{n} \alpha_i y_i = 0$$

- The data points only appear as inner product
- As long as we can calculate the inner product in the feature space, we do not need the mapping explicitly
- Many common geometric operations (angles, distances) can be expressed by inner products
- Define the kernel function K by

$$K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$$

An Example for Φ(.) and K(.,.)

- Suppose ϕ(.) is given as follows

$$\phi(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}) = (1, \sqrt{2}x_1, \sqrt{2}x_2, x_1^2, x_2^2, \sqrt{2}x_1x_2)$$

- An inner product in the feature space is

$$\langle \phi(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}), \phi(\begin{bmatrix} y_1 \\ y_2 \end{bmatrix}) \rangle = (1 + x_1y_1 + x_2y_2)^2$$

- So, if we define the kernel function as follows, there is no need to carry out ϕ(.) explicitly

$$K(\mathbf{x}, \mathbf{y}) = (1 + x_1y_1 + x_2y_2)^2$$

- This use of kernel function to avoid carrying out ϕ(.) explicitly is known as the kernel trick

More on Kernel Functions

- Not all similarity measures can be used as kernel function, however
  - The kernel function needs to satisfy the Mercer function, i.e., the function is “positive-definite”
- This implies that
  - the n by n kernel matrix,
  - in which the (i,j)-th entry is the $K(\mathbf{x}_i, \mathbf{x}_j)$, is always positive definite
- This also means that optimization problem can be solved in polynomial time!

Examples of Kernel Functions

- **Polynomial** kernel with degree *d*

$$K(\mathbf{x},\mathbf{y}) = (\mathbf{x}^T\mathbf{y} + 1)^d$$

- **Gaussian:** Radial basis function kernel with width σ

$$K(\mathbf{x},\mathbf{y}) = \exp(-||\mathbf{x}-\mathbf{y}||^2/(2\sigma^2))$$

  - Closely related to radial basis function neural networks
  - The feature space is infinite-dimensional

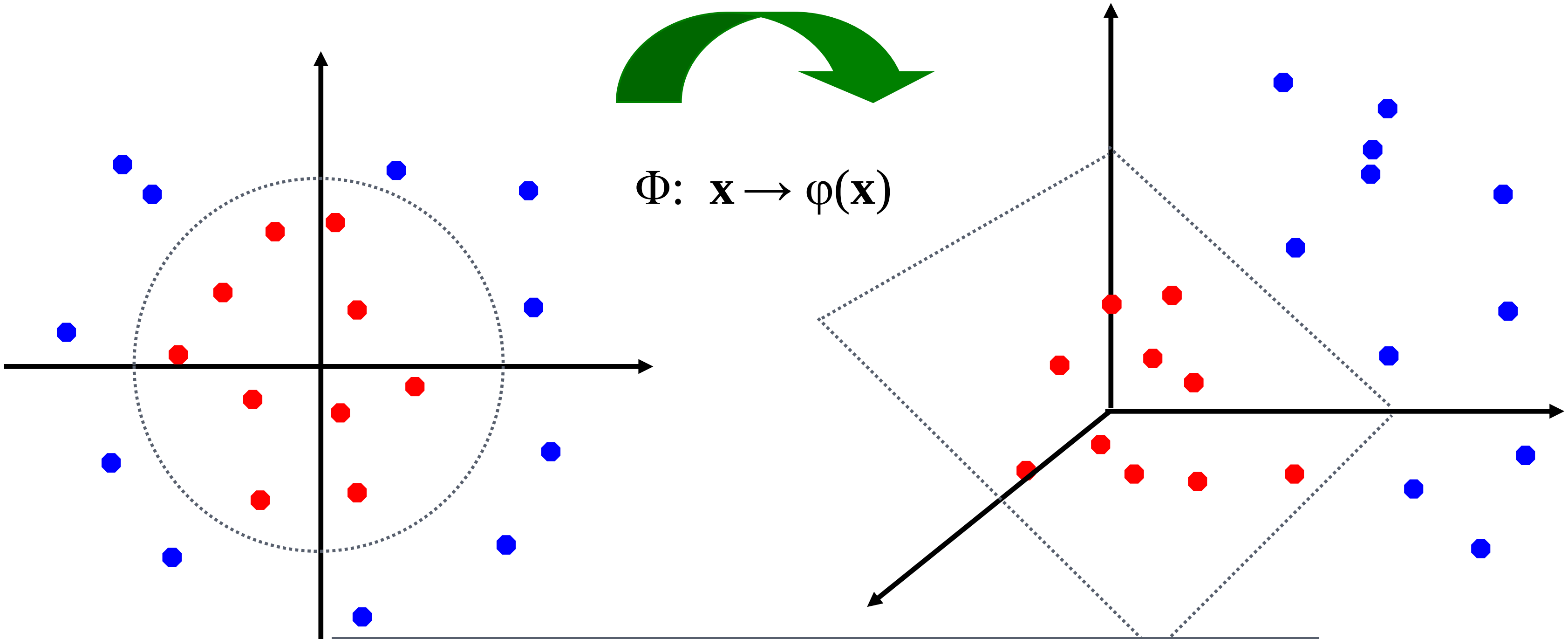
- **Sigmoid** with parameter κ and θ

$$K(\mathbf{x},\mathbf{y}) = \tanh(\kappa\mathbf{x}^T\mathbf{y} + \theta)$$

  - It does not satisfy the Mercer condition on all κ and θ

Non-linear SVMs: Feature spaces

- General idea: the original input space can always be mapped to some higher-dimensional feature space where the training set is separable:



Φ: $\mathbf{x} \rightarrow \varphi(\mathbf{x})$

$$\Phi : \mathbb{R}^2 \rightarrow \mathbb{R}^3$$
$$(x_1, x_2) \mapsto (z_1, z_2, z_3) := (x_1^2, \sqrt{2}\, x_1 x_2, x_2^2)$$

EXAMPLE

- Suppose we have 5 one-dimensional data points
  - $x_1=1$, $x_2=2$, $x_3=4$, $x_4=5$, $x_5=6$, with 1, 2, 6 as class 1 and 4, 5 as class 2 $\Rightarrow$ $y_1=1$, $y_2=1$, $y_3=-1$, $y_4=-1$, $y_5=1$
- We use the polynomial kernel of degree 2
  - $K(x,y) = (xy+1)^2$
  - C is set to 100
- We first find α_i (i=1, …, 5) by

$$\text{max.} \quad \sum_{i=1}^{5} \alpha_i - \frac{1}{2} \sum_{i=1}^{5} \sum_{j=1}^{5} \alpha_i \alpha_j y_i y_j (x_i x_j + 1)^2$$

$$\text{subject to } 100 \geq \alpha_i \geq 0, \sum_{i=1}^{5} \alpha_i y_i = 0$$

Example

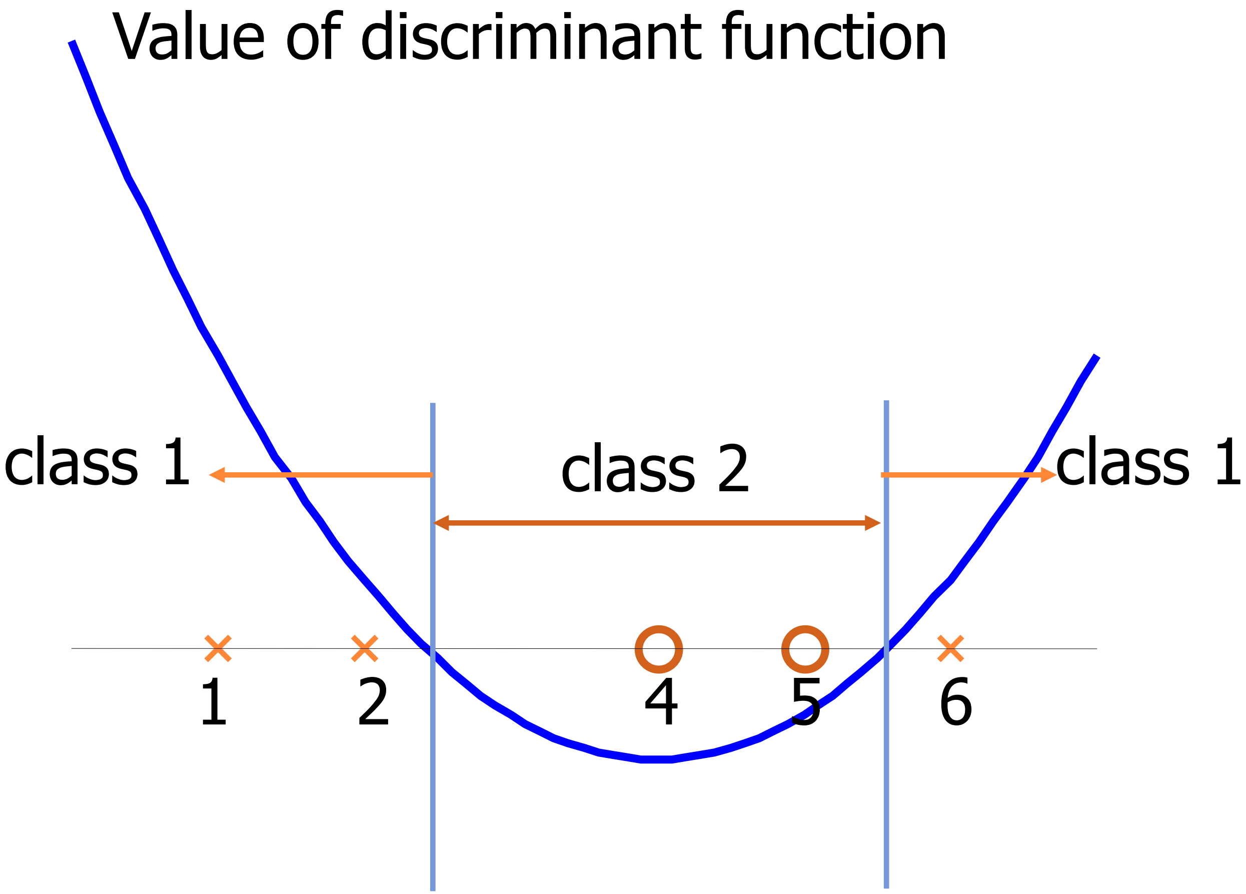
- By using a Quadratic (QP) solver, we get
  - α_1=0, α_2=2.5, α_3=0, α_4=7.333, α_5=4.833
  - Note that the constraints are indeed satisfied
  - The support vectors are {x_2=2, x_4=5, x_5=6}
- The discriminant function is

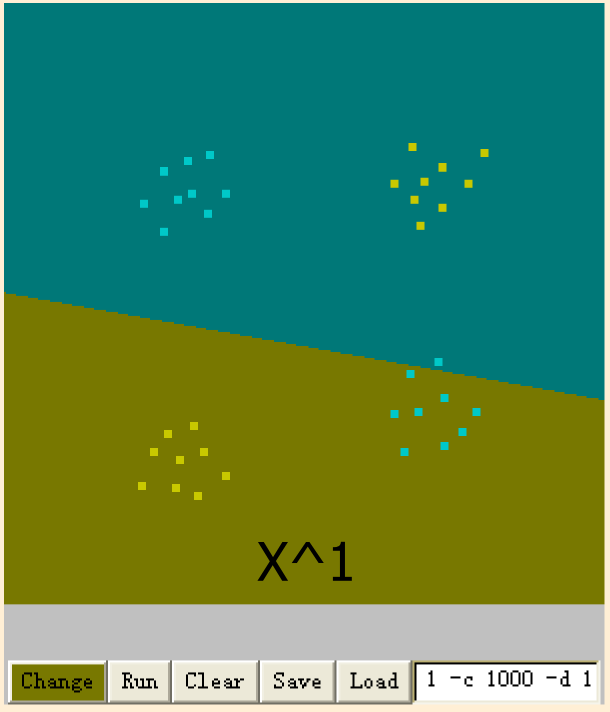
$$\begin{aligned} f(z) \\ &= 2.5(1)(2z+1)^2 + 7.333(-1)(5z+1)^2 + \overset{\alpha_5}{4.833}\overset{y_5}{(1)}\overset{K(z,x_5)}{(6z+1)^2} + b \\ &= 0.6667z^2 - 5.333z + b \end{aligned}$$

- *b* is recovered by solving f(2)=1 or by f(5)=-1 or by f(6)=1, as x_2 and x_5 lie on the line $\phi(\mathbf{w})^T\phi(\mathbf{x}) + b = 1$nd x_4 lies on the line $\phi(\mathbf{w})^T\phi(\mathbf{x}) + b = -1$
- All three give b=9

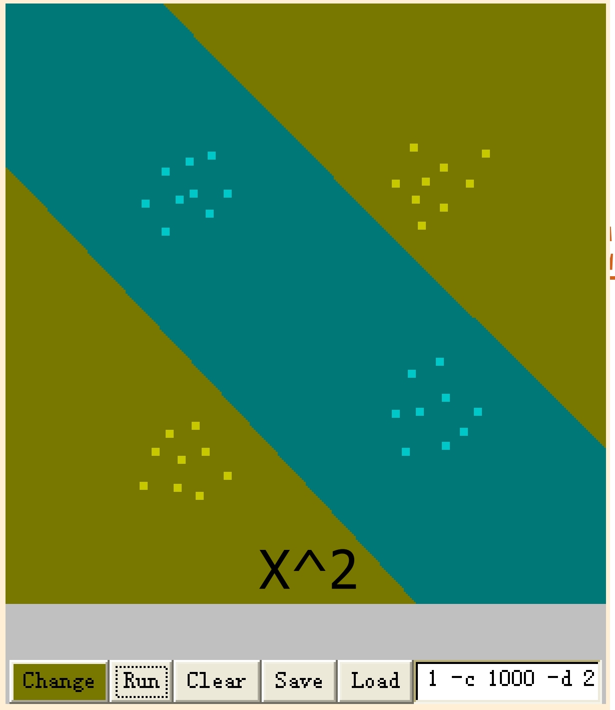
⟹ $f(z) = 0.6667z^2 - 5.333z + 9$

EXAMPLE

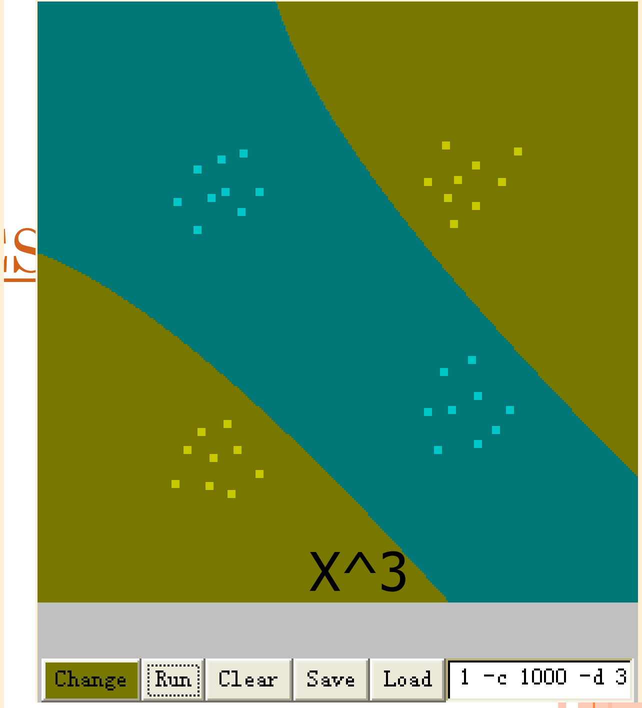

Value of discriminant function
class 1
class 2
class 1
1
2
4
5
6



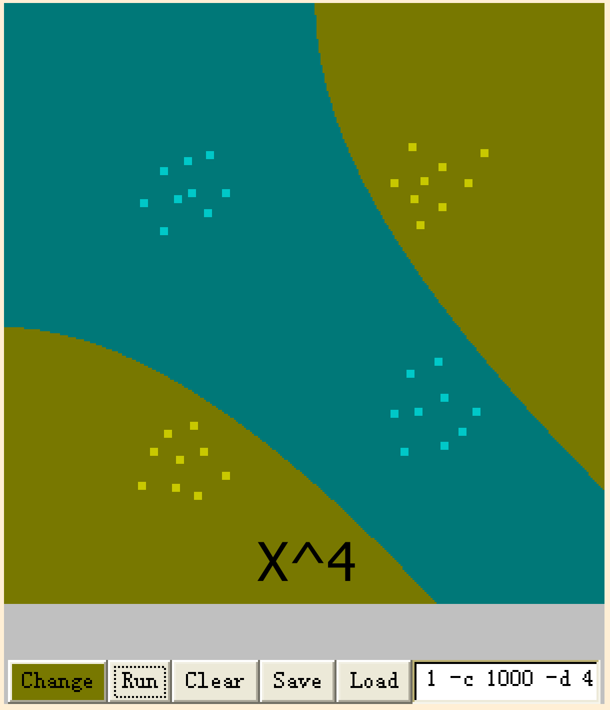
1 -c 1000 -d 1



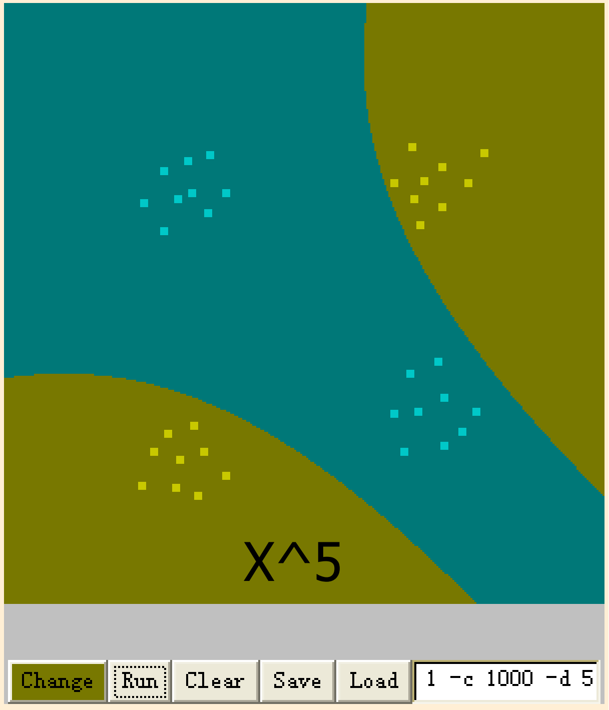
1 -c 1000 -d 2



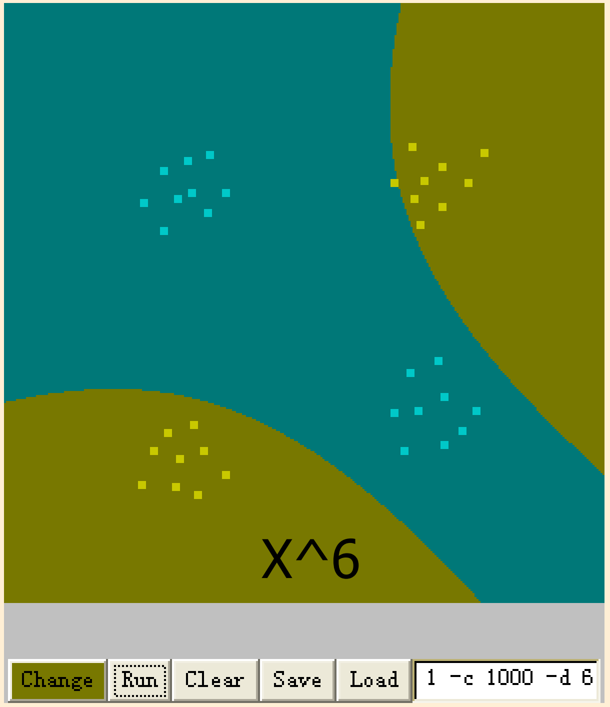
1 -c 1000 -d 3



1 -c 1000 -d 4



1 -c 1000 -d 5



1 -c 1000 -d 6

Choosing the Kernel Function

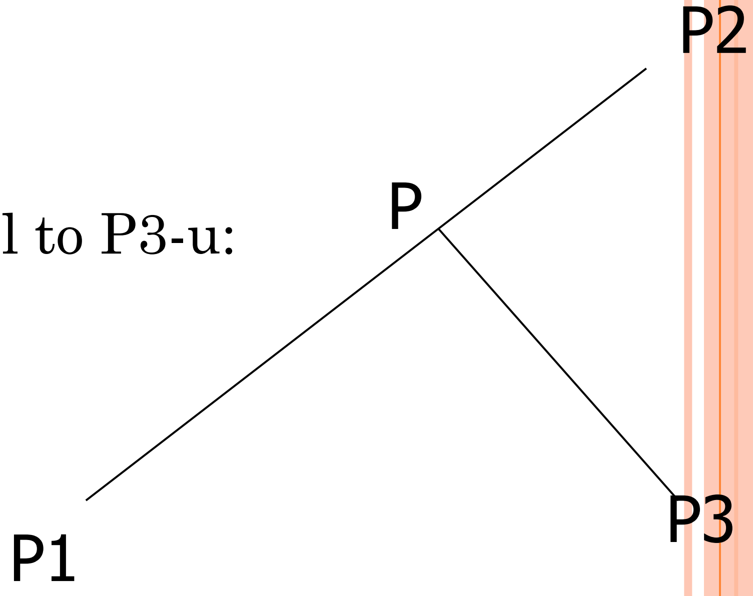
- Probably the most tricky part of using SVM.

SUMMARY: STEPS FOR CLASSIFICATION

- Prepare the pattern matrix
- Select the kernel function to use
- Select the parameter of the kernel function and the value of *C*
  - You can use the values suggested by the SVM software, or you can set apart a validation set to determine the values of the parameter
- Execute the training algorithm and obtain the α_i
- Unseen data can be classified using the α_i and the support vectors

APPENDIX: DISTANCE FROM A POINT TO A LINE

- Equation for the line: let u be a variable, then any point on the line can be described as:
  - $\mathbf{P} = \mathbf{P1} + u\,(\mathbf{P2} - \mathbf{P1})$
- Let the intersect point be u,
- Then, u can be determined by:
  - The two vectors (P2-P1) is orthogonal to P3-u:
  - That is,
    - (P3-P) dot (P2-P1) =0
    - P=P1+u(P2-P1)
  - P1=(x1,y1),P2=(x2,y2),P3=(x3,y3)



$$u = \frac{(x3 - x1)(x2 - x1) + (y3 - y1)(y2 - y1)}{\|p2 - p1\|^2}$$

Distance and margin

$$u = \frac{(x3 - x1)(x2 - x1) + (y3 - y1)(y2 - y1)}{\|p2 - p1\|^2}$$

- $x = x1 + u\,(x2 - x1)$
  $y = y1 + u\,(y2 - y1)$

- The distance therefore between the point **P3** and the line is the distance between P=(x,y) above and **P3**
- Thus,
  - d= |(P3-P)|=