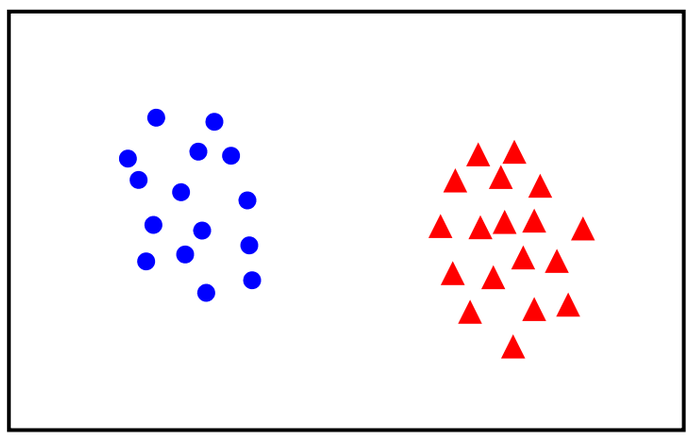
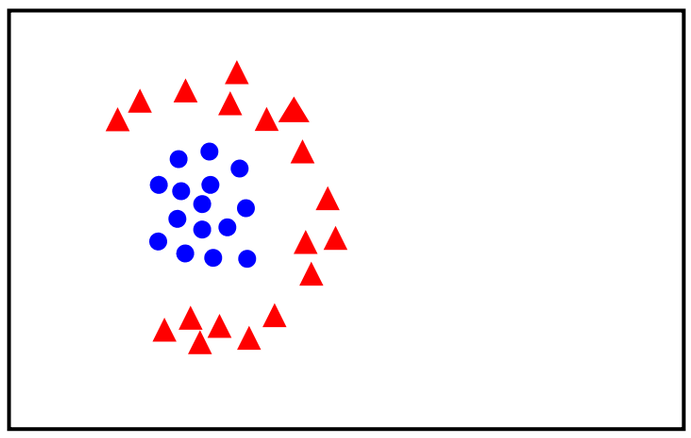
Binary Classification

Given training data $(\mathbf{x}_i, y_i)$ for $i = 1 \ldots N$, with $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \{-1, 1\}$, learn a classifier $f(\mathbf{x})$ such that

$$f(\mathbf{x}_i) \begin{cases} \geq 0 & y_i = +1 \\ < 0 & y_i = -1 \end{cases}$$

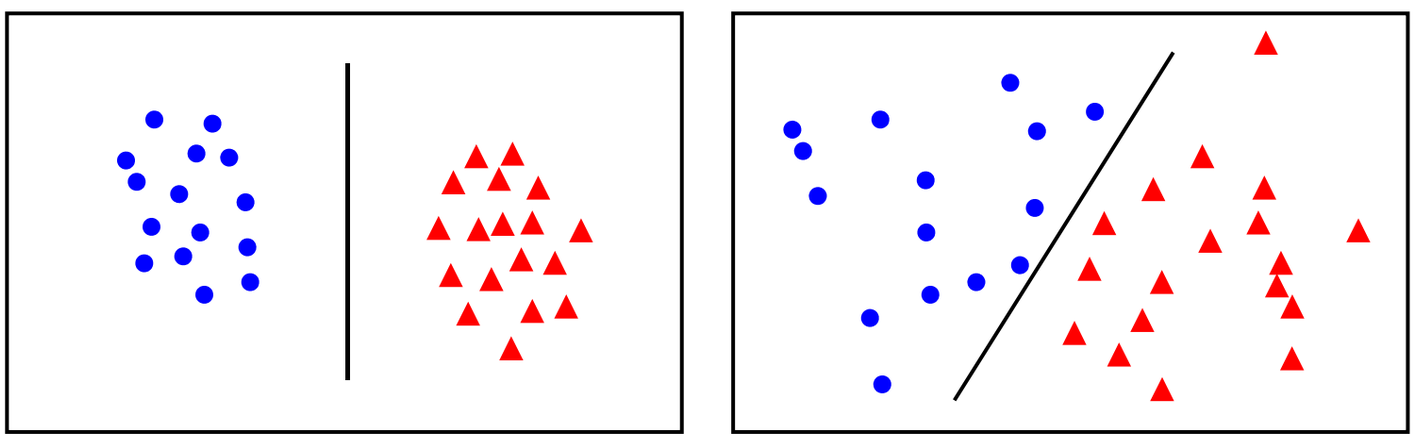
i.e. $y_i f(\mathbf{x}_i) > 0$ for a correct classification.



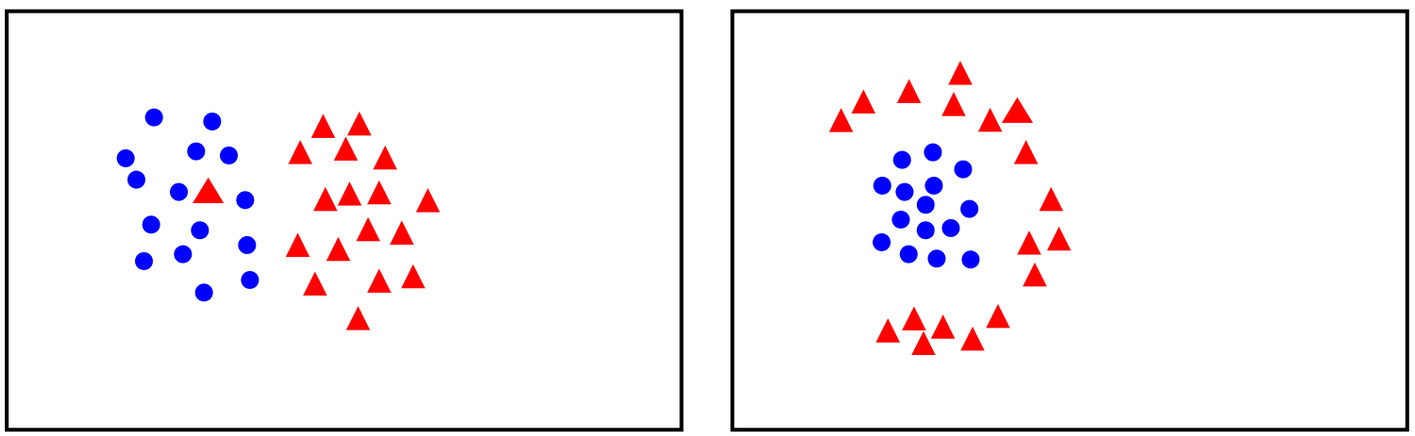


Linear separability

linearly separable



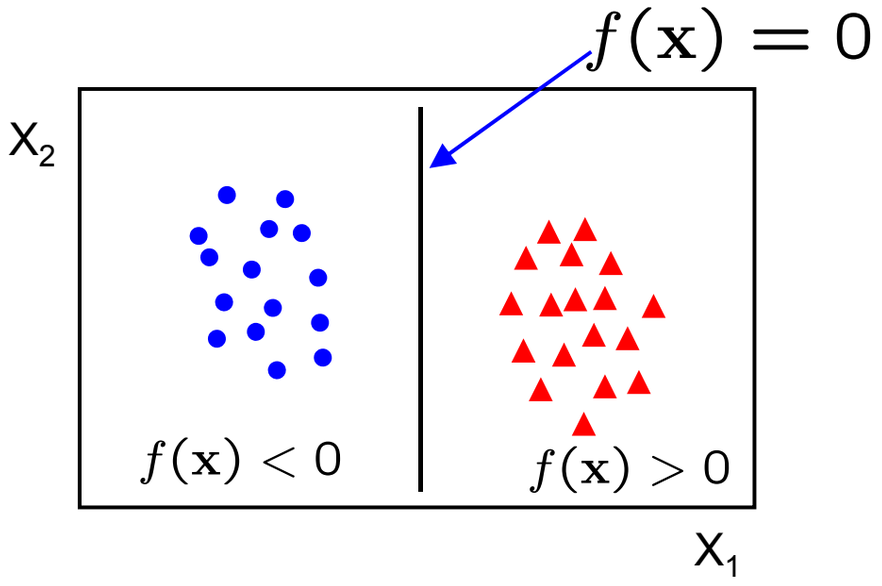
not linearly separable



Linear classifiers

A linear classifier has the form

$$f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + b$$

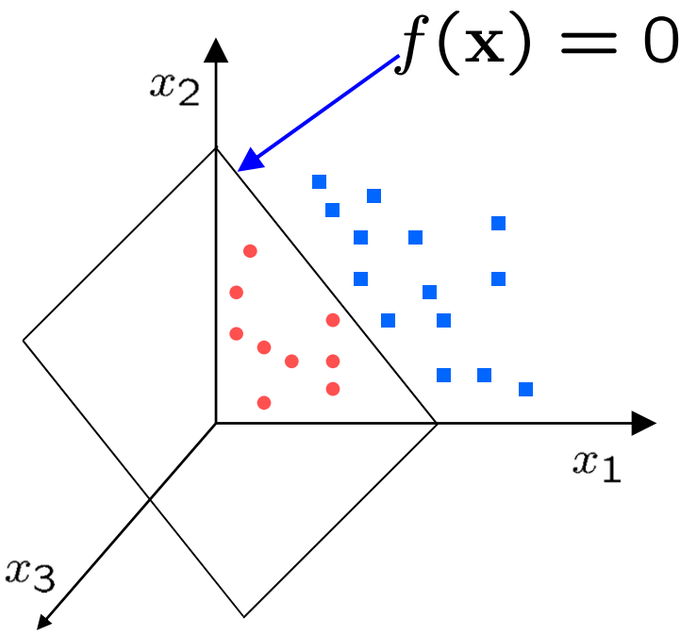


- in 2D the discriminant is a line
- $\mathbf{w}$ is the normal to the line, and b the bias
- $\mathbf{w}$ is known as the weight vector

Linear classifiers

A linear classifier has the form

$$f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + b$$



- in 3D the discriminant is a plane, and in nD it is a hyperplane

For a K-NN classifier it was necessary to `carry' the training data

For a linear classifier, the training data is used to learn **w** and then discarded

Only **w** is needed for classifying new data

The Perceptron Classifier

Given linearly separable data $\mathbf{x}_i$ labelled into two categories $y_i = \{-1,1\}$, find a weight vector **w** such that the discriminant function

$$f(\mathbf{x}_i) = \mathbf{w}^\top \mathbf{x}_i + b$$

separates the categories for i = 1, .., N

• how can we find this separating hyperplane ?

The Perceptron Algorithm

Write classifier as $f(\mathbf{x}_i) = \tilde{\mathbf{w}}^\top \tilde{\mathbf{x}}_i + w_0 = \mathbf{w}^\top \mathbf{x}_i$

$$\text{where } \mathbf{w} = (\tilde{\mathbf{w}}, w_0), \mathbf{x}_i = (\tilde{\mathbf{x}}_i, 1)$$

• Initialize **w** = 0

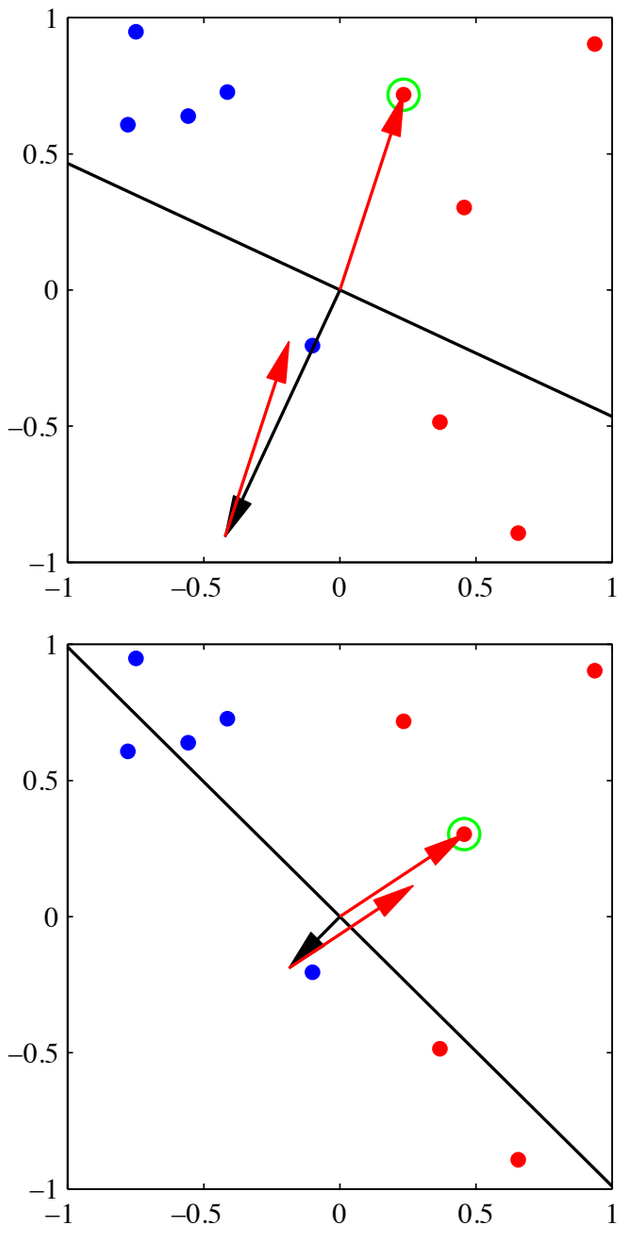
• Cycle though the data points { $\mathbf{x}_i$, y_i }

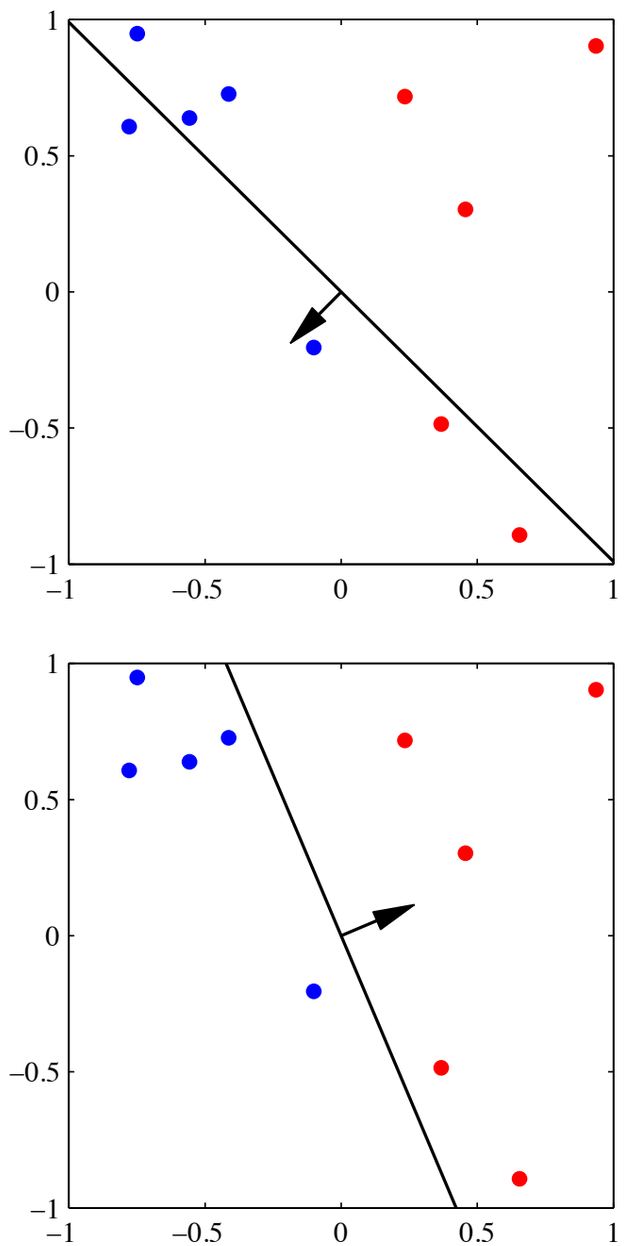
  • if $\mathbf{x}_i$ is misclassified then $\mathbf{w} \leftarrow \mathbf{w} + \alpha \, \text{sign}(f(\mathbf{x}_i)) \, \mathbf{x}_i$

• Until all the data is correctly classified

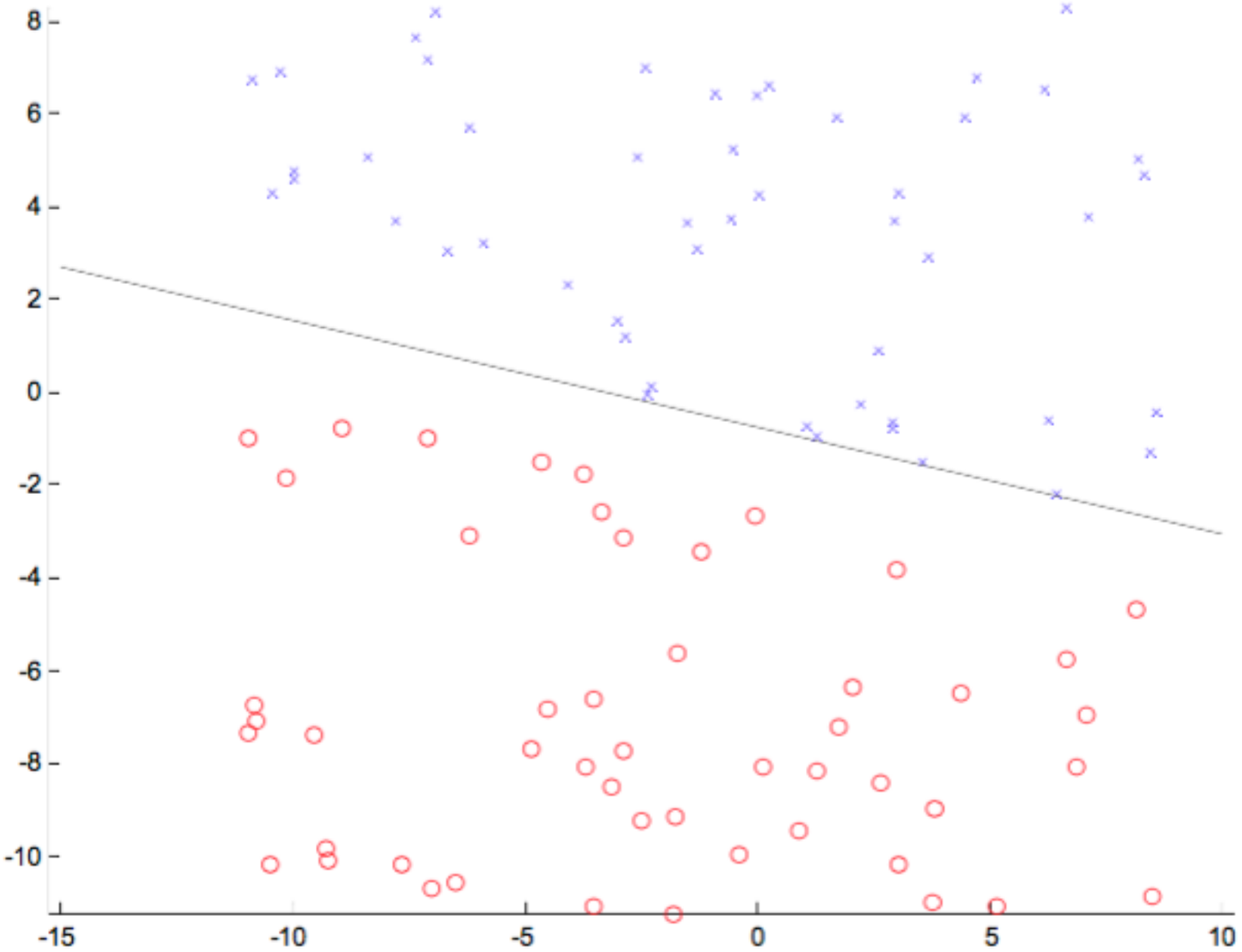
Illustration of two steps in the Perception learning algorithm (Bishop 4.7)

(Green points are misclassified)



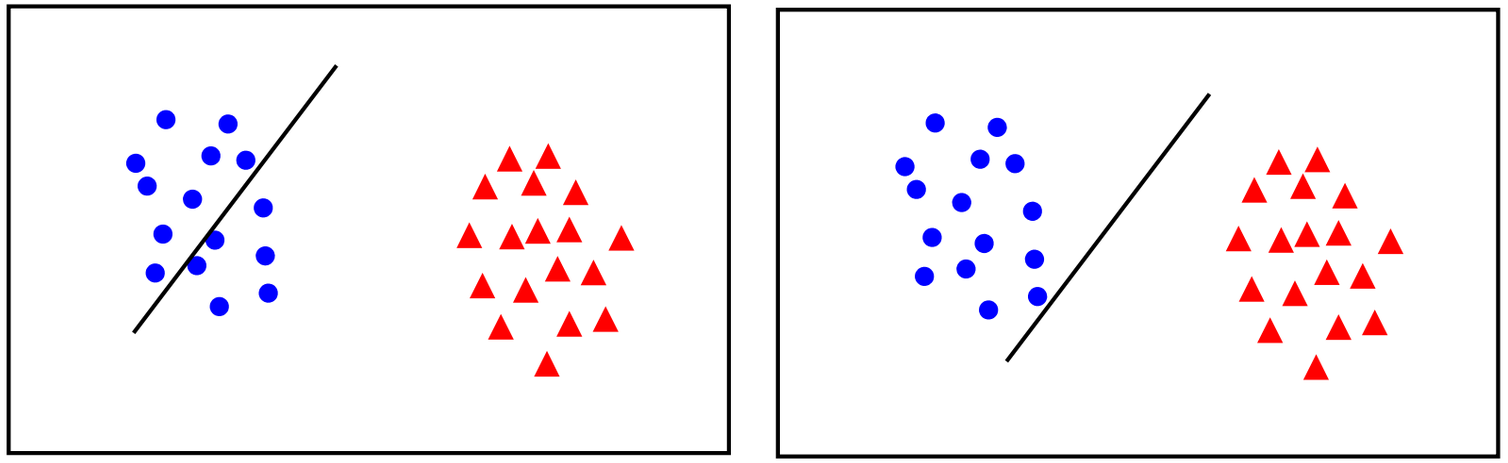


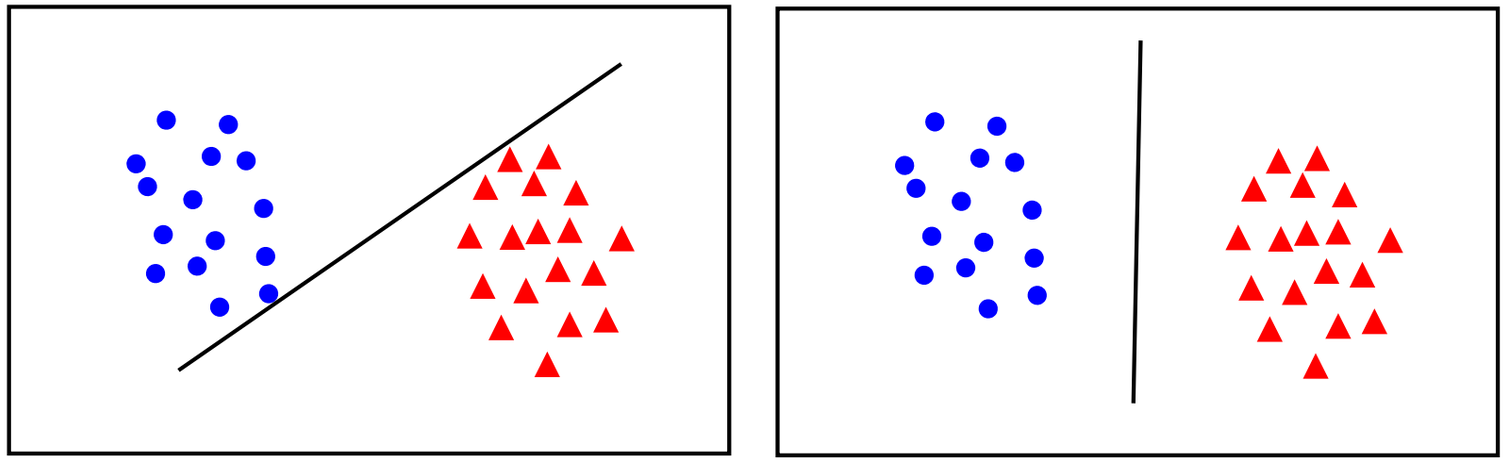
Perceptron example



- if the data is linearly separable, then the algorithm will converge
- convergence can be slow
- separating line close to training data
- we would prefer a larger margin for generalization

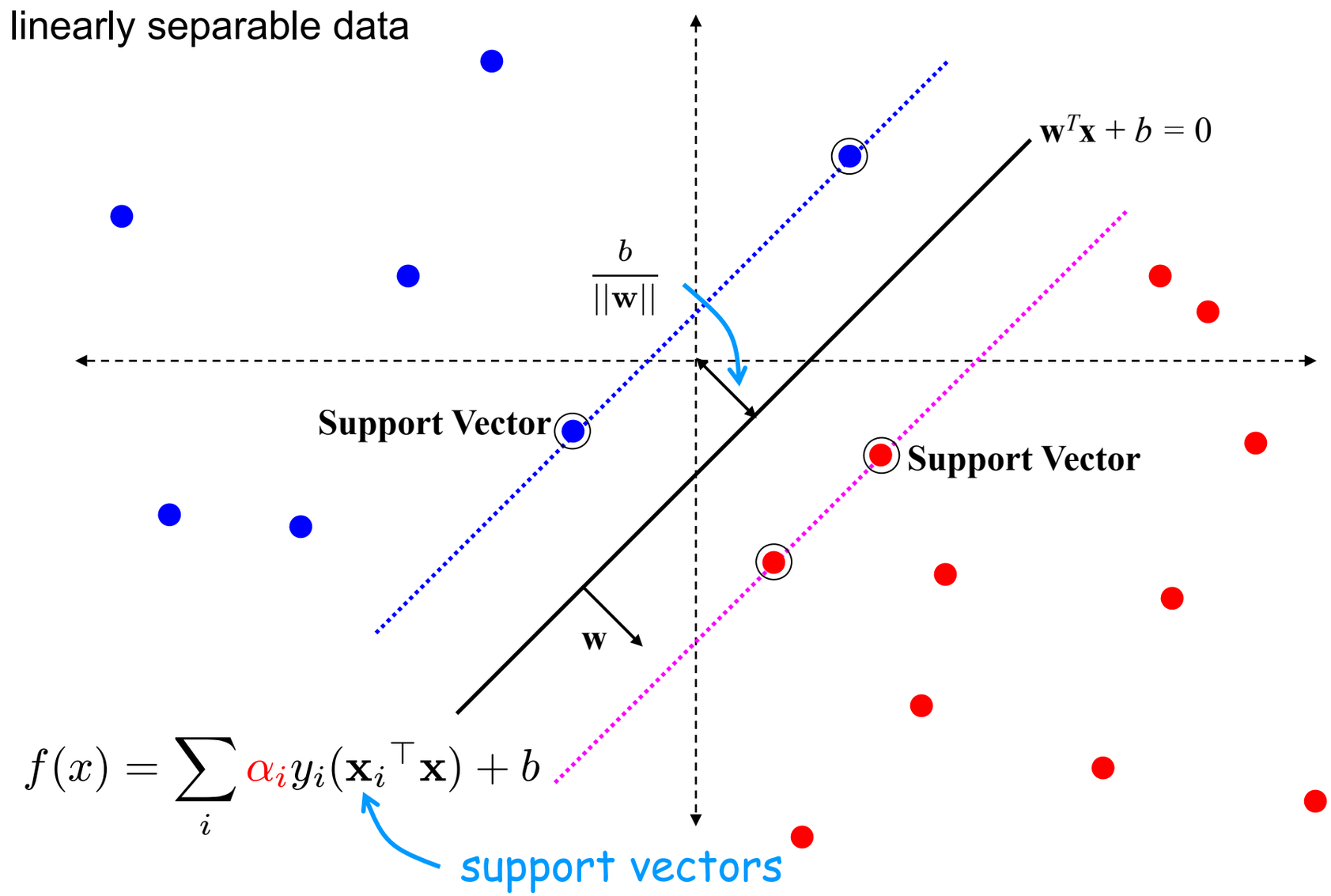
What is the best w?





- maximum margin solution: most stable under perturbations of the inputs

Support Vector Machine



SVM – sketch derivation

- Since $\mathbf{w}^\top \mathbf{x} + b = 0$ and $c(\mathbf{w}^\top \mathbf{x} + b) = 0$ define the same plane, we have the freedom to choose the normalization of $\mathbf{w}$

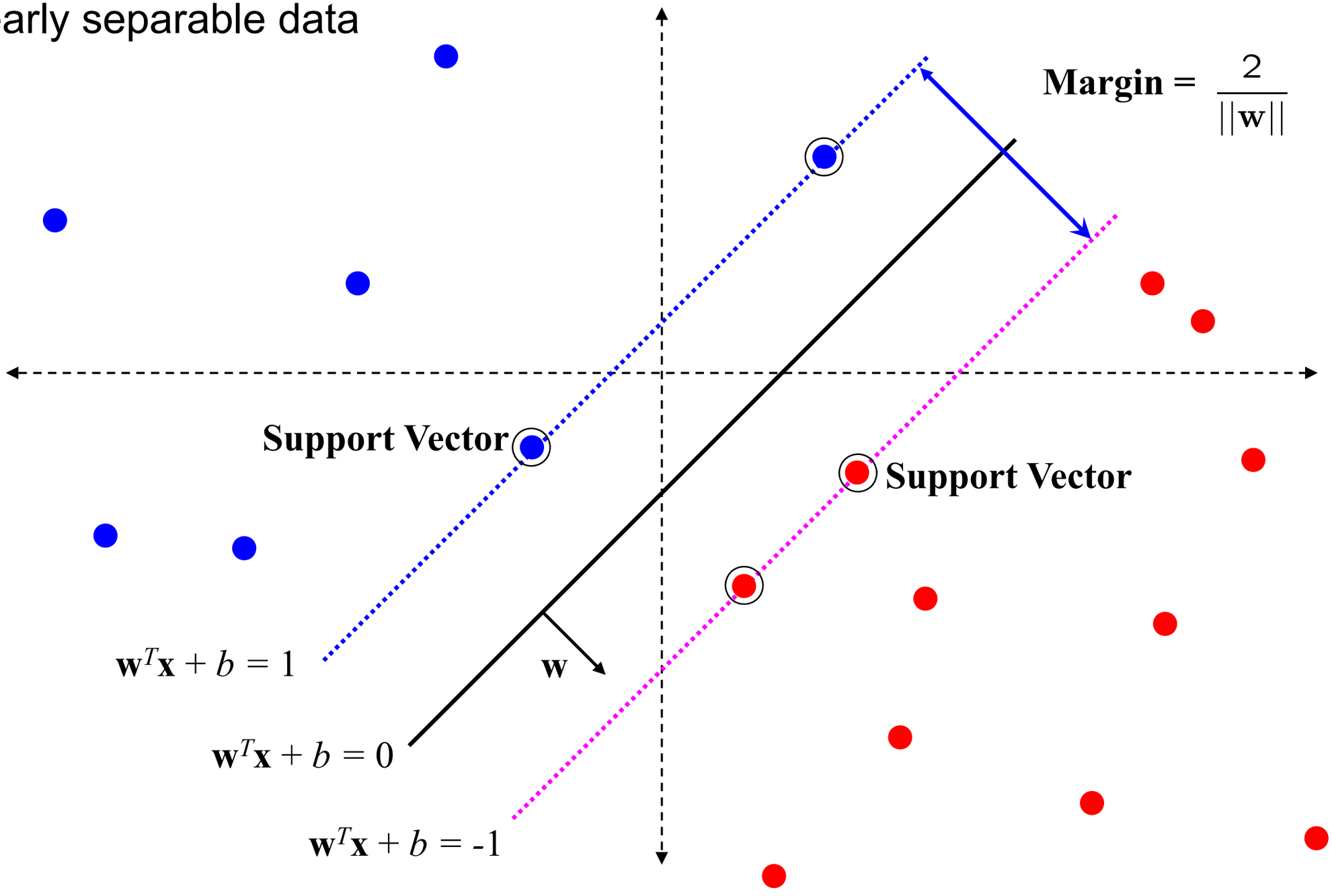
- Choose normalization such that $\mathbf{w}^\top \mathbf{x}_+ + b = +1$ and $\mathbf{w}^\top \mathbf{x}_- + b = -1$ for the positive and negative support vectors respectively

- Then the margin is given by

$$\frac{\mathbf{w}}{||\mathbf{w}||} \cdot \left(\mathbf{x}_+ - \mathbf{x}_-\right) = \frac{\mathbf{w}^\top \left(\mathbf{x}_+ - \mathbf{x}_-\right)}{||\mathbf{w}||} = \frac{2}{||\mathbf{w}||}$$

Support Vector Machine

linearly separable data



SVM – Optimization

- Learning the SVM can be formulated as an optimization:

$$\max_{\mathbf{w}} \frac{2}{||\mathbf{w}||} \text{ subject to } \mathbf{w}^\top \mathbf{x}_i + b \begin{array}{ll} \geq 1 & \text{if } y_i = +1 \\ \leq -1 & \text{if } y_i = -1 \end{array} \text{ for } i = 1 \ldots N$$

- Or equivalently

$$\min_{\mathbf{w}} ||\mathbf{w}||^2 \text{ subject to } y_i \left(\mathbf{w}^\top \mathbf{x}_i + b \right) \geq 1 \text{ for } i = 1 \ldots N$$

- This is a quadratic optimization problem subject to linear constraints and there is a unique minimum

The problem statement

Given a set of training data $\{(\mathbf{x}_i, d_i), i = 1, \ldots, N\}$, minimize

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

subject to the constraint that

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

The problem statement

Given a set of training data $\{(\mathbf{x}_i, d_i), i = 1, \ldots, N\}$, minimize

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

subject to the constraint that

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

Looks like a job for LAGRANGE MULTIPLIERS!

$$J(\mathbf{w}, b, \lambda) = \frac{1}{2}\mathbf{w}^T\mathbf{w} - \sum_{i=1}^{N} \lambda_i(d_i(\mathbf{w}^T\mathbf{x_i} + b) - 1)$$

So that

$$J_{\mathbf{w}} = \mathbf{0} = \mathbf{w} - \sum_{i=1}^{N} \lambda_i d_i \mathbf{x_i} \tag{3}$$

and

$$J_b = 0 = \sum_{i=1}^{N} \lambda_i d_i \tag{4}$$

Now for the DUAL PROBLEM

$$J(\mathbf{w}, b, \lambda) = \frac{1}{2}\mathbf{w}^T\mathbf{w} - \sum_{i=1}^{N} \lambda_i d_i \mathbf{w}^T\mathbf{x_i} + b\sum_{i=1}^{N} \lambda_i d_i + \sum_{i=1}^{N} \lambda_i$$

Note that from (4) third term is zero. Using Eq. (3):

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

Lets enjoy the moment!

DUAL PROBLEM

$$\max Q(\lambda) = \sum_{i=1}^{N} \lambda_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \lambda_i \lambda_j d_i d_j \mathbf{x}_i^T \mathbf{x}_j$$

Subject to constraints

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

$$\lambda_i \geq 0, \quad i = 1, \ldots, N$$

This is *easier to solve* than the original. Furthermore it only depends on the training samples $\{(\mathbf{x}_i, d_i), i = 1, \ldots, N\}$.

Once you have the λ_is, get the $\mathbf{w}$ from

$$\mathbf{w} = \sum_{i=1}^{N} \lambda_i d_i \mathbf{x_i}$$

and the b from a support vector that has $d_i = 1$,

$$b = 1 - \mathbf{w}^T \mathbf{x}_s$$

Almost done ...

KERNEL FUNCTIONS

Now the big bonus occurs because all the machinery we have developed will work if we map the points $\mathbf{x}_i$ to a higher dimensional space, provided we observe certain conventions.

Let $\phi(\mathbf{x}_i)$ be a function that does the mapping. So the new hyperplane is

$$\sum_{i=1}^{N} w_i \phi_i(\mathbf{x}) + b = 0$$

For simplicity in notation define

$$\phi(\mathbf{x}) = (\phi_0(\mathbf{x}), \phi_1(\mathbf{x}), \phi_2(\mathbf{x}), \ldots, \phi_{m_1}(\mathbf{x}))$$

where m_1 is the new dimension size and by convention $\phi_0(\mathbf{x}) = 1$.

Then all the work we did with $\mathbf{x}$ works with $\phi(\mathbf{x})$. The only issue is that instead of $\mathbf{x}_i^T \mathbf{x}_j$ we have a *Kernel function*, $K(\mathbf{x}_i, \mathbf{x}_j)$ where

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

and Kernel functions need to have certain nice properties. :)

Examples

Polynomials

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

Radial Basis Functions

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

TABLE 6.2 XOR Problem

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

$$k(\mathbf{x}, \mathbf{x}_i) = (1 + \mathbf{x}^T\mathbf{x}_i)^2 \tag{6.40}$$

With $\mathbf{x} = [x_1, x_2]^T$ and $\mathbf{x}_i = [x_{i1}, x_{i2}]^T$, we may thus express the kernel $k(\mathbf{x}, \mathbf{x}_i)$ in terms of *monomials* of various orders as follows:

$$k(\mathbf{x}, \mathbf{x}_i) = 1 + x_1^2 x_{i1}^2 + 2x_1 x_2 x_{i1} x_{i2} + x_2^2 x_{i2}^2 + 2x_1 x_{i1} + 2x_2 x_{i2} \tag{6.41}$$

The image of the input vector $\mathbf{x}$ induced in the feature space is therefore deduced to be

$$\boldsymbol{\phi}(\mathbf{x}) = [1, x_1^2, \sqrt{2}x_1 x_2, x_2^2, \sqrt{2}x_1, \sqrt{2}x_2]^T$$

Similarly,

$$\boldsymbol{\phi}(\mathbf{x}_i) = [1, x_{i1}^2, \sqrt{2}x_{i1} x_{i2}, x_{i2}^2, \sqrt{2}x_{i1}, \sqrt{2}x_{i2}]^T, \qquad i = 1, 2, 3, 4 \tag{6.42}$$

Using the definition of Eq. (6.35), we obtain the Gram

$$\mathbf{K} = \begin{bmatrix} 9 & 1 & 1 & 1 \\ 1 & 9 & 1 & 1 \\ 1 & 1 & 9 & 1 \\ 1 & 1 & 1 & 9 \end{bmatrix}$$

$$Q(\alpha) = \alpha_1 + \alpha_2 + \alpha_3 + \alpha_4 - \frac{1}{2}(9\alpha_1^2 - 2\alpha_1\alpha_2 - 2\alpha_1\alpha_3 + 2\alpha_1\alpha_4 + 9\alpha_2^2 + 2\alpha_2\alpha_3 - 2\alpha_2\alpha_4 + 9\alpha_3^2 - 2\alpha_3\alpha_4 + 9\alpha_4^2)$$

Optimizing $Q(\alpha)$ with respect to the four Lagrange multipliers yields the following set of simultaneous equations:

$$\begin{aligned} 9\alpha_1 - \alpha_2 - \alpha_3 + \alpha_4 &= 1 \\ -\alpha_1 + 9\alpha_2 + \alpha_3 - \alpha_4 &= 1 \\ -\alpha_1 + \alpha_2 + 9\alpha_3 - \alpha_4 &= 1 \\ \alpha_1 - \alpha_2 - \alpha_3 + 9\alpha_4 &= 1 \end{aligned}$$

Hence, the optimum values of the Lagrange multipliers are

$$\alpha_{o,1} = \alpha_{o,2} = \alpha_{o,3} = \alpha_{o,4} = \frac{1}{8}$$

This result indicates that in this example, all four input vectors $\{\mathbf{x}_i\}_{i=1}^{4}$ are support vectors. The optimum value of $Q(\alpha)$ is

$$Q_o(\alpha) = \frac{1}{4}$$

Correspondingly, we may write

$$\frac{1}{2}\|\mathbf{w}_o\|^2 = \frac{1}{4}$$

or

$$\|\mathbf{w}_o\| = \frac{1}{\sqrt{2}}$$

$$\mathbf{w}_o = \frac{1}{8}[-\varphi(\mathbf{x}_1) + \varphi(\mathbf{x}_2) + \varphi(\mathbf{x}_3) - \varphi(\mathbf{x}_4)]$$

$$= \frac{1}{8}\left[-\begin{bmatrix}1\\1\\\sqrt{2}\\1\\-\sqrt{2}\\-\sqrt{2}\end{bmatrix} + \begin{bmatrix}1\\1\\-\sqrt{2}\\1\\-\sqrt{2}\\\sqrt{2}\end{bmatrix} + \begin{bmatrix}1\\1\\-\sqrt{2}\\1\\\sqrt{2}\\-\sqrt{2}\end{bmatrix} - \begin{bmatrix}1\\1\\\sqrt{2}\\1\\\sqrt{2}\\\sqrt{2}\end{bmatrix}\right]$$

$$= \begin{bmatrix}0\\0\\-1/\sqrt{2}\\0\\0\\0\end{bmatrix}$$

The first element of $\mathbf{w}_o$ indicates that the bias b is zero.

The optimal hyperplane is defined by

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

Expanding the inner product $\mathbf{w}_o^T \boldsymbol{\phi}(\mathbf{x})$ yields:

$$\left[0, 0, \frac{-1}{\sqrt{2}}, 0, 0, 0\right] \begin{bmatrix} 1 \\ x_1^2 \\ \sqrt{2}x_1x_2 \\ x_2^2 \\ \sqrt{2}x_1 \\ \sqrt{2}x_2 \end{bmatrix} = 0$$

which reduces to

$$-x_1x_2 = 0$$