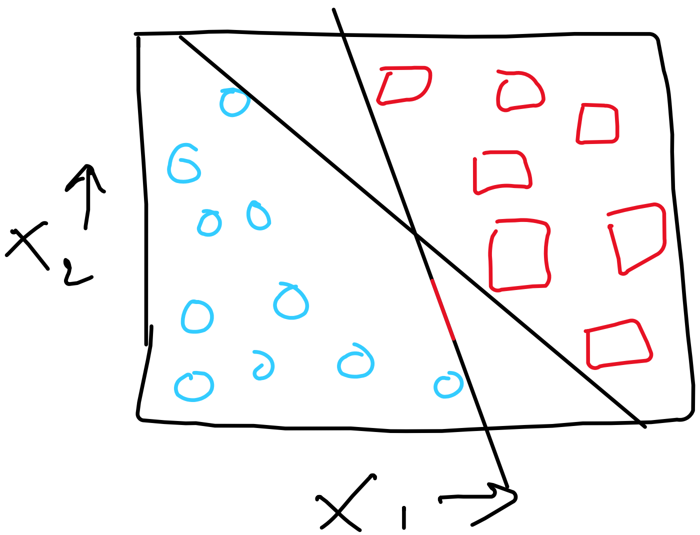
Support vector machine

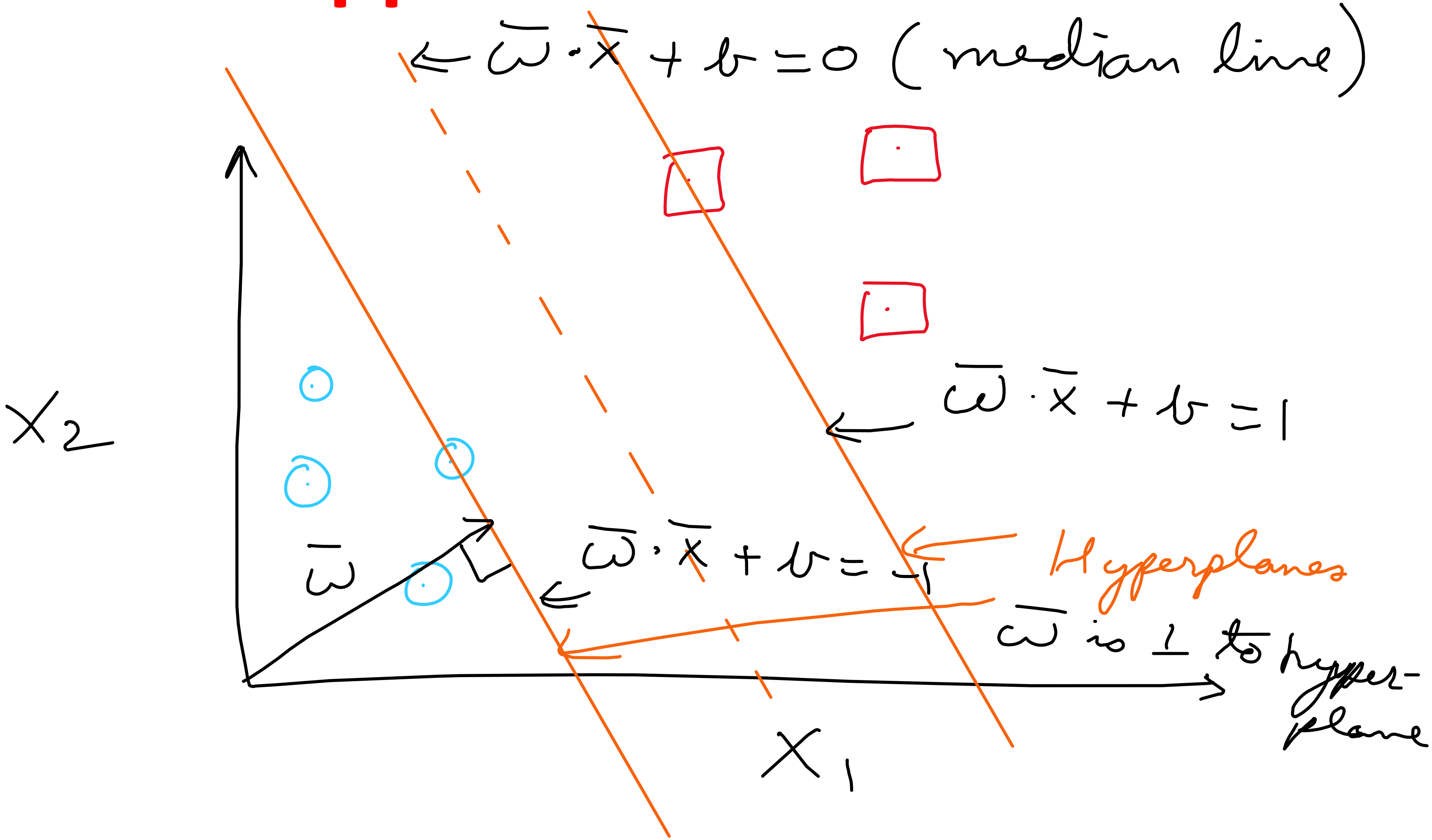
Prof. Asim Tewari
IIT Bombay

Support Vector Classifier

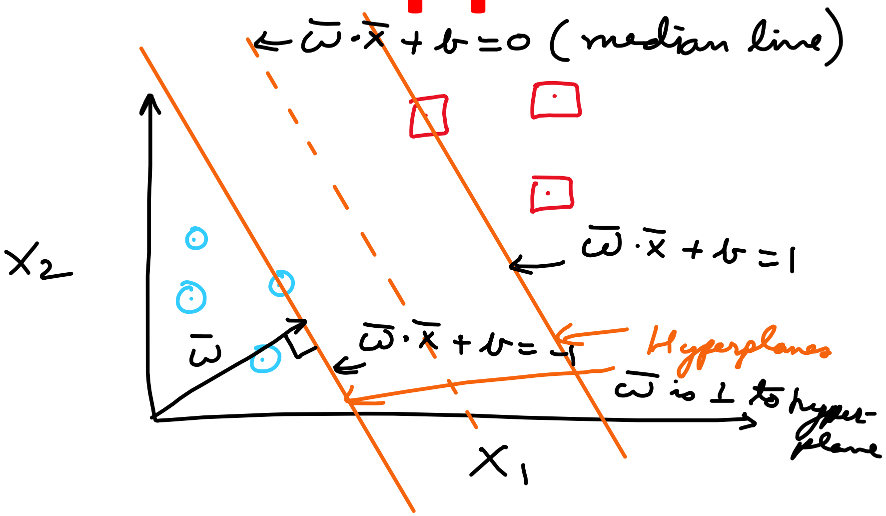


SVC : Maximum - margin hyperplane

Support Vector Classifier



Support Vector Classifier



① Any point above or on $\bar{w}\cdot\bar{x}+b=1$ belongs to □

② Any point on or below $\bar{w}\cdot\bar{x}+b=-1$ belongs to ○

For "□": $\bar{w}\cdot\bar{x}+b \geq 1$

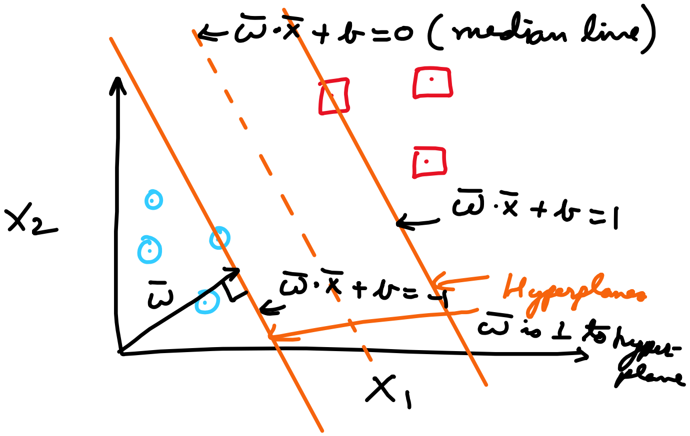
For "○" $\bar{w}\cdot\bar{x}+b \leq -1$

For an unknown $\bar{u}$ the decision rule is based on maximum-margin hyperplane.

$\bar{w}\cdot\bar{u}+b \geq 0 \Rightarrow$ □

$\bar{w}\cdot\bar{u}+b < 0 \Rightarrow$ ○

Support Vector Classifier



Standardize data set

For point i

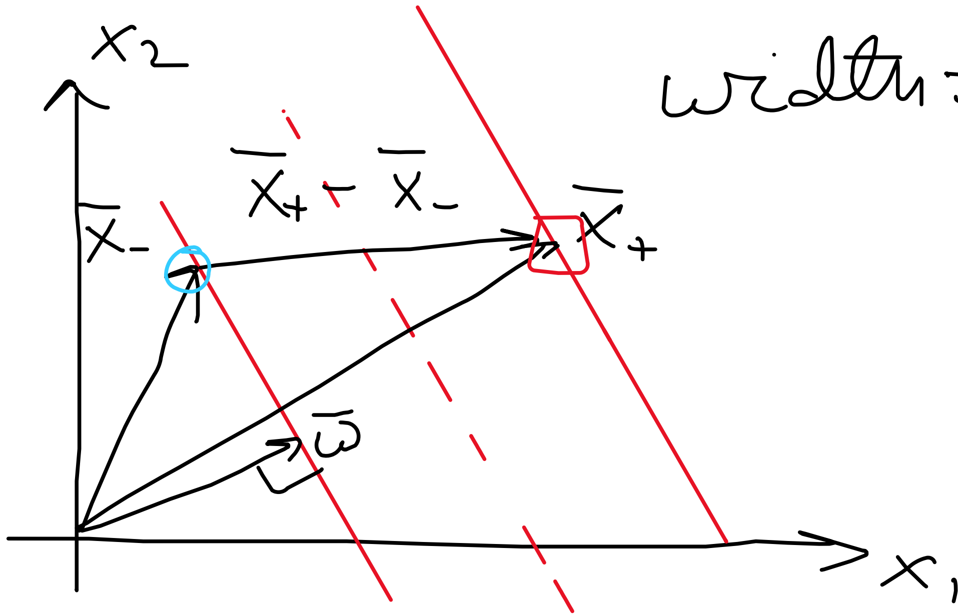
$$y_i = \begin{cases} 1 & \text{if it belongs to } \square \\ -1 & \text{if it belongs to } \circ \end{cases}$$

$$\underbrace{y_i(\bar{\omega}\cdot\bar{x}_i + b)}\geq 1 \qquad \left\{\begin{array}{l} \square \\ \circ \end{array}\right. \text{True for both}$$

$$\Rightarrow \; y_i(\bar{\omega}\cdot\bar{x}_i + b) - 1 \geq 0$$

$$y_i(\bar{\omega}_i\cdot\bar{x}_i + b) - 1 = 0$$

Support Vector Classifier



$$\text{width} = (\bar{x}_+ - \bar{x}_-) \cdot \frac{\bar{\omega}}{\|\bar{\omega}\|}$$

↑

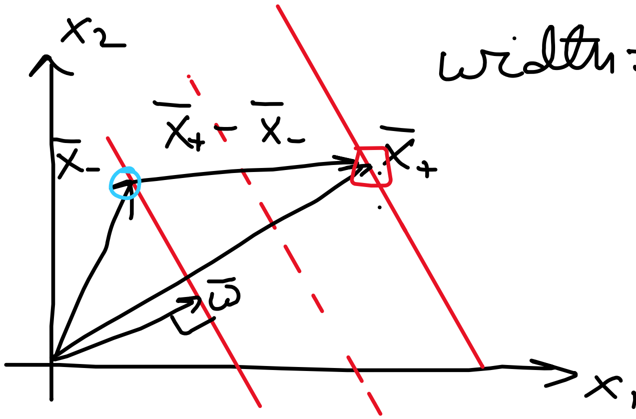
unit vector

$$y_i(\bar{\omega} \cdot \bar{x}_i + b) - 1 = 0$$

we get $\bar{\omega} \cdot \bar{x}_+ = 1 - b$ and $\bar{\omega} \cdot \bar{x}_- = -1 - b$

$$\text{width} = (\bar{x}_+ - \bar{x}_-) \cdot \frac{\bar{\omega}}{\|\bar{\omega}\|} = \frac{\bar{x}_+ \cdot \bar{\omega} - \bar{x}_- \cdot \bar{\omega}}{\|\bar{\omega}\|}$$

Support Vector Classifier



$$\text{width} = \frac{\bar{x}_+ \cdot \bar{\omega} - \bar{x}_- \cdot \bar{\omega}}{\|\bar{\omega}\|}$$

$$= \frac{(1 - b) - (-1 - b)}{\|\bar{\omega}\|}$$

$$\Rightarrow \text{width} = \frac{2}{\|\bar{\omega}\|}$$

So we want two hyperplanes $y_i(\bar{\omega} \cdot \bar{x}_i + b) - 1 = 0$ such that the width $\left(\frac{2}{\|\omega\|}\right)$ is maximum.

Maximize $\frac{2}{\|\omega\|}$

$\Rightarrow$ Minimize $\|\omega\|/2$

Support Vector Classifier

$\equiv$ minimize $||\omega||^2$

minimize $||\omega||/2$ with a constrain

$$y_i(\bar{\omega}\cdot\bar{x}_i + b) - 1 = 0$$

$$L = \frac{1}{2}||\bar{\omega}||^2 - \sum \alpha_i\left[y_i(\bar{\omega}\cdot\bar{x}_i + b) - 1\right]$$

$$\min_{\omega, b}(L) \Rightarrow \frac{\partial L}{\partial \bar{\omega}} = 0, \quad \frac{\partial L}{\partial b} = 0$$

$$\Rightarrow \frac{\partial L}{\partial \omega} = \frac{\partial}{\partial \bar{\omega}}\left[\frac{1}{2}||\omega||^2\right] - \frac{\partial}{\partial \bar{\omega}}\left[\sum \alpha_i y_i(\omega\cdot x_i + b) - 1\right]$$

$$\frac{\partial L}{\partial \omega} = 0 \quad \Rightarrow \bar{\omega} = \sum \alpha_i y_i \bar{x}_i$$

Support Vector Classifier

$$L = \frac{1}{2}\|\bar{\omega}\|^2 - \sum \alpha_i \left[y_i(\bar{\omega}\cdot x_i + b) - 1 \right]$$

$$\frac{\partial L}{\partial \bar{\omega}} = 0 \Rightarrow \bar{\omega} = \sum (\alpha_i y_i \bar{x}_i)$$

$$\frac{\partial L}{\partial b} = 0 \Rightarrow -\sum \alpha_i y_i = 0 \text{ or } \sum \alpha_i y_i = 0$$

$$\therefore L = \frac{1}{2} \underbrace{\left(\sum \alpha_i y_i \bar{x}_i\right) \cdot \left(\sum \alpha_j y_j \bar{x}_j\right)}_{\bar{\omega} \cdot \bar{\omega} = \|\bar{\omega}\|^2}$$

$$- \sum \left[\alpha_i \left(y_i \left(\underbrace{\sum \alpha_j y_j x_j}_{\bar{\omega}} \right) \cdot \bar{x}_i + b \right) - 1 \right]$$

Support Vector Classifier

$$L = \frac{1}{2}\left(\sum \alpha_i y_i \bar{x}_i\right)\cdot\left(\sum \alpha_j y_j \bar{x}_j\right)$$

$$- \left(\sum \alpha_i y_i \bar{x}_i\right)\cdot\left(\sum \alpha_j y_j \bar{x}_j\right)$$

$$- \sum \alpha_i y_i b + \sum \alpha_i$$

$$\Rightarrow L = \sum \alpha_i - \frac{1}{2} \sum_i \sum_j \left(\alpha_i \alpha_j y_i y_j \bar{x}_i \cdot \bar{x}_j\right)$$

dot product

$$L = f\left(\bar{x}_i \cdot \bar{x}_j\right)$$

Support Vector Classifier

$$\Rightarrow L = \sum \alpha_i - \frac{1}{2} \sum_i \sum_j \left(\alpha_i \alpha_j y_i y_j \, \bar{x}_i \cdot \bar{x}_j \right)$$

dot product

$$L = f\left(\bar{x}_i \cdot \bar{x}_j \right)$$

Now if $\sum \alpha_i y_i \underbrace{\bar{x}_i \cdot \bar{u}} + b \geq 0$ Decision Rule $+$ $-$

$$\bar{x}_i \cdot \bar{u}$$

also depends on this 'dot product'.

Maximal Margin Classifier

- **Hyperplane**: In a *p*-dimensional space, a *hyperplane* is a flat affine subspace of hyperplane dimension *p* − 1

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p = 0$$

- If $\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p > 0$

then this tells us that *X* lies to one side of the hyperplane. On the other hand, if

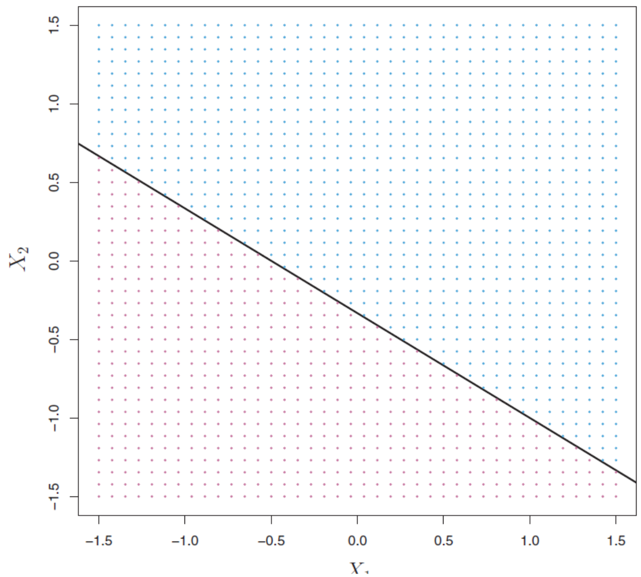
$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p < 0$$

then X lies on the other side of the hyperplane. So we can think of the hyperplane as dividing p-dimensional space into two halves.

Maximal Margin Classifier

- **Hyperplane**: In a *p*-dimensional space, a *hyperplane* is a flat affine subspace of hyperplane dimension *p* − 1

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p = 0$$



The hyperplane $1 + 2X_1 + 3X_2 = 0$ *is shown. The blue region is the set of points for which* $1+2X_1 +3X_2 > 0$*, and the purple region is the set of points for which* $1 + 2X_1 +3X_2 < 0$.

Maximal Margin Classifier

Classification Using a Separating Hyperplane

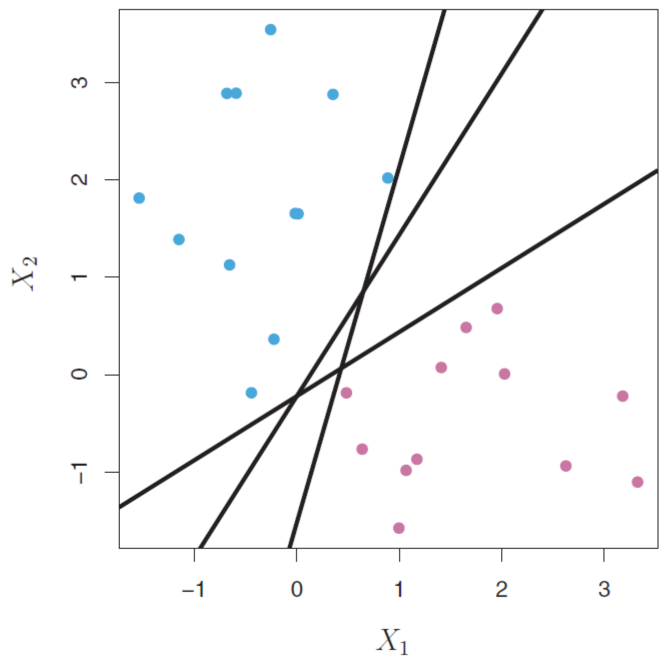
- Suppose that we have a n×p data matrix X, that consists of n training observations in p-dimensional space,

$$x_1 = \begin{pmatrix} x_{11} \\ \vdots \\ x_{1p} \end{pmatrix}, \ldots, x_n = \begin{pmatrix} x_{n1} \\ \vdots \\ x_{np} \end{pmatrix}$$

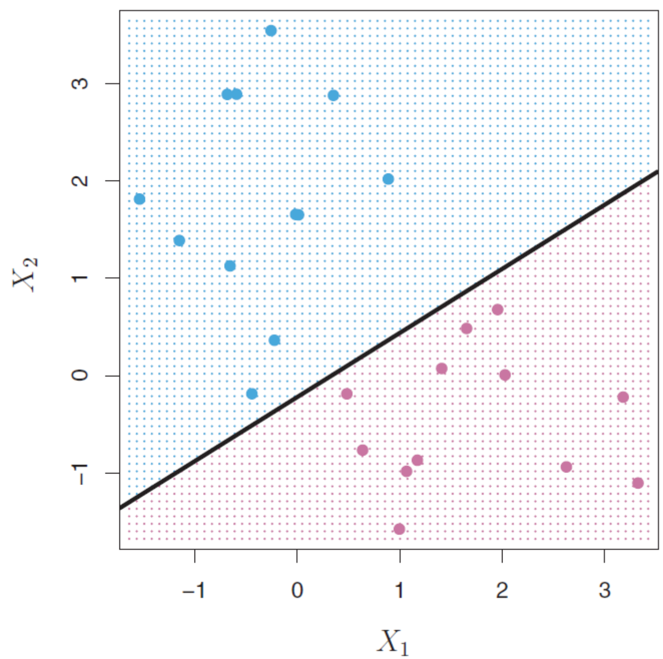
- and that these observations fall into two classes—that is, $y1, \ldots, yn \in \{-1, 1\}$ where −1 represents one class and 1 the other class.

Maximal Margin Classifier

Classification Using a Separating Hyperplane



There are two classes of observations, shown in blue and in purple, each of which has measurements on two variables. Three separating hyperplanes, out of many possible, are shown in black



A separating hyperplane is shown in black. The blue and purple grid indicates the decision rule made by Maximal Margin Classifier *based on this separating hyperplane.*

Maximal Margin Classifier

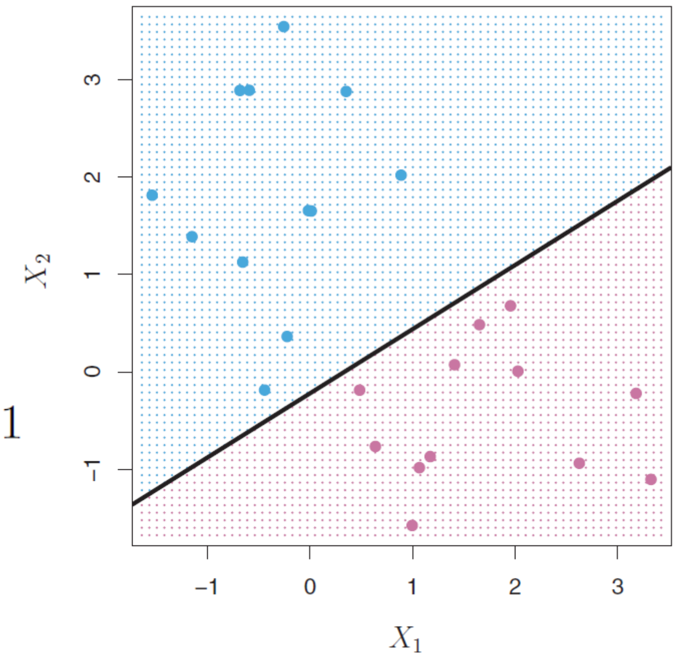
Classification Using a Separating Hyperplane

We can label the observations from the blue class as $y_i = 1$ and those from the purple class as $y_i = -1$. Then a separating hyperplane has the property that

$$\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip} > 0 \text{ if } y_i = 1$$

And

$$\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip} < 0 \text{ if } y_i = -1$$



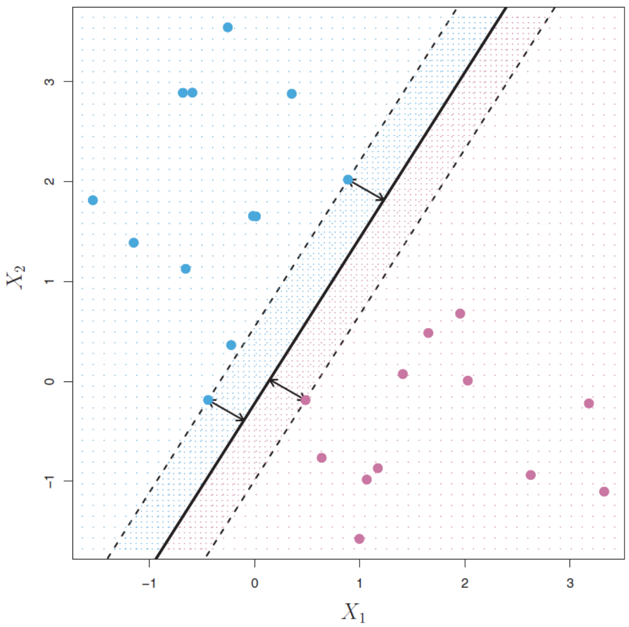
Equivalently, a separating hyperplane has the property that

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip}) > 0$$

Maximal Margin Classifier

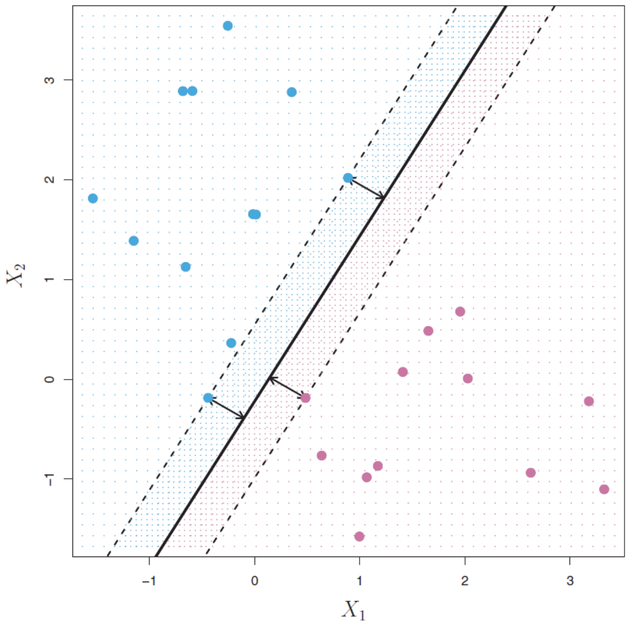
The Maximal Margin Classifier (separable case)

There are two classes of observations, shown in blue and in purple. The maximal margin hyperplane is shown as a solid line. The margin is the distance from the solid line to either of the dashed lines. The two blue points and the purple point that lie on the dashed lines are the support vectors, and the distance from those points to the hyperplane is indicated by arrows. The purple and blue grid indicates the decision rule made by a classifier based on this separating hyperplane.



Maximal Margin Classifier

Construction of the Maximal Margin Classifier (separable Case)



We now consider the task of constructing the maximal margin hyperplane based on a set of n training observations $x_1, \ldots, x_n \in \mathbb{R}_p$ and associated class labels $y_1, \ldots, y_n \in \{-1, 1\}$. Briefly, the maximal margin hyperplane is the solution to the optimization problem

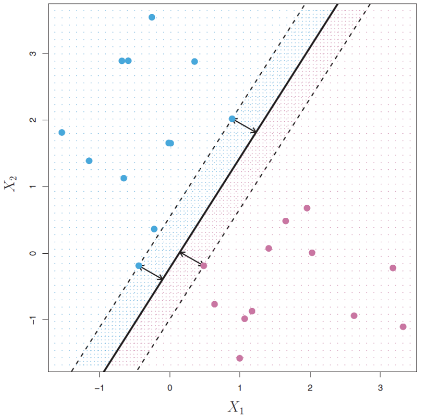
$$\underset{\beta_0,\beta_1,\ldots,\beta_p}{\text{maximize}}\, M$$

$$\text{subject to } \sum_{j=1}^{p} \beta_j^2 = 1,$$

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip}) \geq M \;\; \forall\, i = 1, \ldots, n.$$

Maximal Margin Classifier

Construction of the Maximal Margin Classifier (separable Case)



$$\underset{\beta_0,\beta_1,\ldots,\beta_p}{\text{maximize}}\, M$$

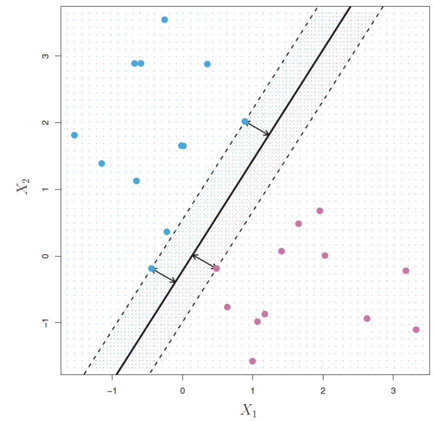
$$\text{subject to } \sum_{j=1}^{p} \beta_j^2 = 1,$$

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip}) \geq M \quad \forall\, i = 1, \ldots, n.$$

- The above constrain guarantees that each observation will be on the correct side of the hyperplane, provided that *M* is positive.
- With this constraint the perpendicular distance from the *i*th observation to the hyperplane is given by $$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip})$$
- Therefore, the constraints ensure that each observation is on the correct side of the hyperplane and at least a distance *M* from the hyperplane.

Maximal Margin Classifier

Construction of the Maximal Margin Classifier (separable Case)



$$\underset{\beta_0,\beta_1,\ldots,\beta_p}{\text{maximize}}\, M$$

$$\text{subject to } \sum_{j=1}^{p} \beta_j^2 = 1,$$

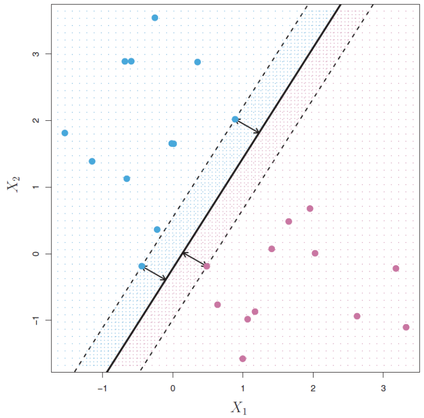
$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip}) \geq M \;\; \forall\, i = 1, \ldots, n.$$

- The above constrain guarantees that each observation will be on the correct side of the hyperplane, provided that *M* is positive.

- Actually, for each observation to be on the correct side of the hyperplane we would simply need $y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip}) > 0$, so the constraint in fact requires that each observation be on the correct side of the hyperplane, with some cushion, provided that M is positive.

Maximal Margin Classifier

Construction of the Maximal Margin Classifier (separable Case)



$$\text{subject to } \sum_{j=1}^{p} \beta_j^2 = 1,$$

This is not really a constraint on the hyperplane, since if $y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + . . . + \beta_p x_{ip}) = 0$ defines a hyperplane, then so does $k\, y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + . . . + \beta_p x_{ip}) = 0$ for any $k \neq 0$. However, this constraint adds meaning to

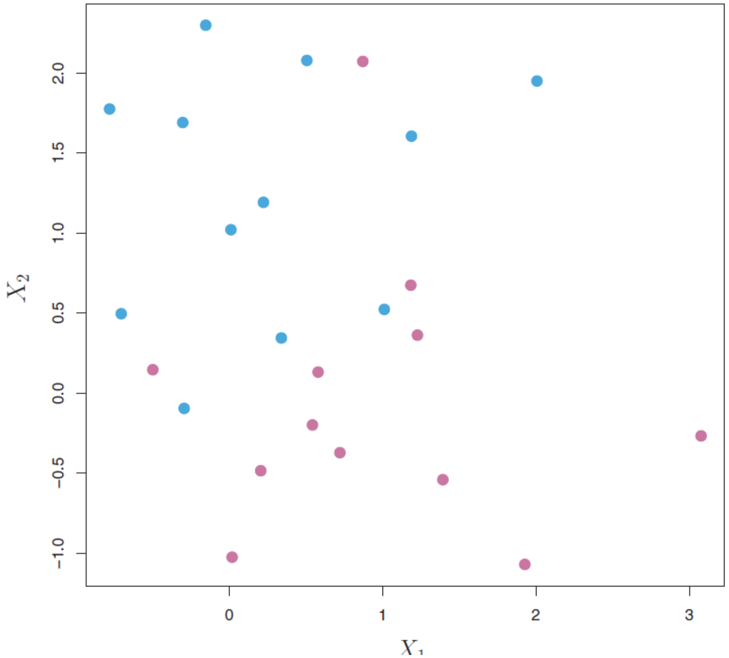
$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip}) \geq M \;\; \forall\, i = 1, \ldots, n.$$

One can show that with this constraint the perpendicular distance from the ith observation to the hyperplane is given by $y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + . . . + \beta_p x_{ip})$

Support Vector Classifiers

*The Maximal Margin Classifier (**Non-separable** Case)*

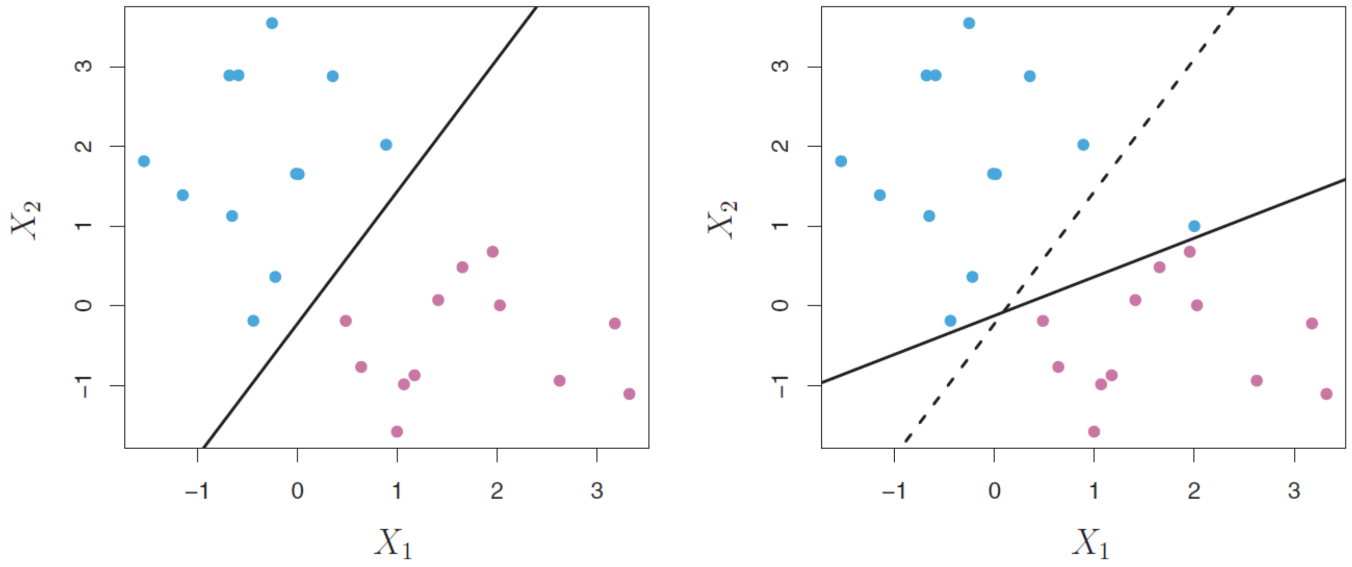
- In this case, the optimization problem has no solution with $M > 0$.
- Even if a separating hyperplane does exist, it would be very sensitive to individual observations.



There are two classes of observations, shown in blue and in purple. In this case, the two classes are not separable by a hyperplane, and so the maximal margin classifier cannot be used.

Support Vector Classifiers

*The Maximal Margin Classifier (**Non-separable** Case)*



The addition of a single observation could leads to a dramatic change in the maximal margin hyperplane.

Support Vector Classifiers

*The Maximal Margin Classifier (**Non-separable** Case)*

Thus, we want a classifier based on a hyperplane that does *not* perfectly separate the two classes, but has:

- Greater robustness to individual observations, and
- Better classification of *most* of the training observations.

The *support vector classifier*, sometimes called a *soft margin classifier*, does exactly this. Rather than seeking the largest possible margin so that every observation is not only on the correct side of the hyperplane but also on the correct side of the margin, we instead allow some observations to be on the incorrect side of the margin, or even the incorrect side of the hyperplane, to achieve **Greater robustness on *most* of the training observations.**

Support Vector Classifiers

Construction of the Support Vector Classifier

In this the hyperplane is chosen to correctly separate most of the training observations into the two classes, but may misclassify a few observations. It is the solution to the optimization problem

$$\underset{\beta_0,\beta_1,\ldots,\beta_p,\epsilon_1,\ldots,\epsilon_n}{\text{maximize}} \quad M$$

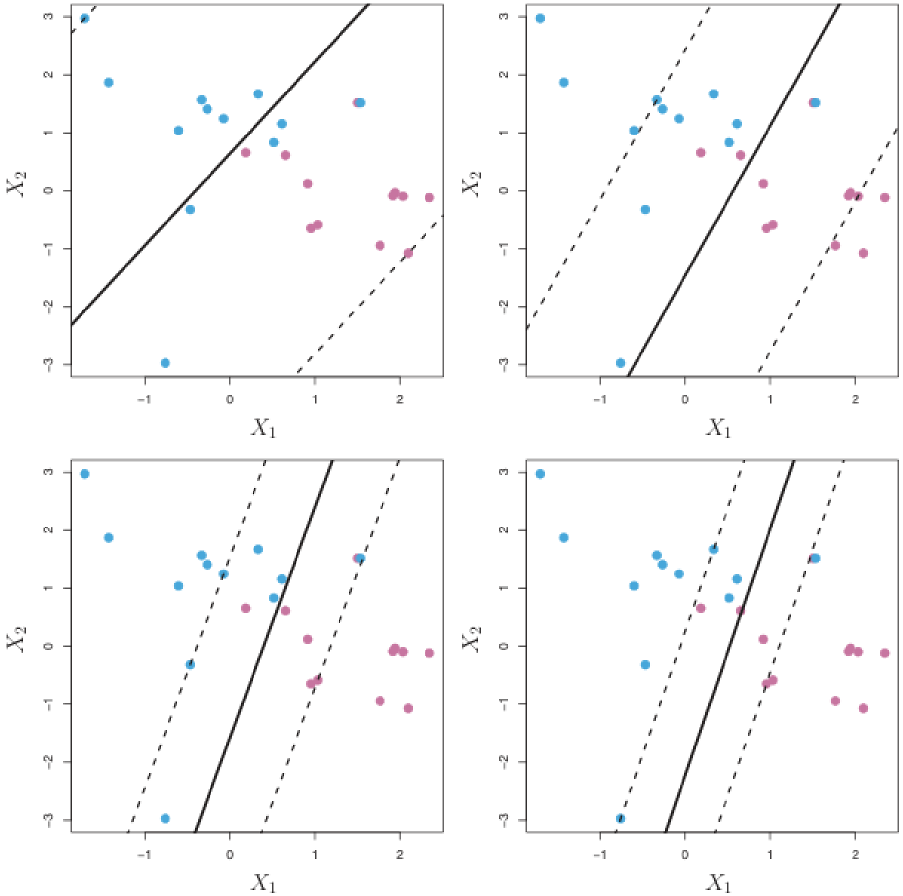
$$\text{subject to } \sum_{j=1}^{p} \beta_j^2 = 1,$$

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip}) \geq M(1 - \epsilon_i)$$

$$\epsilon_i \geq 0, \ \sum_{i=1}^{n} \epsilon_i \leq C,$$

where *C* is a nonnegative tuning parameter. We seek to make M as large as possible. $\varepsilon_1, \ldots, \varepsilon_n$ are *slack variables* that allow individual observations to be on the wrong side of the margin or the hyperplane.

Support Vector Classifiers



A support vector classifier was fit using four different values of the tuning parameter C. The largest value of C was used in the top left panel, and smaller values were used in the top right, bottom left, and bottom right panels. When C is large, then there is a high tolerance for observations being on the wrong side of the margin, and so the margin will be large. As C decreases, the tolerance for observations being on the wrong side of the margin decreases, and the margin narrows.

Mercer's theorem

Mercer's theorem states that for any data set X and any kernel function

$k : \mathbb{R}^p \times \mathbb{R}^p \rightarrow \mathbb{R}$ there exists a mapping $\varphi : \mathbb{R}^p \rightarrow \mathbb{R}^q$ so that

$$k(x_j, x_k) = \varphi(x_j) \cdot \varphi(x_k)^T$$

This means that a mapping from X to X' can be implicitly done by replacing scalar products in X' by kernel function values in X, without explicitly specifying φ, and without explicitly computing X'. Replacing scalar products in X' by kernel functions in X is called the *kernel trick*. Some frequently used kernel functions are

Support Vector Classifier

$$\Rightarrow L = \sum \alpha_i - \frac{1}{2} \sum_i \sum_j \left(\alpha_i \alpha_j y_i y_j \, \bar{x}_i \cdot \bar{x}_j \right)$$

$$L = f\left(\bar{x}_i \cdot \bar{x}_j \right)$$

dot product

Now if $\sum \alpha_i y_i \underbrace{\bar{x}_i \cdot \bar{u}} + b \geq 0$ Decision Rule

$+$

$-$

$$\bar{x}_i \cdot \bar{u}$$

also depends on the "." dot product.

Support Vector Classifier

Decision rule as well as L depends only on the dot product.

∴ For a non-linear hyperplane we can transform the original input space to another space by a

$$\bar{X} \xrightarrow{\phi(\bar{X})} \bar{Z}$$

Support Vector Classifier

$$\bar{X} \xrightarrow{\phi(\bar{x})} \bar{z}$$

You only need

$\bar{X}_i \cdot \bar{X}_j$ $\qquad\qquad$ $\bar{z}_i \cdot \bar{z}_j$

$\Rightarrow$ You only need to know $\phi(\bar{x}_i) \cdot \phi(\bar{x}_j)$

$\equiv \bar{z}_i \cdot \bar{z}_j$

From Mercer's theorem

This is achieved by a Kernal Function

$$K(\bar{x}_i, \bar{x}_j)$$

Support Vector Classifier

Common Kernel Function

Polynomial $K(\bar{x}_i, \bar{x}_j) = (\bar{x}_i \cdot \bar{x}_j)^a$
(homogeneous)

(Non-Homogeneous) $(\bar{x}_i \cdot \bar{x}_j + r)^a$

Gaussian radial basis function:

$$K(\bar{x}_i, \bar{x}_j) = \exp\left(- \frac{\| \bar{x}_i - \bar{x}_j \|}{2\sigma^2} \right)$$

Support Vector Classifier

Hyperbolic tangent:

$$K(\bar{x}_i, \bar{x}_j) = \tanh(k\, \bar{x}_i \cdot \bar{x}_j + c)$$

for some $k > 0$, $c < 0$

The decision boundary can be then given by

$$\bar{z}_i = \phi(\bar{x}_i), \quad \bar{z}_i \cdot \bar{z}_j = K(x_i, x_j)$$

and $\bar{\omega} \cdot \bar{z}(i) = \bar{\omega} \cdot \phi(\bar{x}_i)$

$$\Rightarrow \bar{\omega} \cdot \bar{z}_i = \sum \alpha_j y_j \cdot K(\bar{x}_i, x_j)$$

The Support Vector Machine Kernel Trick

linear kernel

$$k(x_j, x_k) = x_j \cdot x_k^T$$

polynomial kernel

$$k(x_j, x_k) = (x_j \cdot x_k^T)^d, \quad d \in \{2, 3, \ldots\}$$

Gaussian kernel

$$k(x_j, x_k) = e^{-\frac{\|x_j - x_k\|^2}{\sigma^2}}, \quad \sigma > 0$$

hyperbolic tangent kernel

$$k(x_j, x_k) = 1 - \tanh \frac{\|x_j - x_k\|^2}{\sigma^2}, \quad \sigma > 0$$

radial basis function (RBF) kernel

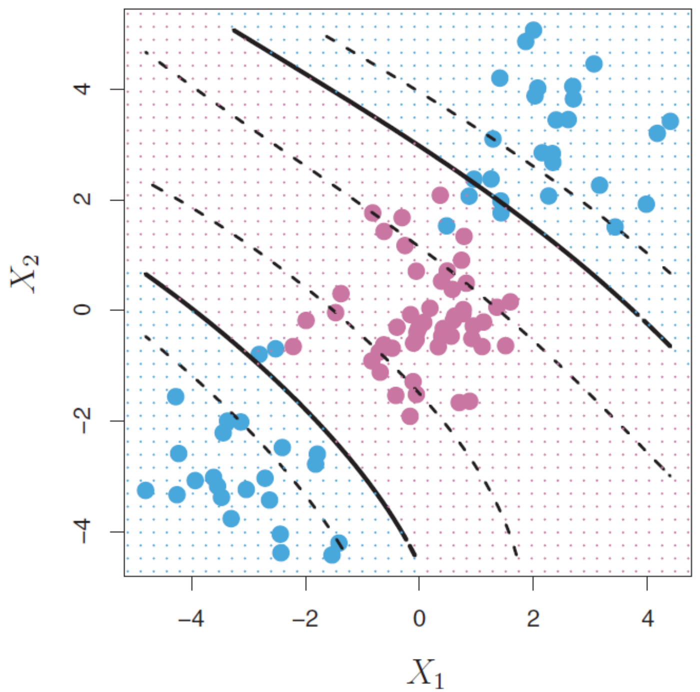
$$k(x_j, x_k) = f\left(\|x_j - x_k\|\right)$$

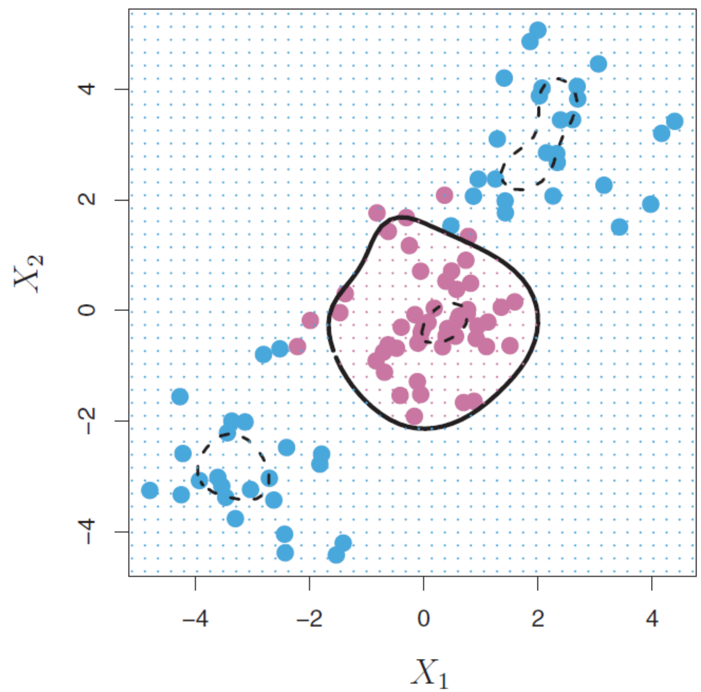
The SVM classification constraints are

$$\sum_{y_j=1} \alpha_j k(x_j, x_k) - \sum_{y_j=2} \alpha_j k(x_j, x_k) + b \geq +1 - \xi_k \quad \text{if } y_k = 1$$

$$\sum_{y_j=1} \alpha_j k(x_j, x_k) - \sum_{y_j=2} \alpha_j k(x_j, x_k) + b \leq -1 + \xi_k \quad \text{if } y_k = 2$$

The Support Vector Machine





An SVM with a polynomial kernel of degree 3 is applied to the non-linear data, resulting in a far more appropriate decision rule

An SVM with a radial kernel is applied. In this example, either kernel is capable of capturing the decision boundary.