Machine Learning II
Week 5 Lecture – 6th November 2019
Support Vector Machines 1

Contents

- Maximal margin classifier
- Support vector classifier with two classes

This and next week's lectures follow James et. al Chapter 9

A support vector machine (SVM) is a **Supervised Learning** method.

It is a **classification** method with K classes.
Usually an SVM is a **binary classifier** so $K{=}2$.
Extending to $K > 2$ is not trivial.

Where does this strange name come from?

Vector Here a vector is a data point, an element (row) in a data set with numeric **values** for p-variables

Support Some data points (vectors) define (support) the boundary between two classification regions and

Machine As in machine learning: a prediction model.

Our starting point is an easy scenario which is similar to linear discriminant analysis (LDA).

Maximal Margin Classifier

For ease, we will assume at first that the data have two predictor variables and one outcome variable.

The “vectors” are two dimensional points in $\mathbb{R}^2$, and each point belongs to one of K=2 classes labelled $\{-1, 1\}$.

Choose an arbitrary line and assign one class to each side of that boundary. It is likely that there are many misclassifications.

We can rotate and shift the line so that the number of misclassifications decrease. If we are very lucky there is a boundary which gives no misclassifications, the two classes are totally separated.

Defining the boundary

Suppose we have a straight line with the form y=$ax + b$, we can express this line in another form

$$\beta_0 + \beta_1 x + \beta_2 y = 0$$

Note that this expression is overparametised, if you multiply all 3 βs by a constant, the same boundary is obtained.

Now consider 3 points: (x_1, y_1) lies on the boundary, (x_2, y_2) lies above the boundary and (x_3, y_3) lies below the boundary.

$$\begin{aligned} y_1 &= ax_1 + b & \Leftrightarrow \quad & \beta_0 + \beta_1 x_1 + \beta_2 y_1 = 0 \\ y_2 &> ax_2 + b & \Leftrightarrow \quad & \beta_0 + \beta_1 x_2 + \beta_2 y_2 > 0 \\ y_3 &< ax_3 + b & \Leftrightarrow \quad & \beta_0 + \beta_1 x_3 + \beta_2 y_3 < 0 \end{aligned}$$

This gives an easy classification rule
Assign (x, y) to class 1 if $f(x, y)$=$\beta_0 + \beta_1 x + \beta_2 y \geqslant 0$
otherwise assign class -1.

As we usually reserve y_i for the outcome variable, we will redefine the i-th point in $\mathbb{R}^2$ as $\boldsymbol{x}_i = (x_{i1}, x_{i2})$

The reason for the β definition of the boundary line is that it easily generalises into higher dimensions.

If each data point is a 3-dimensional vector $\boldsymbol{x}_i = (x_{i1}, x_{i2}, x_{i3})$ then the boundary is a **plane** defined by

$$\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 = 0$$

In higher dimensions the boundary is called a **hyperplane**.

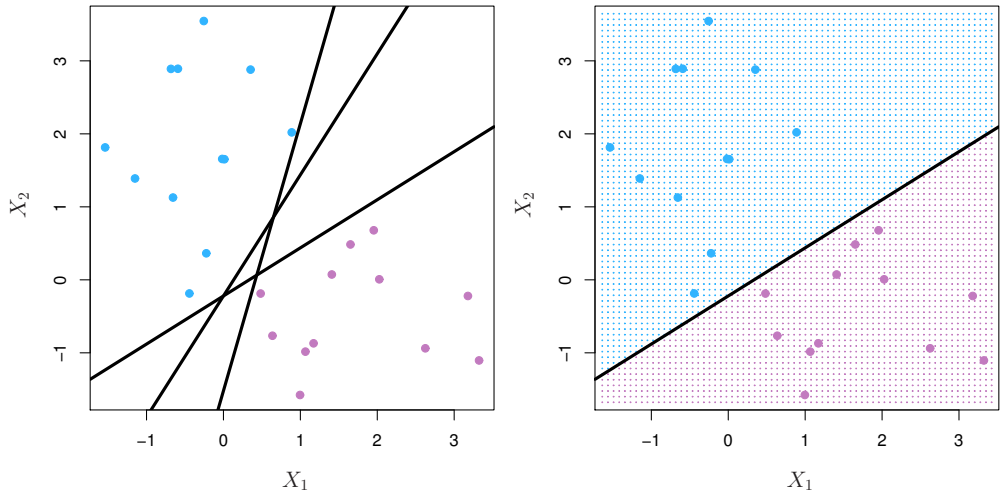
From now on we will use the term hyperplane for the boundary, even if the data are 2 or 3 dimensional.

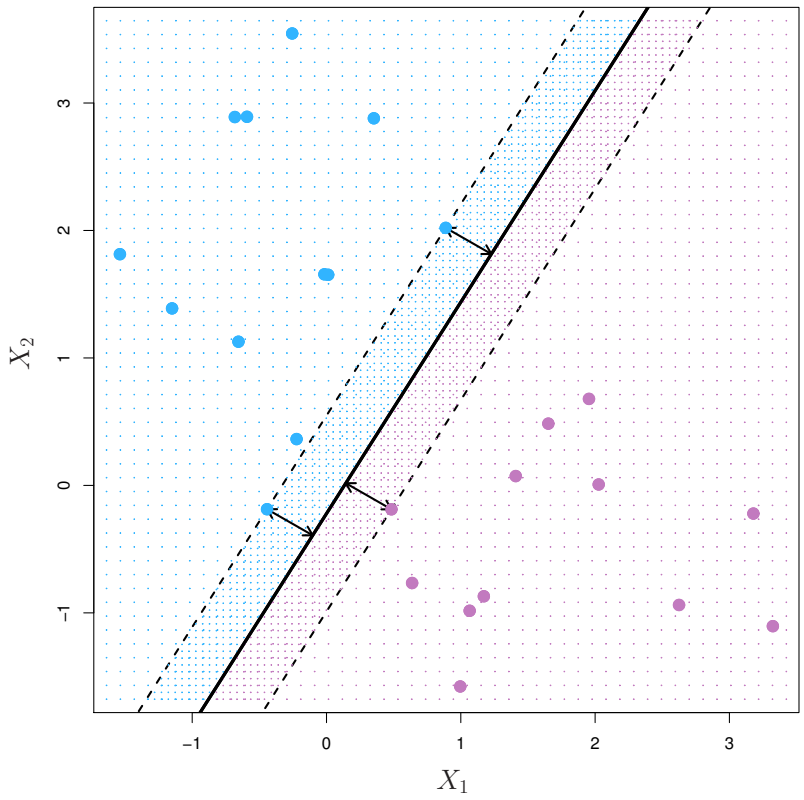
In this lecture, we only consider $\boldsymbol{x}_i \in \mathbb{R}^2$, the adaptation into higer dimensions is straight forward.

If our data are separable, then there will infinitely many hyperplanes which are perfect classifiers (left diagram).

Do we care which hyperplane to use? Yes, because we want to be able to use the method to make the prediction on other data.

The *maximal marginal hyperplane* is the hyperplane which is the furthest away from all data points (right diagram). The distance between the hyperplane and the closest point is called the *margin*





The dotted lines are the margin edges. There are no points in the margin.

Let M be margin distance, i.e. half the margin width. We want to maximise M.

We require that every point is on the correct side of the hyperplane, so:

For y_i=1 $\quad \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} > 0$

For $y_i = -1$ $\quad \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} < 0$

Together $\quad y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2}) > 0$

If we impose the normalising constraint $\sum_{j=1}^{2} \beta_j^2 = 1$, then the distance from the hyperplane $\beta_0 + \beta_1 x_1 + \beta_2 x_2$=0 to the point (x_{i1}, x_{i2}) is

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i1}).$$

To ensure that all points are at least M units away from the hyperplane, we require

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2}) \geqslant M.$$

Margin classifier algorithm

Find the *maximin* solution for $y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2})$. Ie.

$$M = \max_{\beta_0,\beta_1,\beta_2} \left\{ \min_i \{y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2})\} \right\}, \qquad \text{subject to } \sum_{j=1}^{2} \beta_j^2 = 1.$$

In terms of developing the algorithm it is better to re-express the algorithm as

- Choose $\beta_0, \beta_1, \beta_2$ to maximise M, such that
- $y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2}) \geqslant M$ for i=1, . . . n,
- subject to the constraint $\sum_{j=1}^{2} \beta_j^2 = 1$.

The size of the maximal margin can be used as a measure of how confident we are in the maximal margin classifier.

N.B. This is the format in James et al. Many other authors use an alternative equivalent: minimise $\sum_{j=0}^{2} \beta_i^2$ subject to $y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2}) \geqslant 1$.

Usually there are three data points that lie on the edge of the margin, their distance from the optimal hyperplane is M. These points are called the *support vectors*, they 'support' the hyperplane.

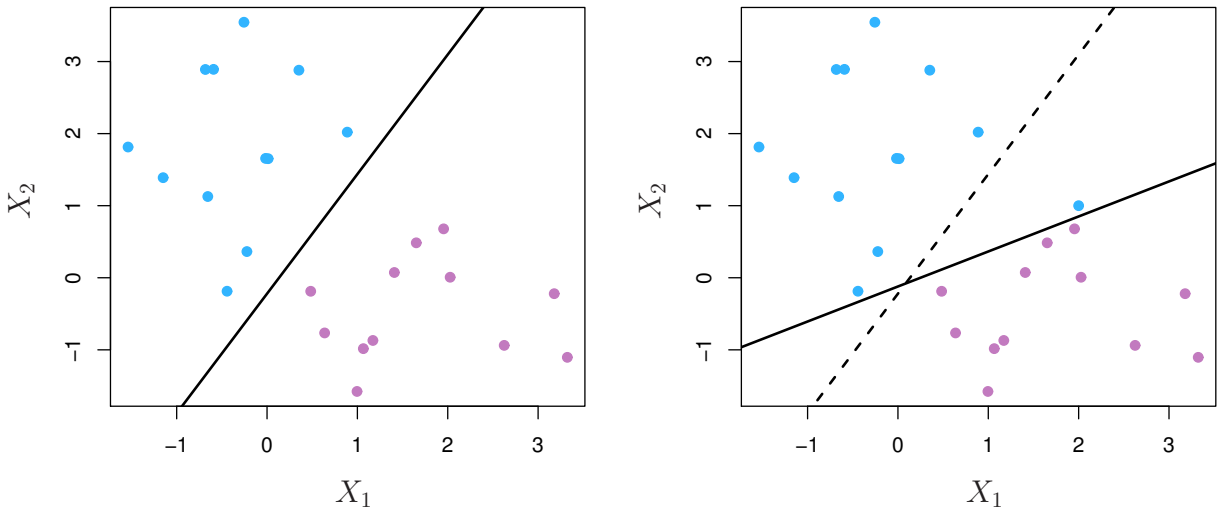
If you remove a non-support vector from the dataset, the optimal hyperplane won't change.

Once the best hyperplane has been found, the predictions are easy to obtain by taking the sign of the hyperplane function.
Ie. for a test observation vector $\boldsymbol{x}_t$

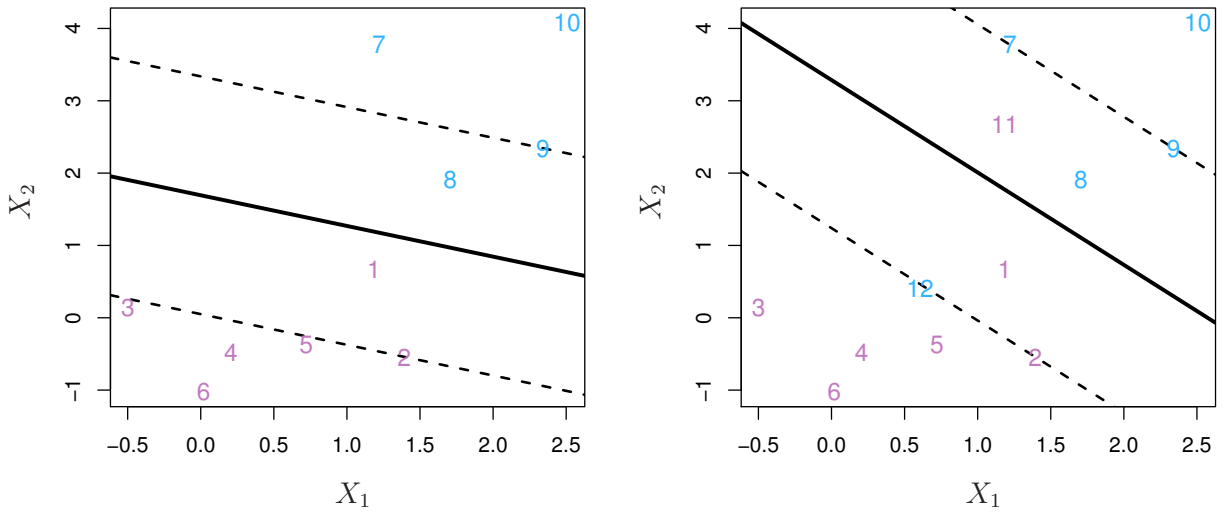
$$\widehat{y}_t = \begin{cases} 1 & \text{for } \beta_0 + \beta_1 x_{t1} + \beta_2 x_{t2} \geqslant 0 \\ -1 & \text{for } \beta_0 + \beta_1 x_{t1} + \beta_2 x_{t2} < 0 \end{cases}$$

Support vector classifier

Usually we don't have two perfectly separated classes. Even if there is a hyperplane that separates the data perfectly, the resulting regions can be quite sensitive to values near the hyperplane.



The *support vector classifier* allows for some points to be in the margin or on the wrong side of the border, the hyperplane has a "soft margin".



The adaptation from the previous algorithm is straight forward. We introduce *slack variables* for each observation $\epsilon_1, \ldots, \epsilon_n$, which allow some of the points to move into the margin.

- $\max_{\beta_0, \beta_1, \beta_2, \epsilon_1, \ldots, \epsilon_n} M$, such that
- $y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2}) \geqslant M(1 - \epsilon_i)$ for i=1, $\ldots$ n,
- subject to the constraints $\sum_{j=1}^{2} \beta_j^2 = 1$, $\epsilon_i \geqslant 0$ and $\sum_{i=1}^{n} \epsilon_i < C$.

C is a positive smoothing parameter.

Many of the ϵ_i will be zero, indicating that the i-th data point is on the correct side of the margin.

If ϵ_i is between zero and one, the i-th data point is in the margin, on the correct side of the hyperplane.

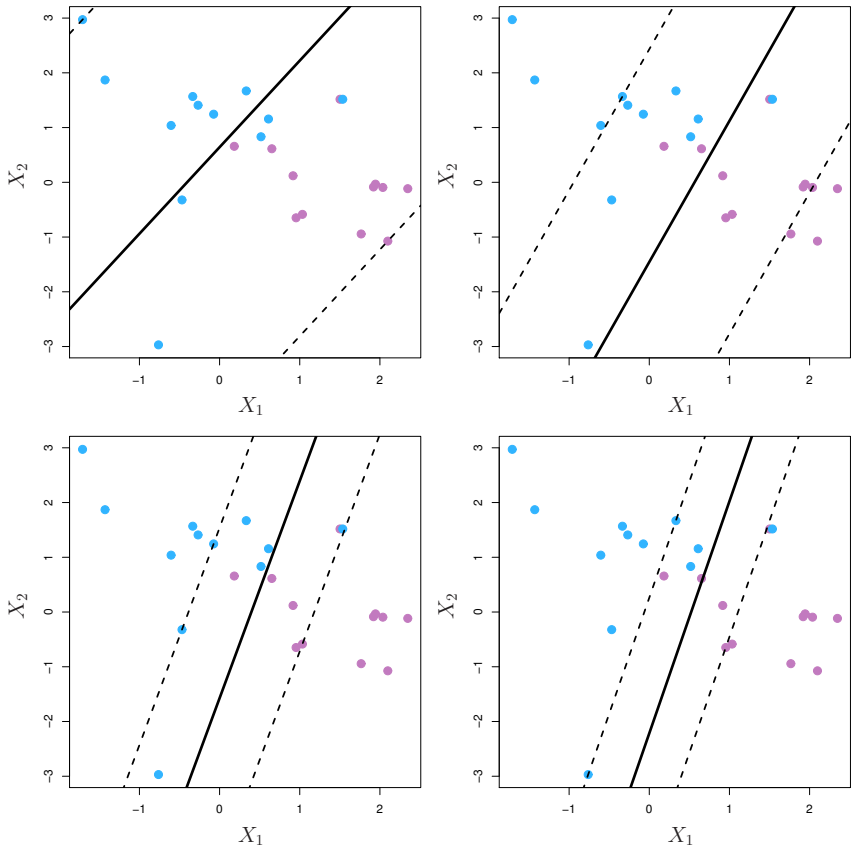
If ϵ_i is between one and two, the i-th data point is in the margin, on the wrong side of the hyperplane.

If ϵ_i is greater than two, the i-th data point is outside the margin, on the wrong side of the hyperplane.

C can be considered as a budget, every point that crosses the margin has a cost ϵ and the total cost cannot exceed C.

If C=0, we return to the maximal margin classifier.

The cost of being on the wrong side of the hyperplane is at least 1 unit, so if we want no more that 5 misclassifications in the training data, we can start by setting C=5.



By decreasing C the width of the margin decreases.

Predicted values from support vector classifier are obtained in exactly the same way as for the maximal margin classifier.

All the points which lie in the margin or on the wrong side are the support vectors. They all contribute to defining the optimal hyperplane.

As before removing a non-support vector has no influence on the hyperplane.

Moving a non-support vector has no influence on the hyperplane unless it crosses over the margin.

This property means that extreme values on the correct side of the hyperplane do not affect the classifier, meaning that the classifier is moderately robust.

LDA gives similar results but is influenced by extreme values in all directions.

Next week

- Support vector machines for non linear boundaries.
- SVMs when $K > 2$.