MONASH University

ETC3250

Business Analytics

Week 5.
Support Vector Machines

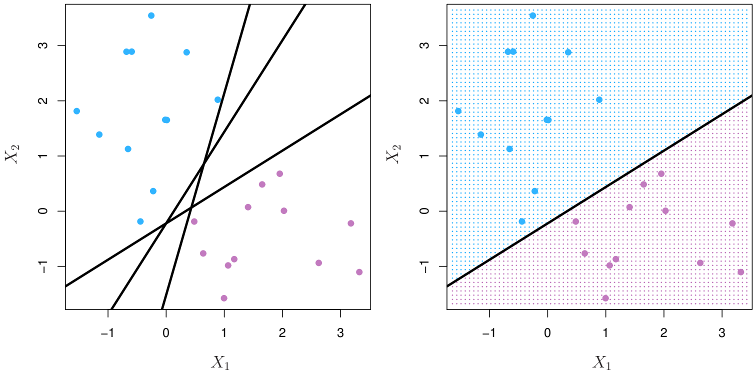
24 August 2017

Outline

Week	Topic	Chapter	Lecturer
1	Introduction to business analytics & R	1	Souhaib
2	Statistical learning	2	Souhaib
3	Regression for prediction	3,7	Tas & David
4	Classification	4	Souhaib
5	Classification	4, 9	Souhaib
	Comparison of classifiers		Souhaib
	Support Vector Machines		Souhaib
6	Resampling methods	5	Souhaib
7	Dimension reduction	6,10	Souhaib
8	Advanced regression	6	Souhaib
9	Advanced learning methods	8	Souhaib
	Semester break		
10	Clustering	10	Souhaib
11	Visualization		Souhaib
12	Data wrangling		Souhaib

Separating Hyperplane

In a p-dimensional space, a **hyperplane** is a flat affine subspace of dimension $p - 1$.



Classification Using a Separating Hyperplane

The equation of p-dimensional hyperplane is given by

$$\beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p = 0.$$

If $x_i \in \mathbb{R}^p$ and $y_i \in \{-1, 1\}$ for $i = 1, \ldots, n$, then

$$\beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip} > 0 \text{ if } y_i = 1,$$
$$\beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip} < 0 \text{ if } y_i = -1.$$

Equivalently,

$$y_i(\beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip}) > 0.$$

Classification Using a Separating Hyperplane

- A new observation is assigned a class depending on **which side** of the hyperplane it is located
- we classify the test observation x^* based on the **sign** of

$$f(x^*) = \beta_0 + \beta_1 x_1^* + \cdots + \beta_p x_p^*$$

- If $f(x^*) > 0$, class 1, and if $f(x^*) < 0$, class -1, i.e. $C(x^*) = \textbf{sign}(f(x^*))$.

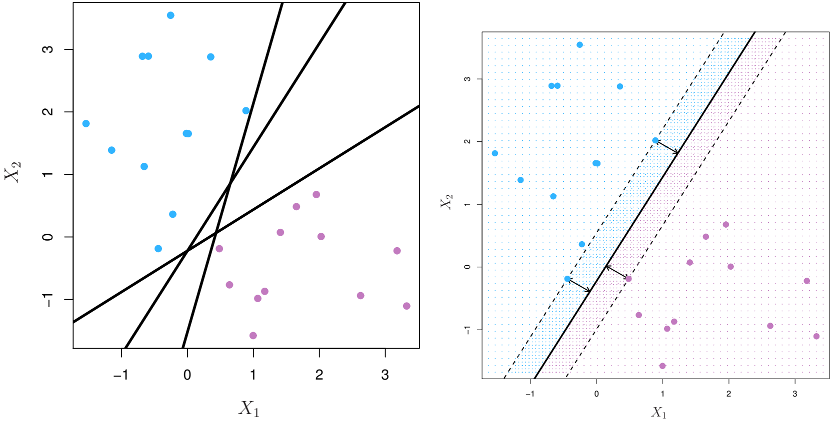
What about the **magnitude** of $f(x^*)$?

- $f(x^*)$ far from zero $\rightarrow x^*$ lies far from the hyperplane + **more confident** about our classification
- $f(x^*)$ close to zero $\rightarrow x^*$ near the hyperplane + **less confident** about our classification

The Maximal Margin Classifier

- If our data can be perfectly separated using a hyperplane, then there will in fact exist an **infinite number of such hyperplanes**.
- The **margin** is the (perpendicular) distance from each training observation to a given separating hyperplane
- The **optimal separating hyperplane** (or **maximal margin hyperplane**) is the separating hyperplane for which the margin is *largest*
- We can then classify a test observation based on which side of the maximal margin hyperplane it lies. This is known as the **maximal margin classifier**.

The Maximal Margin Classifier



Support vectors

- The **support vectors** are equidistant from the maximal margin hyperplane and lie along the dashed lines indicating the width of the margin.
- They **support** the maximal margin hyperplane in the sense that if these points were moved slightly then the maximal margin hyperplane would move as well
- The maximal margin hyperplane depends directly on the support vectors, but **not on the other observations**

The Maximal Margin Classifier

If $x_i \in \mathbb{R}^p$ and $y_i \in \{-1, 1\}$ for $i = 1, \ldots, n$, then the maximal margin hyperplane is the solution to the following optimization problem ($M \geq 0$):

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

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

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

The constraints ensure that each observation is on the **correct side** of the hyperplane and **at least a (margin) distance** M from the hyperplane.

Non-seperable case? robustness?

The Maximal Margin Classifier

If $x_i \in \mathbb{R}^p$ and $y_i \in \{-1, 1\}$ for $i = 1, \ldots, n$, then the maximal margin hyperplane is the solution to the following optimization problem ($M \geq 0$):

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

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

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

The constraints ensure that each observation is on the **correct side** of the hyperplane and **at least a (margin) distance** M from the hyperplane.

Non-seperable case? robustness?

The Maximal Margin Classifier

If $x_i \in \mathbb{R}^p$ and $y_i \in \{-1, 1\}$ for $i = 1, \ldots, n$, then the maximal margin hyperplane is the solution to the following optimization problem ($M \geq 0$):

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

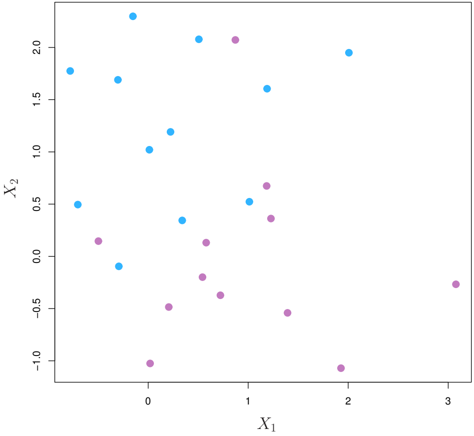
$$\text{subject to} \; \sum_{j=1}^{p} \beta_j^2 = 1,$$

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

The constraints ensure that each observation is on the **correct side** of the hyperplane and **at least a (margin) distance** *M* from the hyperplane.

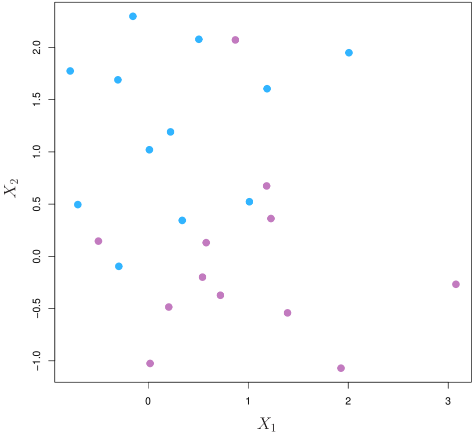
Non-seperable case? robustness?

Non-separable case



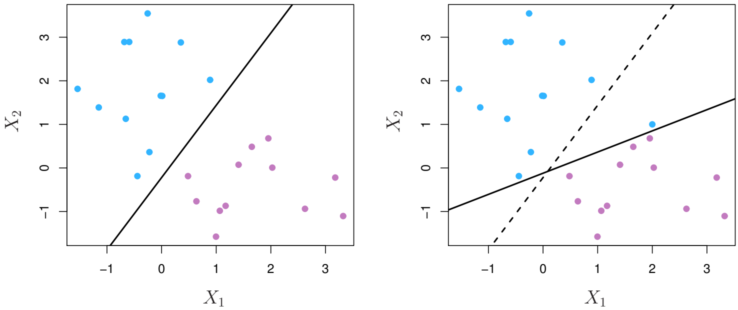
The optimization problem has no solution with $M > 0$.

Non-separable case



The optimization problem has no solution with $M > 0$.

Lack of robustness



Sensitivity to individual observations:

- → might overfit the training data.
- → small margin (measure of our confidence).

Support Vector Classifier

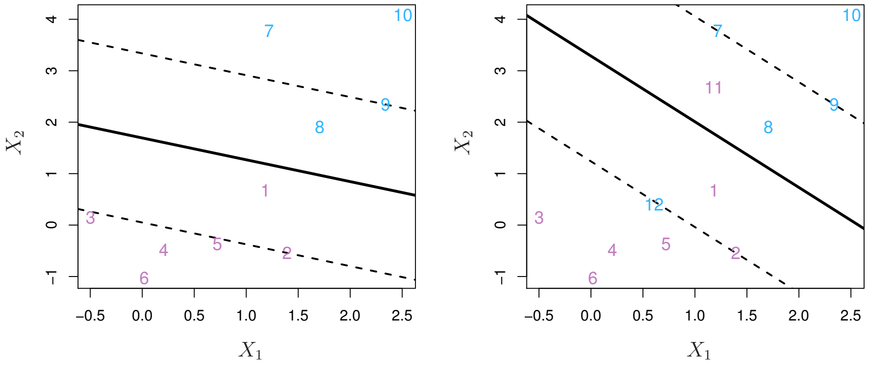
The *support vector classifier* (SVC) is based on a hyperplane that does not **perfectly separate** the two classes, in the interest of

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

The SVC is also called a *soft margin classifier* because the margin can be violated by some of the training observations.

- Most of the observations are on the **correct side of the margin**.
- Few observations are on the **wrong side of the margin**.
- Few observations are on the **wrong side of the hyperplane**.

Support Vector Classifier



Support Vector Classifier

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

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

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

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

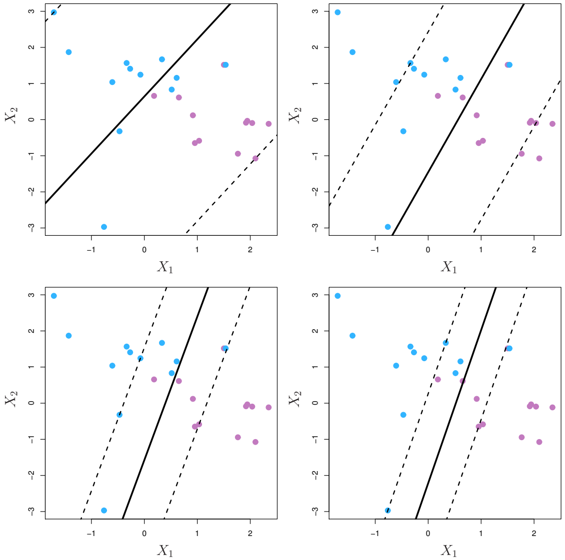
where ε_i tells us where the *i*th observation is located and C is a nonnegative **tuning parameter**.

- $\varepsilon_i = 0$: *correct side of the margin*,
- $\varepsilon_i > 0$: *wrong side of the margin* (violation of the margin),
- $\varepsilon_i > 1$: *wrong side of the hyperplane*.

Tuning parameter

- C determines the **number** and **severity** of the violations to the margin (and to the hyperplane) that we will **tolerate**
  - $C = 0$: no budget for violations to the margin, and $\varepsilon_1 = \cdots = \varepsilon_n = 0$
  - $C > 0$: no more than C observations can be on the *wrong side of the hyperplane*
- C increases/decreases $\rightarrow$, margin wider/more narrow
- An observation that lies strictly on the correct side of the margin **does not affect** the classifier
- Only observations that either lie on the margin or that violate the margin (known as support vectors) **will affect** the classifier

Tuning parameter



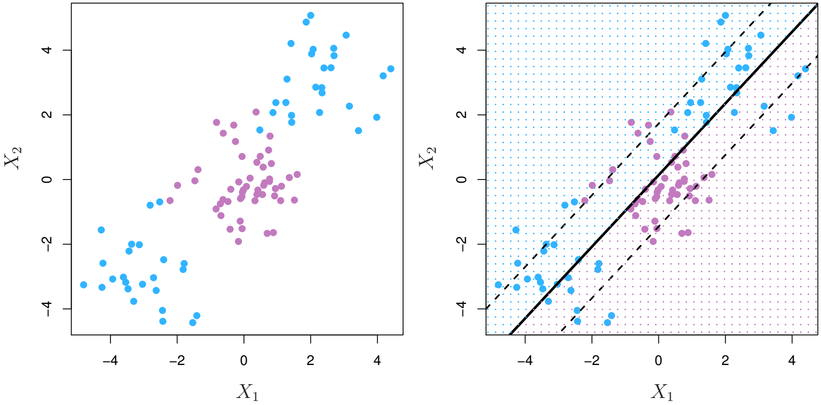
Decreasing values of the tuning parameter C.

Bias and variance tradeoff

- C controls the bias-variance trade-off of the classifier
  - C large: high bias but low variance
  - C small: low bias but high variance
- When C is large, many observations violate the margin, and so there are many support vectors. Many observations are involved in determining the hyperplane.
- When C is small, then there will be fewer support vectors and hence the resulting classifier will have low bias but high variance
- In practice, C is generally chosen via cross-validation

The dependence on a few support vectors means that it is quite *robust to observations that are far away from the hyperplane* (compared to LDA, for example).

Non-linear Decision Boundaries



Enlarged feature space

In regression, we consider **enlarging the feature space** using functions of the predictors such as *quadratic* and *cubic terms*, in order to address this non-linearity.

1. Obtain new features by computing **nonlinear transformations** of original features:

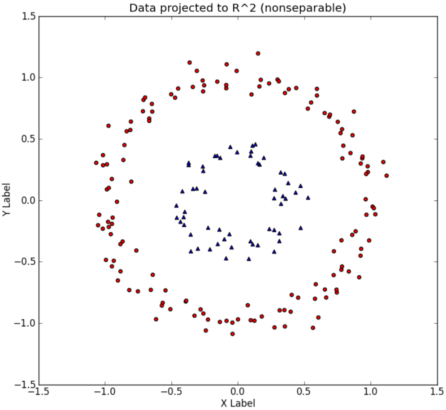
$$X = (X_1, X_2, \ldots, X_p) \xrightarrow{\Phi} Z = (Z_1, Z_2, \ldots, Z_{\tilde{p}})$$

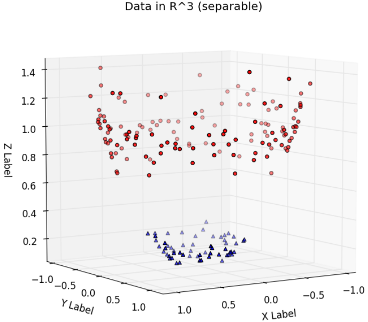
   where $Z_j = \phi_j(X)$.

2. Run the support vector classifier using the **new features**:

$$f(Z) = \beta_0^{'} + \beta_1^{'} Z_1 + \beta_2^{'} Z_2 + \ldots \beta_{\tilde{p}}^{'} Z_{\tilde{p}}$$

Enlarged feature space





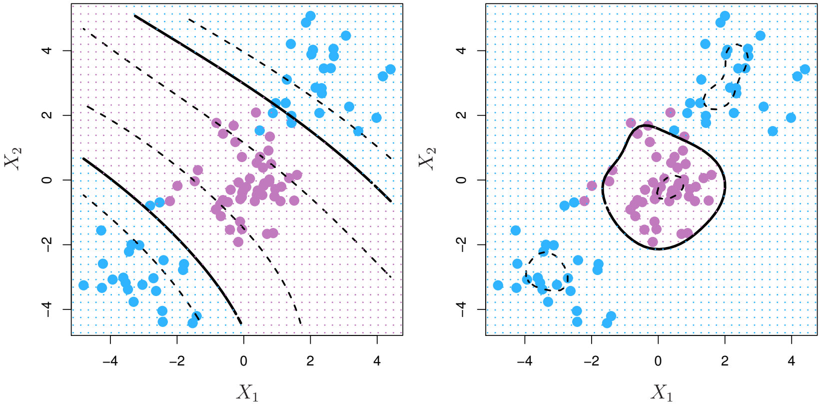
source: `http://www.eric-kim.net/eric-kim-net/posts/1/kernel_trick.html`

Original feature space: X_1, X_2

Enlarged feature space: $Z_1 = X_1, Z_2 = X_2$ and $Z_3 = X_1^2 + X_2^2$

Non-linear Decision Boundaries

The decision boundary of the support vector classifier will be **linear** in the enlarged feature space, but (generally) **nonlinear** in the original feature space



Support vector machines

- There are many possible ways to enlarge the feature space, and that unless we are careful, we could end up with a **huge number of features**.
- The support vector machine (SVM) is an extension of the support vector support classifier that results from enlarging the feature space in a way that leads to **efficient computations**
- The support vector machines (SVM) uses the **kernel trick**

Support vector machines

The **inner product** of two p-vectors a and b is defined as

$$\langle a, b \rangle = \sum_{i=1}^{p} a_i b_i.$$

It can be shown that the **linear support vector classifier** can be represented as

$$f(x) = \beta_0 + \sum_{i=1}^{n} \alpha_i \langle x, x_i \rangle$$

which depends on n parameters $\alpha_i, i = 1, \dots, n$.
In representing the linear classifier $f(x)$, and in computing its coefficients, all we need are **inner products**.
$\rightarrow$ **Is it also the case with an enlarged feature space?**

Support vector machines

It can be shown that the **support vector machine classifier** can be represented as

$$f(x) = \beta_0 + \sum_{i=1}^{n} \alpha_i \, \langle \Phi(x), \Phi(x_i) \rangle.$$

The previous expression involves $\Phi(x)$ only through inner products.

Can we compute $\langle \Phi(x), \Phi(x^{'}) \rangle$ without transforming x and $x^{'}$?

The kernel trick

Consider for example $K(x,x^{'}) = (1 + x^T x^{'})^2$, where $x \in \mathbb{R}^2$.

$$
\begin{aligned}
&K(x,x^{'})\\
&= (1 + x^T x^{'})^2\\
&= (1 + x_1 x_1^{'} + x_2 x_2^{'})^2\\
&= 1 + x_1^2 x_1^{'2} + x_2^2 x_2^{'2} + 2x_1 x_1^{'} + 2x_2 x_2^{'} + 2x_1 x_1^{'} x_2 x_2^{'}\\
&= \langle (1, x_1^2, x_2^2, \sqrt{2}x_1, \sqrt{2}x_2, \sqrt{2}x_1 x_2), (1, x_1^{'2}, x_2^{'2}, \sqrt{2}x_1^{'}, \sqrt{2}x_2^{'}, \sqrt{2}x_1^{'} x_2^{'}) \rangle
\end{aligned}
$$

$\rightarrow$ it is an **inner product** in an enlarged space.

$\rightarrow$ computing $(1 + x^T x^{'})^2$ is significantly **faster** than computing the inner product in the enlarged space.

$\rightarrow$ we **do not need** to specify the transformation $\Phi(x)$ at all, but require only knowledge of the **kernel function**

The kernel trick

Consider for example $K(x, x') = (1 + x^T x')^2$, where $x \in \mathbb{R}^2$.

$$
\begin{aligned}
& K(x, x') \\
& = (1 + x^T x')^2 \\
& = (1 + x_1 x_1' + x_2 x_2')^2 \\
& = 1 + x_1^2 x_1'^2 + x_2^2 x_2'^2 + 2x_1 x_1' + 2x_2 x_2' + 2x_1 x_1' x_2 x_2' \\
& = \langle (1, x_1^2, x_2^2, \sqrt{2}x_1, \sqrt{2}x_2, \sqrt{2}x_1 x_2), (1, x_1'^2, x_2'^2, \sqrt{2}x_1', \sqrt{2}x_2', \sqrt{2}x_1' x_2') \rangle
\end{aligned}
$$

$\rightarrow$ it is an **inner product** in an enlarged space.

$\rightarrow$ computing $(1 + x^T x')^2$ is significantly **faster** than computing the inner product in the enlarged space.

$\rightarrow$ we **do not need** to specify the transformation $\Phi(x)$ at all, but require only knowledge of the **kernel function**

Examples of kernels

Linear kernel:

$$K(x_i, x_{i'}) = \sum_{j=1}^{p} x_{ij} x_{i'j}$$

Polynomial kernel of degree *d*:

$$K(x_i, x_{i'}) = (1 + \sum_{j=1}^{p} x_{ij} x_{i'j})^d$$

where *d* is a positive polynomial integer.

Radial kernel:

$$K(x_i, x_{i'}) = \exp(-\gamma \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2)$$

where γ is a positive constant.

SVM and logistic regression