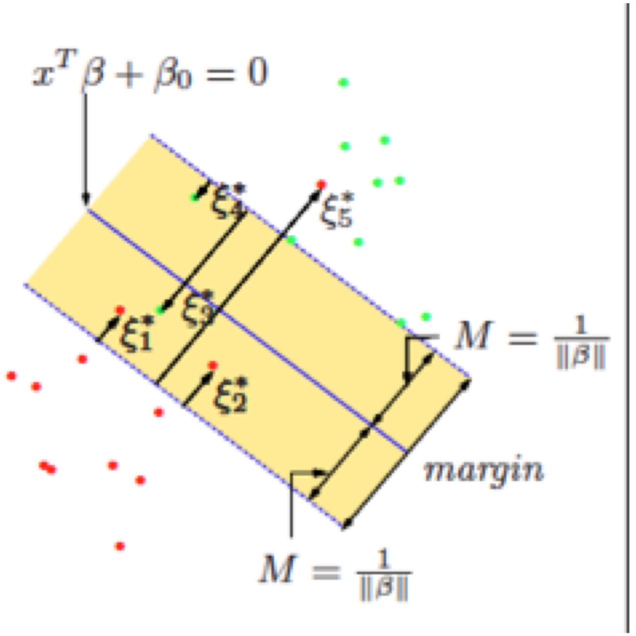
Separating Hyperplanes & Support Vector Machines



Statistical Data Mining I

Rachael Hageman Blair

Outline

- Revisit Chapter 4: Separating Hyperplanes
- Rosenblatt’s Perceptron Algorithm
- Maximal Margin Classifier
- Support Vector Classifier
- Support Vector Machines
- Conclusions

Motivation

- **Linear Methods we have discussed:**

  Linear regression, linear discriminant analysis, logistic regression, and separating hyperplanes.

- **In reality**:

  It is unlikely the true underlying function f(x) is linear in X.

  In regression problems, the relationships are likely nonlinear and non-additive.

- Linear assumptions are convenient and often provide a good approximation.

Separating Hyperplanes

- Construct linear decision boundaries that explicitly try to separate the data into classes as much as possible.
- Good separation is defined mathematically.
- Even when the training data can be perfectly separated by hyperplanes, LDA or other linear methods developed under a statistical framework may not achieve perfect separation.

Separating Hyperplanes

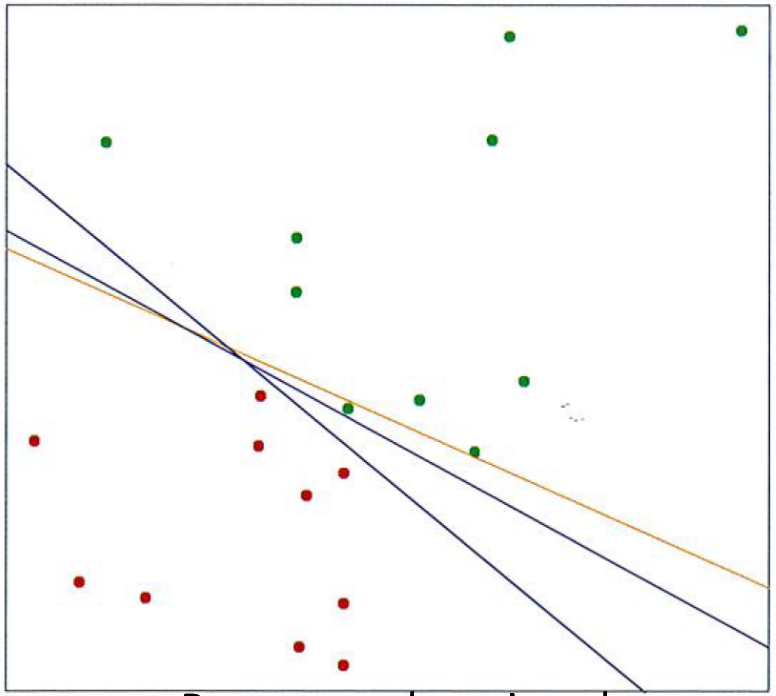


FIGURE 4.14. *A toy example with two classes separable by a hyperplane. The orange line is the least squares solution, which misclassifies one of the training points. Also shown are two blue separating hyperplanes found by the perceptron learning algorithm with different random starts.*

Linear Algebra Re-cap

- A hyperplane or an *affine set L* is defined by the linear equation: $L = \left\{ x : f(x) = \beta_0 + \beta^T x = 0 \right\}.$
- For any two points x_1 and x_2 lying in *L*, $\beta^T (x_1 - x_2) = 0$, *and* $\beta^* = \beta / \|\beta\|$ is a vector normal to the surface *of L.*
- For any point x_0 in *L*, $\beta^T x_0 = -\beta_0$.
- The signed distance of any point x to *L* is given by:

$$\beta^{*T} \left(x - x_0 \right) = \frac{1}{\|\beta\|} \left(\beta^T x + \beta_0 \right) = \frac{1}{\|f'(x)\|} f(x).$$

Hence $f(x)$ is proportional to the signed distance from x to the hyperplane defined by $f(x) = 0$.

Separating Hyperplanes

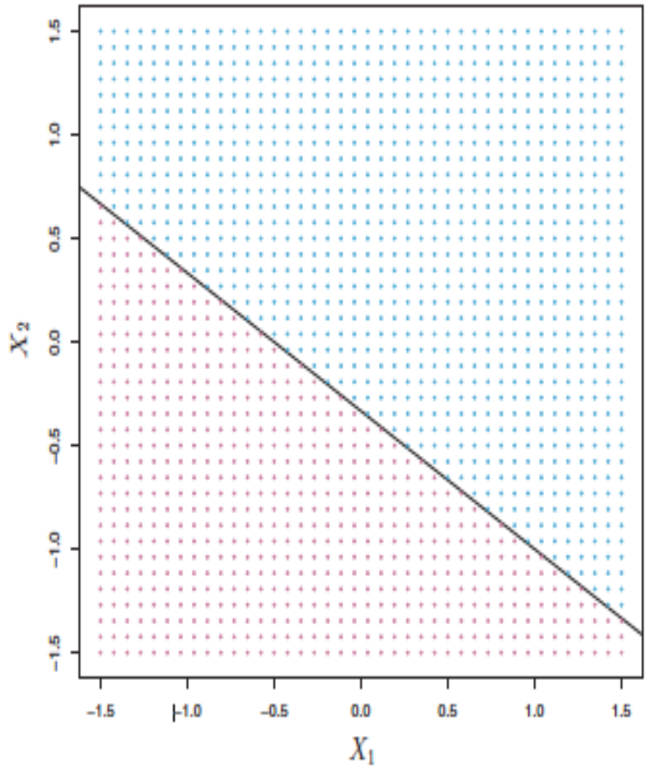


FIGURE 9.1. *The hyperplane* $1 + 2X_1 + 3X_2 = 0$ *is shown. The blue region is the set of points for which* $1 + 2X_1 + 3X_2 > 0$*, and the purple region is the set of points for which* $1 + 2X_1 + 3X_2 < 0$.

Rosenblatt's Perceptron Learning

- **Goal:** find a separating hyperplane by minimizing the distance of misclassified points to the decision boundary.
- Code the two classes by y_i=1 and y_i=-1.
- y_i is classified correctly if:

$$\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip} > 0 \text{ if } y_i = 1, \tag{9.6}$$

$$\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip} < 0 \text{ if } y_i = -1. \tag{9.7}$$

Equivalently, the separating hyperplane has the property:

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

Rosenblatt's Perceptron Learning

- Since the signed distance from x_i to the decision boundary is given as $\frac{\beta^T x_i + \beta_0}{\|\beta\|}$, thus the distance from a misclassified x_i to the decision boundary is:

$$\frac{y_i\left(\beta^T x_i + \beta_0\right)}{\|\beta\|}.$$

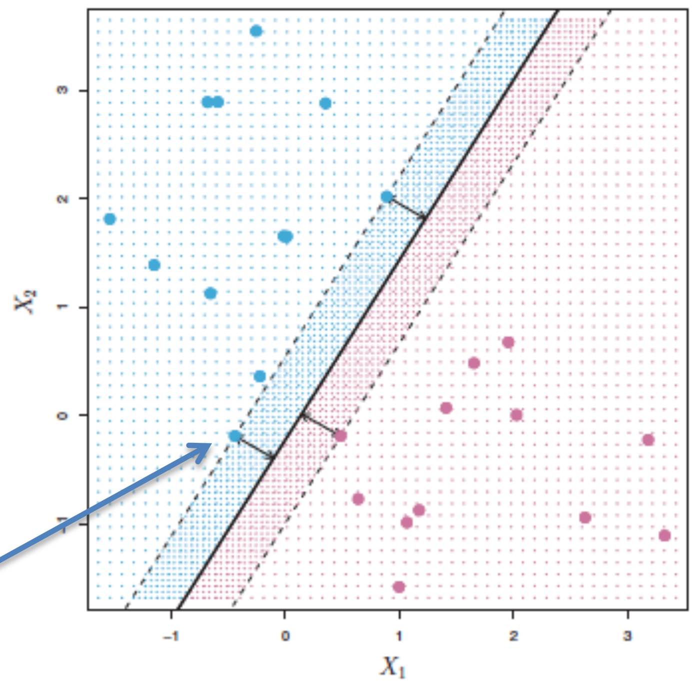
If this is BIG, then we are far away from the Decision boundary…. We are "well classified", Or "well misclassified.

If this is small, then less certain of the classification

Rosenblatt's Perceptron Learning

- The classification of an observation, x^*, is based on:

$$f(x^*) = sign(\beta_0 + \beta_1 x_1^* + \beta_2 x_2^* + \cdots + \beta_p x_p^*).$$



Support vectors

Rosenblatt's Perceptron Learning

The constrained optimization problem:

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

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

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

Optimal Separating Hyperplanes

- The constraints define an empty "slab" or "cushion" (margin) around the linear decision boundary of thickness

$$1/\|\beta\|.$$

- The problem is to find the parameters β_0 and β that maximize the thickness of the "slab".

(See Chapter 4 In Elements of Statistical Learning for solution to the optimization).

Optimal Separating Hyperplanes

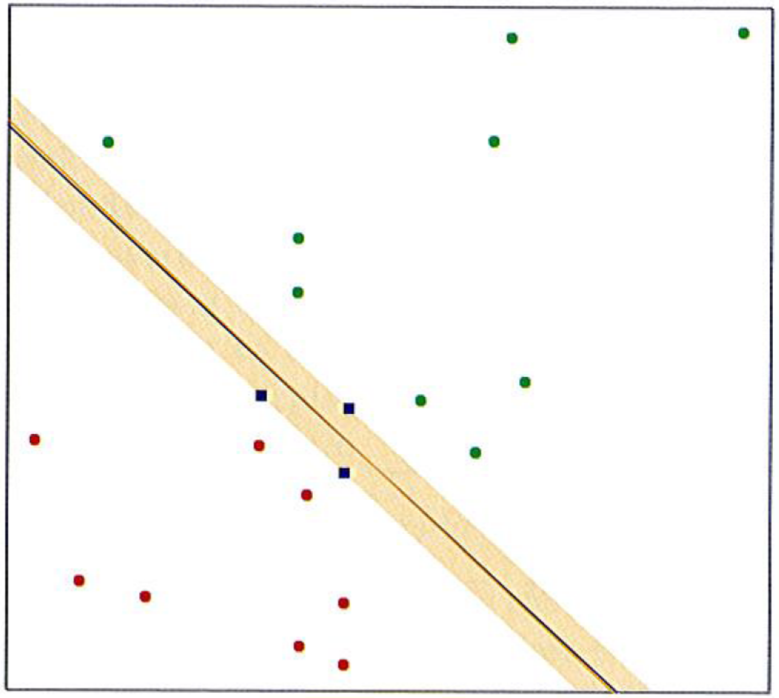


FIGURE 4.16. *The same data as in Figure 4.14. The shaded region delineates the maximum margin separating the two classes. There are three support points indicated, which lie on the boundary of the margin, and the optimal separating hyperplane (blue line) bisects the slab. Included in the figure is the boundary found using logistic regression (red line), which is very close to the optimal separating hyperplane (see Section 12.3.3).*

Optimal Separating Hyperplanes

- The optimal separating function produces a function $\hat{f}(x) = x^T \hat{\beta} + \hat{\beta}_0$ for classifying new observations:

$$\hat{G}(x) = \text{sign}\hat{f}(x).$$

- **Note:** none of the training points will fall into the margin by construction. This is not the case for the test data.

  The larger the margin in the training data, the more likely there will be better separation in the test data.

Support Vector Classifier

Note: The maximal margin classifier exists if and only if there is a separating hyperplane.

Motivation: the non-separable case.

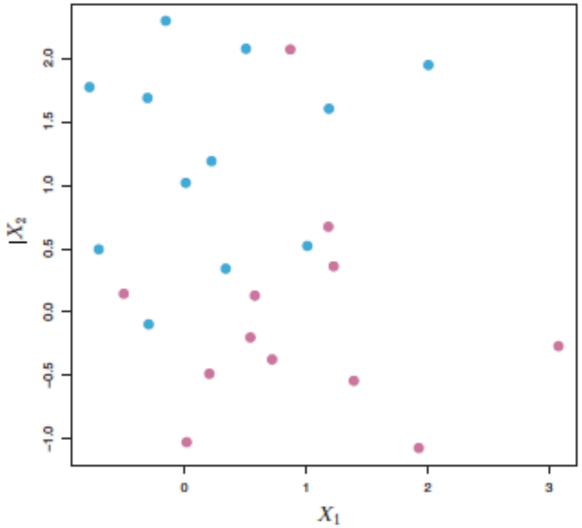
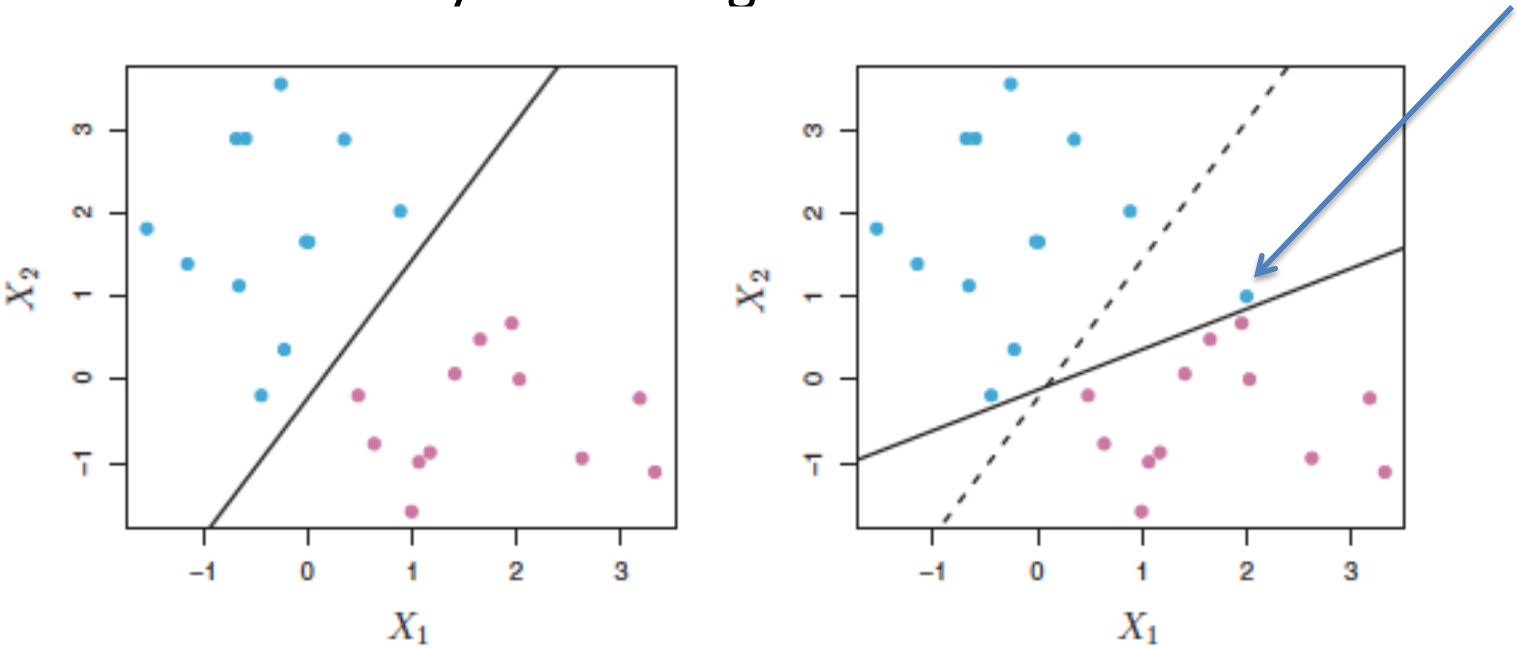


FIGURE 9.4. *There are two classes of observations, shown in blue and in purple. In this case, the two classes are not separable by a hyperplane, and so the maximal margin classifier cannot be used.*

Support Vector Classifier

Motivation: sensitivity to training data.

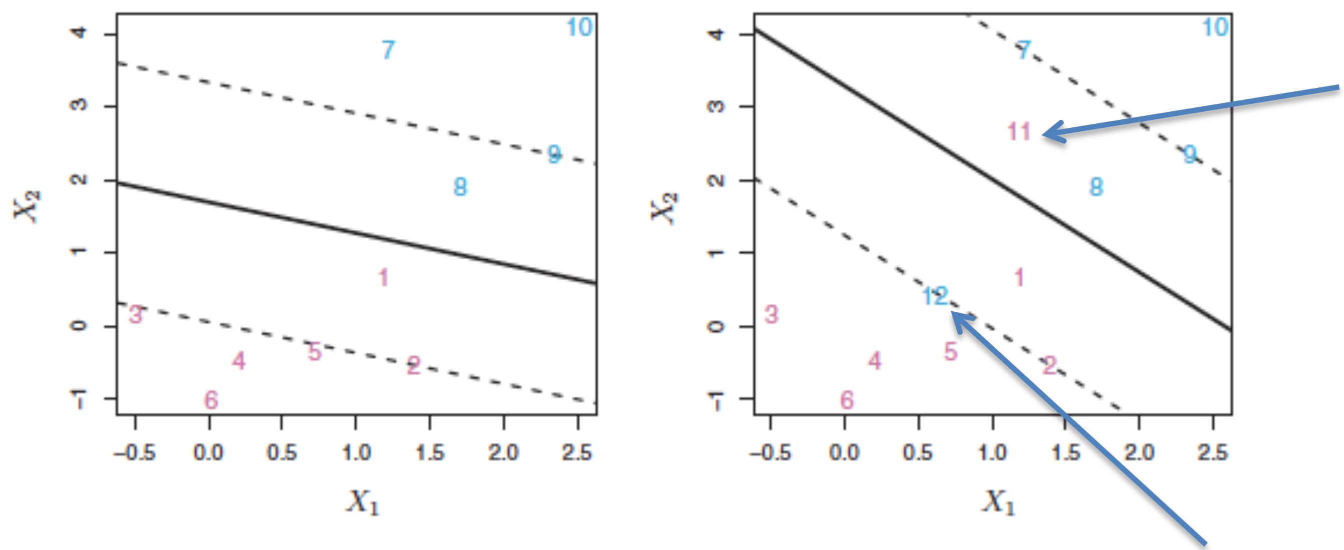
New point



Support Vector Classifier

- Relax out need to have every observation on the correct side of the hyperplane and the correct side of the margin.

- Create a “soft margin”, which allows points to be on the wrong side of the margin, and even the hyperplane.



Support Vector Classifier

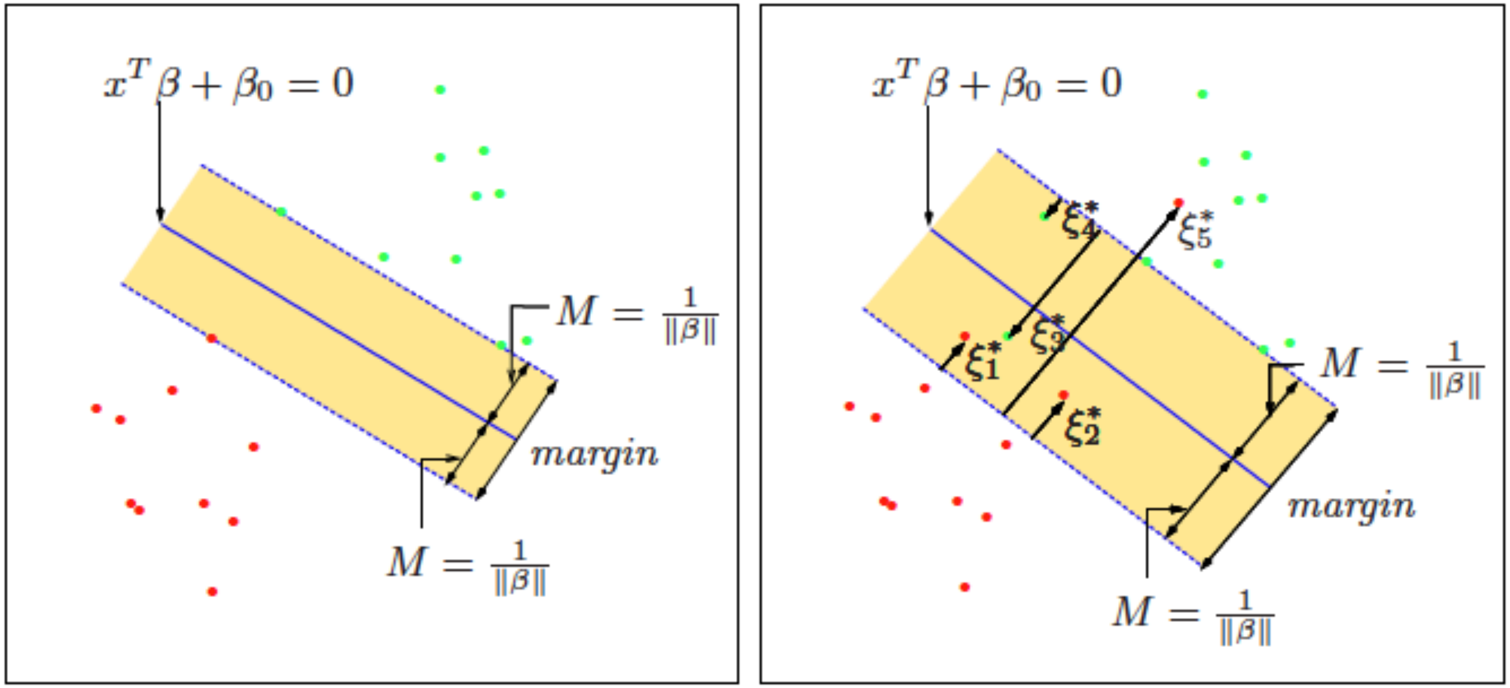


FIGURE 12.1. *Support vector classifiers. The left panel shows the separable case. The decision boundary is the solid line, while broken lines bound the shaded maximal margin of width $2M = 2/\|\beta\|$. The right panel shows the nonseparable (overlap) case. The points labeled ξ_j^* are on the wrong side of their margin by an amount $\xi_j^* = M\xi_j$; points on the correct side have $\xi_j^* = 0$. The margin is maximized subject to a total budget $\sum \xi_i \leq$ constant. Hence $\sum \xi_j^*$ is the total distance of points on the wrong side of their margin.*

Support Vector Classifier

The optimization problem:

Margin

$$\max_{\beta,\varepsilon} M,$$

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

Slack Variables

$$y_i\left(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip}\right) \geq M\left(1 - \varepsilon_i\right),$$

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

Where C is a non-negative tuning parameter.

Support Vector Classifier

$$\max_{\beta,\varepsilon} M,$$

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

$$y_i\left(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip}\right) \geq M\left(1 - \varepsilon_i\right),$$

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

- Slack Variables: tells us where the i^{th} observation is located relative to the hyperplane and margin.

  $\varepsilon_i = 0$ implies the i^{th} observation is on the correct side of the margin.

  $\varepsilon_i > 1$ implies that the i^{th} observation is on the wrong side of the hyperplane.

Support Vector Classifier

$$\max_{\beta,\varepsilon} M,$$

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

$$y_i\left(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip}\right) \geq M\left(1 - \varepsilon_i\right),$$

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

C can be thought of as a "budget" for The amount that the margin can be violated by n observations.

- Slack Variables: tells us where the i^th observation is located relative to the hyperplane and margin.

  $\varepsilon_i = 0$ implies the i^th observation is on the correct side of the margin.

  $\varepsilon_i > 1$ implies that the i^th observation is on the wrong side of the hyperplane.

Support Vector Classifier

- Importantly, only observations that lie directly on the margin or that violate the margin will affect the hyperplane, and hence the classifier obtained.

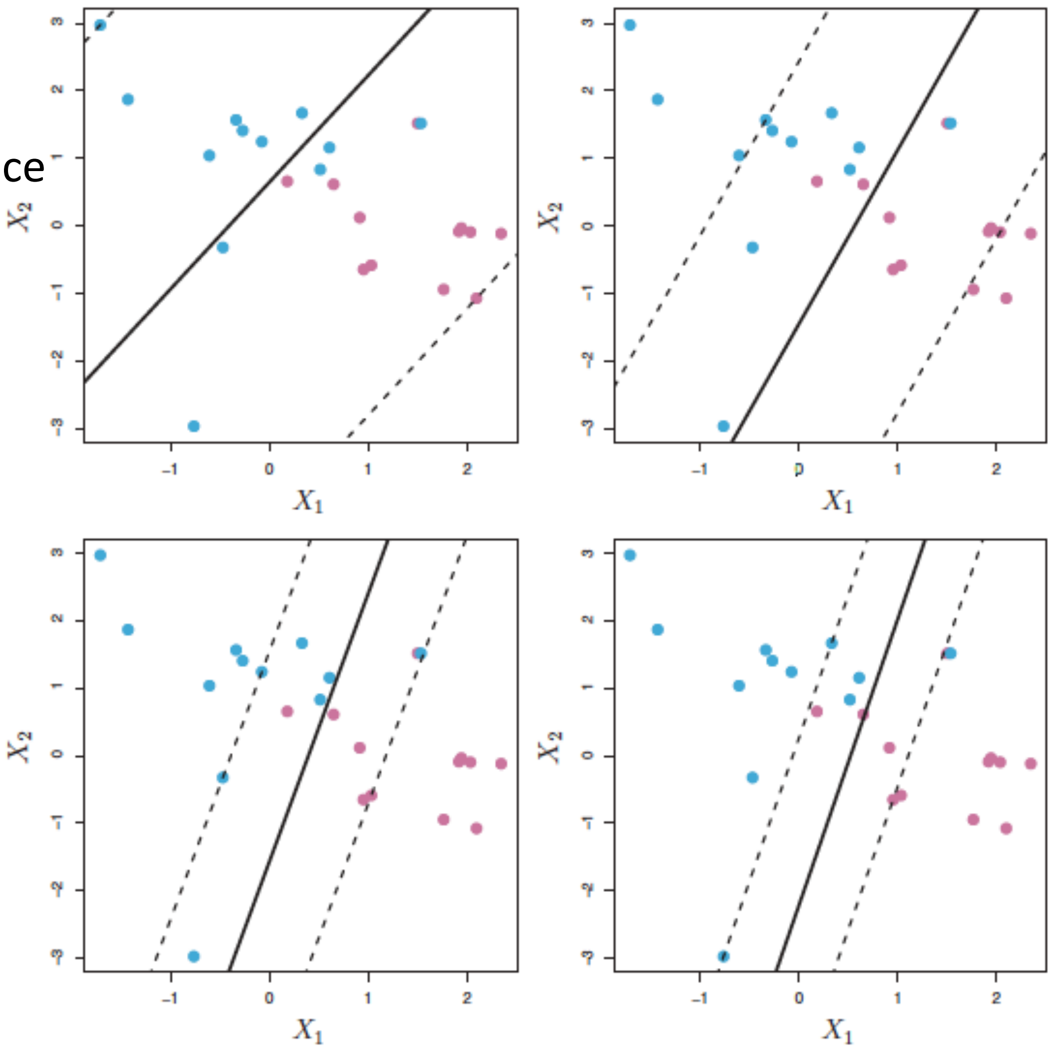
- In other words… points that classified well **do not matter** they do not participate in the classification of new points.

- **Support Vectors**: observations that lie on the margin, or on the wrong side of the margin (or hyperplane) for their class.

Support Vector Classifier

Large C
Low bias
High variance

Smaller C



Even
Smaller C

Smallest C
High bias
Low variance

Support Vector Classifier

Our training data consists of *N* pairs $(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)$,
With $x_i \in \Re^p$ and $y_i \in \{-1, 1\}$.
Define a hyperplane by:

$$\{x : f(x) = x^T\beta + \beta_0 = 0\},$$

where β is a unit vector: $\|\beta\| = 1$. A classification rule induced by $f(x)$ is:

$$G(x) = sign\left[x^T\beta + \beta_0\right].$$

Support Vector Machines

- As usual, sometimes linear decision boundaries do not capture the model space well.
- Support Vector Machines (SVMs) aim to enlarge the feature space (recall QDA).
- Non-linear decision boundaries are achieved via. Basis expansions.

Recall: an enlarged feature space:

$$X_1, X_2, \ldots, X_p \longrightarrow X_1, X_2, \ldots, X_p, X_1^2, X_1^2, X_2^2, \ldots, X_p^2$$

the optimization problem naturally extends.

Support Vector Machines

The "extended" optimization problem:

$$\underset{\beta_0,\beta_{11},\beta_{12}\ldots,\beta_{p1},\beta_{p2},\epsilon_1,\ldots,\epsilon_n}{\text{maximize}} \quad M$$

$$\text{subject to } y_i\left(\beta_0 + \sum_{j=1}^{p}\beta_{j1}x_{ij} + \sum_{j=1}^{p}\beta_{j2}x_{ij}^2\right) \geq M(1-\epsilon_i),$$

$$\sum_{i=1}^{n}\epsilon_i \leq C,\;\; \epsilon_i \geq 0,\;\; \sum_{j=1}^{p}\sum_{k=1}^{2}\beta_{jk}^2 = 1.$$

- The main idea: we want to enlarge the feature space to accommodate a non-linear boundary.

  Kernel approach

Support Vector Machines

Optimization: extremely technical (see 12.2.1, ESL).

Main ideas:

- Solution involves the inner products of the observations as opposed to the observations themselves.

  The inner product of $x_i, x_{i'}$ is given by: $\langle x_i, x_{i'} \rangle = \sum_{j=1}^{p} x_{ij} x_{i'j}.$

- The linear support vector classifier can be represented as:

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

  where there are n parameters $\alpha_i, \;\; i = 1, \ldots, n.$

The inner product
Between a new point
And each of the
training points

Support Vector Machines

- The linear support vector classifier can be represented as:

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

where there are n parameters $\alpha_i, \ \ i = 1, \ldots, n.$

The inner product
Between a new point
And each of the
training points

$\alpha_i \neq 0$
Only for support vectors

Support Vector Machines

- The linear support vector classifier can be represented as:

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

where there are n parameters $\alpha_i, \quad i = 1, \ldots, n.$

The inner product
Between a new point
And each of the
training points

$\alpha_i \neq 0$
Only for support vectors

- Estimation of parameters, $f(x) = \beta_0, \alpha_1, \ldots, \alpha_n$ requires C(n,2) inner products between all pairs of training observations.

Support Vector Machines

- The solution can be simplified:

$$f(x) = \beta_0 + \sum_{i \in S} \alpha_i \langle x, x_i \rangle.$$

Kernel: a generalization of the inner product: $K(x_i, x_{i'})$.
The kernel conveys the similarity of two observations.

Simplest Example: inner product (aka linear kernel)

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

yields the support vector classifier.

Support Vector Machines

- Polynomial kernel of degree d”

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

More flexible decision boundary, equivalently, fitting a Support vector classifier in a higher dimensional space.

- Resulting function: $f(x) = \beta_0 + \sum_{i \in S} \alpha_i K\left(x, x_i\right).$

Support Vector Classifier + Non-linear Kernel → Support Vector Machine

Support Vector Machines

- Polynomial kernel of degree d"

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

Consider for example a feature space with two inputs X_1 and X_2, and a polynomial kernel of degree 2. Then

$$\begin{aligned} K(X, X') &= (1 + \langle X, X' \rangle)^2 \\ &= (1 + X_1 X_1' + X_2 X_2')^2 \\ &= 1 + 2X_1X_1' + 2X_2X_2' + (X_1X_1')^2 + (X_2X_2')^2 + 2X_1X_1'X_2X_2'. \end{aligned} \tag{12.23}$$

This blows up to high dimensions fast....

imagine big p, and/or big d --- > overfitting

Support Vector Machines

- Radial Kernel:

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

By design, far away training observations play a weak role in the classification. Considered a very "local method".

Similar in spirit to exponential loss.

Support Vector Machines

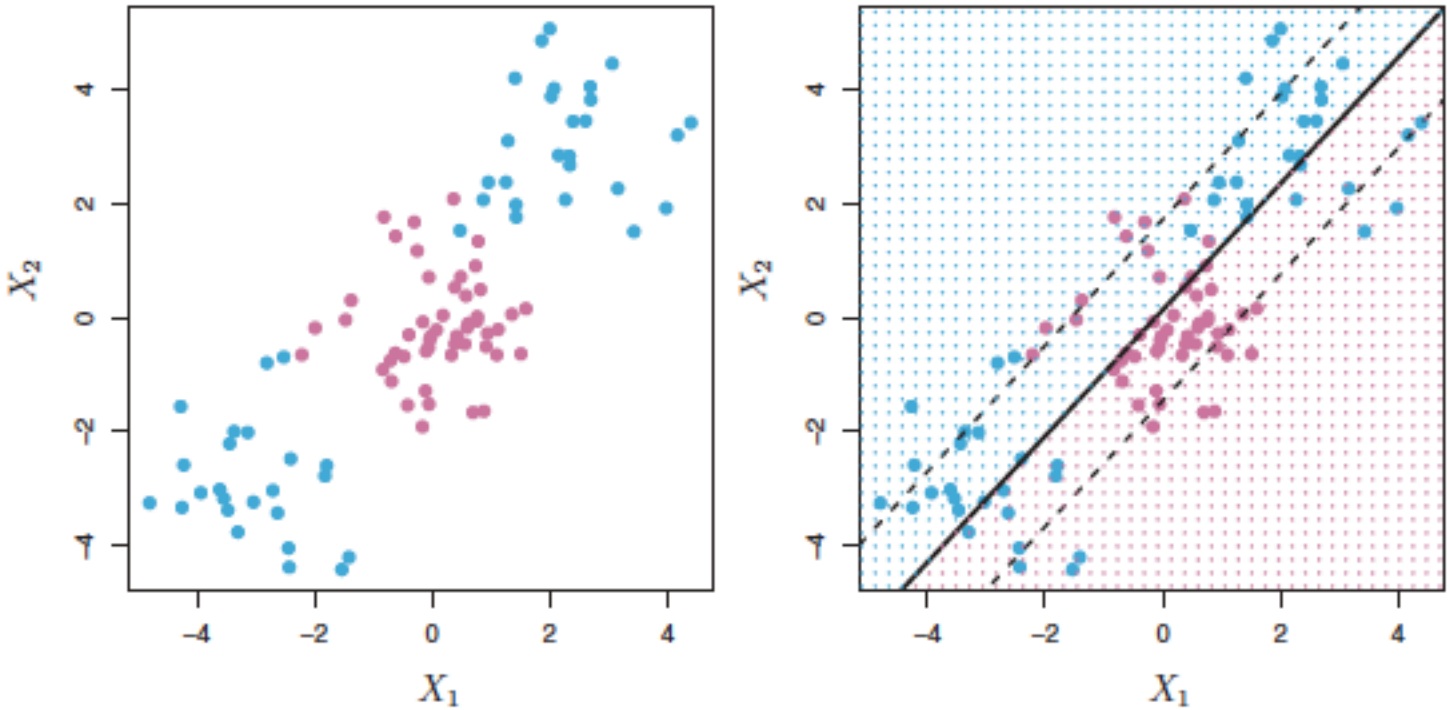


FIGURE 9.8. Left: *The observations fall into two classes, with a non-linear boundary between them.* Right: *The support vector classifier seeks a linear boundary, and consequently performs very poorly.*

Support Vector Machines

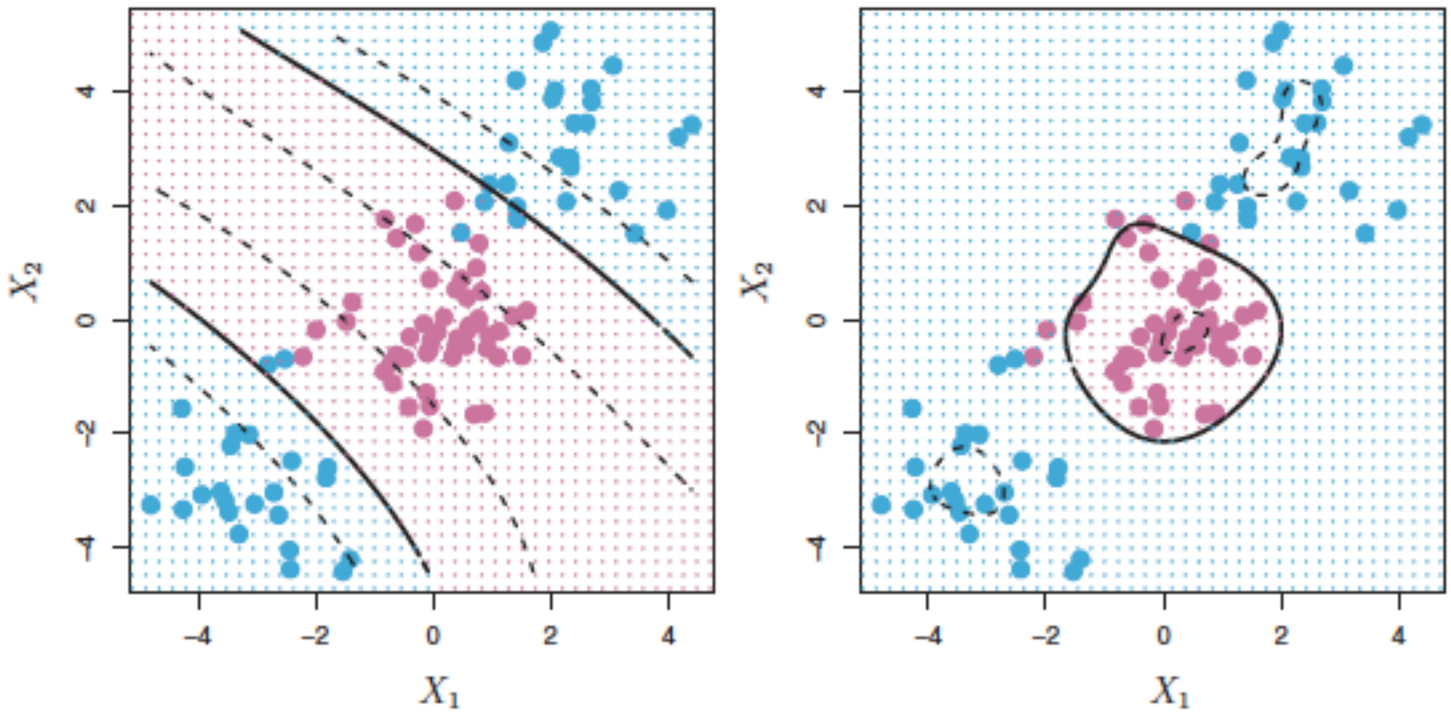
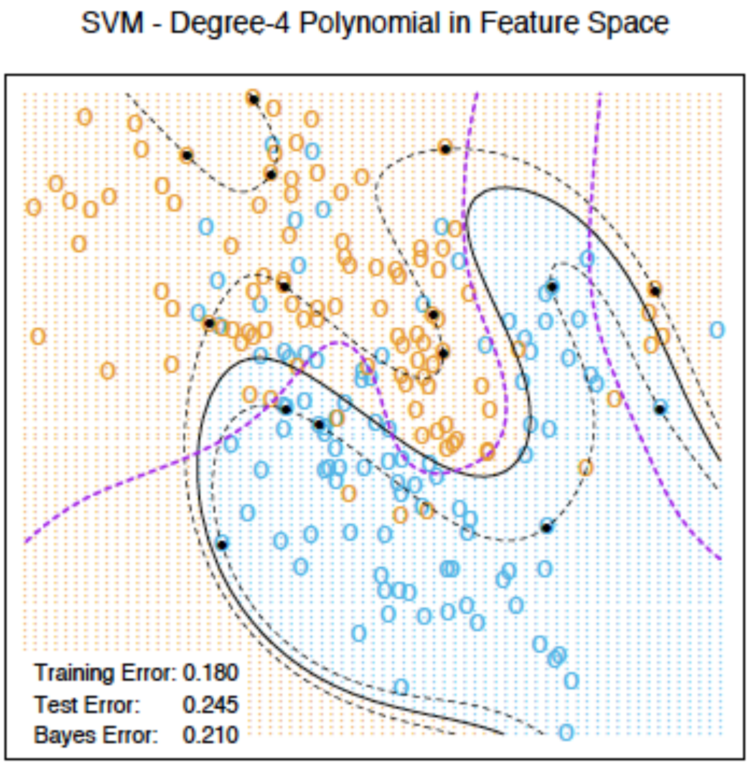
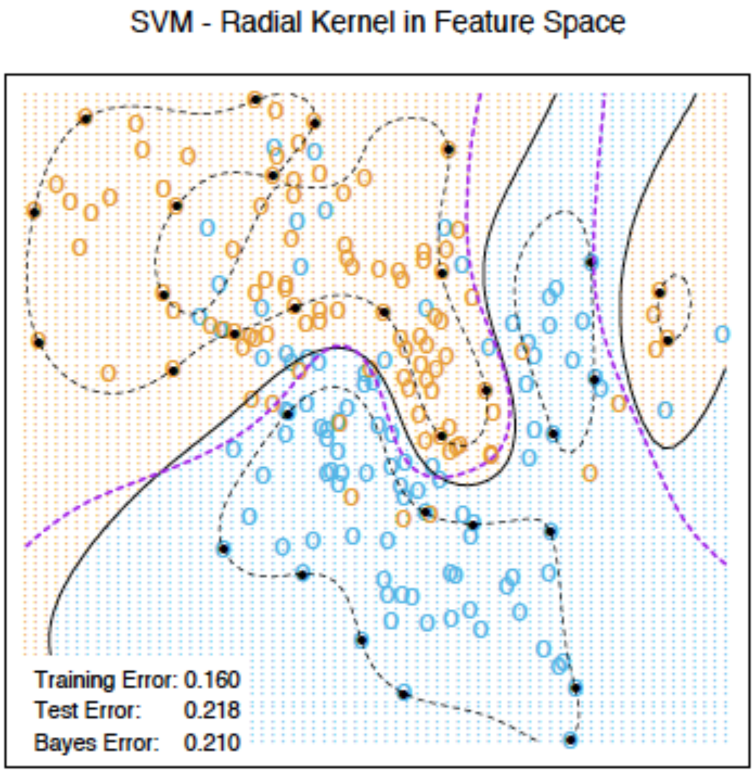


FIGURE 9.9. Left: *An SVM with a polynomial kernel of degree 3 is applied to the non-linear data from Figure 9.8, resulting in a far more appropriate decision rule.* Right: *An SVM with a radial kernel is applied. In this example, either kernel is capable of capturing the decision boundary.*

Support Vector Machines

- Mixture of Gaussians:





Example

- Heart data: 13 predictors
- Aim: Predict heart disease.
- Data: split into test and training

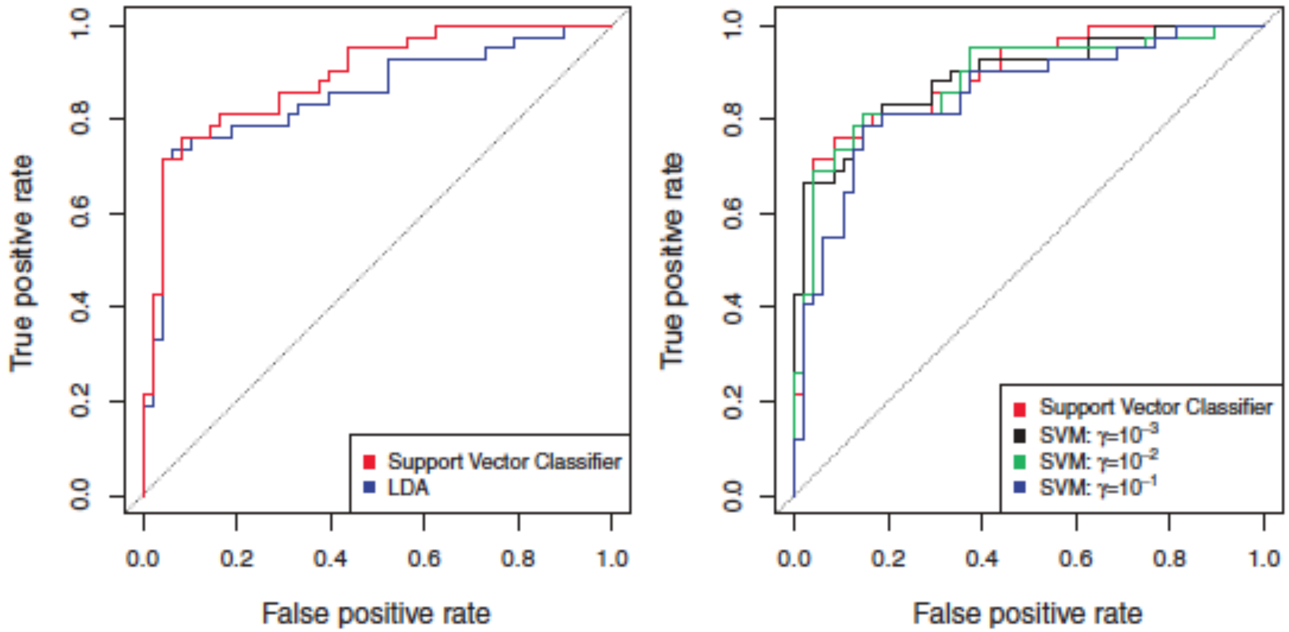


FIGURE 9.11. *ROC curves for the test set of the* Heart *data.* Left: *The support vector classifier and LDA are compared.* Right: *The support vector classifier is compared to an SVM using a radial basis kernel with* $\gamma = 10^{-3}$, 10^{-2}, *and* 10^{-1}.

Conclusions

- Ability to deal with multiple classes is viewed as a limitation. Work arounds: one-vs.-one, and one-vs.-all.
- Extensions include support vector regression.
- Other methods can be used in connection with non-linear kernels, SVMs is the most popular for this purpose.
- As with other methods, the choice of tuning parameter, C, is critical.
- When working with non-linear kernels, additional parameters controlling the complexity of the kernel have to be set as well.