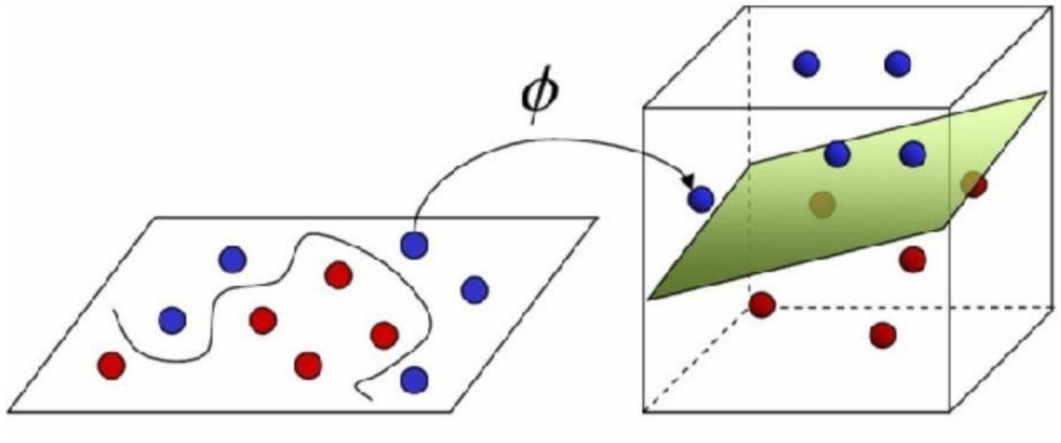
University of Ljubljana, Faculty of Computer and Information Science

Kernel methods



Prof Dr Marko Robnik-Šikonja

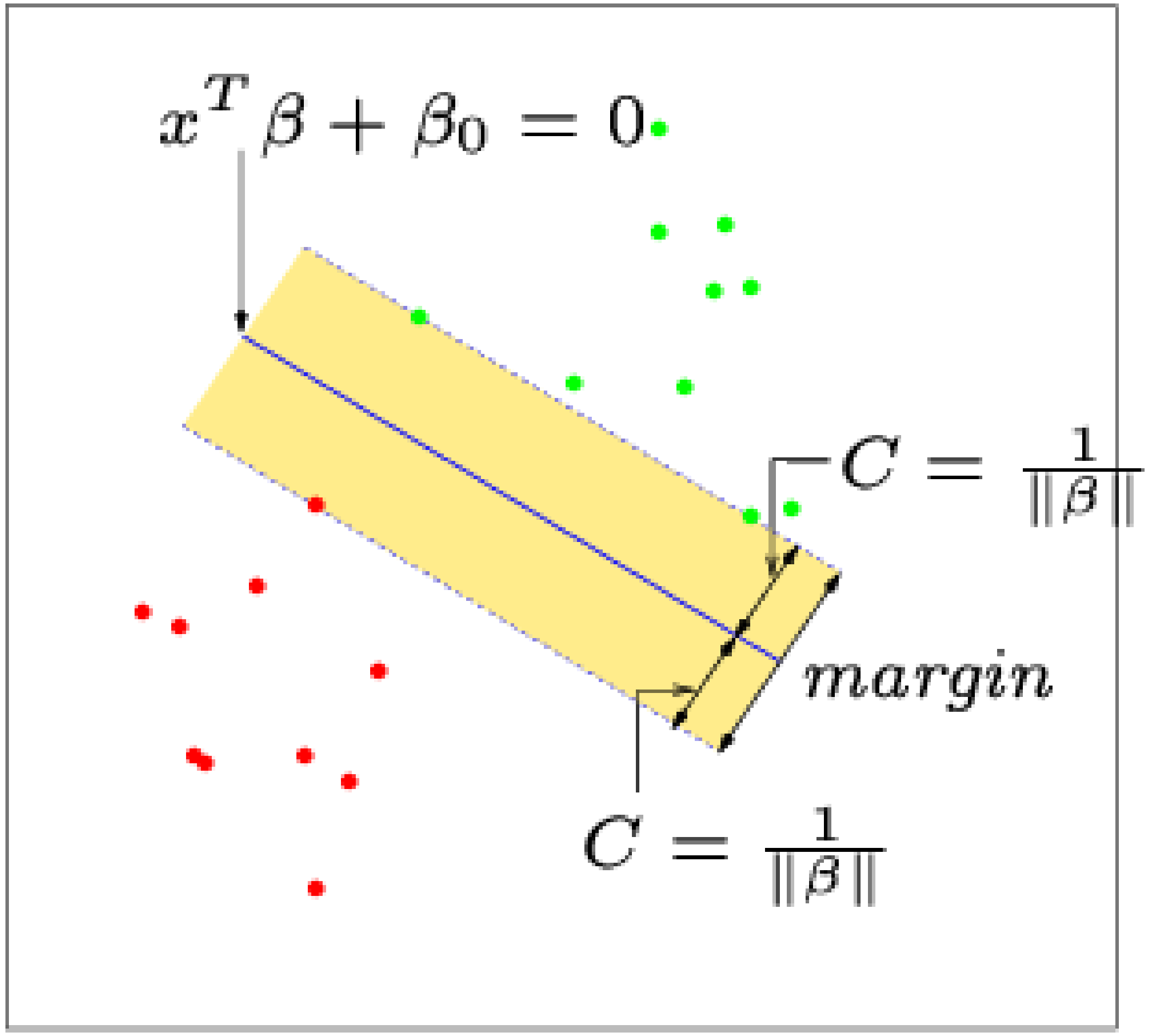
Intelligent Systems, November 2022

Support vector machines

- Imagine a situation where you have a two class classification problem with two predictors X_1 and X_2.
- Suppose that the two classes are “linearly separable” i.e. one can draw a straight line in which all points on one side belong to the first class and points on the other side to the second class.
- Then a natural approach is to find the straight line that gives the biggest separation between the classes, i.e. the points are as far from the line as possible
- This is the basic idea of support vector classifiers.

An illustration

- *C* is the minimum perpendicular distance between each point and the separating line.
- We find the line which maximizes *C*.
- This line is called the “optimal separating hyperplane”
- The classification of a point depends on which side of the line it falls on.



More than two dimensions

- This idea works just as well with more than two predictor variables.
- For example, with three predictors you want to find the plane that produces the largest separation between the classes.
- With more than three dimensions it becomes hard to visualize a plane but it still exists. In general they are caller hyper-planes.

Non-separating classes

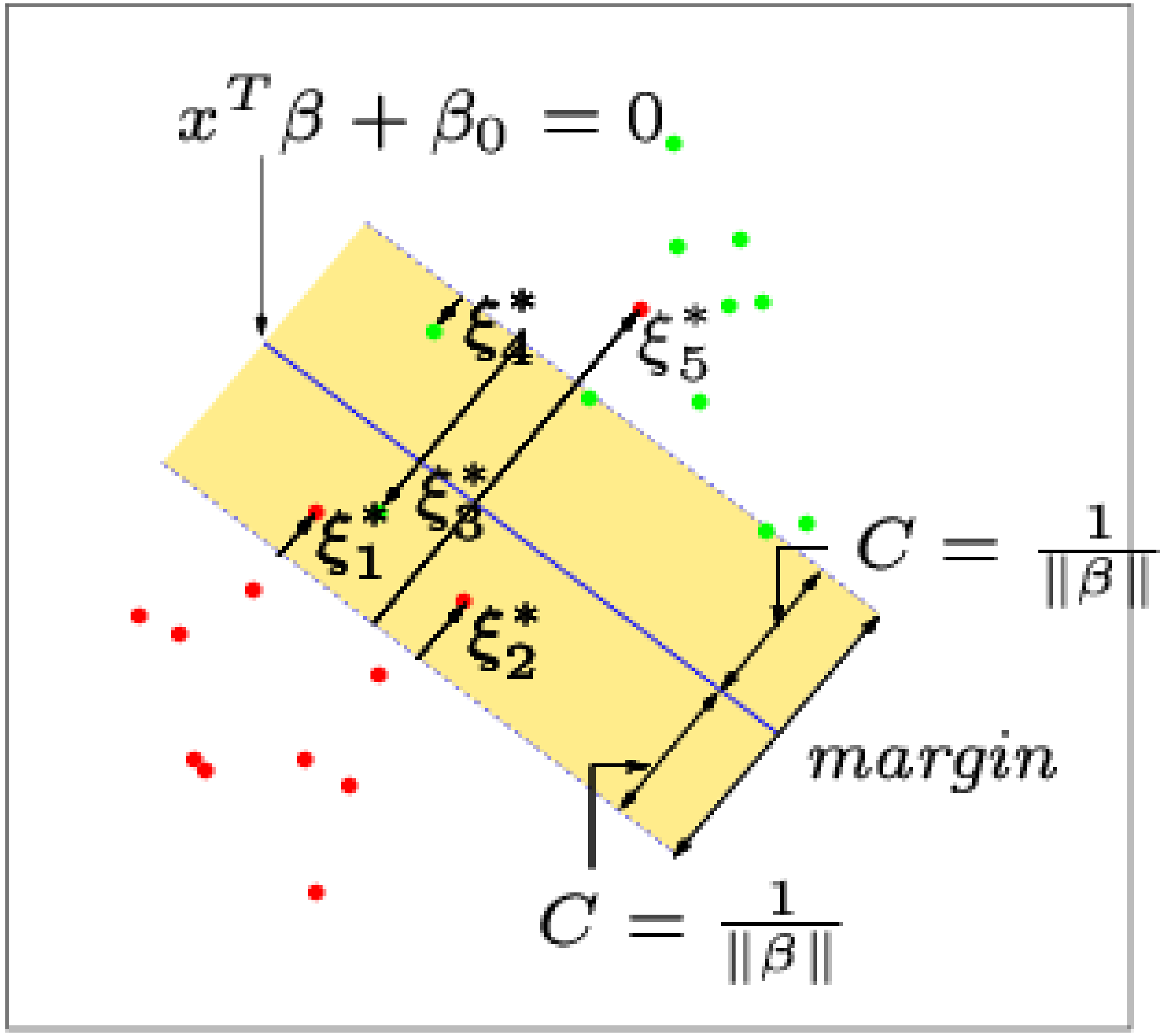
- In practice, it is not usually possible to find a hyper-plane that perfectly separates two classes.
- In other words, for any straight line or plane that we draw, there will always be at least some points on the wrong side of the line.
- In this situation, we try to find the plane that gives the best separation between the points that are correctly classified, subject to the points on the wrong side of the line not being off by too much.
- It is easier to see with a picture!

Non-separating example

- Let ξ^*_i represent the amount that the *i*-th point is on the wrong side of the margin (the dashed line).
- Then we want to maximize *C*, subject to

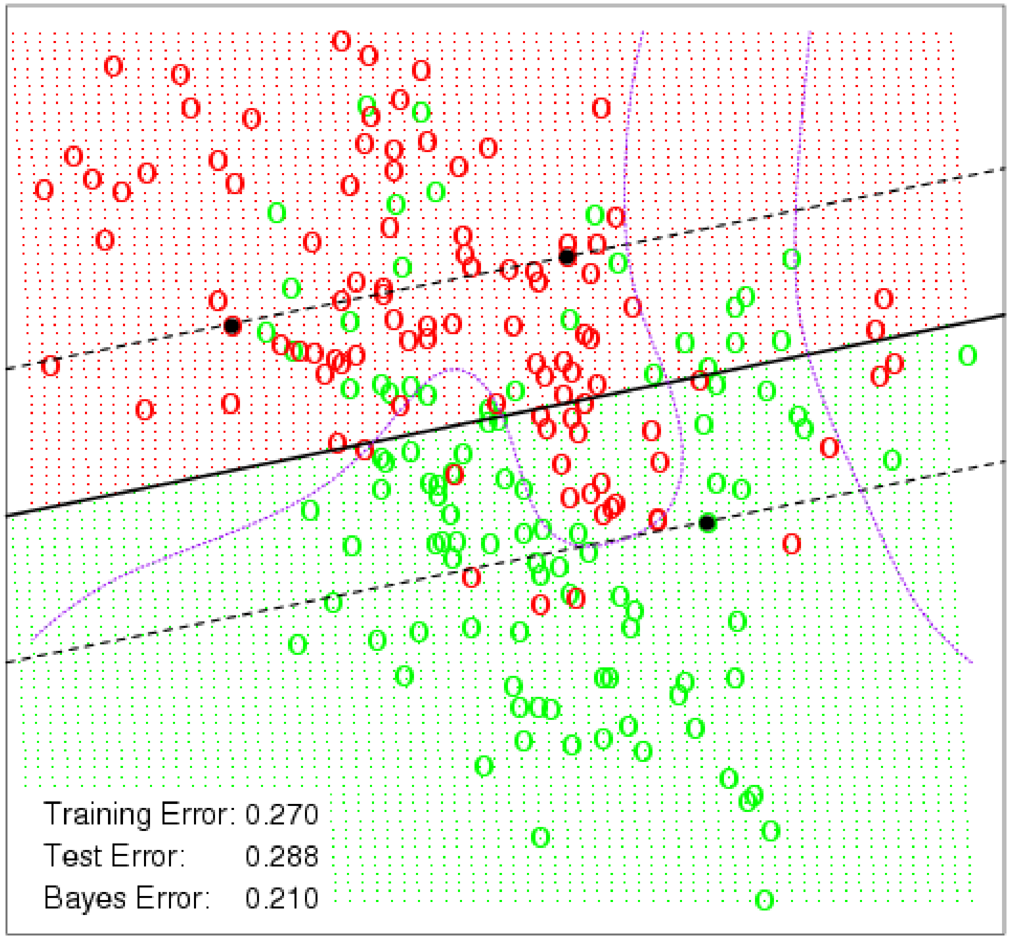
$$\frac{1}{C}\sum_{i=1}^{n} \xi_i^* \leq \text{Constant}$$

- The constant is a tuning parameter that we choose.



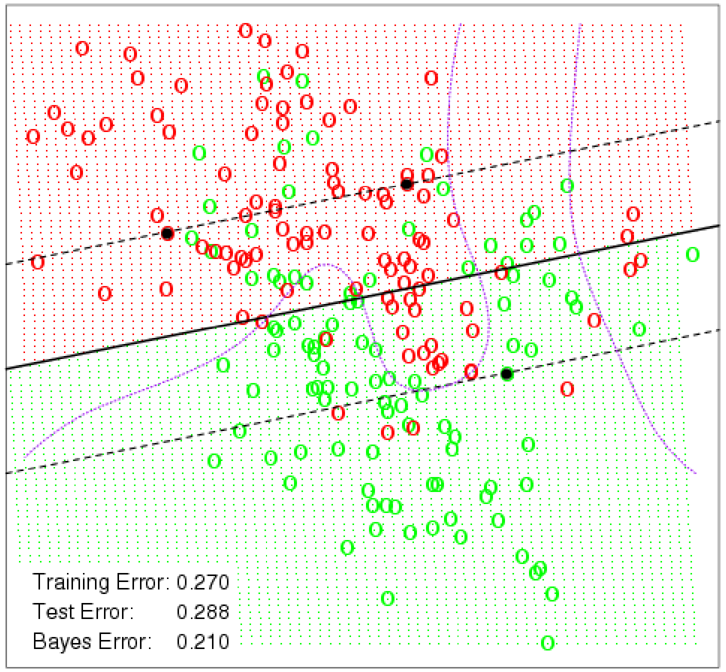
A simulation example with a small constant

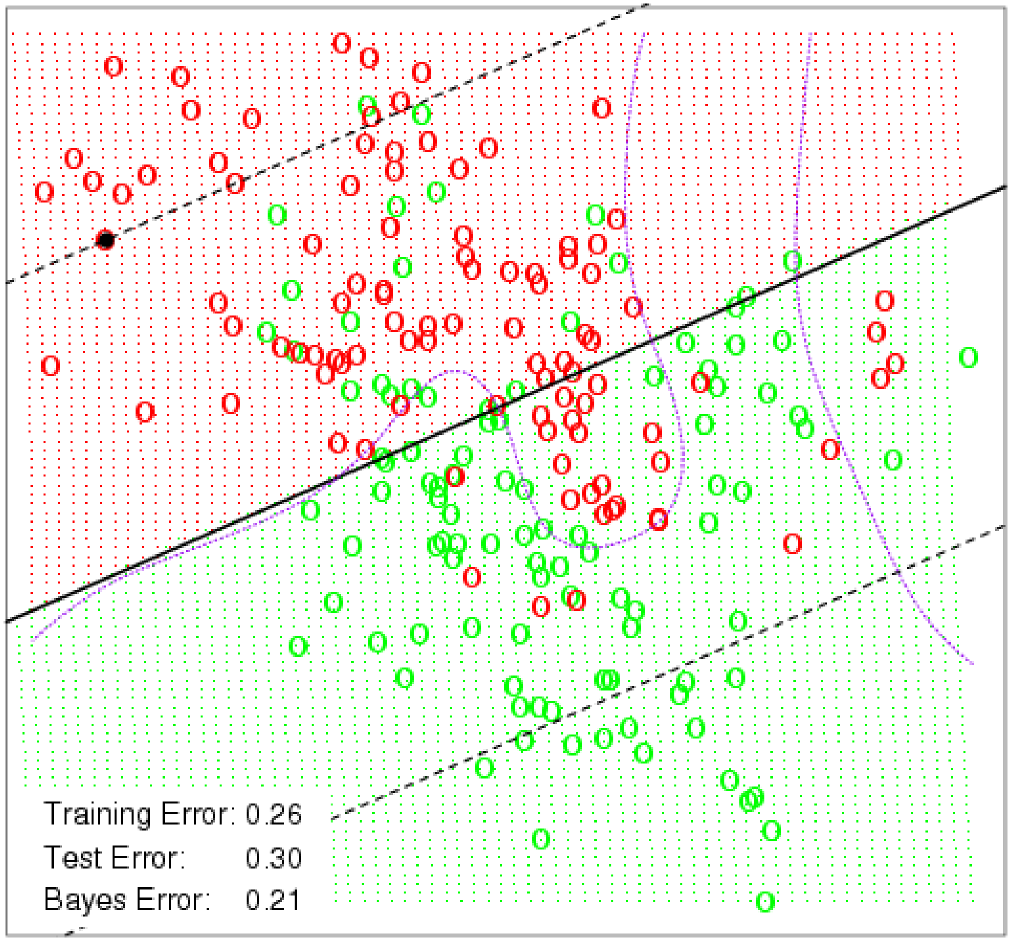
- The distance between the dashed lines represents the margin or 2C.
- The purple lines represent the Bayes decision boundaries



The same example with a larger constant

- Using a larger constant allows for a greater margin and creates a slightly different classifier.
- Notice, however, that the decision boundary must always be linear.





Non-linear support vector classifier

- The support vector classifier is fairly easy to think about. However, because it only allows for a linear decision boundary, it may not be all that powerful.
- Recall that linear regression is extended to non-linear regression using a basis function i.e.

$$Y_i = \beta_0 + \beta_1 b_1(X_i) + \beta_2 b_2(X_i) + \cdots + \beta_p b_p(X_i) + \varepsilon_i$$

A basis approach

- Conceptually, we can take a similar approach with the support vector classifier.
- The support vector classifier finds the optimal hyper-plane in the space spanned by $X_1, X_2, \ldots, X_p$.
- Instead, we can create transformations (or a basis) $b_1(x), b_2(x), \ldots, b_M(x)$ and find the optimal hyper-plane in the space spanned by $b_1(\boldsymbol{X}), b_2(\boldsymbol{X}), \ldots, b_M(\boldsymbol{X})$.
- This approach produces a linear plane in the transformed space but a non-linear decision boundary in the original space.
- This is called the support vector machine classifier.

Basis example

- Suppose we use polynomials as bases
$$X_1, X_2, X_1^2, X_2^2, X_1X_2, X_1^3, X_1X_2^2, \ldots$$
- We go from p dimensional space to *M>p* dimensional space and fit the SVM classifier in the enlarged space
- For the bases $(X_1, X_2, X_1^2, X_2^2, X_1X_2)$ this gives a non-linear classifier in the original space
$$\beta_1X_1 + \beta_2X_2 + \beta_3X_1^2 + \beta_4X_2^2 + \beta_5X_1X_2 = 0$$

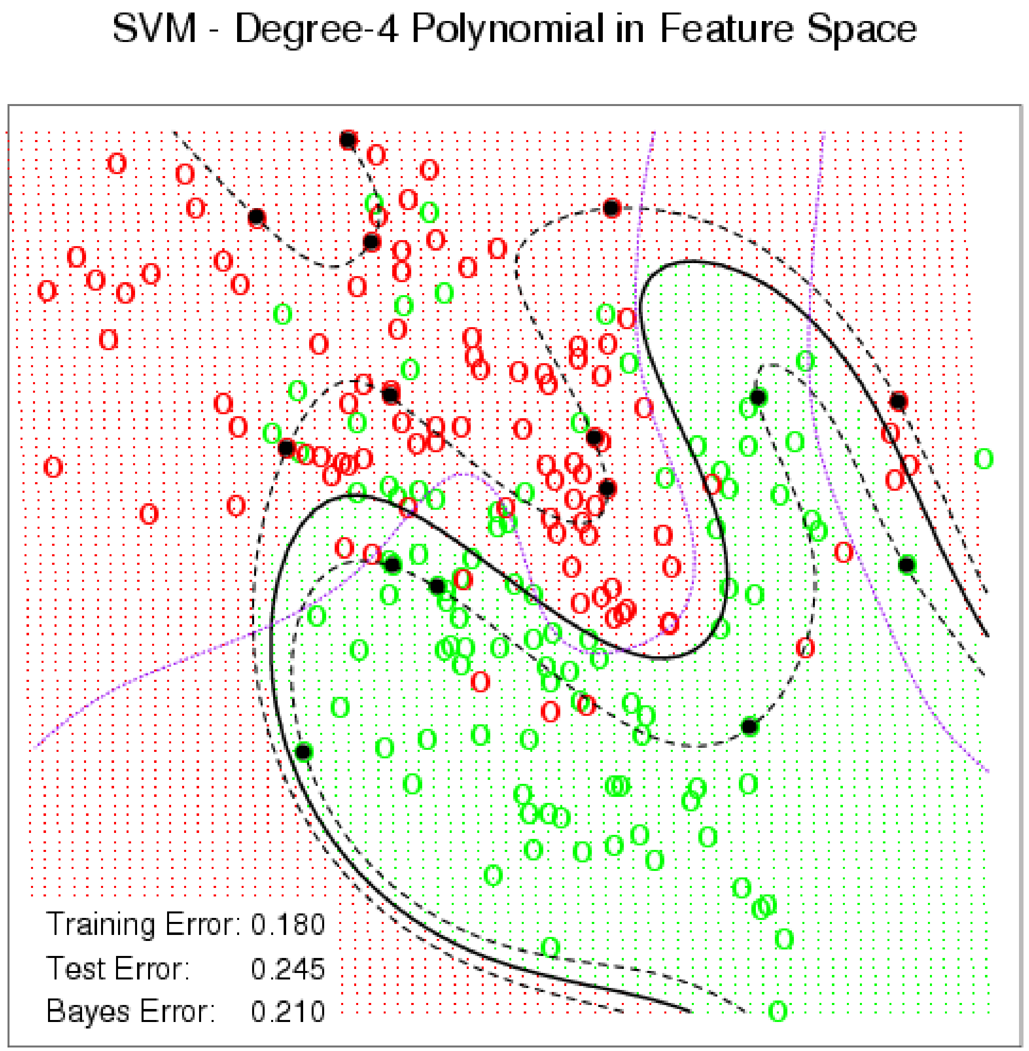
In reality

- While conceptually the basis approach is how the support vector machine works, there are some technical details which means that we don't actually choose $b_1(x), b_2(x), \ldots, b_M(x)$.
- Instead we choose a kernel function which takes the place of the basis.
- Kernel operates on inner products between instances
- Common kernel functions include
  - Linear
  - Polynomial
  - Radial Basis
  - Sigmoid

Polynomial kernel on Sim data

- Using a polynomial kernel, we now allow SVM to produce a non-linear decision boundary.
- Notice that the test error rate is a lot lower.

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

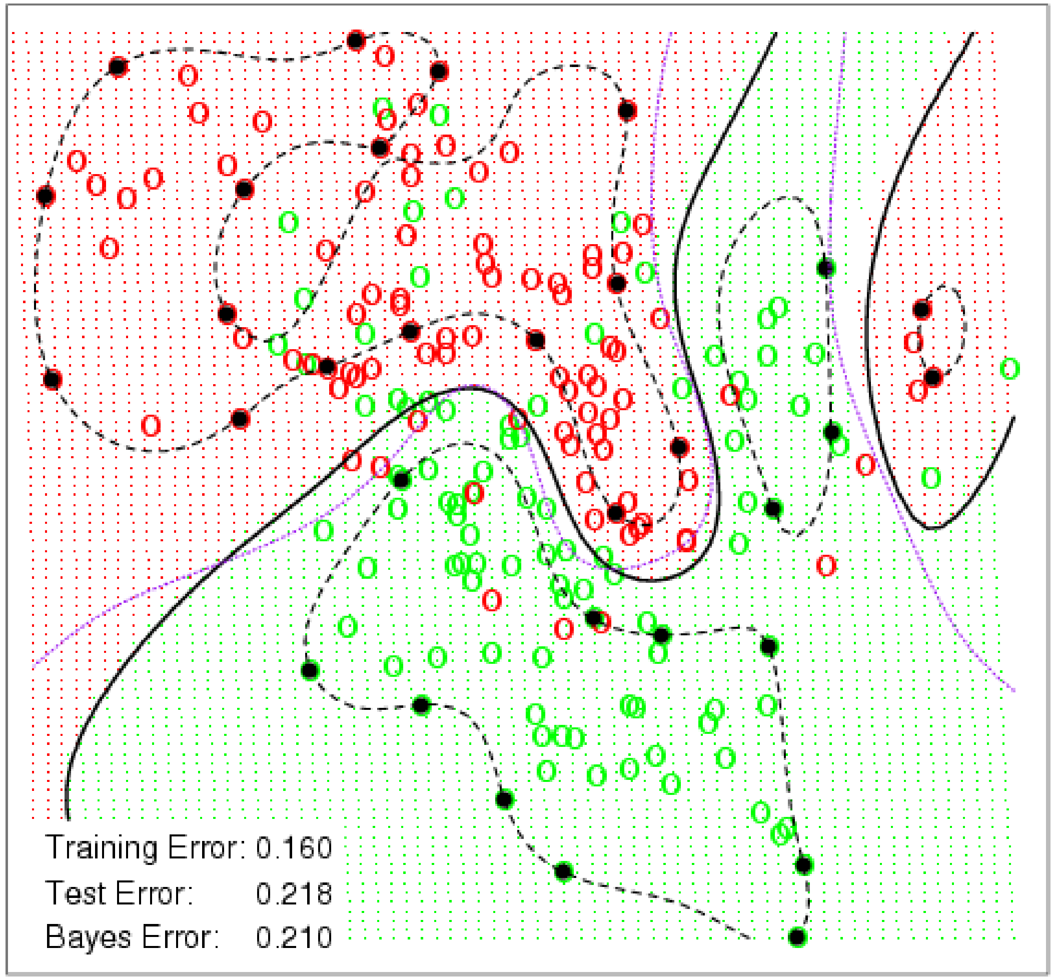


Radial basis kernel

- Using a radial basis kernel you often get an even lower error rate.

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

SVM - Radial Kernel in Feature Space

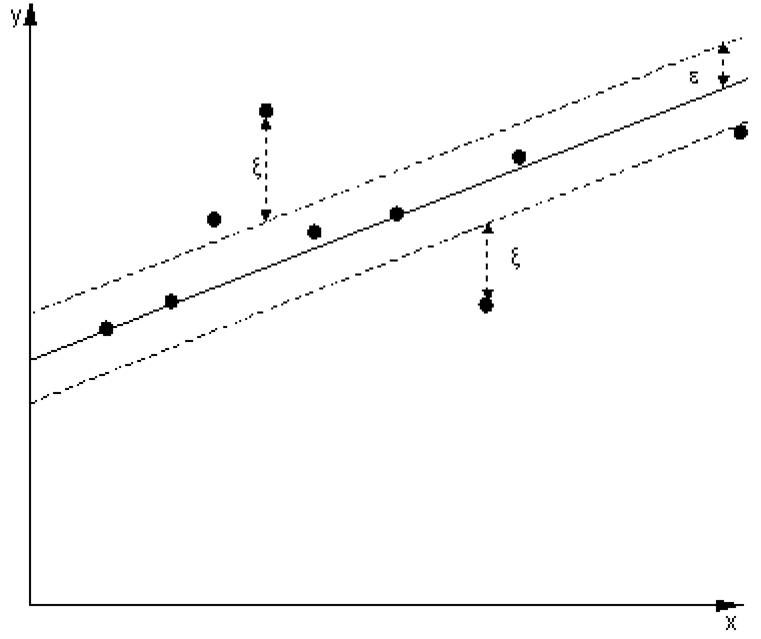


SVM for more than two classes

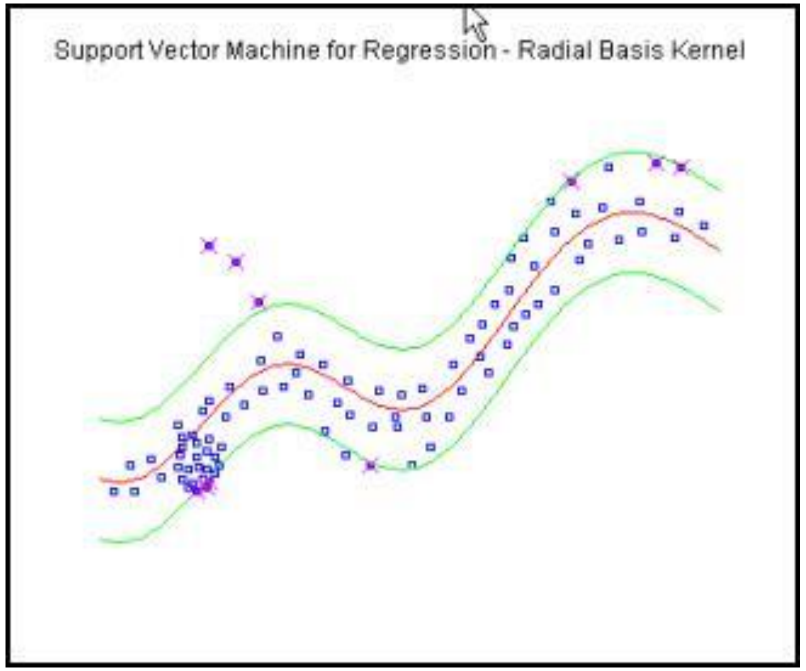
- The SVM as defined works for $K = 2$ classes. What do we do if we have $K > 2$ classes?
- One Versus All (OVA)
  Fit K different 2-class SVM classifiers $f_k(x)$, $k = 1,\dots,K$; each class versus the rest. Classify new x to the class for which $f_k(x)$ is largest.
- One Versus One (OVO)
  Fit all $\binom{K}{2}$ pairwise classifiers $f_{uv}(x)$. Classify new x to the class that wins the most pairwise competitions.
- Which to choose?
  If K is not too large, use OVO.

SVM for regression

- As in classification, seek and optimize the generalization bounds given for regression.
- The loss function ignores errors which are situated within the certain distance of the true value; it is often called – epsilon intensive – loss function.



for linear SVM regression


Support Vector Machine for Regression - Radial Basis Kernel