Support Vector Machines

Reference: The Elements of Statistical Learning,
by T. Hastie, R. Tibshirani, J. Friedman, Springer

Separating Hyperplanes

- Construct linear decision boundaries that explicitly try to separate the data into different classes

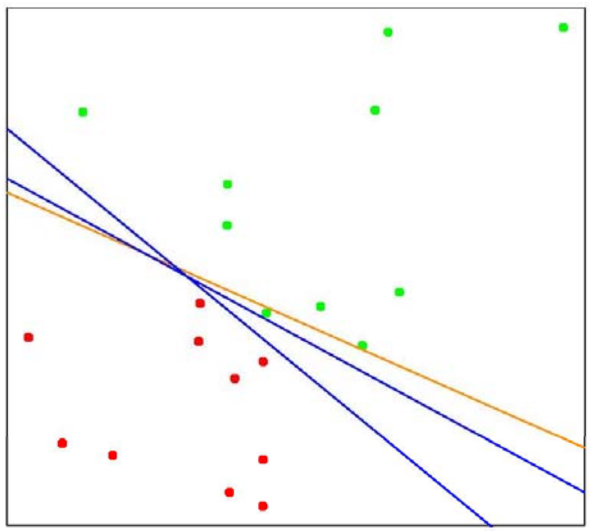


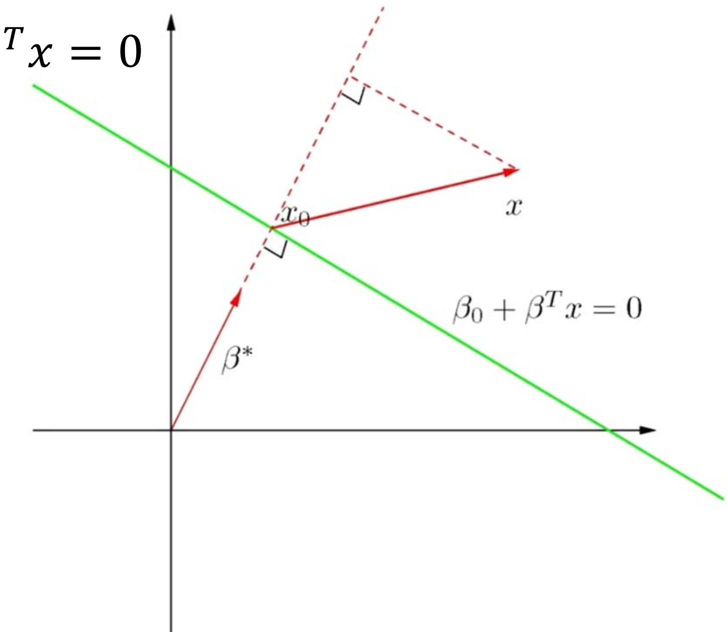
FIGURE 4.14. *A toy example with two classes separable by a hyperplane. The orange line is the least squares solution, which misclassifies one of the training points. Also shown are two blue separating hyperplanes found by the* perceptron learning algorithm *with different random starts.*

Separating Hyperplanes

- Construct classifiers that use a linear combination of input features and return the sign were called *perceptrons*
    - Perceptrons set the foundations for neural network models
- Hyperplane or affine set L defined by equation: $f(x) = \beta_0 + \beta^T x = 0$
- Since in $\mathbb{R}^2$, this is a line

Separating Hyperplanes

$f(x) = \beta_0 + \beta^T x = 0$



Properties:

- For any two points x_1 and x_2 lying in L, $\beta^T(x_1 - x_2) = 0$
  → $\beta^* = \frac{\beta}{\|\beta\|}$ 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:

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

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

Optimal Separating Hyperplanes

- $\{x_1, \ldots, x_N\}$: our training dataset in d-dimension
- $y_i \in \{1,-1\}$: class label
  - Note that the label value is 1 and -1 (not 1 and 0)
- Hyperplane defined by equation:

$$f(x) = x^T\beta + \beta_0 = 0$$

- A classification rule is:

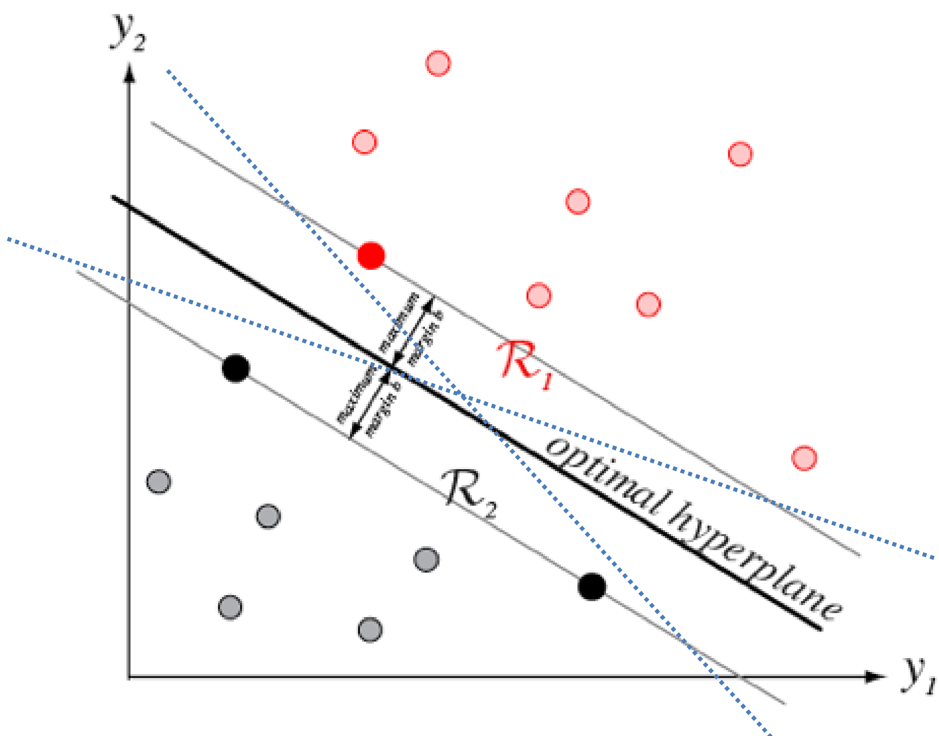
$$G(x) = \text{sign}[x^T\beta + \beta_0]$$

- Since the classes are separable, we have

$$y_i f(x_i) > 0 \quad \forall i$$

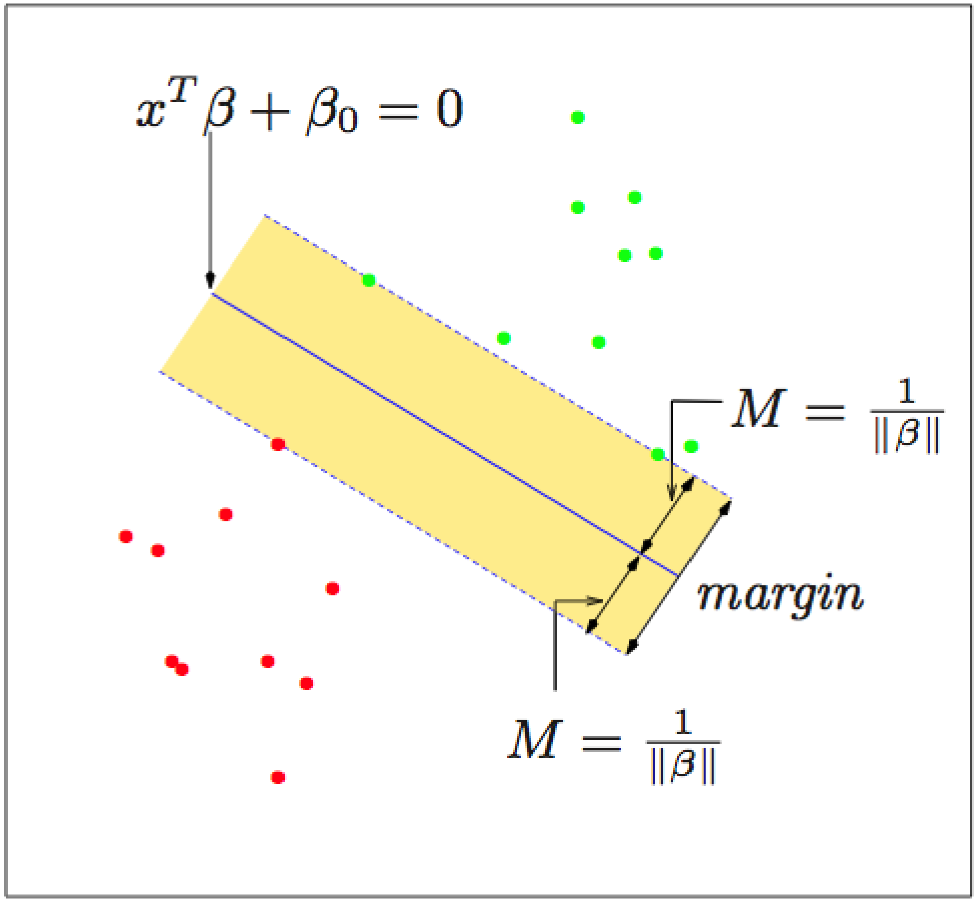
Optimal Separating Hyperplanes

- Find the optimal separating hyperplane
- Separates the two classes and maximizes the distance to the closest point from either class
- Leads to better classification performance on test data



Optimal Separating Hyperplanes

- The border is M away from the hyperplane.
- The band is 2M wide and it is called *margin*



- Try to maximize the margin:

$$\max_{\beta,\beta_0,||\beta||=1} M$$

$$\text{subject to } y_i(x_i^T\beta + \beta_0) \geq M,\ i = 1, \ldots, N,$$

Optimal Separating Hyperplanes

- Consider the optimization problem:

$$\max_{\beta, \beta_0, ||\beta||=1} M$$

$$\text{subject to } y_i(x_i^T \beta + \beta_0) \geq M, \; i = 1, \ldots, N.$$

- This can ensure that all the points are at least a signed distance M from the decision boundary
- We can get rid of ‖ β ‖=1 by replacing the conditions with:

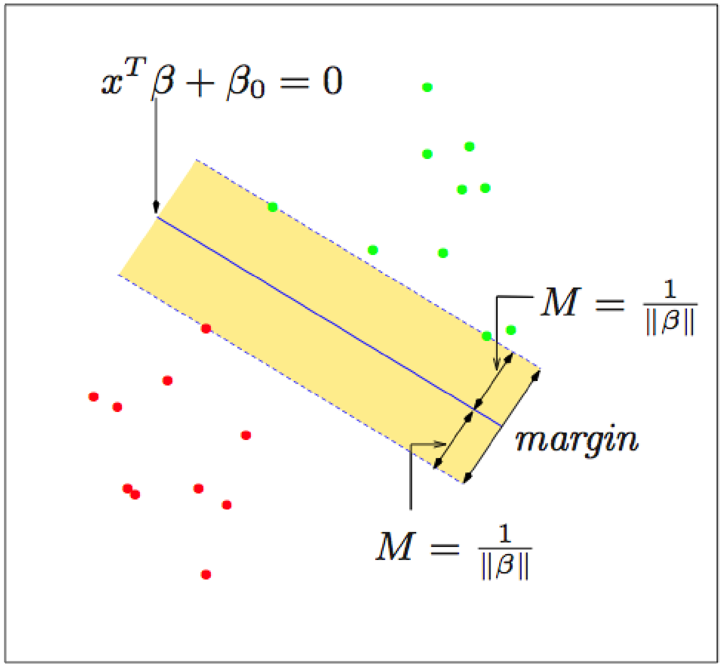
$$\frac{1}{||\beta||} y_i(x_i^T \beta + \beta_0) \geq M,$$

- Equivalently (redefine β_0)

$$y_i(x_i^T \beta + \beta_0) \geq M||\beta||.$$

Optimal Separating Hyperplanes

- Equivalently Since for any β and β_0 satisfying these inequalities, any positively scaled multiple satisfies them too, we can arbitrarily set $\| \beta \| = {}^{1}/_{M}$
- As a result, the optimization is equivalent to:



$$\min_{\beta,\beta_0} \frac{1}{2}||\beta||^2$$

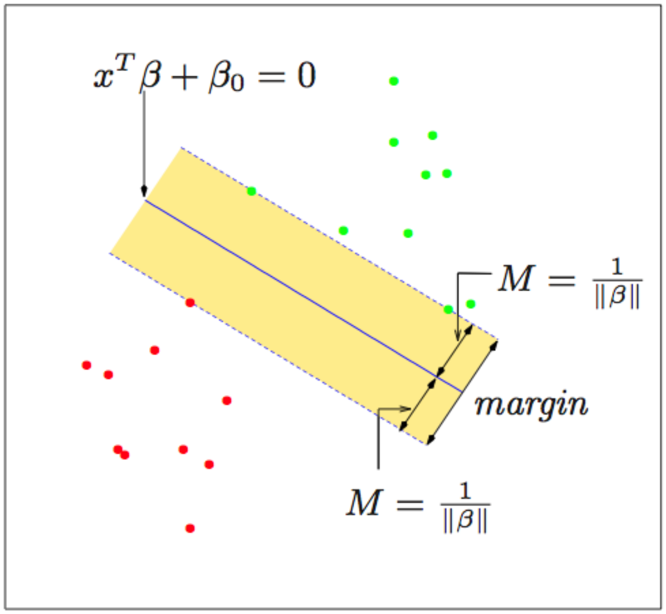
$$\text{subject to } y_i(x_i^T\beta + \beta_0) \geq 1,\ i = 1, \ldots, N.$$

Optimal Separating Hyperplanes

$$\min_{\beta,\beta_0} \frac{1}{2}||\beta||^2$$

$$\text{subject to } y_i(x_i^T\beta + \beta_0) \geq 1,\ i = 1, \ldots, N.$$

- The constraints define an empty margin around the linear decision boundary of thickness $^1/_{\|\beta\|}$
- We choose β and β_0 to maximize the thickness of the margin



Optimal Separating Hyperplanes

$$\min_{\beta,\beta_0} \frac{1}{2}||\beta||^2$$

$$\text{subject to } y_i(x_i^T\beta + \beta_0) \geq 1,\ i = 1, \ldots, N.$$

- A convex optimization problem (quadratic criterion with linear inequality constraints)
- The Lagrange function, to be minimized w.r.t. β and β_0, is:

$$L_P = \frac{1}{2} \parallel \beta \parallel^2 - \sum_{i=1}^{N} \alpha_i [y_i(x_i^T\beta + \beta_0) - 1]$$

Optimal Separating Hyperplanes

- Setting the derivatives to zero, we obtain:

$$\beta = \sum_{i=1}^{N} \alpha_i y_i x_i$$

$$0 = \sum_{i=1}^{N} \alpha_i y_i$$

- Substituting into the Lagrange function, we obtain Wolfe dual:

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{k=1}^{N} \alpha_i \alpha_k y_i y_k x_i^T x_k$$

subject to $\alpha_i \geq 0$ and $\sum_{i=1}^{N} \alpha_i y_i = 0$

- The solution is obtained by maximizing L_D
- Standard software can be used

Optimal Separating Hyperplanes

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{k=1}^{N} \alpha_i \alpha_k y_i y_k x_i^T x_k$$

subject to $\alpha_i \geq 0$ and $\sum_{i=1}^{N} \alpha_i y_i = 0$

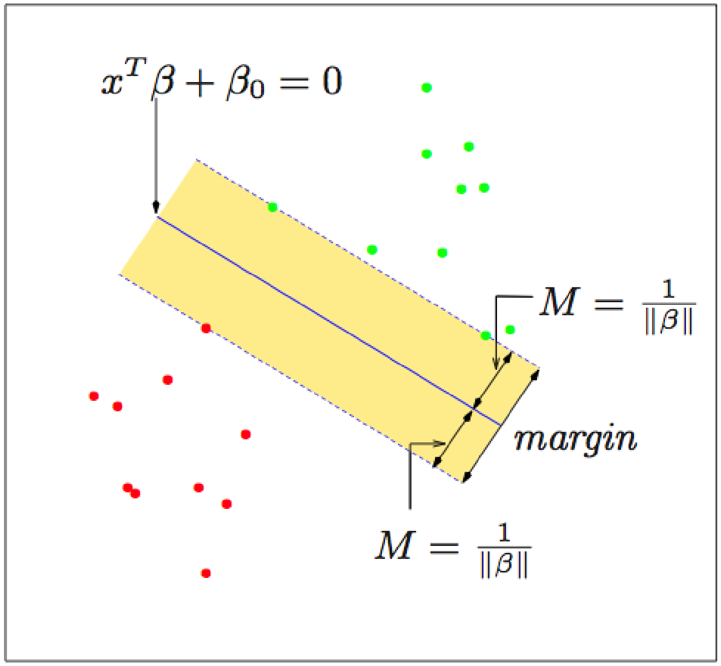
- The solution must satisfy the Karush-Kuhn-Tucker conditions, which include the previous equations and

$$\alpha_i \left[y_i \left(x_i^T \beta + \beta_0 \right) - 1 \right] = 0 \ \ \forall i$$

- From these, we can see that
  - If $\alpha_i > 0$, then $y_i \left(x_i^T \beta + \beta_0 \right) = 1$, or in other words, x_i is on the boundary of the slab;
  - If $y_i \left(x_i^T \beta + \beta_0 \right) > 1$, x_i is not on the boundary of the slab, and $\alpha_i = 0$

Optimal Separating Hyperplanes

- Recall that: $\beta = \sum_{i=1}^{N} \alpha_i y_i x_i$
- We can see that the solution vector β is defined in terms of a linear combination of the support points x_i
- Those points defined to be on the boundary of the slab via $\alpha_i > 0$



- Likewise, β_0 is obtained by solving the above equation for any of the support points

Optimal Separating Hyperplanes

- The hyperplane produces a function:
  $\hat{f}(x) = x^T\hat{\beta} + \hat{\beta}_0 = 0$
- For classifying new observations:
  $\hat{G}(x) = \text{sign}\hat{f}(x)$

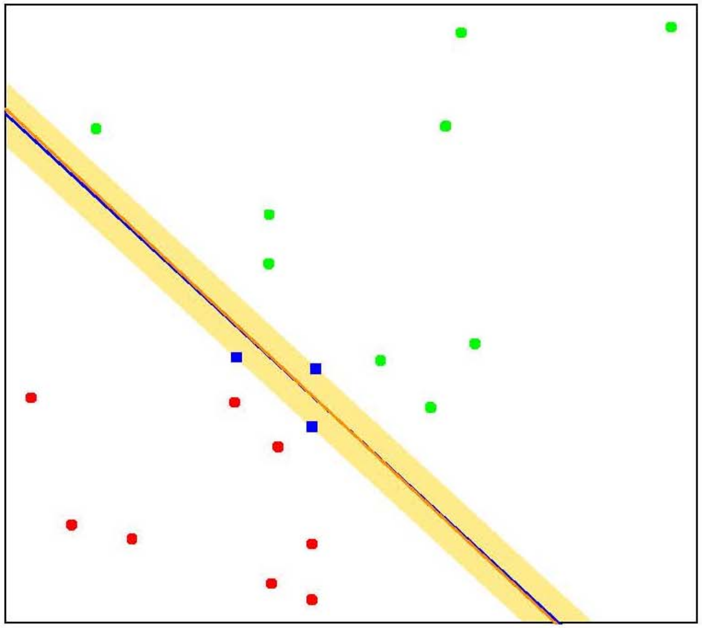


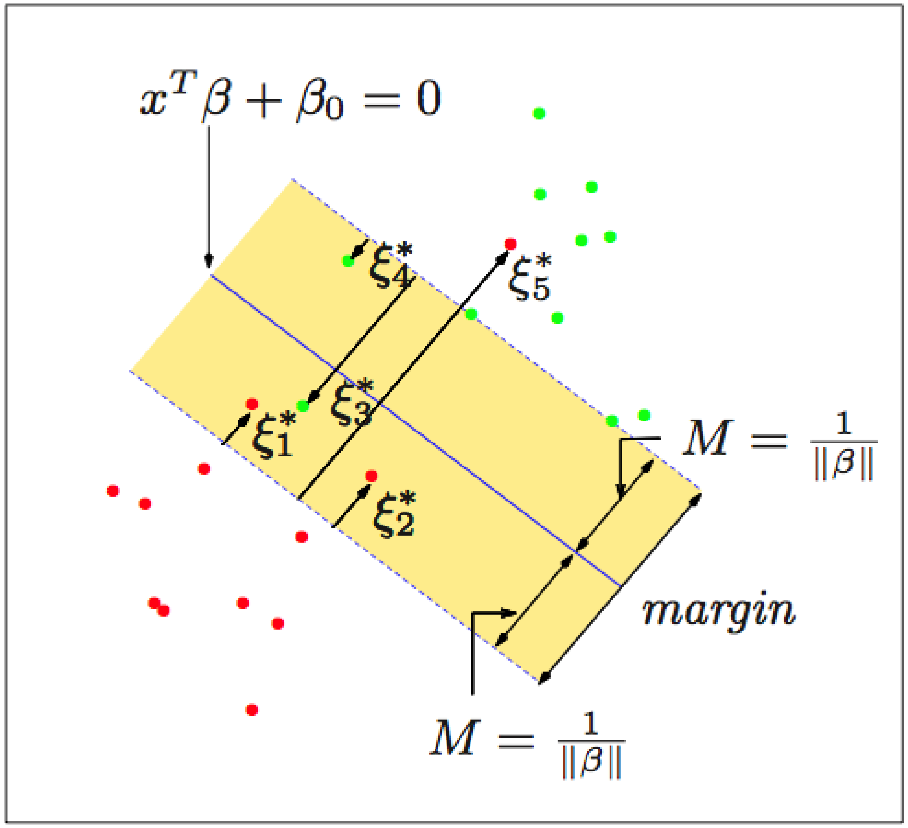
FIGURE 4.16. *The same data as in Figure 4.14. The shaded region delineates the maximum margin separating the two classes. There are three support points indicated, which lie on the boundary of the margin, and the optimal separating hyperplane (blue line) bisects the slab. Included in the figure is the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

- Support vectors suggest that hyperplane focuses more on points that count.

Non-Separable Cases

When two classes are not linearly separable, allow ***slack variables*** for the points on the wrong side of the border:

$$\xi = (\xi_1, \xi_2, \dots, \xi_N)$$



Two natural ways to modify constraint:

$$\max_{\beta, \beta_0, \|\beta\|=1} M$$

$$\text{subject to } y_i(x_i^T\beta + \beta_0) \geq M - \xi_i, \quad i = 1, \dots, N,$$

$$\text{or}$$

$$y_i(x_i^T\beta + \beta_0) \geq M(1 - \xi_i),$$

$$\forall i,\ \xi_i \geq 0,\ \textstyle\sum_{i=1}^N \xi_i \leq \text{constant}.$$

Non-Separable Cases

$$\max_{\beta, \beta_0, \|\beta\|=1} M$$

$$\text{subject to } y_i(x_i^T\beta + \beta_0) \geq M - \xi_i, \quad i = 1, \ldots, N,$$

$$\text{or}$$

$$y_i(x_i^T\beta + \beta_0) \geq M(1 - \xi_i),$$

$$\forall i, \ \xi_i \geq 0, \ \textstyle\sum_{i=1}^{N} \xi_i \leq \text{constant}.$$

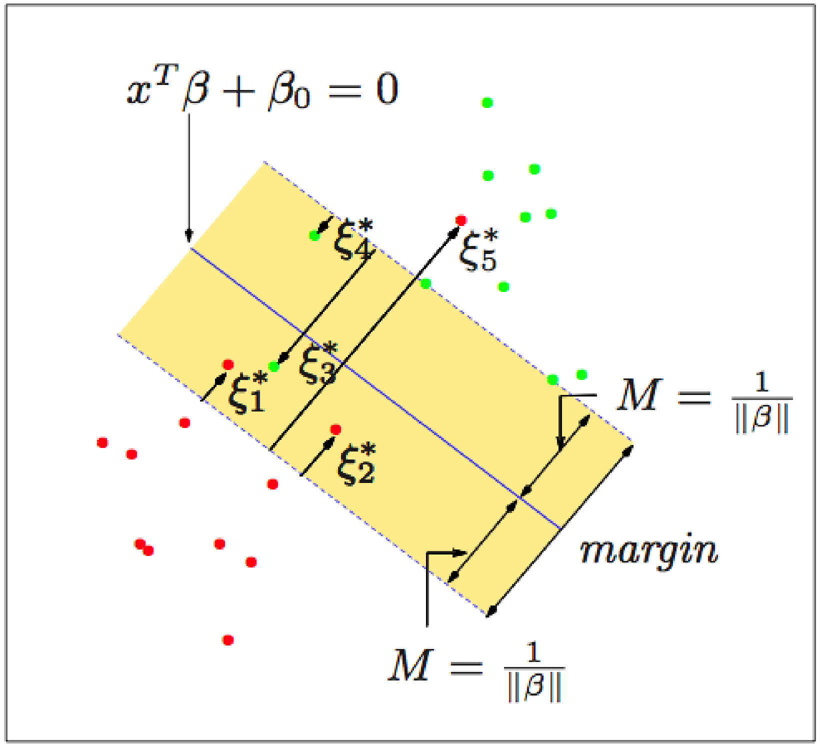
- For the constraint related to margin
  - The first choice results in a nonconvex optimization problem
  - The second choice is a convex optimization problem leading to the well-known support vector classifier

Non-Separable Cases

The optimization problem becomes:

$$\min \|\beta\| \quad \text{subject to} \begin{cases} y_i(x_i^T\beta + \beta_0) \geq 1 - \xi_i \ \forall i, \\ \xi_i \geq 0, \ \sum \xi_i \leq \text{constant}. \end{cases}$$

- $\xi=0$ when the point is on the correct side of the margin;
- $\xi>1$ when the point passes the hyperplane to the wrong side;
- $0<\xi<1$ when the point is in the margin but still on the correct side.



Non-Separable Cases

$$\min \|\beta\| \quad \text{subject to} \begin{cases} y_i(x_i^T\beta + \beta_0) \geq 1 - \xi_i \ \forall i, \\ \xi_i \geq 0, \ \sum \xi_i \leq \text{constant}. \end{cases}$$

- When a point is outside the boundary, $\xi=0$. It does not play a big role in determining the boundary ---- not forcing any special class of distribution.

Non-Separable Cases

$$\min \|\beta\| \quad \text{subject to} \begin{cases} y_i(x_i^T\beta + \beta_0) \geq 1 - \xi_i \ \forall i, \\ \xi_i \geq 0, \ \sum \xi_i \leq \text{constant}. \end{cases}$$

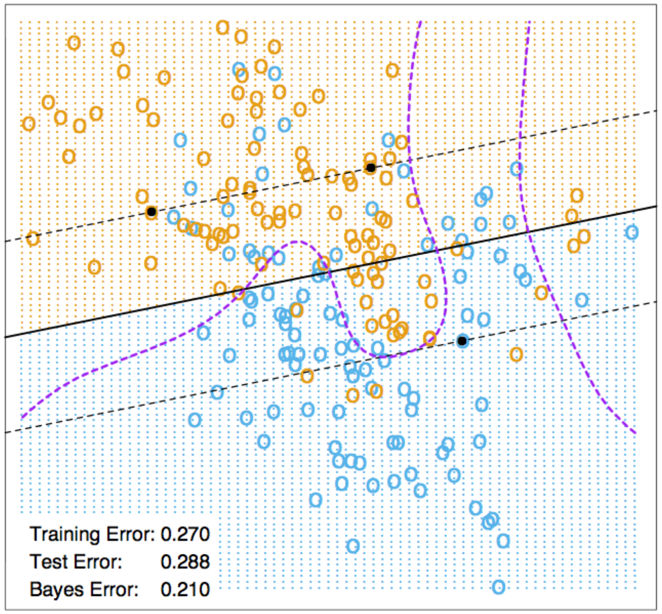
equivalent

$$\min_{\beta, \beta_0} \frac{1}{2}\|\beta\|^2 + C\sum_{i=1}^{N} \xi_i$$

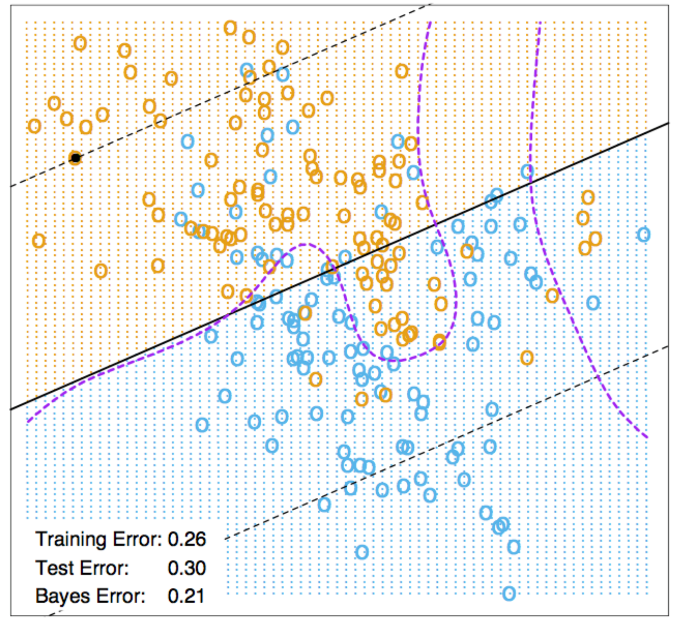
$$\text{subject to} \quad \xi_i \geq 0, \ y_i(x_i^T\beta + \beta_0) \geq 1 - \xi_i \ \forall i,$$

C replaces the “constant” and it can be regarded as a cost parameter

Effect of Cost Parameter



$C = 10000$



$C = 0.01$

Figure 12.2: *The linear support vector boundary for the mixture data example with two overlapping classes, for two different values of* γ. *The broken lines indicate the margins, where* $f(x) = \pm 1$. *The support points* ($\alpha_i > 0$) *are all the points on the wrong side of their margin. The black solid dots are those support points falling exactly on the margin* ($\xi_i = 0$, $\alpha_i > 0$). *In the upper panel* 62% *of the observations are support points, while in the lower panel* 85% *are.*

Support Vectors:

- Points on the wrong side of the boundary
- Points on the correct side of the boundary, but close to it.

Non-Separable Cases

Computation

- The Lagrange function is:

$$L_P = \frac{1}{2}\|\beta\|^2 + C\sum_{i=1}^{N}\xi_i - \sum_{i=1}^{N}\alpha_i[y_i(x_i^T\beta + \beta_0) - (1-\xi_i)] - \sum_{i=1}^{N}\mu_i\xi_i,$$

which we minimize w.r.t. β, β_0, ξ_i (12.9)

- Take derivatives of β, β_0, ξ_i, set to zero:

$$\beta = \sum_{i=1}^{N}\alpha_i y_i x_i, \quad (12.10)$$

$$0 = \sum_{i=1}^{N}\alpha_i y_i, \quad (12.11)$$

$$\alpha_i = C - \mu_i, \ \forall i, \quad (12.12)$$

and positivity constraints: $\alpha_i, \ \mu_i, \ \xi_i \geq 0 \ \forall i$

Non-Separable Cases

Computation

Substitute 12.10~12.12 into 12.9, the Lagrangian dual objective function:

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{i'=1}^{N} \alpha_i \alpha_{i'} y_i y_{i'} x_i^T x_{i'},$$

maximize L_D subject to $0 \leq \alpha_i \leq C$ and $\sum_{i=1}^{N} \alpha_i y_i = 0$.

Karush-Kuhn-Tucker conditions include

$$\alpha_i [y_i (x_i^T \beta + \beta_0) - (1 - \xi_i)] = 0, \quad (12.14)$$

$$\mu_i \xi_i = 0, \quad (12.15)$$

$$y_i (x_i^T \beta + \beta_0) - (1 - \xi_i) \geq 0, \quad (12.16)$$

Non-Separable Cases

Computation

- From $\beta = \sum_{i=1}^{N} \alpha_i y_i x_i$, The solution of β has the form:

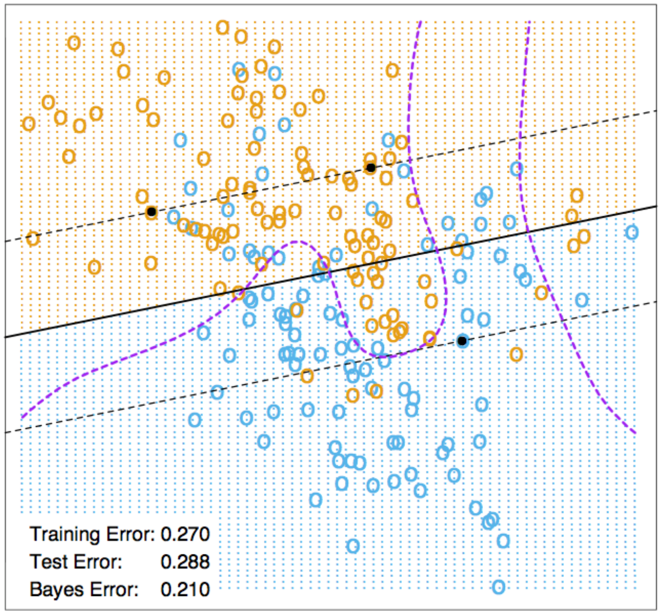
$$\hat{\beta} = \sum_{i=1}^{N} \hat{\alpha}_i y_i x_i,$$

- Non-zero coefficients $\hat{\alpha}_i$ only for those points *i* for which

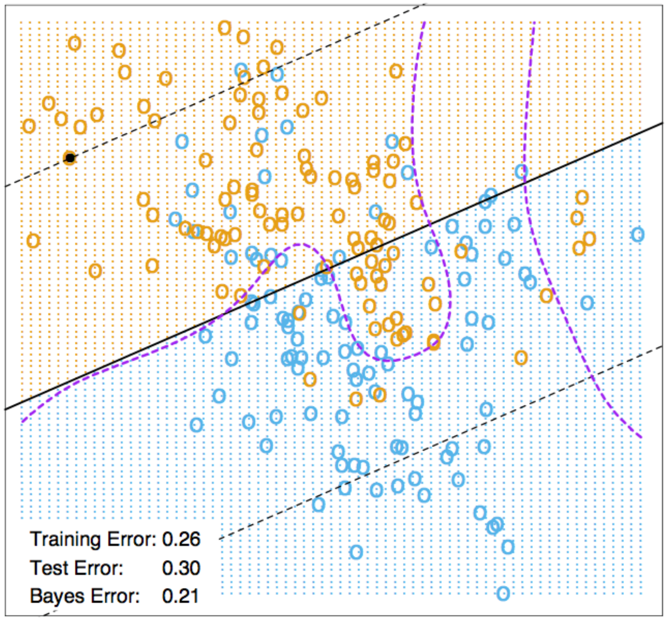
$$[y_i(x_i^T \beta + \beta_0) - (1 - \xi_i)] = 0,$$

  - These are called "support vectors". Some will lie on the edge of the margin $(0 < \hat{\alpha}_i < C; \hat{\xi}_i = 0)$
  - The remainder have $0 < \hat{\xi}_i$ $\hat{\alpha}_i = C$ They are on the wrong side of the margin.

Effect of Cost Parameter



$C = 10000$

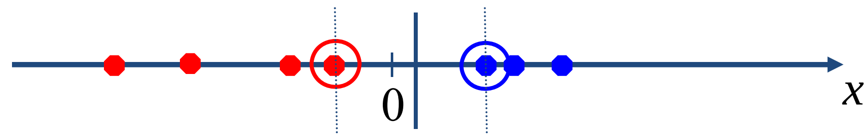


$C = 0.01$

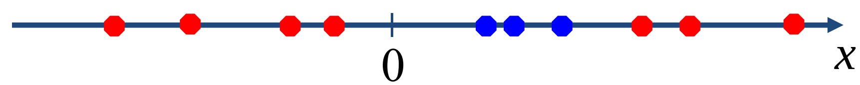
- Larger values of C focus attention more on (correctly classified) points near the decision boundary
- Smaller values of C involve data further away
- Either way, misclassified points are given weights, no matter how far away.

Non-linear SVM

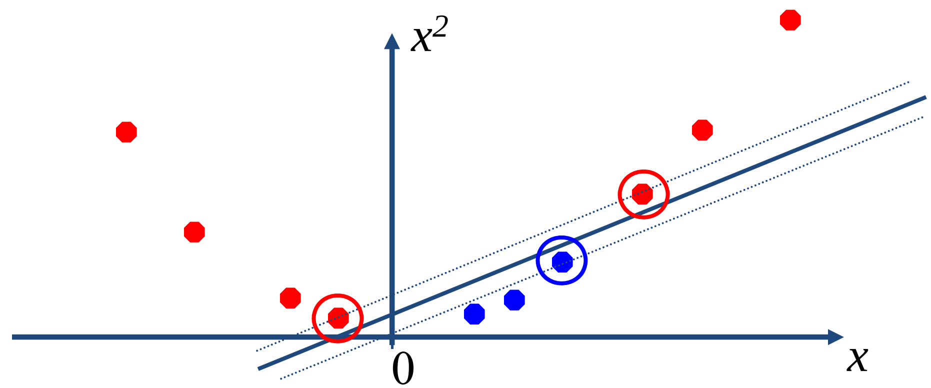
- Datasets that are linearly separable with noise work out great:



- But what are we going to do if the dataset is just too hard?



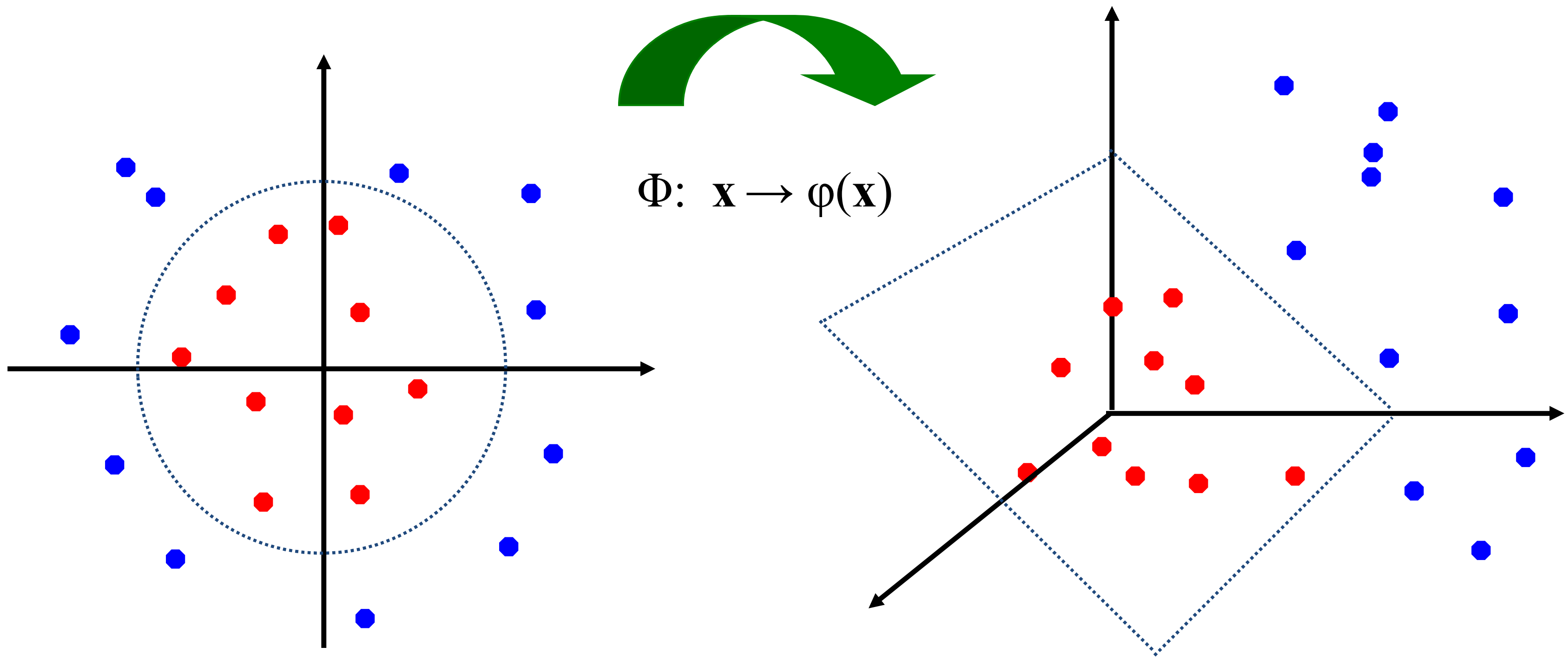
- How about… mapping data to a higher-dimensional space:



Non-linear SVM

Feature Space

- General idea: the original input space can be mapped to some higher-dimensional feature space where the training set is separable:



Non-linear SVM

Kernels

- Since $\hat{\beta} = \sum_{i=1}^{N} \hat{\alpha}_i y_i x_i$,

  the classifying function will have the form:

$$\hat{f}(x) = \sum_{i=1}^{N} \hat{\alpha}_i y_i x^T x_i + \hat{\beta}_0$$

$$= \sum_{i=1}^{N} \hat{\alpha}_i y_i \langle x, x_i \rangle + \hat{\beta}_0$$

- Note that most $\hat{\alpha}_i$ are zero. It relies on an *inner product* between test point x and the support vectors x_i (non-zero $\hat{\alpha}_i$)

Non-linear SVM
Kernels

- Enlarge the feature space to make the procedure more flexible
- Basis functions (mapping function)

$$h(x_i) = (h_1(x_i), h_2(x_i), \ldots, h_M(x_i)),$$

- Use the same procedure to construct support vector classifier

$$\hat{f}(x) = h(x)^T \hat{\beta} + \hat{\beta}_0.$$

- The decision is made by

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

Non-linear SVM

Kernels

Recall in linear space:

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{i'=1}^{N} \alpha_i \alpha_{i'} y_i y_{i'} x_i^T x_{i'}$$

With new basis:

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{i'=1}^{N} \alpha_i \alpha_{i'} y_i y_{i'} \underline{\langle h(x_i), h(x_{i'}) \rangle}.$$

- The inner products can be computed very efficiently

Non-linear SVM
Kernels

- Recall that

$$\hat{\beta} = \sum_{i=1}^{N} \hat{\alpha}_i y_i x_i \quad \longrightarrow \quad \hat{\beta} = \sum_{i=1}^{N} \hat{\alpha}_i y_i h(x_i)$$

- The model can go through similar transformation

$$\hat{f}(x) = \sum_{i=1}^{N} \hat{\alpha}_i y_i \langle x, x_i \rangle + \hat{\beta}_0$$

$$\downarrow$$

$$\hat{f}(x) = \sum_{i=1}^{N} \hat{\alpha}_i y_i \langle h(x), h(x_i) \rangle + \hat{\beta}_0$$

- It involves $h(x)$ only through inner products.

Non-linear SVM

Kernels

- In fact, we need not specify the transformation $h(x)$ at all, but require only knowledge of the kernel function:

$$K(x, x') = \langle h(x), h(x') \rangle$$

- It computes inner products in the transformed space. We don't need to know what $h(x)$ itself is!
  - It is also called "Kernel trick"
- Some commonly used kernels:

$$d\text{th-Degree polynomial: } K(x, x') = (1 + \langle x, x' \rangle)^d,$$

$$\text{Radial basis: } K(x, x') = \exp(-\gamma \|x - x'\|^2),$$

$$\text{Neural network: } K(x, x') = \tanh(\kappa_1 \langle x, x' \rangle + \kappa_2).$$

Non-linear SVM
Kernels

- Example: Consider a feature space with two inputs X_1 and X_2, and a polynomial kernel of degree 2. Then
$$K(X, X') = (1 + \langle X, X' \rangle)^2$$
$$= (1 + X_1X_1' + X_2X_2')^2$$
$$= 1 + 2X_1X_1' + 2X_2X_2' + (X_1X_1')^2 + (X_2X_2')^2 + 2X_1X_1'X_2X_2'$$

- Then, $M = 6$ and the mapping $h(X)$ consists of:
$$h_1(X) = 1,\ h_2(X) = \sqrt{2}X_1,\ h_3(X) = \sqrt{2}X_2,$$
$$h_4(X) = X_1^2,\ h_5(X) = X_2^2,\ h_6(X) = \sqrt{2}X_1X_2$$

- The inner product in the transformed space can be expressed in terms of a kernel function K in the original space
$$K(X, X') = \langle h(X), h(X') \rangle = (1 + \langle X, X' \rangle)^2$$

Non-linear SVM
Kernels

As a result

$$\hat{f}(X) = \sum_{i=1}^{N} \hat{\alpha}_i y_i K(x, x_i) + \hat{\beta}_0$$

Recall that $\hat{f}(X)$ depends only on the support vectors, i.e. $\hat{\alpha}_i \neq 0$

Non-linear SVM
Kernels

- The role of the parameter C is clearer in an enlarged feature space
- A large value of C will discourage any positive ξ_i, and lead to an overfit wiggly boundary in the original feature space
- A small value of C will encourage a small value of $||\beta||$, which in turn causes $f(x)$ and hence the boundary to be smoother

Non-linear SVM

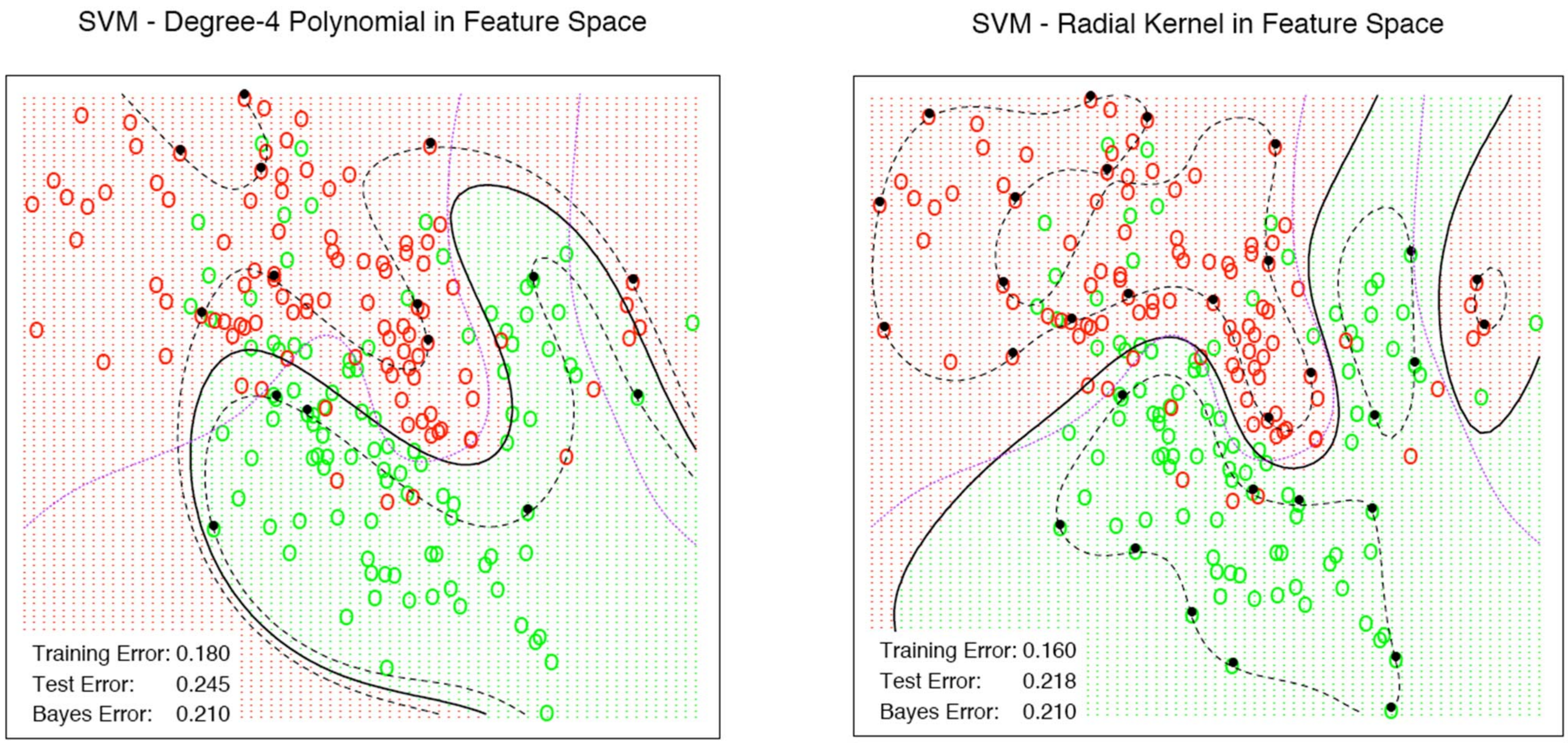


FIGURE 12.3. *Two nonlinear SVMs for the mixture data. The upper plot uses a 4th degree polynomial kernel, the lower a radial basis kernel (with $\gamma = 1$). In each case C was tuned to approximately achieve the best test error performance, and $C = 1$ worked well in both cases. The radial basis kernel performs the best (close to Bayes optimal), as might be expected given the data arise from mixtures of Gaussians. The broken purple curve in the background is the Bayes decision boundary.*

Non-linear SVM

Kernels

- K(x,x') can be seen as a similarity measure between x and x' .
- The decision is made essentially by a weighted sum of similarity of the object to all the support vectors.

$$f(x) = \sum_{x_i \in S} \alpha_i y_i K(x, x_i) + \beta_0$$

S: the set of support vectors