STATS 235: Modern Data Analysis
Support Vector Machines

Babak Shahbaba

Department of Statistics, UCI

Separating hyperplanes

- As discussed before, decision boundaries for linear classifiers are hyperplanes.
- A hyperplane in the column space of x has the following form:

$$\mathcal{L} : \beta_0 + \beta^\top x = 0$$

Separating hyperplanes

- For any point x_0 in the hyperplane, $\beta^\top x_0 = -\beta_0$.
- For any two points, x_1 and x_2, on the hyperplane, we have

$$\beta^\top (x_2 - x_1) = 0$$

- Therefore, $\beta/||\beta||$ is the normal vector to the hyperplane.
- For each point, x, in the space, the signed distance from the hyperplane is

$$\beta^\top (x - x_0)/||\beta|| = (\beta_0 + \beta^\top x)/||\beta||$$

- This is the projection of $(x - x_0)$ on the normal vector.

Separating hyperplanes

- A hyperplane divides the space into two subsets: a subset that includes points for which $\beta_0 + \beta^\top x > 0$, and another subset where $\beta_0 + \beta^\top x < 0$.
- For binary classification problems, $y \in \{-1, +1\}$, we would like to find a hyperplane such that the data points for each class fall on the same side.
- This way, we can classify observations according to the sign of $\beta^\top x + \beta_0$; that is, we would like to find a hyperplane so $\text{sign}(\beta_0 + \beta^\top x)$ matches the labels:

$$y\text{sign}(\beta_0 + \beta^\top x) = 1$$

- In general, there would be some observations that fall on the wrong side of the hyperplane: $y\text{sign}(\beta_0 + \beta^\top x) = -1$
- To find the best hyperplane, we can minimize the misclassification rate.

Perceptron

- Perceptrons do this by minimizing the following function

$$D(\beta_0, \beta) = -\sum_{i \in M} y_i(\beta_0 + x_i^\top \beta)$$

where M is the set of misclassified observations.

- The above quantity is non-negative and proportional to the distance of misclassified cases from the hyperplane.

- By differentiation with respect to each parameter we have

$$\partial D/\partial \beta_0 = -\sum_{i \in M} y_i$$

$$\partial D/\partial \beta = -\sum_{i \in M} y_i x_i$$

Perceptron

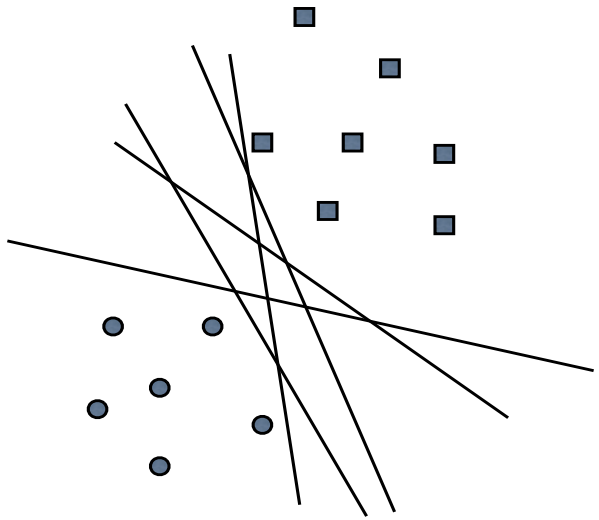
- To estimate parameters, we can start with some initial values for (β_0, β) and move in the direction of negative gradient.
- Rosenblatt's algorithm does this by using *stochastic* gradient descent approach.
- The algorithm visits misclassified observations in some sequence and updates the parameters as follows

$$\begin{pmatrix} \beta_0 \\ \beta \end{pmatrix} \leftarrow \begin{pmatrix} \beta_0 \\ \beta \end{pmatrix} + \rho \begin{pmatrix} y_i \\ y_i x_i \end{pmatrix}$$

where ρ is the stepsize.

Linearly separable problems

- We call a classification problem *linearly separable* if there is a hyperplane in the input space that can completely separates the observed classes.
- When a classification problem is linearly separable, there will be an infinite number of hyperplane decision boundaries that separate the classes completely.



Optimal separating hyperplanes

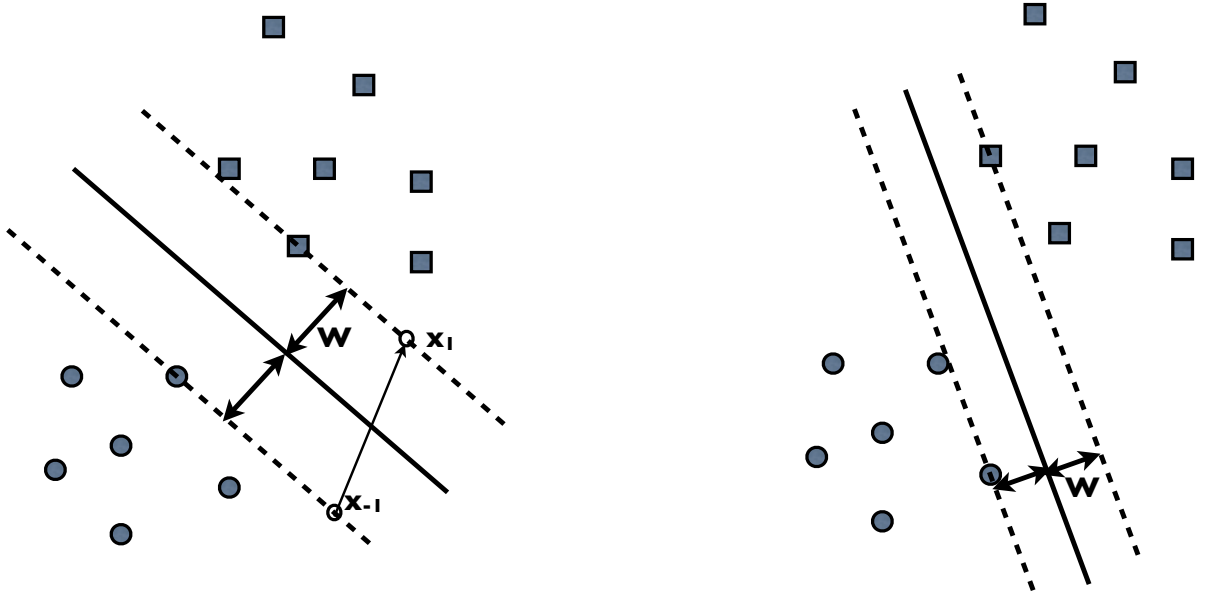
- How can we choose among all possible solutions (i.e., hyperplanes)?
- One strategy is to find the one with the maximum distance to the closest points from each class.
- The distance to the closest points is called the *margin*.
- This approach provides a unique response, and the resulting hyperplane is likely to perform well on future observations.
- The resulting hyperplane is called *optimal separating hyperplane*.
- The observed points with minimum distance to the hyperplane are called *support vectors*.

Optimal separating hyperplanes

- We want to maximize w subject to $y_i(\beta_0 + x_i^\top \beta) \geq w$; that is, the distance of any point from the hyperplane must be at least w.

$$\max_{\beta,\beta_0} \quad w$$

$$\text{Subject to} \quad y_i(\beta_0 + x_i^\top \beta)/||\beta|| \geq w, \quad i = 1, \ldots, n$$



Optimal separating hyperplanes

- Consider two points x_1 and x_{-1} on the boundaries defining the margins.
- The margin, w, is half of the perpendicular distance between these two points.
- Recall that $\beta/||\beta||$ is the normal vector to the hyperplane.
- Therefore, we can obtain the margin by projecting $(x_1 - x_{-1})$ on $\beta/||\beta||$

$$w = \frac{1}{2}(x_1 - x_{-1})^\top \beta/||\beta||$$

Optimal separating hyperplanes

- We can rescale β_0 and β so the $|\beta_0 + x^\top \beta|$ (i.e., the distance from the hyperplane) is equal to 1 for points defining the margins,

$$\begin{aligned} \beta_0 + x_1^\top \beta &= 1 \\ \beta_0 + x_{-1}^\top \beta &= -1 \end{aligned}$$

- This way, we can write the constraints as $y_i(\beta_0 + x_i^\top \beta) \geq 1$ for $i = 1, \ldots, n$.

- Also,

$$\begin{aligned} \beta_0 + x_1^\top \beta - \beta_0 + x_{-1}^\top \beta &= 2 \\ \beta^\top (x_1 - x_{-1}) &= 2 \end{aligned}$$

- As the result, $w = 1/||\beta||$

Optimal separating hyperplanes

- So maximizing the margin is equivalent to minimizing $||\beta||$.
- We can rewrite the optimization problem as

$$\begin{aligned} &\text{minimize} && ||\beta|| \\ &\text{Subject to} && y_i(\beta_0 + x_i^\top \beta) \geq 1, \quad i = 1, \ldots, n \end{aligned}$$

- Alternatively, we can use Lagrange multipliers and minimize the following objective function (called the primal function):

$$L_P = \frac{1}{2}||\beta||^2 - \sum_{i=1}^{n} \alpha_i [y_i(\beta_0 + x_i^\top \beta) - 1], \quad \alpha_i \geq 0$$

Optimal separating hyperplanes

- By setting the derivatives to zero, we have

$$\begin{aligned} \beta &= \sum_{i=1}^{n} \alpha_i y_i x_i \\ 0 &= \sum_{i=1}^{n} \alpha_i y_i \end{aligned}$$

- By substituting these values in the primal function, we obtain the *dual* problem, which we need to *maximize* (See Boyd and Vandenberghe, and my supplementary notes on constraint optimization)

$$\begin{aligned} \text{maximize} \quad & L_D = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^\top x_j \\ \text{Subject to} \quad & \sum_{i=1}^{n} \alpha_i y_i = 0, \qquad \alpha_i = 0 \end{aligned}$$

Optimal separating hyperplanes

- After we find the optimum values, $\hat{\alpha}$, we can write the estimates of β as follows:

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

- Then, the estimate of y is

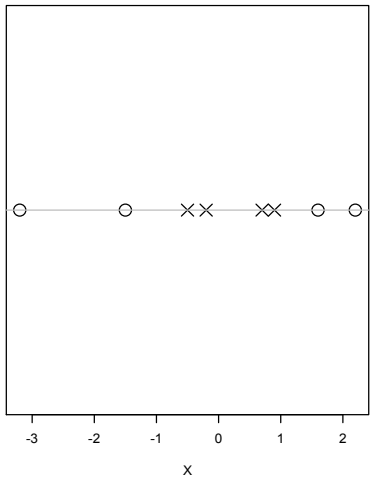
$$\begin{aligned} \hat{y}_i &= \text{sign}\left(\hat{\beta}_0 + x_i^\top \hat{\beta}\right) \\ &= \text{sign}\left(\hat{\beta}_0 + \sum_{i=1}^{n} \hat{\alpha}_i y_i x_i^\top x\right) \end{aligned}$$

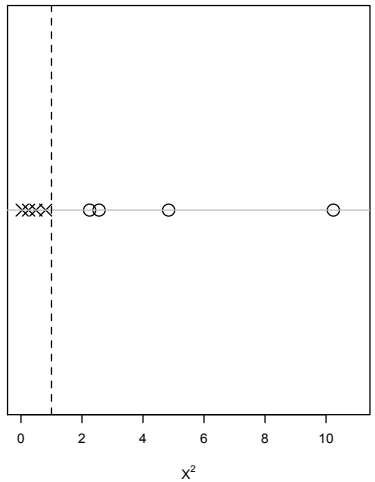
- Note that similar to the dual problem, the estimates depend on the input variables, x, only through their inner product.

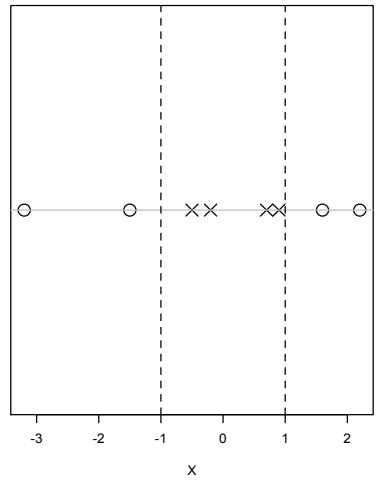
Expanding the feature space

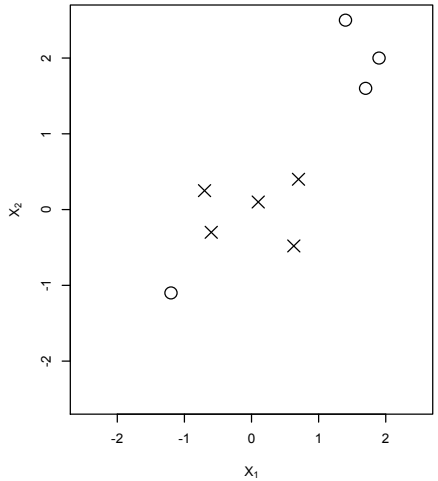
- So far, we assumed that the two classes are completely separable using linear boundaries. Of course, this is not always the case.
- When the classes are not completely separable using linear boundaries, we can expand the feature space using basis functions (analogous to what we did for splines) and find optimal hyperplanes in the augmented space.

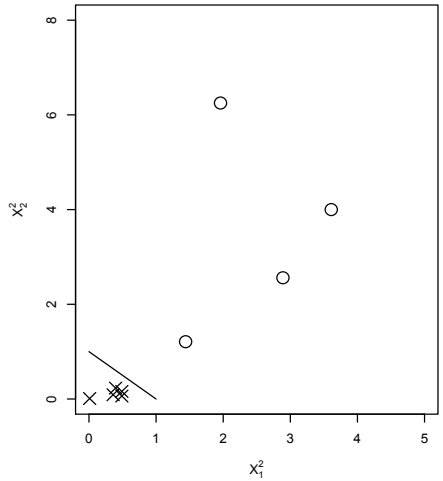
Example

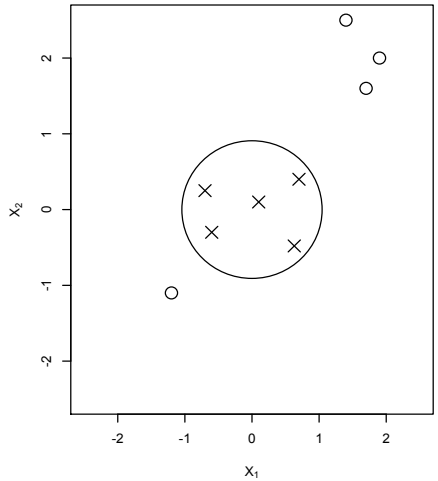

-3
-2
-1
0
1
2
x


0
2
4
6
8
10
x²


-3
-2
-1
0
1
2
x


-2
-1
0
1
2
x₂
-2
-1
0
1
2
x₁


0
2
4
6
8
x₂²
0
1
2
3
4
5
x₁²


-2
-1
0
1
2
x₂
-2
-1
0
1
2
x₁

The kernel trick

- Finding the solution for the above model (and most linear models) involves obtaining the inner product $x^\top x$.
- After we expand the feature space using basis functions $h(x) = (h_1(x), \ldots, h_M(x))$, finding the solution involves obtaining the inner products $K(x_j, x_k) = \langle h(x_j), h(x_k) \rangle$.
- We refer to $K(x_j, x_k)$ as the *kernel matrix*, which is symmetric and positive semidefinite.
- Therefore, it seems that we need to transform the original variables using functions $h(x) = (h_1(x), \ldots, h_M(x))$, and find the inner product of the transformed variables, $K(x_j, x_k)$.

The kernel trick

- Mercer's theorem indicates that iff a symmetric function $K(x, x')$ is positive semidefinite, it can be expressed as an inner product $K(x, x') = \langle h(x), h(x') \rangle$.
- Therefore, instead of first mapping x to a larger space and then taking the inner product, we can directly use the equivalent kernel function.
- Then,

$$\begin{aligned} \hat{y}_i &= \text{sign}\left(\hat{\beta}_0 + \hat{\beta}^\top x_i\right) \\ &= \text{sign}\left(\hat{\beta}_0 + \sum_{i=1}^{n} \hat{\alpha}_i y_i K(x, x_i)\right) \end{aligned}$$

The kernel trick

- In practice, we can use kernels that result in some nonlinear functions in the space of original variables, x.

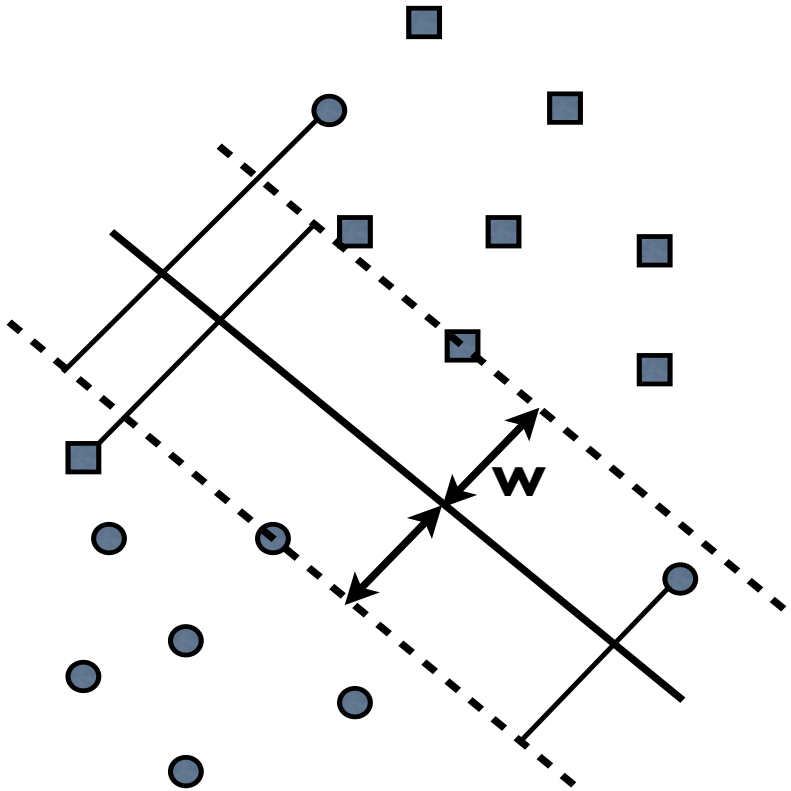
- Some popular choices of kernels are

$$\begin{aligned} d^{th} \text{ Degree polynomial:} \quad & K(x, x') = (1 + \langle x, x' \rangle)^d \\ \text{Radial basis:} \quad & K(x, x') = \exp(-||x - x'||^2/c) \\ \text{Sigmoid:} \quad & K(x, x') = \tanh(\kappa_1 \langle x, x' \rangle + \kappa_2) \end{aligned}$$

Support vector machines

- Enlarging the space arbitrarily until we find optimal hyperplanes can lead to overfitting.
- A more reasonable solution is provided by *support vector machines* that allows for enlarging the feature space through the kernel trick, but it also allows overlap between classes while penalizes large overlaps.
- For this, we introduce a set of slack variables, $\xi = (\xi_1, \ldots, \xi_n)$, where ξ_i proportional to the amount by which a point is on the wrong side of its margin.
- For points that fall on the correct side of their margins $\xi = 0$.

Support vector machines



Support vector machines

- Now we can modify the optimization problem for optimal hyperlanes as follows:

$$\begin{aligned} &\min \frac{1}{2}||\beta||^2 \\ &\text{Subject to} \quad y_i(\beta_0 + x_i^\top \beta) \geq 1 - \xi_i, \quad \forall i \\ &\qquad\qquad\qquad\quad \xi_i \geq 0, \\ &\qquad\qquad\qquad\quad \sum \xi_i \leq t \end{aligned}$$

for some constant t.

Support vector machines

- Alternatively, we can minimize

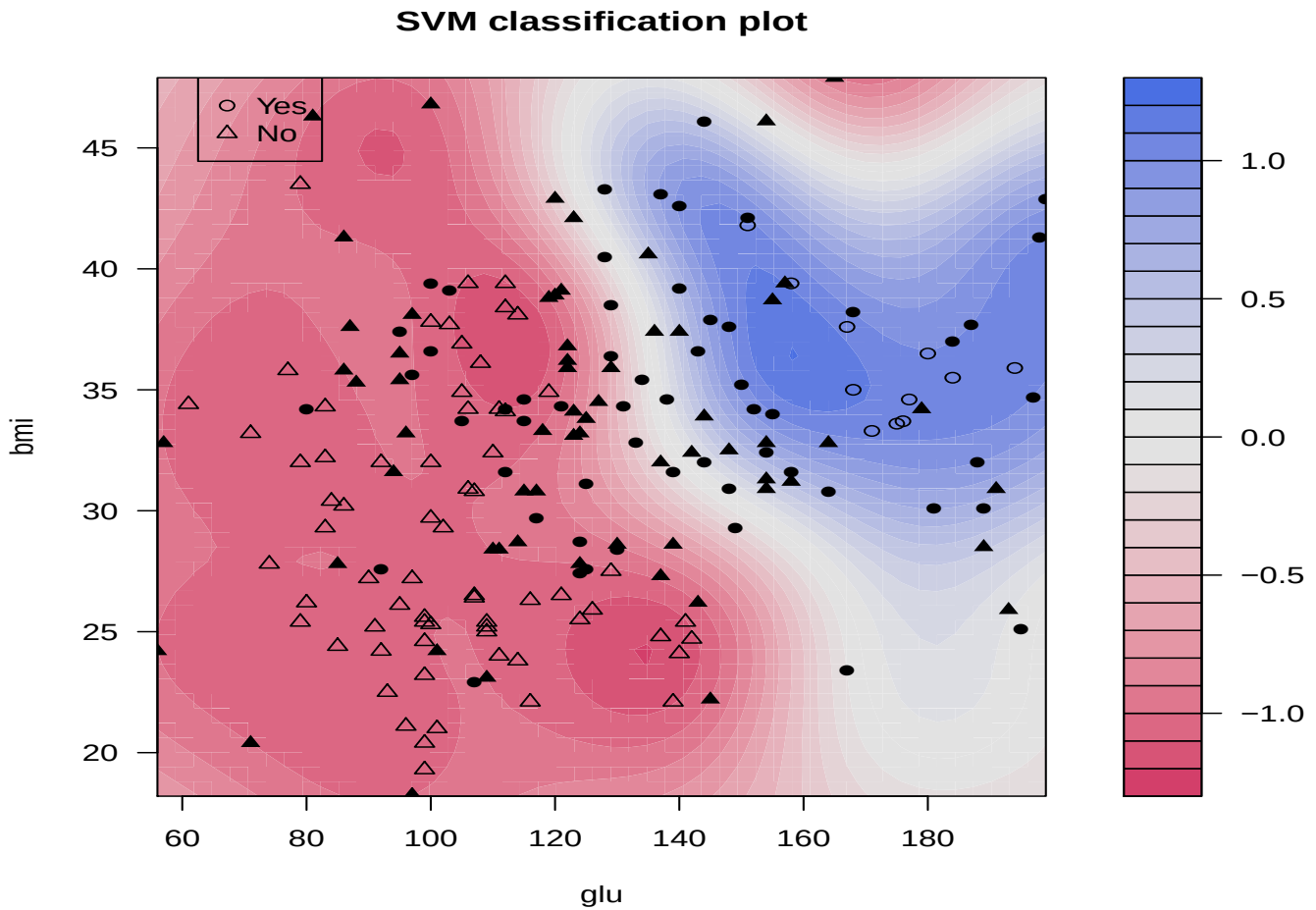
$$\frac{1}{2}||\beta||^2 + C\sum_i \xi_i - \sum_i \alpha_i[y_i(\beta_0 + x_i^\top \beta) - (1-\xi_i)] - \sum_i \mu_i \xi_i$$

- In this case, C represents the regularization parameter, which controls our tolerance for the number misclassifications during the training.

- It is common to set $C = 1/(\nu n)$, where $0 < \nu < 1$, and refer to the resulting model as ν-SVM classifier.

- As before, $\hat{y}$ depends on the input variables through $x^\top x$, which we usually replace by a kernel, $K(x, x')$.

Example: Pima Indian Women



Predicting diabetes among Pima Indian women