Lecture 22 : Support vector machines

Reading: Chapter 9

STATS 202: Data mining and analysis

November 13, 2019

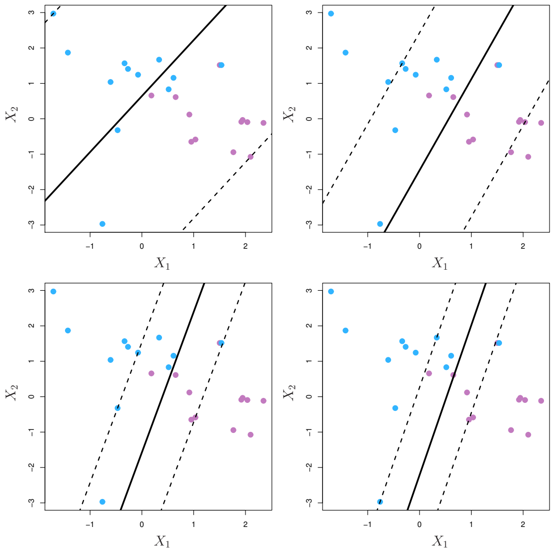
Review of support vector classifier

- The **support vector classifier** defines a hyperplane and two margins.
- **Goal:** to maximize the width of the margins, with some budget C for "violations of the margins", i.e.

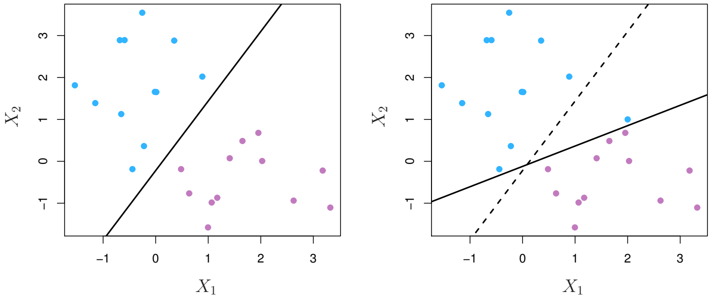
$$\sum_{\substack{x_i \text{ on the wrong} \\ \text{side of the margin}}} \text{Distance from } x_i \text{ to the margin} \;\leq\; C.$$

- The only points that affect the orientation of the hyperplane are those at the margin or on the wrong side of it.
- Low budget $C \iff$ Few samples used $\iff$ High variance $\iff$ Tendency to overfit.

Tuning the budget, C (high to low)



If the budget is too low, we tend to overfit



Maximal margin classifier, $C = 0$. Adding one observation dramatically changes the classifier.

Finding the support vector classifier

The problem can be reduced to the optimization:

$$\hat{\alpha} \;=\; \arg\max_{\alpha} \;\; \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{i'=1}^{n} \alpha_i \alpha_{i'} y_i y_{i'} (x_i \cdot x_{i'})$$

subject to $0 \leq \alpha_i \leq D$ for all $i = 1, \ldots, n,$

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

$$\hat{w} = \sum_{i=1}^{n} \alpha_i y_i x_i, \qquad \hat{w} \cdot x_0 = \sum_{i=1}^{n} \alpha_i y_i (x_i \cdot x_0)$$

Key fact about the support vector classifier

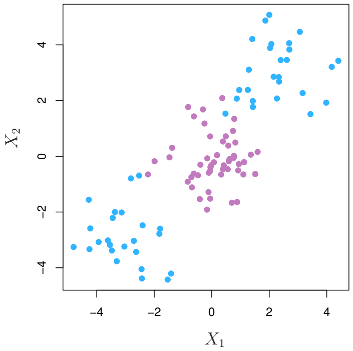
To **find the hyperplane** and **make predictions** all we need to know is the dot product between any pair of input vectors:

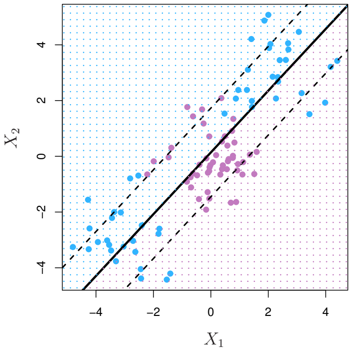
$$K(i,k) = (x_i \cdot x_k) = \langle x_i, x_k \rangle = \sum_{j=1}^{p} x_{ij} x_{kj}$$

We call this the **kernel matrix**.

How to deal with non-linear boundaries?

The support vector classifier can only produce a linear boundary.





How to deal with non-linear boundaries?

- In **logistic regression**, we dealt with this problem by adding transformations of the predictors.
- The original decision boundary is a line:

$$\log\left[\frac{P(Y=1|X)}{P(Y=0|X)}\right] = \beta_0 + \beta_1 X_1 + \beta_2 X_2 = 0.$$

- With a quadratic predictor, we get a quadratic boundary:

$$\log\left[\frac{P(Y=1|X)}{P(Y=0|X)}\right] = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_1^2 = 0.$$

How to deal with non-linear boundaries?

- With a **support vector classifier** we can apply a similar trick.
- The original decision boundary is the hyperplane defined by:

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 = 0.$$

- If we expand the set of predictors to the 4D space (X_1, X_2, X_1^2, X_2^2), the decision boundary becomes:

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_1^2 + \beta_4 X_2^2 = 0.$$

- This is in fact a linear boundary in the augmented variable set (X_1, X_2, X_1^2, X_2^2) but a quadratic boundary in (X_1, X_2).

How do we expand the space of predictors?

- **Idea:** Add polynomial terms up to degree d:

  $$Z = (X_1, X_1^2, \ldots, X_1^d, X_2, X_2^2, \ldots, X_2^d, \ldots, X_p, X_p^2, \ldots, X_p^d).$$

- Does this make the computation more expensive?
- Recall that all we need to compute is the dot product:

  $$x_i \cdot x_k = \langle x_i, x_k \rangle = \sum_{j=1}^{p} x_{ij} x_{kj}.$$

- With the expanded set of predictors, we need:

  $$z_i \cdot z_k = \langle z_i, z_k \rangle = \sum_{j=1}^{p} \sum_{\ell=1}^{d} x_{ij}^{\ell} x_{kj}^{\ell}.$$

Kernels

The **kernel matrix** defined by $K(i,k) = \langle z_i, z_k \rangle$ for a set of linearly independent vectors $z_1, \ldots, z_n$ is always **positive semi-definite**, i.e. it is symmetric and has no negative eigenvalues.

Theorem:

If K is a positive definite $n \times n$ matrix, there exist vectors $(z_1, \ldots, z_n)$ in some space $\mathbf{Z}$, such that $K(i,k) = \langle z_i, z_k \rangle$.

The kernel trick

Example: Suppose that you want to include all linear and quadratic features, including pairwise interactions, in your feature expansion:

$$\begin{aligned}\Phi(X) = &(1, \sqrt{2}X_1, \ldots, \sqrt{2}X_p, \\ &X_1^2, \ldots, X_p^2, \\ &\sqrt{2}X_1X_2, \sqrt{2}X_1X_3, \ldots, \sqrt{2}X_{p-1}X_p)\end{aligned}$$

Computing $K(i,k)$ via $\langle \Phi(X_i), \Phi(X_k) \rangle$ takes order p^2 time.

Equivalently we can compute

$$K(i,k) = k(x_i, x_k) = (1 + \langle x_i, x_k \rangle)^2$$

which takes $O(p)$ time!

$k(x_i, x_k) = (1 + \langle x_i, x_k \rangle)^2$ is a degree 2 **polynomial kernel**.

The kernel trick

Expand the set of predictors:

- Find a mapping Φ which expands the original set of predictors $X_1, \ldots, X_p$. For example,

$$\Phi(X) = (X_1, X_2, X_1^2, X_2^2)$$

- For each pair of samples, compute:

$$K(i,k) = \langle \Phi(x_i), \Phi(x_k) \rangle.$$

Define a kernel:

- Find a positive definite function $k(\cdot,\cdot)$. For example:

$$k(x_i, x_k) = \left(1 + \langle x_i, x_k \rangle\right)^2 .$$

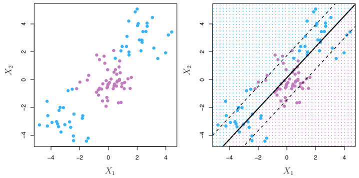
- For each pair of samples, compute:

$$K(i,k) = k(x_i, x_k).$$

- Often much easier!

How are kernels defined?

- Proving that a bilinear function $k(\cdot, \cdot)$ is positive definite (PD) is not always easy.
- However, we can easily define PD kernels by combining those we are familiar with:
  - Sums and products of PD kernels are PD.
- Intuitively, a kernel $k(x_i, x_k)$ defines a *similarity* between the samples x_i and x_k. This intuition can guide our choice in different problems.



Common kernels

- The polynomial kernel:

$$k(x_i, x_k) = (1 + \langle x_i, x_k \rangle)^d$$

- The radial basis kernel:

$$k(x_i, x_k) = \exp\Big(-\gamma \underbrace{\sum_{j=1}^{p} (x_{ip} - x_{kp})^2}_{\text{Euclidean } d(x_i, x_k)} \Big)$$