Support Vector Machines

Fabio G. Cozman - fgcozman@usp.br

November 20, 2018

Linearly separable problems

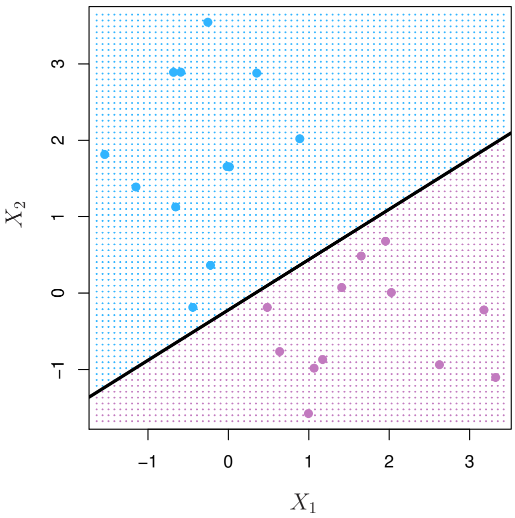
- Consider a classification problem with a binary class variable Y with values 1 and -1.

Linearly separable problems

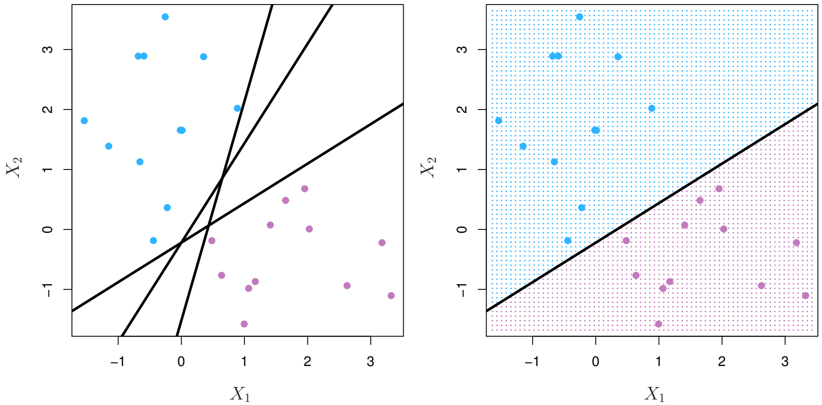
- Consider a classification problem with a binary class variable Y with values 1 and -1.
- Suppose the classes are linearly separable by an hyperplane:

$$H(X) = \beta_0 + \beta_1 X_1 + \cdots + \beta_n X_n = 0.$$

Example



Separating the labels given a dataset

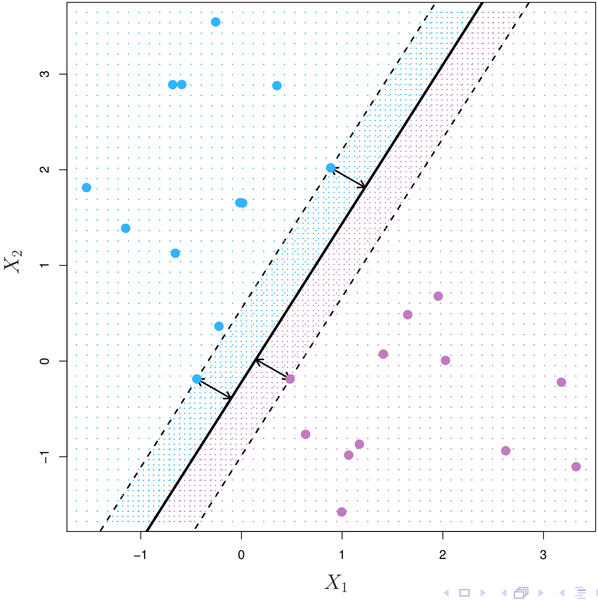


Maximal margin classifier

- In case there are many possible hyperplanes that separate the data: take the hyperplane with maximum *margin*.

- Margin: the distance from the hyperplane to the closest training point.

Example: hyperplane with maximal margin



Support vectors and distance

- For a hyperplane with maximal margin, the support vectors are the points that are closest to it.

- For such a hyperplane, the distance between the *i*th observation to the hyperplane is

$$\frac{y_j(\beta_0 + \beta_1 x_{1,j} + \cdots + \beta_n x_{n,j})}{\sqrt{\beta_1^2 + \beta_2^2 + \cdots + \beta_n^2}}.$$

Note:

- If a problem is linearly separable, there is a hyperplane such that

$$y_j(\beta_0 + \beta_1 x_{1,j} + \cdots + \beta_n x_{n,j}) > 0$$

for all observations.

Finding the hyperplane

- Find $\beta_0, \beta_1, \ldots, \beta_n$ that maximize M subject to:

$$\sum_{i=1}^{n} \beta_i^2 = 1,$$

and

$$y_j(\beta_0 + \beta_1 x_{1,j} + \cdots + \beta_n x_{n,j}) \geq M$$

for each j.

Equivalent problem

- Find $\beta_0, \beta_1, \ldots, \beta_n$ that minimize

$$\sum_{i=1}^{n} \beta_i^2,$$

subject to

$$y_j(\beta_0 + \beta_1 x_{1,j} + \cdots + \beta_n x_{n,j}) \geq 1$$

for each j.

Quadratic programming

- These optimization problems can be solved (relatively) quickly using quadratic programming.
- Optimum is guaranteed to be found.

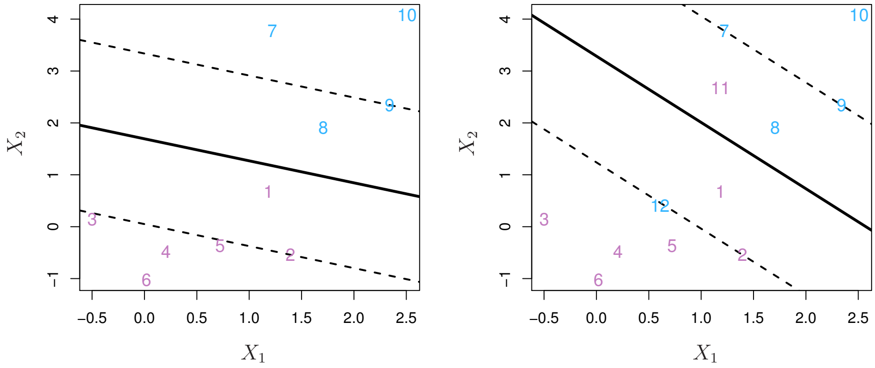
Non-separable problem

- In practice, problems are non-separable (and we cannot hope to deal only with separable problems).
  - If problem is non-separable, there is no solution with $M > 0$.

Non-separable problem

- In practice, problems are non-separable (and we cannot hope to deal only with separable problems).
  - If problem is non-separable, there is no solution with $M > 0$.
- Problem must be relaxed.
  - Consider a *soft margin*: let some observations violate the margin.

Soft margin



The "soft" optimization problem

- Find $\beta_0, \beta_1, \ldots, \beta_n$ and $\epsilon_1, \ldots, \epsilon_N$ that maximize M subject to:

$$\sum_{i=1}^{n} \beta_i^2 = 1,$$

and

$$y_j(\beta_0 + \beta_1 x_{1,j} + \cdots + \beta_n x_{n,j}) \geq M(1 - \epsilon_j)$$

for each j, and

$$\epsilon_j \geq 0, \qquad \sum_{j=1}^{N} \epsilon_j \leq C.$$

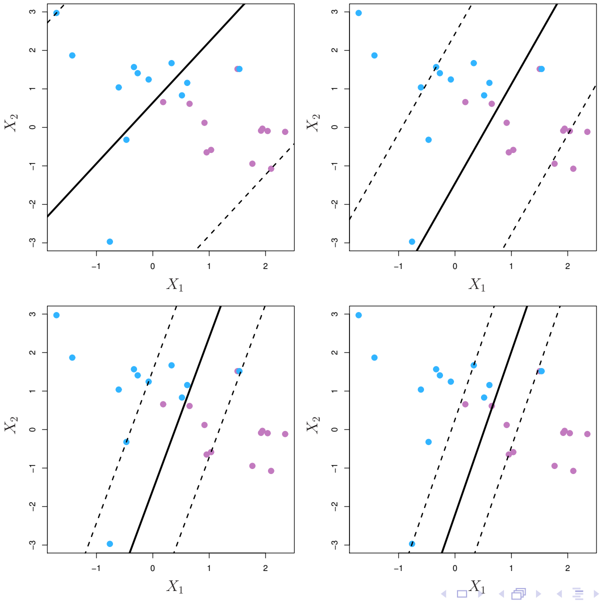
A few points

- The latter problem can be solved by quadratic programming.
- Important: only observations inside the margin affect the hyperplane.
  - Only the boundary matters.
  - SVMs are closer to discriminative classifiers than generative ones.

A few points

- The latter problem can be solved by quadratic programming.
- Important: only observations inside the margin affect the hyperplane.
  - Only the boundary matters.
  - SVMs are closer to discriminative classifiers than generative ones.
- Hyperparameter C is usually set by cross-validation.
  - If C is zero, no relaxation; the larger C, the more training points can be "inside" the margin.

Changing C



Non-linear boundaries

- One can capture non-linear boundaries by enlarging the features, say with

$$X_1, X_1^2, X_2, X_2^2, \ldots, X_n, X_n^2,$$

and perhaps other functions of features.

Non-linear boundaries

- One can capture non-linear boundaries by enlarging the features, say with

$$X_1, X_1^2, X_2, X_2^2, \ldots, X_n, X_n^2,$$

and perhaps other functions of features.

- The structure of SVMs lets one do this efficiently for very large enlarged spaces, using *kernels*.

The basic insight (first part)

- It turns out that SVMs can be learned just by handling inner products

$$x' \cdot x'',$$

where x' and x'' are two observations (note: $a \cdot b = \sum_j a_j b_j$).

The basic insight (second part)

- Suppose we have functions $\phi(X) = [\phi_1(X), \phi_2(X), \ldots, \phi_m(X)]$.

- We would need

$$\phi(x') \cdot \phi(x''),$$

for observations x' and x''.

The basic insight (third part)

- We would need to compute $\phi(x') \cdot \phi(x'')$ for all pairs of observations.

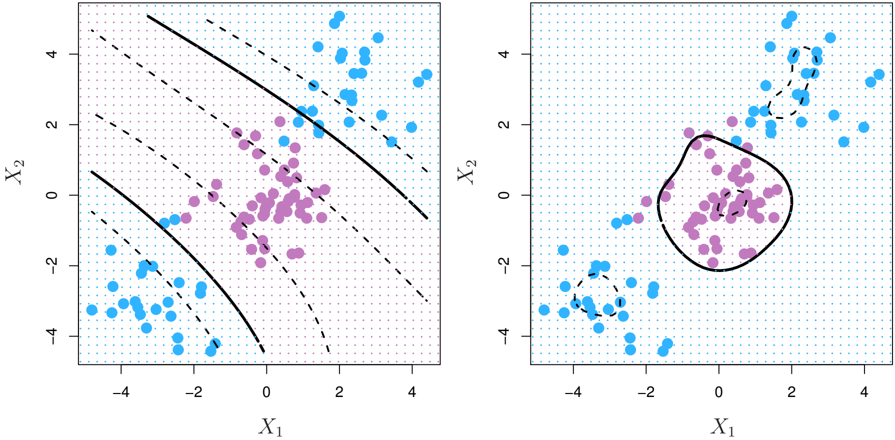
The basic insight (third part)

- We would need to compute $\phi(x') \cdot \phi(x'')$ for all pairs of observations.
- Instead, we can just select a function $K(X', X'')$ and use it whenever a product is needed.
- Such a function is called a *kernel*.

Popular kernels

- Polynomial: $K(X', X'') = (1 + X' \cdot X'')^d$.
- Radial basis: $K(X', X'') = \exp(-\gamma ||X' \cdot X''||)$.
- Neural: $K(X', X'') = \tanh(\gamma_1 (X' \cdot X'') + \gamma_2)$.

Example: Kernels



A final note

Some of the figures in this presentation are taken from *An Introduction to Statistical Learning, with applications in R* (Springer, 2013) with permission from the authors: G. James, D. Witten, T. Hastie and R. Tibshirani.