Chapter 9: Support vector machines

Yu-Tzung Chang and Hsuan-Wei Lee

Department of Political Science, National Taiwan University

2019.01.03.

Outline

- Maximal margin classifier
- Support vector classifiers
- Support vector machines
- SVMs with more than 2 classes
- Relationship to logistic regression

Support vector machines

Here we approach the two-class classification problem in a direct way:

We try and find a plane that separates the classes in feature space.

If we cannot, we get creative in two ways:

- We soften what we mean by "separates".
- We enrich and enlarge the feature space so that separation is possible.

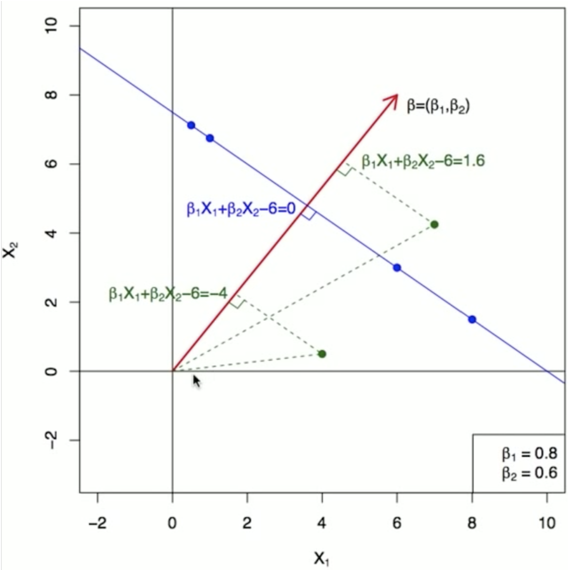
What is a hyperplane?

- A hyperplane in p dimensions is a flat affine subspace of dimension $p - 1$.
- In general, the equation for a hyperplane has the form:

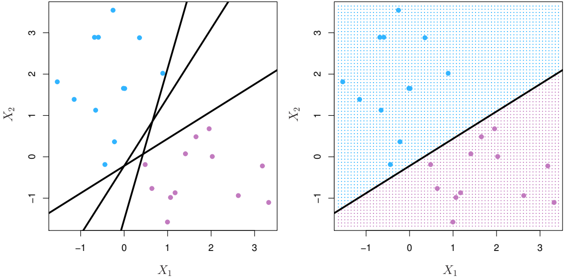
$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p = 0.$$

- In $p = 2$ dimensions a hyperplane is a line.
- If $\beta_0 = 0$, the hyperplane goes through the origin, otherwise not.
- The vector $\beta = (\beta_1, \beta_2, \ldots, \beta_p)$ is called the normal vector – it points in a direction orthogonal to the surface of a hyperplane.

Hyperplane in 2 dimensions



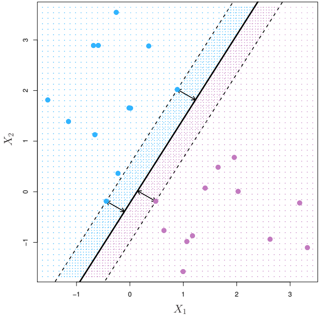
Separating hyperplanes



- If $f(X) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p$, then $f(X) > 0$ for points on one side of the hyperplane, and $f(X) < 0$ for points on the other side.
- If we code the colored points as $Y_i = +1$ for blue, say, and $Y_i = -1$ for mauve, then if $Y_i \cdot f(X_i) > 0$ for all i, $f(X) = 0$ defines a *separating hyperplane*.

Maximal margin classifier

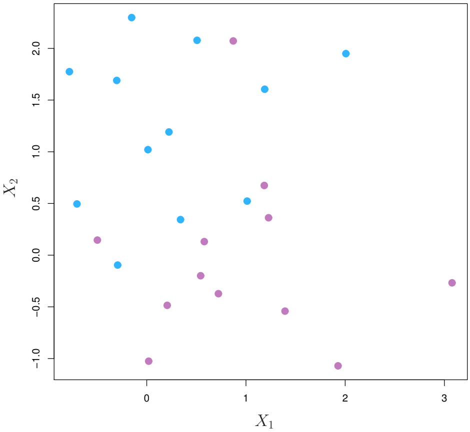
- Among all separating hyperplanes, find the one that makes the biggest gap or margin between the two classes.
- Constrained optimization problem
  $\text{maximize}_{\beta_0,\beta_1,\ldots,\beta_p} M$
  subject to $\sum_{j=1}^{p} \beta_j^2 = 1$,
  $y_i(\beta_0 + \beta_1 X_{is1} + \beta_2 X_{i2} + \cdots + \beta_p X_{ip}) \geqslant M, \ \forall i = 1, \ldots, N.$



- Note: this can be rephrased as a convex quadratic program, and solved efficiently.

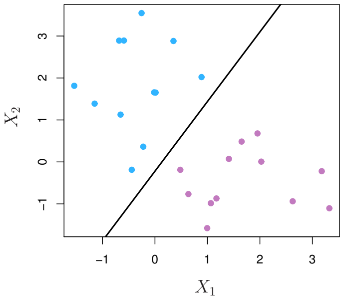
Non-separable data

The data are not separable by a linear boundary. This is often the case unless $N < p$.



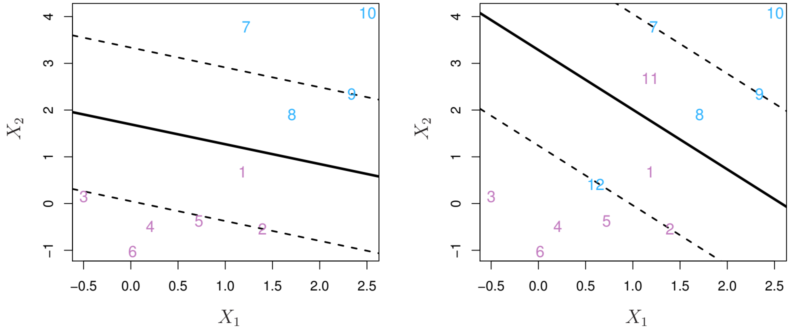
Noisy data

Sometimes the data are separable, but noisy. This can lead to a poor solution for the maximal-margin classifier.



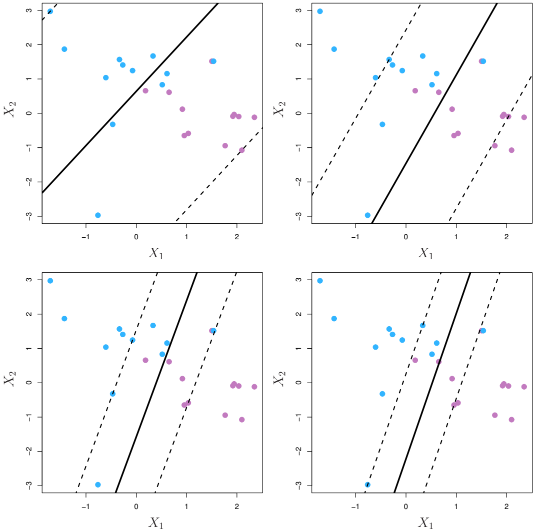
The *support vector classifier* maximizes a *soft* margin.

Support vector classifier

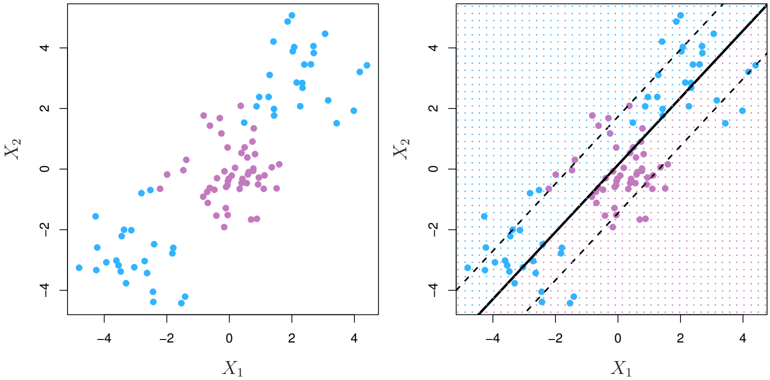


$\text{maximize}_{\beta_0,\beta_1,\ldots,\beta_p,\epsilon_1,\ldots,\epsilon_n} M$ subject to $\sum_{j=1}^{p} \beta_j^2 = 1$,
$y_i(\beta_0 + \beta_1 X_{i1} + \beta_2 X_{i2} + \cdots + \beta_p X_{ip}) \geqslant M(1 - \epsilon_i)$,
$\epsilon_i \geqslant 0, \sum_{i=1}^{n} \epsilon_i \leqslant C$.

C is a regularization parameter



Linear boundary can fail



Sometime a linear boundary simply won't work, no matter what value of C.

Feature expansion

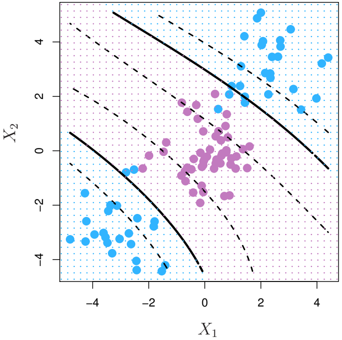
- Enlarge the space of features by including transformations, e.g. $X_1^2, X_1^3, X_1X_2, X_1X_2^2, \ldots$. Hence go from a p-dimensional space to a $N < p_{new}$ dimensional space.
- Fit a support vector classifier in the enlarged space.
- This results in non-linear decision boundaries in the original space.

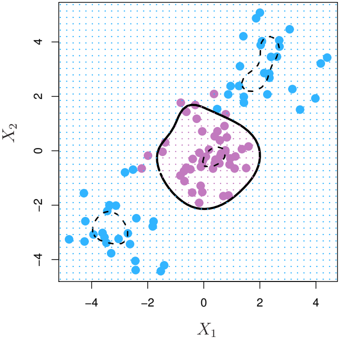
Example: Suppose we use $(X_1, X_2, X_1^2, X_2^2, X_1X_2)$ instead of just (X_1, X_2). Then the decision boundary would be of the form

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_1^2 + \beta_4 X_2^2 + \beta_5 X_1 X_2 = 0$$

This leads to nonlinear decision boundaries in the original space (quadratic conic sections).

Cubic polynomials





Here we use a basis expansion of cubic polynomials. From 2 variables to 9. The support vector classifier in the enlarged space solves the problem in the lower-dimensional space.

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_1^2 + \beta_4 X_2^2 + \beta_5 X_1 X_2$$
$$+ \beta_6 X_1^3 + \beta_7 X_2^2 + \beta_8 X_1 X_2^2 + \beta_9 X_1^2 X_2 = 0$$

Nonlinearities and kernels

- Polynomials (especially high-dimensional ones) get wild rather fast.
- There is a more elegant and controlled way to introduce nonlinearities in support vector classifiers – through the use of *kernels*.
- Need to understand the role of *inner products* in support vector classifiers.

Inner products and support vectors

- $\langle x_i, x_{i'} \rangle = \sum_{j=1}^{p} x_{ij} x_{i'j}$ – *inner products between vectors*
- The linear support vector classifier can be represented as

$$f(x) = \beta_0 + \sum_{i=1}^{n} \alpha_i \langle x, x_i \rangle$$

  with n parameters.

- To estimate the parameters $\alpha_1, \ldots, \alpha_n$ and β_0, all we need are the $\binom{n}{2}$ inner products $\langle x_i, x_{i'} \rangle$ between all pairs of training observations.
- It turns out that most of the $\hat{\alpha}_i$ can be zero:

$$f(x) = \beta_0 + \sum_{i \in S} \hat{\alpha}_i \langle x, x_i \rangle$$

  where S is the *support set* of indices i such that $\hat{\alpha}_i > 0$.

Kernels and support vector machines

- If we can compute inner-products between observations, we can fit a SV classifier. This can be quite abstract.
- Some special *kernel functions* can do this for us. E.g.

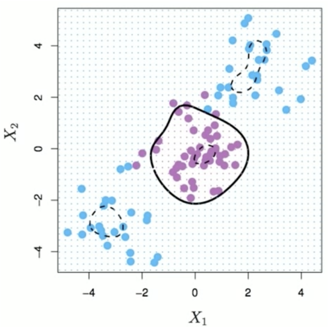
$$K(x_i, x_{i'}) = \left(1 + \sum_{j=1}^{p} x_{ij} x_{i'j}\right)^d$$

  computes the inner-products needed for d dimensional polynomials – $\binom{p+d}{d}$ basis functions.
- The solution has the form

$$f(x) = \beta_0 + \sum_{i \in S} \hat{\alpha}_i K(x, x_i).$$

Radial kernel

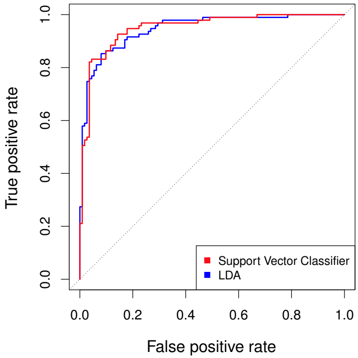
$$K(x_i, x_{i'}) = \exp(-\gamma \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2).$$

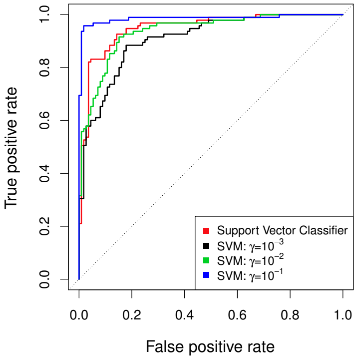


$$f(x) = \beta_0 + \sum_{i \in S} \hat{\alpha}_i K(x, x_i).$$

Implicit feature space; very high dimensional. Controls variances by squashing down most dimensions severely.

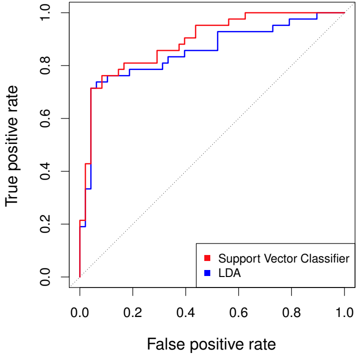
Example: Heart data (training)

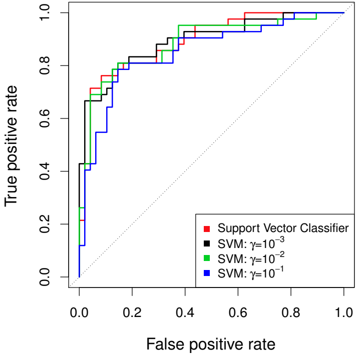




ROC curve is obtained by changing the threshold 0 to threshold t in $\hat{f}(X) > t$, and recording *false positive* and *true positive* rates as t varies. Here we see ROC curves on training data.

Example: Heart data (test)





SVMs: more than 2 classes?

The SVM as defined works for $K = 2$ classes. What do we do if we have $K > 2$ classes?

- *OVA* One versus All. Fit K different 2-class SVM classifiers $\hat{f}_k(x), k = 1, \ldots, K$; each class versus the rest. Classify x^* to the class for which $\hat{f}_k(x^*)$ is largest.
- *OVO* One versus One. Fit all $\binom{K}{2}$ pairwise classifiers $\hat{f}_{kl}(x)$. Classify x^* to the class that wins the most pairwise competitions.

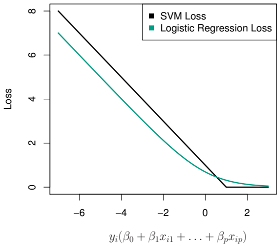
Which one to choose? If K is not too large, use OVO.

Support vector versus logistic regression?

With $f(X) = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p$ can rephrase support vector classifier optimization as

$$\text{minimize}_{\beta_0,\beta_1,\ldots,\beta_p} \left\{ \sum_{i=1}^{n} \max[0, 1 - y_i f(x_i)] + \lambda \sum_{j=1}^{p} \beta_j^2 \right\}$$

This has the form *loss plus penalty*.



The loss is known as the *hinge loss*. It's very similar to "loss" in logistic regression (negative log-likelihood).

Which to use: SVM or logistic regression

- When classes are (nearly) separable, SVM does better than LR. So does LDA.
- When not, LR (with ridge penalty) and SVM very similar.
- If you wish to estimate probabilities, LR is the choice.
- For nonlinear boundaries, kernel SVMs are popular.Can use kernels with LR and LDA as well, but computations are more expensive.