Machine learning with Python

Support Vector Machines

bluemetrica

Outline

1. Maximal Margin classifier
2. Support vector classifiers
3. Support vector machines
4. 2 and more classes SVM's

SVM

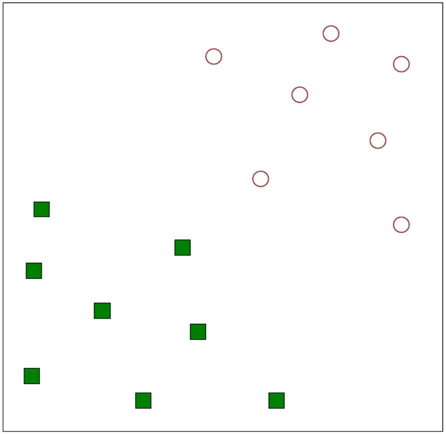
- The goal of a SVM is to create a flat boundary called a hyperplane, which divides the space to create fairly homogeneous partitions on either side
- A Support Vector Machine (SVM) can be imagined as a surface that creates a boundary between points of data plotted in multidimensional space
- SVM's have recently exploded in popularity that is due to excellent performance and implementation of award winning SVM algorithms in popular languages like R.
- SVMs can be adapted for use with nearly any type of learning task, including both classification and numeric prediction SVMs are most easily understood when used for binary classification, therfore we explain SVM's for binary classification problems

Notable applications

- Classification of microarray gene expression data in the field of bioinformatics to identify cancer or other genetic diseases
- Text categorization such as identification of the language used in a document or the classification of documents by subject matter
- The detection of rare yet important events like combustion engine failure, security breaches, or earthquakes

Classification problem

- Try to find a hyperplane that separates two classess



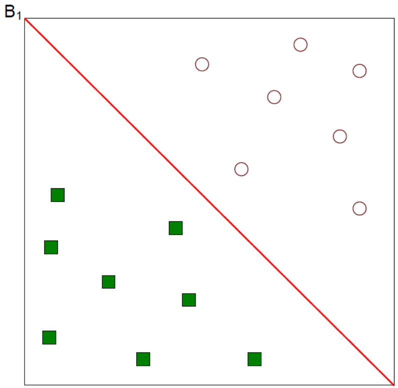
Hyperplane

- A hyperplane in p dimensions is a flat subspace of dimension $p - 1$.
- The general equation for a hyperplane is as following:

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_p X_p = 0 \tag{1}$$

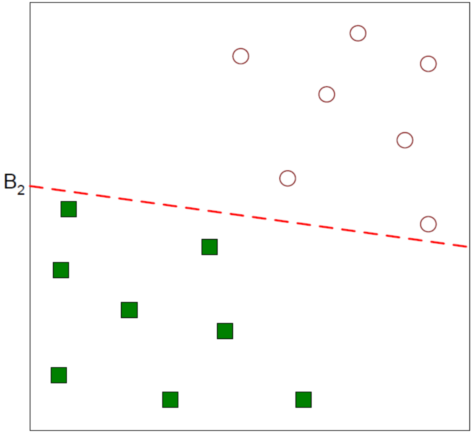
- In the case where $p = 2$ hyperplane is a line
- Sometimes it is impossible to find a plane that separates exactly the classes in feature space. In that case we:
  - soften separation by allowing for some errors
  - enrich and enlarge the feature space

Separating Hyperplanes



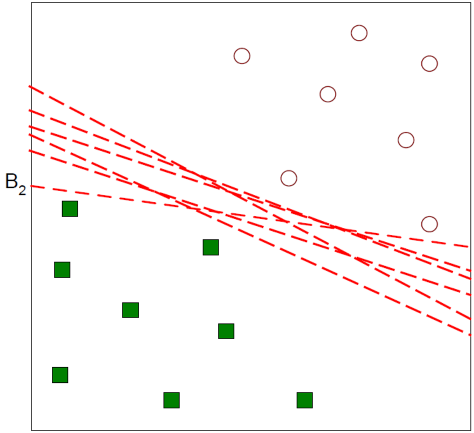
- Assuming $f(X) = \beta_0 + \beta_1 X_1 + \beta_2 X_2$ then $f(X) > 0$ for points on one side of the hyperplane, and $f(X) < 0$ for points on the other.
- Coding circle points as $Y_i = +1$ for white and $Y_i = -1$ for green then $Y_i * f(X_i) > 0$ for all. $f(X) = 0$ defines a separating hyperplane

Separating Hyperplanes



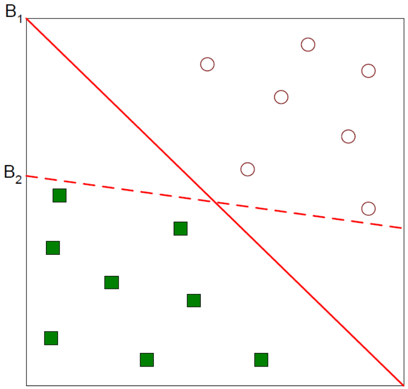
Another possible solution

Separating Hyperplanes



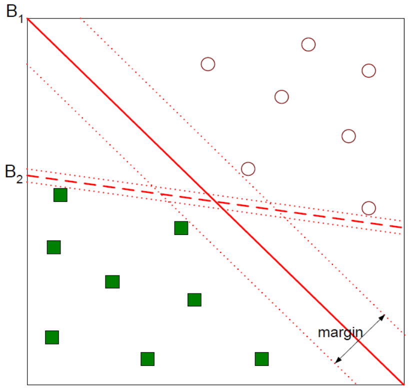
Many possible solutions

Separating Hyperplanes



- Which hyperplane is better? $B1$ or $B2$?
- What does it mean better?

Maximal Margin Classifier



- Better hyperplane is the one that maximizes margin

Maximal Margin Classifier

Construction of the maximal margin hyperplane:

- n training observations $x_1, ..., x_n \in \mathbb{R}^p$
- associated class labels $y_1, ..., y_n \in \{-1, 1\}$
- Maximal margin hyperplane is the solution to the following optimization problem :
  - maximize M
  - changing parameters $\beta_0, \beta_1, ..., \beta_p$
  - subject to:

$$\sum_{j=1}^{p} \beta_j^2 = 1 \tag{2}$$

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \beta_p x_{ip}) \geqslant M \quad \forall i = 1, ..., n \tag{3}$$

Maximal Margin Classifier

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \beta_p x_{ip}) \geqslant M \quad \forall i = 1, ..., n$$

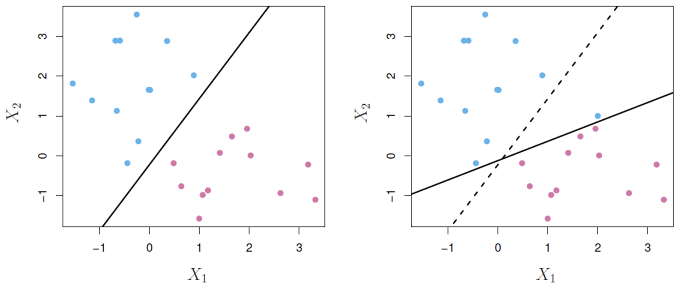
- This constraint guarantees that each observation will be on the correct side of the hyperplane, provided that M is positive.

$$\sum_{j=1}^{p} \beta_j^2 = 1$$

- This constraint is not really a constraint on the hyperplane but it adds meaning to the previous constraint
- With this constraint the perpendicular distance from the ith observation to the hyperplane is given by $y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + ... + \beta_p x_{ip})$.
- Both constraints ensure that each observation is on the correct side of the hyperplane and at least a distance M from the hyperplane.

MMC Issue

- There are situations in which Maximal Margin Classifier lead to undesirable solution



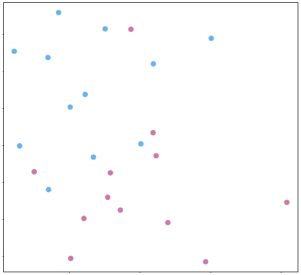
The addition of a single observation in the right-hand panel graph leads to a dramatic change in the maximal margin hyperplane resulting in a small margin

MMC Issue

Why a small margin in MMC is problematic?

- The distance of an observation from the hyperplane can be seen as a measure of our confidence in correct classification - small margin less confident classfication
- Apart from that this kind of great sensitivity to a change in a single observation suggests that it may have overfit the training data.
- To provide greater robustness and better classsification of most of the training observations we consider a classifier based on a hyperplane that does not perfectly separate two classes
- In other words we may accept that some observations might be misclassified for the greater good in apropriate classification of the remaining observations

Non-separable data



The dataset may not be separable by a linear boundary

- In this case the optimization problem for maximal margin classifier has no solution with $M > 0$
- We can extend the concept of a separating hyperplane in order to develop a hyperplane that almost separates the classes, using a so-called soft margin.
- The generalization of the maximal margin classifier to the non-separable case is known as the support vector classifier

Support Vector Classifiers

- The support vector classifier, also called a soft margin classifier allow some observations to be on the incorrect side of the margin, or even the incorrect side of the hyperplane.
- The margin is soft because it can be violated by some of the training observations.
- With soft margin we can provide:
    - Robustness
    - Better overall classification
    - Classification for non separable data

Support Vector Classifier

Construction of the Support Vector hyperplane:

- n training observations $x_1, ..., x_n \in \mathbb{R}^p$
- associated class labels $y_1, ..., y_n \in \{-1, 1\}$
- Support Vector Classifier is the solution to the following optimization problem :
  - maximize M
  - changing parameters $\beta_0, \beta_1, ..., \beta_p, \epsilon_1, ..., \epsilon_n$
  - subject to:

$$\sum_{j=1}^{p} \beta_j^2 = 1 \tag{4}$$

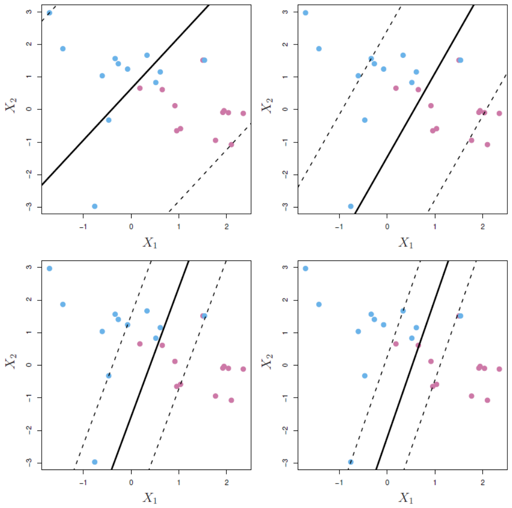
$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \beta_p x_{ip}) \geqslant M(1 - \epsilon_i) \tag{5}$$

$$\epsilon_i \geqslant 0, \qquad \sum_{i=1}^{n} \epsilon_i \leqslant C \tag{6}$$

Support Vector Classifier

- $\epsilon_1, ..., \epsilon_n$ are slack variables that allow observations to be on the wrong side of the margin
- ϵ_i tells us where observation is located,
- $\epsilon_i = 0$ it is on the correct side of the margin
- $\epsilon_i > 0$ wrong side of the margin or $\epsilon_i > 1$ wrong side of the hyperplane
- C is a nonnegative tuning parameter
- C bounds the sum of the ϵ's, and so it determines the number and severity of the violations to the margin
- $C = 0$ - there are no violations
- $C > 0$ - no more than C observations can be on the wrong side of the hyperplane
- In practice C is treated as a tuning paramater that is chosen via cross-validation
- C controls the bias-variance trade off. Small C low bias but high variance

C tuning parameter

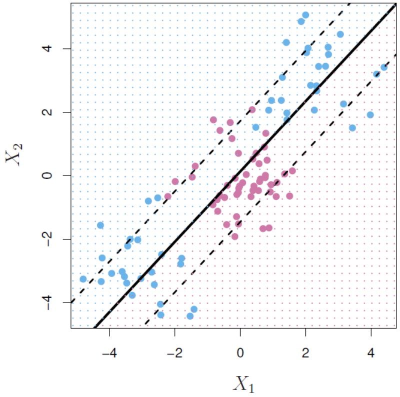


Support vectors

- Only observations that either lie on the margin or that violate the margin will affect the hyperplane
- An observation that lies strictly on the correct side of the margin does not affect the support vector classifier
- Observations that lie directly on the margin, or on the wrong side of the margin for their class, are known as support vectors.
- When the tuning parameter C is large, then the margin is wide, many observations violate the margin, and so there are many support vectors.
- The fact that the support vector classifier's decision rule is based only on a potentially small subset of the training observations means that it is quite robust to the behavior of observations that are far away from the hyperplane

Nonlinear problem

- In some situations linear boundary can fail



Nonlinear decision boundaries

- We could address the problem in the case of the support vector classifier, by enlarging the feature space using quadratic, cubic, and even higher-order polynomial functions of the predictors.
- We can also enlarge the feature space with higher order polynomial terms or with interactions terms
- There are many possible ways to enlarge the feature space
- but computations could become very quickly unmanageable
- There is more elegant and controlled way to introduce nonlinearities in support vector classifiers - kernels

Support Vector Machine

- SVM is an extension of the support vector classifier that results from enlarging the feature space in a specific way, using kernels.
- Idea of SVM is to enlarge our feature space in order to accommodate a non-linear boundary between the classes
- The kernel approach is simply an efficient computational approach for enacting this idea.
- When the support vector classifier is combined with a non-linear kernel the resulting classifier is known as a support vector machine

Support Vector Machine

- Solution to the support vector classifier problem involves only the inner products of the observations
- The inner product of two observations x_i, $x_{i'}$ is given by:

$$\langle x_i, x_{i'} \rangle = \sum_{j=1}^{p} x_{ij} x_{i'j} \tag{7}$$

- The linear support vector classifier can be presented as:

$$f(x) = \beta_0 + \sum_{i=1}^{n} \alpha_i \langle x, x_i \rangle \tag{8}$$

- In order to evaluate the function $f(x)$, we need to compute the inner product between the new point x and each of the training points x_i
- However α_i is nonzero only for the support vectors in the solution
- Assuming S is the collection of indices of these support points, we can rewrite any solution function of the form:

$$f(x) = \beta_0 + \sum_{i \in S} \alpha_i \langle x, x_i \rangle \tag{9}$$

Support Vector Machine

- Suppose that we replace the inner product $\langle x_i, x_{i'} \rangle$ with generalization in the form:

$$K(x_i, x_{i'}) \tag{10}$$

- K - kernel is a function that quantifies the similarity of two observations
- If we assume kernel function to be of the form as below, then it gives us back the support vector classifier:

$$K(x_i, x_{i'}) = \sum_{j=1}^{p} x_{ij} x_{i'j} \tag{11}$$

$$f(x) = \beta_0 + \sum_{i \in S} \alpha_i K(x, x_i) = \beta_0 + \sum_{j=1}^{p} x_{ij} x_{i'j} \tag{12}$$

Non-linear Kernels

- Different from linear kernels lead to a much more flexible decision boundary
- Polynomial kernel of degree d adds a simple nonlinear transformation of the data:

$$K(x_i, x_{i'}) = (1 + \sum_{j=1}^{p} x_{ij}x_{i'j})^d \tag{13}$$

  - Polynomial kernel fits a support vector classifier in a higher-dimensional space involving polynomials of degree d, rather than in the original feature space
- Radial kernel takes the following form

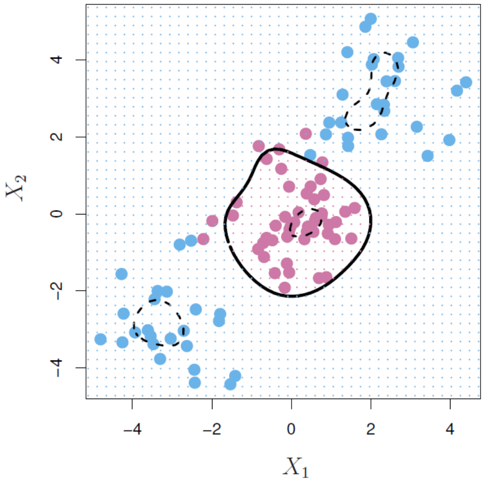
$$K(x_i, x_{i'}) = exp(-\gamma \sum_{j=1}^{p} (x_{ij}x_{i'j})^2) \tag{14}$$

Advantages of kernels

Why we do not simply enlarge the feature space using functions of the original features?

- Computational - using kernels, one need only compute $K(x_i, x_{i'})$ for all distinct pairs i, i'. This can be done without explicitly working in the enlarged feature space
- For some cases enlarged feature space is so large that computations are intractable
- For some kernels, such as the radial kernel, the feature space is implicit and infinite-dimensional, so we could never do the computations
- There are many other possible non-linear kernels

Radial Kernel



Kernel choice

- There is no reliable rule to match a kernel to a particular learning task.
- The fit depends heavily on the concept to be learned as well as the amount of training data and the relationships among the features.
- A bit of trial and error is required by training and evaluating several SVMs on a validation dataset.
- In many cases, the choice of kernel is arbitrary, as the performance may vary slightly.

SVM Strenghts

SVMs with nonlinear kernels are extremely powerful classifiers:

- Can be used for classification or numeric prediction problems
- Not overly influenced by noisy data and not very prone to overfitting
- May be easier to use than neural networks, particularly due to the existence of several well-supported SVM algorithms
- Gaining popularity due to its high accuracy and high-profile in winning data mining competitions

SVM Weaknesses

Like all methods SVMs have some downsides:

- Finding the best model requires testing of various combinations of kernels and model parameters
- Can be slow to train, particularly if the input dataset has a large number of features or examples
- Results in a complex black box model that is difficult, if not impossible, to interpret