View on GitHub

stats-learning-notes

Notes from Introduction to Statistical Learning

Previous: Chapter 8 - Tree-Based Methods

Chapter 9 - Support Vector Machines

Support vector machines (SVMs) are a generalization of a simple and intuitive classifier called the maximal margin classifier. Support vector machines improve upon maximal margin classifiers by utilizing a support vector classifier which overcomes a limitation of the maximal margin classifier which requires that classes must be separable by a linear boundary. The use of the support vector classifier allows support vector machines to be applied to a wider range of cases than the maximal margin classifier. Support vector machines extend the support vector classifier to accommodate non-linear class boundaries.

Support vector machines are intended for the binary classification setting in which there are two classes, but can be extended to handle more than two classes.

Maximal Margin Classifier

In a p-dimensional space, a hyperplane is a flat affine subspace of dimension $p - 1$. For example, in two dimensions, a hyperplane is a flat one-dimensional subspace, or in other words, a line. In three dimensions, a hyperplane is a plane.

The word affine indicates that the subspace need not pass through the origin.

A p-dimensional hyperplane is defined as

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

which means that any $X = (X_1,\ X_2,\ \ldots,\ X_p)^T$ for which the above hyperplane equation holds is a point on the hyperplane.

If $X = (X_1,\ X_2,\ \ldots,\ X_p)^T$ doesn't fall on the hyperplane, then it must fall on one side of the hyperplane or the other. As such, a hyperplane can be thought of as dividing a p-dimensional space into two partitions. Which side of the hyperplane a point falls on can be computed by calculating the sign of the result of plugging the point into the hyperplane equation.

Classification Using a Separating Hyperplane

Consider an $n \times p$ data matrix X that consists of n training observations in p-dimensional space,

$$X_1 = \begin{pmatrix} X_{11} \\ \vdots \\ X_{1p} \end{pmatrix}, \ldots, X_n = \begin{pmatrix} X_{n1} \\ \vdots \\ X_{np} \end{pmatrix}$$

where each of these observations fall into two classes, or $y_1, \ldots, y_n \in -1, 1$ where -1 represents one class and 1 represents the other. Also available is a test observation constituted of a p-vector of observed features, $x^* = (x_1^*, x_2^*, \ldots, x_p^*)$.

Suppose that it is possible to construct a hyperplane that separates the training observations perfectly according to their class labels.

Such a hyperplane would have a property that

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ \ldots\ + \beta_p X_p < 0 \text{ if } y_i = -1$$

and

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ \ldots\ + \beta_p X_p > 0 \text{ if } y_i = 1.$$

More concisely, a separating hyperplane would have the property that

$$y_i(\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ \ldots\ + \beta_p X_p) > 0 \text{ for all } i = 1, \ \ldots, \ n.$$

If a separating hyperplane exists it can be used to construct a classifier that assigns a test observation to a class depending on which side of the hyperplane the observation is located.

That is, a test observation is classified based on the sign of

$$f(x^*) = \beta_0 + \beta_1 x_1^* + \beta_2 x_2^* + \ \ldots\ + \beta_p x_p^*.$$

If $f(x^*)$ is positive then the test observation is assigned to class 1. If $f(x^*)$ is negative then the test observation is assigned to class -1. Additionally, the magnitude of $f(x^*)$ can be used as a measure of confidence in the class assignment for x_*. If $f(x^*)$ is far from zero, it is far from the hyperplane and more confidence in the classification can be had. Conversely, If $f(x^*)$ is near to zero then the classification is less certain.

A classifier based on a separating hyperplane has a linear decision boundary.

The Maximal Margin Classifier

If the data can be perfectly separated using a hyperplane, then there will in fact exist an infinite number of such hyperplanes. In order to construct a classifier based on a separating hyperplane it is necessary to decide which of the infinite possible separating hyperplanes to use.

The maximal margin hyperplane (also known as the optimal separating hyperplane) is the separating hyperplane which is farthest from the training observations in terms of perpendicular distance. The minimal distance from the observations to the hyperplane is known as the margin. The maximal margin hyperplane is the hyperplane that has the farthest minimum distance to the training observations. When a maximal margin hyperplane is used to classify test observations it is known as a maximal margin classifier. The maximal margin classifier is often successful, but can overfit when p is large.

The maximal margin classifier classifies a test observation x^* based on the sign of

$$f(x^*) = \beta_0 + \beta_1 x_1^* + \ \ldots\ + \beta_p x_p^*$$

where $\beta_0, \ \beta_1, \ \ldots, \ \beta_p$ are the coefficients of the maximal margin hyperplane.

The maximal margin hyperplane represents the mid-line of the widest gap between the two classes.

Though it might seem that the maximal margin hyperplane depends on all of the training observations, there are actually relatively few training observations that affect the maximal margin hyperplane. Those observations which are constituted of p-dimensional vectors, are those observations that would cause the maximal margin hyperplane to move if they were moved in some dimension. These observations are known as support vectors since, in a sense, they support the maximal margin hyperplane and give it its shape.

The maximal margin hyperplane is the solution to the optimization problem of maximizing $M_{\beta_0, \beta_1, \ldots, \beta_p}$ such that

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

and

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ \ldots \ + \beta_p x_{ip}) \geq M \, \forall \, i = 1, \ i = 2, \ \ldots, \ i = n.$$

The constraint

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ \ldots \ + \beta_p x_{ip}) \geq M \, \forall \, i = 1, \ i = 2, \ \ldots, \ i = n$$

guarantees each observation will be on the correct side of the hyperplane, assuming M is positive. More specifically, ensuring each observation is on the correct side of the hyperplane actually entails a looser constraint:

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ \ldots \ + \beta_p x_{ip}) > 0$$

however, the maximal margin hyperplane aims to maximize the cushion between the hyperplane and the observations.

It is worth noting that

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ \ldots \ + \beta_p x_{ip})$$

doesn't really constrain the hyperplane since if $\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ \ldots \ + \beta_p x_{ip} = 0$ defines a hyperplane, then so does the same formula multiplied by a non-zero scalar value. But, combined with the constraint that

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

the constraints ensure that the perpendicular distance from the ith observation to the hyperplane is given by

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ \ldots \ + \beta_p x_{ip}).$$

All together, the optimization problem chooses $\beta_0, \beta_1, \ \ldots, \ \beta_p$ to maximize M while ensuring that each observation is on the right side of the hyperplane and at least distance M from the hyperplane, yielding a maximal margin hyperplane.

If no separating hyperplane exists, no maximal margin hyperplane exists either. However, a soft margin can be used to construct a hyperplane that almost separates the classes. This generalization of the maximal margin classifier is known as the support vector classifier.

In general, a perfectly separating hyperplane can be undesirable because it can be very sensitive to individual observations. This sensitivity can also be an indication of overfitting.

A classifier based on a hyperplane that doesn't perfectly separate the two classes can offer greater robustness to variations in individual observations and better classification of most training observations at the cost of misclassifying a few training observations.

The support vector classifier, sometimes called a soft margin classifier, allows some observations to fall on both the wrong side of the margin and the wrong side of the hyperplane.

This flexibility allows the support vector classifier to utilize a hyperplane that solves the optimization problem of maximizing $M_{\beta_0, \beta_1, \ldots, \beta_p}$ subject to

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

and

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip}) > M(1 - \epsilon_i),$$

where $\epsilon_i \geq 0$ and $\sum_{i=1}^{n} \epsilon_i \leq C$ where C is a non-negative tuning parameter.

Like the maximal margin classifier, M is the width of the margin and the focus of the optimization. $\epsilon_1, \ldots, \epsilon_M$ are slack variables that allow individual variables to fall on the wrong side of the margin and/or hyperplane. As with the maximal margin classifier, once an optimal solution has been found, a test observation can be classified based on the sign of

$$f(x^*) = \beta_0 + \beta_1 x_1^* + \beta_2 x_2^* + \ldots + \beta_p x_p^*.$$

The slack variables, $\epsilon_i, \ldots, \epsilon_n$ encode the location of the ith observation relative to the hyperplane and relative to the margin. When $\epsilon_i = 0,$ the ith observation is on the correct side of the margin. When ϵ_i is greater than zero, the ith observation is on the wrong side of the margin and is said to have violated the margin. When ϵ_i is greater than 1, the observation is on the wrong side of the hyperplane.

The tuning parameter C limits the sum of the slack variables, $\epsilon_i, \ldots, \epsilon_n,$ and so determines the number and severity of the violations to the margin and hyperplane that will be tolerated. When C is zero, no budget is available for violations, which means $\epsilon_i = 0, \ldots, \epsilon_n = 0,$ in which case the solution, if one exists, is the same as the maximal margin classifier. When C is greater than zero, no more than C observations may be on the wrong side of the hyperplane since ϵ_i will be greater than zero and $\sum_{i=1}^{n} \epsilon_i \leq C.$ As C increases, the margin gets wider and more tolerant of violation. As C decreases, the margin gets narrower. Like other tuning parameters, $C,$ is generally selected using cross validation.

Similar to other tuning parameters, C also controls the bias-variance trade-off of the statistical learning model. When C is small, margins will be narrow and rarely violated which amounts to a classifier that is highly fit to the data with low bias, but high variance. Conversely, when C is large, the margin is large and more frequently violated which amounts to a classifier that is more loosely fit with potentially higher bias, and potentially lower variance. Interestingly, only the observations that lie on the margin or violate the margin affect the hyperplane and classifier obtained. Observations directly on the margin or on the wrong side of the margin for their class are known as support vectors since the observations do affect the shape of the support vector classifier. This behavior is the reason why the tuning parameter C controls the bias-variance trade-off of the support vector classifier. When C is large, the margin is wide and more frequently violated which means many support vectors contribute to shaping the hyperplane. This results in low variance but potentially high bias. Conversely, when C is small there will be fewer support vectors and the resulting classifier will have low bias, but high variance.

Because the support vector classifier is based on only a small subset of the training observations (the support vectors), it is robust to the behavior of those observations that are far from the hyperplane.

When class boundaries are non-linear, the feature space can be enlarged using functions of the predictors such as quadratic, cubic, or interaction terms to address the non-linearity. However, computations can become unmanageable in the support vector classifier case and because of this, support vector machines were introduced to allow for enlarging the feature space used by the support vector classifier in a way that leads to efficient calculations.

The support vector machine is an extension of the support vector classifier that results from enlarging the feature space using kernels to accommodate a non-linear boundary between classes.

The exact details of how the support vector classifier is computed are somewhat technical, but it turns out that the solution to the support vector classifier involves only the inner-products of the observations, not the observations themselves. The inner-product of two p-vectors a and b is defined as $\langle a, b \rangle = \sum_{i=1}^{r} a_i b_i$. As such, the inner product of two observations x_i and $x_{i'}$ is given by

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

It can be shown that the linear support vector can be represented as

$$f(x) = \beta_0 + \sum \alpha_i \langle x_i, x_{i'} \rangle$$

where there are n parameters, $\alpha_1, \ldots, \alpha_n$, one per training observation.

To estimate the parameters $\alpha_1, \ldots, \alpha_n$ and β_0 requires the $\binom{n}{2}$ inner products, $\langle x_i, x_{i'} \rangle$ between all pairs of training observations. The notation $\binom{n}{2}$ means $\frac{n(n-1)}{2}$ and gives the number of pairs among a set of n items.

Though it might seem like it is necessary to compute the inner product between a new point x and each of the training observations x_i to evaluate $f(x)$, as it turns out, α_i is zero for all observations except for the support vectors, so only the inner product of the support points need to be calculated.

Given a collection of the indices of the support points, the linear support classifier can be represented by

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

which generally requires far less computational resources.

Going further, it is possible to generalize the support vector classifier by replacing the inner product operations with generalizations of the inner products in the form of

$$K(x_i, x_{i'})$$

where K is some function known as a kernel. A kernel is a function that quantifies the similarity of two observations. For example, a kernel of

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

would be equivalent to the support vector classifier. This kernel is known as a linear kernel because it is linear in the features. In this case, the linear kernel quantifies the similarity of a pair of observations using Pearson (standard) correlation.

Another popular kernel function is

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

where d is a positive integer. This kernel is known as a polynomial kernel of degree d. A polynomial kernel leads to a much more flexible decision boundary than using a linear kernel. It essentially amounts to fitting a support vector classifier in a higher dimensional space involving polynomials of degree d, instead of the original feature space.

When the support vector classifier is combined with a non-linear kernel, the resulting classifier is known as a support vector machine. The non-linear function underlying the support vector machine has the form

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

Another popular non-linear kernel is the radial kernel given by

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

wher γ is a positive constant. The radial kernel has very local behavior, in that only nearby training observations affect the classification of a test observation. This is because if a given test observation, $x^* = (x_1^*, \ldots, x_p^*)^T$ is far from a training observation x_i in terms of Euclidean distance, then $\sum_{j=1}^{p}(x_j^* - x_{ij})^2$ will be large and thus $\exp(-\gamma \sum_{j=1}^{p}(x_{ij} - x_{i'j})^2)$ will be very tiny and as such x_i will have virtually no effect on $f(x^*)$.

Using a kernel has a computational advantage over simply enlarging the feature space using functions of the original features. Using kernels, it is only necessary to compute $K(x_i, x_{i'})$ for $\binom{n}{2}$ distinct i, i'. In some cases, an enlarged feature space can be implicit and of infinite dimension, kernels allow for working solutions in these cases.

Support vector machines don't extend well to more than two classes, but two popular approaches for extending SVMs to the K-class case are one-versus-one and one-versus-all.

Assuming $K > 2$, one-versus-one, or all-pairs, constructs $\binom{K}{2}$ SVMs, each of which compares a pair of classes. A test observation would be classified using each of the $\binom{K}{2}$ classifiers, with the final classification given by the class most frequently predicted by the $\binom{K}{2}$ classifiers.

Assuming $K > 2$, one-versus-all fits K SVMs, each time comparing one of the K classes to the remaining $K - 1$ classes. Assuming a test observation x^* and coefficients $\beta_{0k}, \beta_{1k}, \ldots, \beta_{pk}$, resulting from fitting an SVM comparing the kth class (coded as $+1$) to the others (coded as -1), the test observation is assigned to the class for which

$$\beta_{0k} + \beta_{1k}X_1^* + \ldots + \beta_{pk}X_p^*$$

is largest, as this amounts to the highest level of confidence.

As a final note, it is important to standardize variables when using the maximal margin classifier, the support vector classifier, and the support vector machine.

Next: Chapter 10 - Unsupervised Learning

stats-learning-notes maintained by tdg5

Published with GitHub Pages