Machine Learning and Big Data Analytics Section 8

Emily Mower
November 2018

Note that the material in these notes draws on the excellent and more thorough treatment of these topics in Introduction to Statistical Learning by Gareth James, Daniela Witten, Trevor Hastie and Robert Tibshirani.

1 Support Vector Machines

In this class, we will see a series of methods that fall under the support vector umbrella but which vary significantly in their flexibility. We start with maximal margin classifiers, which are the most rigid and only work when the data can be perfectly separated with a linear decision boundary. Then, we will cover support vector classifiers, which still uses a linear decision boundary, but which can accommodate data that cannot be perfectly separated by such a decision boundary. We will then cover support vector machines, which flexibly transform the original data to allow for decision boundaries that are non-linear in the original feature space (though they remain linear in the original feature space). Like support vector classifiers, support vector machines will also accommodate data that is not perfectly separable. Finally, we will touch upon extending the support vector ideas to multiclass settings.

1.1 Maximal Margin Classifier

Maximal margin classifiers use a separating hyperplane to divide the feature space in two, with the idea being that all observations in one class lie on one side of the separating hyperplane while all observations of the other class lie on the other side.

1.1.1 Separating Hyperplanes

A hyperplane is a flat affine subspace that has one fewer dimensions than the feature space. The first part of the definition (“flat affine subspace”) means that the subspace can be described by a linear equation and does not need to pass through the origin. The second part of the definition (about the dimensionality) means that with \( p \) features, the hyperplane will have \( p - 1 \) dimensions.

When we say "separating hyperplane," we mean a hyperplane that separates the observations in one class from observations in the other by slicing the feature space in two.

Imagine a dataset with only two features, \( X_1 \) and \( X_2 \). The feature space is a plane, which can be divided by a line (which is a hyperplane). Suppose you had three features, \( X_1, X_2, \) and \( X_3 \). The feature space is 3D and can be divided in two by a plane (which is also a hyperplane). We can continue to generalize to higher dimensional feature spaces.

The separating hyperplane that divides the feature space with \( p \) features can be described by a linear function of the following form

\[
\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_p x_p = 0
\]

1.1.2 Classifications with Separating Hyperplanes

We can then use this equation as our classifier. Specifically, let \( f(x) \) represent the left hand side of the previous equation. To identify the class for observation \( x^* \), you first calculate \( f(x^*) \)

\[
f(x^*) = \beta_0 + \beta_1 x_{1^*} + \beta_2 x_{2^*} + \ldots + \beta_p x_{p^*}
\]

If the resulting value is negative, the point lies to one side of the hyperplane and is assigned to class \( y = -1 \). If it’s positive, the point lies on the other side of the hyperplane and is assigned...
to class $y = +1$. If the point is far from the separating hyperplane, we are quite confident in our classification. If it’s close, we have much more uncertainty. In this sense, the sign of $f(x^\star)$ gives us the class and the magnitude of $f(x^\star)$ gives us our level of confidence. Given the linear form of the separating hyperplane, the decision boundary will also be linear.

Note that we are now using $+1$ and $-1$ for our binary classes, whereas up to this point we have used 0 and 1.

1.1.3 Maximal Margin Classifier

The maximal margin classifier can be used when the data is perfectly separable, meaning that there exists a separating hyperplane such that all observations from one class fall on one side of the hyperplane and all observations from the other class fall on the other side of the hyperplane. However, when one such separating hyperplane exists, usually infinite such separating hyperplanes exist. The maximal margin classifier provides a disciplined way to choose among them.

Specifically, it measures the distance between each point and the separating hyperplane. The minimum distance between all points and the separating hyperplane is referred to as the margin. The maximal margin classifier selects that separating hyperplane that leads to the largest possible margin, that is, the largest possible distance between the separating hyperplane and the closest training points. Note that as you increase the distance between the decision boundary and the closest point from one class, you must decrease the distance between the decision boundary and the closest point(s) from the other class. Therefore, the margin will always be defined by at least one point from each class. These points are called “support” points or vectors and the model is fully defined by them, so if they move the model changes. Other points can move however they like outside the margins, and as long as they do not cross the margins, the model will not change.

When the margin is large, it suggests the classes are very well separated and performance on a test set should be good. When the margin is small, the classes are only barely separated and the exact decision boundary may not do as well in the test set.

The book uses the imagery of a slab to describe the decision boundary and margins – imagine the widest possible slab you can fit between the classes (this slab is straight and has even width throughout). The decision boundary is the midline of this slab. The margins (one on each side of the decision boundary) are the surfaces of the slab and run parallel to the decision boundary at equal distances, where the distance (margin) is determined by the distance of the closest training points to the decision boundary.

1.2 Support Vector Classifier

Sometimes the data cannot be perfectly separated by a separating hyperplane. In such cases, the support vector classifier offers a generalization of the maximal margin classifier.

The support vector classifier is extremely similar to the maximal margin classifier, except that it allows some points to cross the margin and even the decision boundary (e.g. be misclassified). Because some points can cross the margin under the support vector classifier, we call the margin a “soft margin” compared to the “hard margin” of the maximal margin classifier, which does not permit any points to cross it.

The support vector classifier improves empirically relative to the maximal margin classifier in three key ways. One, it works for data that cannot be perfectly separated by a hyperplane. Two, it is more robust to individual points, meaning that because it allows points to cross the margin, movement in points near the margin will not have the same dramatic impact on the margin and decision boundary that they would have had under the maximal margin classifier. Third, related to this point, the support vector classifier yields better predictions for most training observations.

The support vector classifier works by creating slack variables $\epsilon_i$ and allowing a misclassification “budget” $C$. The slack variables $\epsilon_i$ will be zero for all observations that fall on the correct side of the margin. If the observation is on the wrong side of the margin, $\epsilon_i > 0$ and if it is on the wrong side of the hyperplane, $\epsilon_i > 1$. The sum of the $\epsilon_i$'s must be no greater than the budget $C$, so when the budget $C = 0$, no observations will be allowed on the wrong side of the margin. In this way, the maximal margin classifier is a special case of the support vector classifier. More generally, no more than $C$ observations can be misclassified, because $\epsilon_i > 1$ when the observation is misclassified and $C$ is the sum of the $\epsilon_i$'s.

The budget $C$ can be seen as a tuning parameter and is thus generally found through cross-validation. As $C$ goes to zero, the classifier converges to the maximal margin classifier, so it is less
tolerant of violations across the margin and thus the margins will shrink. As $C$ gets large, it will become more tolerant of violations and the margin will increase.

In the support vector classifier, only observations that lie on the margin or that violate the margin will affect the hyperplane. Just as with the maximal margin classifier, points that lie on the correct side of the margin will not be used to define the hyperplane. As $C$ increases, there are more violations and so more support vectors compared to when $C$ is small. Given that there are more support vectors used when $C$ is large, large $C$ leads to lower variance though higher bias compared to small $C$ (which uses a small number of support vectors). Intuitively, this is because the model is not as sensitive to the exact training points since more of them are used in defining the hyperplane when $C$ is large.

One nice property of the support vector classifier compared to some other classification methods we have seen this semester, such as LDA and QDA, is that the model is not sensitive to points that lie far away from the decision boundary, as long as they are on the correct side. Logistic regression also shares this attractive property.

1.3 Support Vector Machine

The support vector machine extends the support vector classifier to the case where the decision boundary is non-linear. It is somewhat analogous to using polynomial regression when the linearity assumption of linear regression does not hold. Specifically, it works with a transformed feature space and finds a decision boundary that is linear in the transformed space, but which is non-linear in the original space.

However, the way in which the support vector machine transforms the original space is new. It does the transformation using a kernel $K(x_i, x_i')$, which is a generalization of the inner product $<x_i, x_i'>$ (e.g. dot product). The hyperplane is then defined by

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

The values of $\alpha_i$ are only non-zero for support vectors (points that lie on or across the margin). Some popular kernels include the polynomial kernel and the radial kernel. Note that the support vector classifier is a special case of the support vector machine where the kernel is a polynomial kernel of degree $d = 1$. Just like the support vector classifier, it maintains the principle of a budget $C$, though now we use $\alpha_i$ instead of $\epsilon_i$.

The argument for why we use a kernel rather than an enlarged feature space has to do with computational efficiency. Using the kernel only requires computing the kernel (the generalized inner product) for each unique pair $(x_i, x_i')$ in the training data; it does not require explicitly working in a transformed feature space, which may be computationally intractable. For example, suppose we wanted to expand our feature space from $x_1$ and $x_2$ to also include $x_3 = x_1^2 + x_2^2$. Instead of explicitly calculating $x_3$, we only need to adjust the inner product between two points $a$ and $b$ from

$$K(a, b) = x_{1,a}x_{1,b} + x_{2,a}x_{2,b}$$

to

$$K(a, b) = x_{1,a}x_{1,b} + x_{2,a}x_{2,b} + (x_{1,a}^2 + x_{1,b}^2)(x_{2,a}^2 + x_{2,b}^2)$$

The kernel function can take many forms, including polynomial and radial functions. Different kernel functions allow for different levels of model flexibility. However, note that as with many other models we’ve seen this semester, the more flexible we make the kernel, the more likely we are to fit the training data well (low bias) but risk overfitting the training data (high variance). Therefore, in the support vector machine, a very flexible kernel function leads to low bias but high variance.

1.4 Extensions to Multiclass Problems

The idea of support vector machines does not generalize easily to the multiclass setting, but two options have been proposed.
One is called the \textit{one-versus-one} approach, where a collection of models is built that each evaluate the question of whether the observation belongs to class \textit{a} or class \textit{b}. This is repeated with all possible pairs of classes in the data. For a given test point, you tally how many times the observation is assigned to each of the \( K \) classes and assign whichever class was assigned most often.

The other is called the \textit{one-versus-all} approach, where \( K \) models are built and each model compares the class at hand to a collection of the other \( K - 1 \) classes, coded collectively as -1. For a given test point, you determine which \( f_k(x) \) is largest (most confidence) and assign that class.