CS142: Machine Learning Spring 2017

Lecture 10

Instructor: Pedro Felzenszwalb *Scribe: Tyler Dae Devlin*

Large Margin Separators

In the last lecture we proved that the perceptron learning algorithm is guaranteed to find a linear separator (if such a separator exists). But every data set that is linearly separable with a nonzero margin admits an infinite number of linear decision boundaries that each attain a misclassification rate of 0. The perceptron is assured to find one of these, but there is no guarantee as to how "good" this separator will be. This motivates the development of algorithms that aim to find a separator that has as large a margin as possible.

The margin is defined to be the minimum distance between the data and the decision boundary. If we let x_+ denote the training point that closest to the boundary on the $+1$ side, then the distance between x_+ and the boundary is given by

$$d_+ = \left(\frac{w}{\|w\|}\right)^T x_+.$$

Similarly, if x_- is the closest negatively classified point, we have

$$d_- = -\left(\frac{w}{\|w\}}\right)^T x_-.$$

Suppose T is a separable data set of n examples. We want to find the vector w that *maximizes* the margin

$$\min_{i\in\{1,\ldots,n\}} \left[y_i \left(\frac{w}{\|w\|}\right)^T x_i \right].$$

Consider the set of separators parametrized by vectors w such that for all $i \in \{1,\ldots,n\}$, $y_i w^T x_i > 0$. This is precisely the set of separators that perfectly separate the data with nonzero margin. We can think of the inequalities $y_i w^T x_i > 0$ as a set of linear constraints on w. These constraints are equivalent to

$$y_i w^T x_i \geq \epsilon$$

for some $\epsilon > 0$. We can always scale w so that the above is equivalent to

$$y_i w^T x_i \geq 1.$$

The solution set is

$$S = \{w : y_i w^T x_i \geq 1 \text{ for all } i\}.$$

If $y_i w^T x_i = 1$ for some i, the distance between x_i and the margin is

$$d_i = y_i \left(\frac{w}{\|w\|} \right)^T x_i = \frac{1}{\|w\|}.$$

If, on the other hand, $y_i w^T x_i > 1$, then

$$d_i > \frac{1}{\|w\|}.$$

Note that if our separator is such that the closest point is more than a distance of $1/\|w\|$ away from the boundary, we can always adjust w to decrease that distance. So $1/\|w\|$ is precisely the margin. To maximize the margin, we minimize $\|w\|$ subject to the constraints. Equivalently, we can minimize the squared norm $\|w\|^2$. In summary, we are faced with the following optimization problem:

$$\begin{array}{rl} \text{minimize} & \|w\|^2 \\ \text{subject to} & y_i w^T x_i \geq 1 \text{ for all } i. \end{array}$$

So far we have been assuming that we are given a data set that is perfectly separable. If this is not the case, then the solution set S is empty and the optimization is infeasible. To handle this case, we relax our constraints to require instead that $y_i w^T x_i \geq 1 - \epsilon_i$ for some $\epsilon_i \geq 0$. Of course, we want to keep the ϵ_i terms small, so we reformulate our optimization problem as follows:

$$\begin{array}{rl} \text{minimize} & \|w\|^2 + C \sum_{i=1}^{n} \epsilon_i \\ \text{subject to} & y_i w^T x_i \geq 1 - \epsilon_i. \end{array}$$

The parameter $C \in \mathbb{R}^+$ determines the tradeoff between minimizing error and maximizing the margin; choosing C is an empirical matter. We can solve this optimization problem via quadratic programming.

Note that this is a constrained minimization over a very high-dimensional space: there is one variable for each point in the dataset, plus the D components of w.

Unconstrained optimization

Fixing w, we have $\epsilon_i = \max\{0, 1 - y_i w^T x_i\}$. This equation ensures that ϵ_i is the smallest possible value that satisfies the ith constraint. Now choose

$$w^* = \underset{w}{\operatorname{argmin}} \|w\|^2 + C \sum_i \max\{0, 1 - y_i w^T x_i\}. \tag{1}$$

This is an unconstrained optimization problem in $\mathbb{R}^D$ that is equivalent to the previous constrained optimization in $\mathbb{R}^{D+N}$. There are two advantages to this reformulation. First, unconstrained optimization problems are generally easier than their constrained counterparts. Second, the dimensionality of the solution space is vastly has vastly decreased. Note, however, that we can no longer use quadratic programming to solve this optimization problem.

Support vector machines

The *support vectors* are those x_i that satisfy the constraint $y_i w^T x_i \geq 1 - \epsilon_i$ with equality. If you remove all the training data points other than the support vectors, the solution remains unchanged. Thus, these x_i can be thought of as "supporting" or "holding up" the separator. Support vector machines minimize a sum of $\|w\|^2$ and the so-called *hinge loss* defined in general by $H(t) = \max\{0, 1 - t\}$. By letting the argument $t = y_i w^T x_i$, we recover equation (1).

Let's compare the hinge loss H to the more familiar 0,1-loss function L. By again letting $t = y_i w^T x_i$, we can see that the natural way to define 0,1-loss in this context is

$$L(t) = \begin{cases} 0 & t > 0 \\ 1 & t < 0. \end{cases}$$

Then

$$\sum_{i=1}^{n} L(y_i w^t x_i)$$

is exactly the number of errors that the linear separator defined by w makes on the training data set.

If we have separable data, we can always find a perfect linear separator. If we have nonseparable data, ideally we would like to minimize the misclassification rate as given by the 0,1-loss above. Unfortunately this is an NP-hard problem, so we opt to minimize the hinge loss instead. Note that as far as error functions go, the hinge loss is somewhat weirdly behaved. For example, the loss increases without bound as the distance between a misclassified point and the boundary goes up.

Convexity

A function f is convex if for all w_1, w_2,

$$\alpha f(w_1) + (1-\alpha) f(w_2) \geq f(\alpha w_1 + (1-\alpha) w_2).$$

Convex functions are nice to work with because they are easy to minimize (just roll downhill). Furthermore sums of convex functions are convex.