Machine Learning Lecture Notes

Predrag Radivojac

March 6, 2015

1 Perceptron

As before, we will consider linear binary classifiers in $\mathbb{R}^k$, where $\mathcal{X} = \{1\} \times \mathbb{R}^k$ and $\mathcal{Y} = \{-1, +1\}$. However, in contrast to the logistic regression approach where we used the logistic (sigmoid) function to model the posetrior probability $p(y|\mathbf{x})$, here we are interested in a simpler task of directly finding a linear decision surface $\mathbf{w}^T\mathbf{x} = 0$ (line, plane, hyperplane) that separates positive and negative examples available in the training set $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$.

The perceptron is a simple machine or function $f : \mathcal{X} \to \mathcal{Y}$ defined as

$$f(\mathbf{x}) = \begin{cases} +1 & \mathbf{w}^T\mathbf{x} \geq 0 \\ \\ -1 & \mathbf{w}^T\mathbf{x} < 0 \end{cases}, \tag{1}$$

where, as before, $\mathbf{w} = (w_0, w_1, \ldots, w_k) \in \mathbb{R}^{k+1}$ is a vector of weights we seek to determine through training and $\mathbf{x} = (x_0 = 1, x_1, \ldots, x_k)$ is a data point from the input space $\mathcal{X}$.

Because $f(\mathbf{x})$ is non-differentiable, we cannot use differential calculus to optimize a pre-defined error or cost function. Instead, we will propose a weight update rule (based on the gradient descent method) and subsequently demonstrate that the training algorithm finds a good solution under certain (strong) assumptions.

First, we will assume that the data points from $\mathcal{D}$ are presented to the learning algorithm one at a time and the weights will be updated in the stochastic (incremental) mode. If the data set is infinitely large (say, we consider a data stream), the examples are presented to the learner, the weight update is made (if needed) and the example is deleted. On the other hand, if the data set is finite the learning algorithm can loop over the data set; i.e. once all n points have been presented to the learner, we start from the beginning. Second, and this is a strong assumption, we will consider that the positive and negative examples in $\mathcal{D}$ are linearly separable. Finally, for simplicity of downstream analysis, we will assume that the initial weight vector $\mathbf{w}^{(0)} = \mathbf{0}$.

For each data point $\mathbf{x}$ drawn from the data set at step t we have to consider two cases: (i) $\mathbf{x}$ is correctly classified and (ii) $\mathbf{x}$ is incorrectly classified using the decision surface determined by the current set of weights $\mathbf{w}^{(t)}$. If $\mathbf{x}$ is correctly classified, we do not need to update the current set of weights; i.e. $\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)}$. On the other hand, if $\mathbf{x}$ is incorrectly classified, there are two possibilities: (i) if $\mathbf{x}$ was classified as negative (and its true class label is positive) we will update the weights using $\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} + \mathbf{x}$; (ii) if $\mathbf{x}$ was classified as positive (and its true class label is negative), we will update the weight vector using $\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \mathbf{x}$.

Why are those decisions good? Consider the first scenario of incorrect classification (underclassification); that is, when $\mathbf{w}^{(t)T}\mathbf{x} < 0$. We will investigate the updated inner product between the weight vector and input $\mathbf{x}$ used in Eq. (1); that is

$$\begin{aligned}
\mathbf{w}^{(t+1)T}\mathbf{x} &= \mathbf{w}^{(t)T}\mathbf{x} + \mathbf{x}^T\mathbf{x} \\
&= \mathbf{w}^{(t)T}\mathbf{x} + ||\mathbf{x}||^2 \\
&\geq \mathbf{w}^{(t)T}\mathbf{x},
\end{aligned}$$

and, therefore, the weight update rule is moving the inner product $\mathbf{w}^{(t)T}\mathbf{x}$ (and the prediction $f(\mathbf{x})$ in the right direction. A similar equation can be constructed for the second case of incorrect classification (overclassification). Our hypothesis is that if the algorithm always moves the decision boundary in the right direction, it will eventually find a separation hyperplane.

The update rules for correct and incorrect classification (both underclassification and overclassification) can be combined as follows

$$\mathbf{w}^{(t+1)} = \begin{cases} \mathbf{w}^{(t)} & \mathbf{x} \text{ is correctly classified} \\ \\ \mathbf{w}^{(t)} + y\mathbf{x} & \mathbf{x} \text{ is incorrectly classified} \end{cases}, \tag{2}$$

where y is the class label of a data point $\mathbf{x}$. We refer to the weight update rule from Eq. (2) as the *perceptron training rule*.

We will now prove that, if the data points in $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$ are linearly separable, the perceptron training algorithm converges and finds a separation hyperplane in a finite number of steps (alternatively, if the data set is infinitely large, that the number of incorrect classificatin the algorithm will make during training is finite). To do this, we will turn the entire data set into a set of positive examples $\mathcal{D}^+$ by using $\mathbf{x}_i \leftarrow -\mathbf{x}_i$ and $y_i \leftarrow -y_i$, whenever the original class label is negative. This does not affect the weight update because the product $y_i\mathbf{x}_i$ is unchanged, but it does simplify the weight update rule to $\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} + \mathbf{x}$ for the learning from $\mathcal{D}^+$.

The assumption that data points are linearly separable corresponds to the fact that there exists at least one solution vector, say $\mathbf{w}_0$, as well as a positive constant ε such that $\mathbf{w}_0^T\mathbf{x} > \varepsilon$ for $\forall\mathbf{x} \in \mathcal{D}^+$. Without loss of generality, we will also assume that $\mathbf{w}_0$ has unit length; i.e. $||\mathbf{w}_0|| = 1$. We will now look at the weight vector updated after the ℓ-th misclassified example, $\mathbf{w}^{(\ell)}$, and calculate the cosine of the angle between $\mathbf{w}_0$ and $\mathbf{w}^{(\ell)}$ as

$$\cos(\mathbf{w}_0, \mathbf{w}^{(\ell)}) = \frac{\mathbf{w}_0^T \mathbf{w}^{(\ell)}}{||\mathbf{w}_0|| \cdot ||\mathbf{w}^{(\ell)}||}. \tag{3}$$

We will first investigate the growth of the numerator and denominator with the number of updates ℓ. Considering that $\mathbf{x}^{(\ell)}$ is the ℓ-th misclassified example, we first look at the numerator

$$\begin{aligned} \mathbf{w}_0^T \mathbf{w}^{(\ell)} &= \mathbf{w}_0^T \left(\mathbf{w}^{(\ell-1)} + \mathbf{x}^{(\ell)} \right) \\ &= \mathbf{w}_0^T \mathbf{w}^{(\ell-1)} + \mathbf{w}_0^T \mathbf{x}^{(\ell)} \\ &> \mathbf{w}_0^T \mathbf{w}^{(\ell-1)} + \varepsilon, \end{aligned}$$

given that $\mathbf{w}_0^T \mathbf{x} > \varepsilon$ for $\forall \mathbf{x} \in \mathcal{D}^+$. By repeating this approach recursively for $\mathbf{w}^{(\ell-1)}, \mathbf{w}^{(\ell-2)}, \ldots, \mathbf{w}^{(1)}$ and taking that $\mathbf{w}^{(0)} = \mathbf{0}$, we see that

$$\mathbf{w}_0^T \mathbf{w}^{(\ell)} > \ell \varepsilon. \tag{4}$$

Next, we will look at the norm of $\mathbf{w}^{(\ell)}$

$$\begin{aligned} ||\mathbf{w}^{(\ell)}||^2 &= \left(\mathbf{w}^{(\ell-1)} + \mathbf{x}^{(\ell)} \right)^T \left(\mathbf{w}^{(\ell-1)} + \mathbf{x}^{(\ell)} \right) \\ &= ||\mathbf{w}^{(\ell-1)}||^2 + 2\mathbf{w}^{(\ell-1)T} \mathbf{x}^{(\ell)} + ||\mathbf{x}^{(\ell)}||^2 \\ &< ||\mathbf{w}^{(\ell-1)}||^2 + ||\mathbf{x}^{(\ell)}||^2, \end{aligned}$$

where we used that $\mathbf{w}^{(\ell-1)T} \mathbf{x}^{(\ell)} < 0$ because the (positive) data point was misclassified. From here, we see that

$$\begin{aligned} ||\mathbf{w}^{(\ell)}||^2 &< \sum_{l=1}^{\ell} ||\mathbf{x}^{(\ell)}||^2 \\ &\leq \ell M^2, \end{aligned} \tag{5}$$

where

$$M = \max_{\mathbf{x}_i \in \mathcal{D}^+} ||\mathbf{x}_i||.$$

We introduced the constant M to indicate that the norm of the input vectors is bounded. It is now important to observe the following:

1. In Eq. (4) we showed that the numerator in Eq. (3) grows at least linearly with ℓ
2. In Eq. (5) we showed that the denominator in Eq. (3) grows at most linearly with $\sqrt{\ell}$

However, the two conditions become incompatible as $\ell \to \infty$ because the cosine function cannot be larger than 1. Therefore, we conclude that there must exist some number of updates, say $\ell_{\max}$, for which

$$\ell_{\max}\varepsilon < \mathbf{w}^T\mathbf{w}^{(\ell_{\max})} \leq ||\mathbf{w}^{(\ell_{\max})}|| < \sqrt{\ell_{\max}M^2}.$$

This, in turn, leads to the upper limit on the number of updates

$$\ell_{\max} < \frac{M^2}{\varepsilon^2}.$$

Under the aforementioned assumptions, the linear decision boundary (i.e. the underlying concept) will be learned after a finite number of $\ell_{\max}$ updates, which guarantees the convergence of the perceptron training algorithm. Therefore, the perceptron training algorithm will find a separating hyperplane.

In practice, it is useful to modify the update rule to

$$\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} + \eta y\mathbf{x},$$

where $\eta \in (0, 1]$ in order to avoid rapid movements of the separating hyperplane during training. The convergence of the perceptron training algorithm can be readily proved when $\eta \in (0, 1]$.

1.0.1 Problems with the perceptron algorithm

While interesting in the context of learning theory, the perceptron training algorithm has a strong assumption (linear separability of the data) that is necessary for the algorithm to converge. If the data are not linearly separable, the algorithm never stops and the cycles in weight updates can be hard to detect. Therefore, a better learning algorithm is needed to correct for this deficiency. This can be readily corrected by the Pocket Algorithm.