10-601 Machine Learning

Maria-Florina Balcan Spring 2015

Plan: Perceptron algorithm for learning linear separators.

1 Learning Linear Separators

Here we can think of examples as being from $\{0,1\}^n$ or from R^n. Given a training set of labeled examples (that is consistent with a linear separator),we can find a hyperplane $w \cdot x = w_0$ such that all positive examples are on one side and all negative examples are on other. I.e., $w \cdot x > w_0$ for positive x's and $w \cdot x < w_0$ for negative x's. We can solve this using linear programming. The sample complexity results for classes of finite VC-dimension together with known results about linear programming imply that the class of linear separators is efficiently learnable in the PAC (distributional) model. Today we will talk about the Perceptron algorithm.

1.1 The Perceptron Algorithm

One of the oldest algorithms used in machine learning (from early 60s) is an online algorithm for learning a linear threshold function called the Perceptron Algorithm.

We present the Perceptron algorithm in the *online learning* model. In this model, the following scenario is repeats:

1. The algorithm receives an unlabeled example.
2. The algorithm predicts a classification of this example.
3. The algorithm is then told the correct answer.

We will call whatever is used to perform step (2), the algorithm's "current hypothesis."

As mentioned, the Perceptron algorithm is an online algorithm for learning linear separators. For simplicity, we'll use a threshold of 0, so we're looking at learning functions like:

$$w_1x_1 + w_2x_2 + ... + w_nx_n > 0.$$

We can simulate a nonzero threshold with a "dummy" input x_0 that is always 1, so this can be done without loss of generality.

The Perceptron Algorithm:

1. Start with the all-zeroes weight vector $\mathbf{w}_1 = \mathbf{0}$, and initialize t to 1.
2. Given example $\mathbf{x}$, predict positive iff $\mathbf{w}_t \cdot \mathbf{x} > 0$.
3. On a mistake, update as follows:
   - Mistake on positive: $\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t + \mathbf{x}$.
   - Mistake on negative: $\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t - \mathbf{x}$.

   $t \leftarrow t + 1$.

So, this seems reasonable. If we make a mistake on a positive $\mathbf{x}$ we get $\mathbf{w}_{t+1} \cdot \mathbf{x} = (\mathbf{w}_t + \mathbf{x}) \cdot \mathbf{x} = \mathbf{w}_t \cdot \mathbf{x} + ||\mathbf{x}||^2$, and similarly if we make a mistake on a negative $\mathbf{x}$ we have $\mathbf{w}_{t+1} \cdot \mathbf{x} = (\mathbf{w}_t - \mathbf{x}) \cdot \mathbf{x} = \mathbf{w}_t \cdot \mathbf{x} - ||\mathbf{x}||^2$. So, in both cases we move closer (by $||\mathbf{x}||^2$) to the value we wanted.

We will show the following guarantee for the Perceptron Algorithm :

Theorem 1 *Let $\mathcal{S}$ be a sequence of labeled examples consistent with a linear threshold function $\mathbf{w}^* \cdot \mathbf{x} > 0$, where $\mathbf{w}^*$ is a unit-length vector. Then the number of mistakes M on $\mathcal{S}$ made by the online Perceptron algorithm is at most $(R/\gamma)^2$, where*

$$R = \max_{\mathbf{x} \in \mathcal{S}} ||\mathbf{x}||, \ \textit{and} \ \gamma = \min_{\mathbf{x} \in \mathcal{S}} |\mathbf{w}^* \cdot \mathbf{x}|.$$

Note that since $\mathbf{w}^*$ is a unit-length vector, the quantity $|\mathbf{w}^* \cdot \mathbf{x}|$ is equal to the distance of $\mathbf{x}$ to the separating hyperplane $\mathbf{w}^* \cdot \mathbf{x} = 0$. The parameter "$\gamma$" is often called the "margin" of $\mathbf{w}^*$, or more formally, the L_2 margin because we are measuring Euclidean distance.

Proof of Theorem 1. We are going to look at the following two quantities $\mathbf{w}_t \cdot \mathbf{w}^*$ and $||\mathbf{w}_t||$.

Claim 1: $\mathbf{w}_{t+1} \cdot \mathbf{w}^* \geq \mathbf{w}_t \cdot \mathbf{w}^* + \gamma$. That is, every time we make a mistake, the dot-product of our weight vector with the target increases by at least γ.

> Proof: if $\mathbf{x}$ was a positive example, then we get $\mathbf{w}_{t+1} \cdot \mathbf{w}^* = (\mathbf{w}_t + \mathbf{x}) \cdot \mathbf{w}^* = \mathbf{w}_t \cdot \mathbf{w}^* + \mathbf{x} \cdot \mathbf{w}^* \geq \mathbf{w}_t \cdot \mathbf{w}^* + \gamma$ (by definition of γ). Similarly, if $\mathbf{x}$ was a negative example, we get $(\mathbf{w}_t - \mathbf{x}) \cdot \mathbf{w}^* = \mathbf{w}_t \cdot \mathbf{w}^* - \mathbf{x} \cdot \mathbf{w}^* \geq \mathbf{w}_t \cdot \mathbf{w}^* + \gamma$.

Claim 2: $||\mathbf{w}_{t+1}||^2 \leq ||\mathbf{w}_t||^2 + R^2$. That is, every time we make a mistake, the length squared of our weight vector increases by at most R^2.

> Proof: if $\mathbf{x}$ was a positive example, we get $||\mathbf{w}_t + \mathbf{x}||^2 = ||\mathbf{w}_t||^2 + 2\mathbf{w}_t \cdot \mathbf{x} + ||\mathbf{x}||^2$. This is less than $||\mathbf{w}_t||^2 + ||\mathbf{x}||^2$ because $\mathbf{w}_t \cdot \mathbf{x}$ is negative (remember, we made a mistake on $\mathbf{x}$), and this in turn is at most $||\mathbf{w}_t||^2 + R^2$. Same thing (flipping signs) if $\mathbf{x}$ was negative but we predicted positive.

Claim 1 implies that after M mistakes, $\mathbf{w}_{M+1} \cdot \mathbf{w}^* \geq \gamma M$. On the other hand, Claim 2 implies that after M mistakes, $||\mathbf{w}_{M+1}||^2 \leq R^2 M$. Now, all we need to do is use the fact that $\mathbf{w}_{M+1} \cdot \mathbf{w}^* \leq ||\mathbf{w}_{M+1}||$, since $\mathbf{w}^*$ is a unit-length vector. So, this means we must have $\gamma M \leq R\sqrt{M}$, and thus $M \leq (R/\gamma)^2$. ■

Discussion: In order to use the Perceptron algorithm to find a consistent linear separator given a set S of labeled examples that is linearly separable by margin γ, we do the following. We repeatedly feed the whole set S of labeled examples into the Perceptron algorithm up to $(R/\gamma)^2 + 1$ rounds, until we get to a point where the current hypothesis is consistent with the whole set S. Note that by theorem 1, we are guaranteed to reach such a point. The runnning time is then polynomial in $|S|$ and $(R/\gamma)^2$.

In the worst case, γ can be exponentially small in n. On the other hand, if we're lucky and the data is well-separated, γ might even be large compared to $1/n$. This is called the "large margin" case. (In fact, the latter is the more modern spin on things: namely, that in many natural cases, we would hope that there exists a large-margin separator.) In fact, one nice thing here is that the mistake-bound depends on just a purely geometric quantity: the amount of "wiggle-room" available for a solution and doesn't depend in any direct way on the number of features in the space.

So, if data is separable by a large margin, then the Perceptron is a good algorithm to use.

1.2 Additional More Advanced Notes

Guarantee in a distributional setting: In order to get a distributional guarantee we can do the following.[1] Let $M = (R/\gamma)^2$. For any ϵ, δ, we draw a sample of size $(M/\epsilon) \cdot \log(M/\delta)$. We then run Perceptron on the data set and look at the sequence of hypotheses produced: h_1, h_2, ... For each i, if h_i is consistent with following $1/\epsilon \cdot \log(M/\delta)$ examples, then we stop and output h_i. We can argue that with probability at least $1 - \delta$, the hypothesis we output has error at most ϵ. This can be shown as follows. If h_i was a bad hypothsis with true error greater than ϵ, then the chance we stopped and output h_i was at most δ/M. So, by union bound, there's at most a δ chance we are fooled by *any* of the hypotheses.

Note that this implies that if the margin over the whole distribution is $1/poly(n)$, the Perceptron algorithm can be used to PAC learn the class of linear separators.

What if there is no perfect separator? What if only *most* of the data is separable by a large margin, or what if $\mathbf{w}^*$ is not perfect? We can see that the thing we need to look at is Claim 1. Claim 1 said that we make "γ amount of progress" on every mistake. Now it's possible there will be mistakes where we make very little progress, or even negative progress. One thing we can do is bound the total number of mistakes we make in terms of the total distance we would have to move the points to make them actually separable by margin γ. Let's call that TD_γ. Then, we get that after M mistakes, $\mathbf{w}_{M+1} \cdot \mathbf{w}^* \geq \gamma M - \mathrm{TD}_\gamma$. So,

[1] This is not the most sample efficient online to PAC reduction, but it is the simplest to think about.

combining with Claim 2, we get that $R\sqrt{M} \geq \gamma M - \mathrm{TD}_\gamma$. We could solve the quadratic, but this implies, for instance, that $M \leq (R/\gamma)^2 + (2/\gamma)\mathrm{TD}_\gamma$. The quantity $\frac{1}{\gamma}\mathrm{TD}_\gamma$ is called the total *hinge-loss* of w^*.

So, this is not too bad: we can't necessarily say that we're making only a small multiple of the number of mistakes that $\mathbf{w}^*$ is (in fact, the problem of finding an approximately-optimal separator is NP-hard), but we can say we're doing well in terms of the "total distance" parameter.

Perceptron for approximately maximizing margins. We saw that the perceptron algorithm makes at most $(R/\gamma)^2$ mistakes on any sequence of examples that is linearly-separable by margin γ, i.e., any sequence for which there exists a unit-length vector $\mathbf{w}^*$ such that all examples $\mathbf{x}$ satisfy $\ell(\mathbf{x})(\mathbf{w}^* \cdot \mathbf{x}) \geq \gamma$, where $\ell(\mathbf{x}) \in \{-1, 1\}$ is the label of $\mathbf{x}$.

Suppose we are handed a set of examples $\mathcal{S}$ and we want to actually *find* a large-margin separator for them. One approach is to directly solve for the maximum-margin separator using convex programming (which is what is done in the SVM algorithm). However, if we only need to *approximately* maximize the margin, then another approach is to use Perceptron. In particular, suppose we cycle through the data using the Perceptron algorithm, updating not only on mistakes, but also on examples $\mathbf{x}$ that our current hypothesis gets correct by margin less than $\gamma/2$. Assuming our data is separable by margin γ, then we can show that this is guaranteed to halt in a number of rounds that is polynomial in R/γ. (In fact, we can replace $\gamma/2$ with $(1-\epsilon)\gamma$ and have bounds that are polynomial in $R/(\epsilon\gamma)$.)

The Margin Perceptron Algorithm(γ):

1. Initialize $\mathbf{w}_1 = \ell(\mathbf{x})\mathbf{x}$, where $\mathbf{x}$ is the first example seen and initialize t to 1.
2. Predict positive if $\frac{\mathbf{w}_t \cdot \mathbf{x}}{||\mathbf{w}_t||} \geq \gamma/2$, predict negative if $\frac{\mathbf{w}_t \cdot \mathbf{x}}{||\mathbf{w}_t||} \leq -\gamma/2$, and consider an example to be a margin mistake when $\frac{\mathbf{w}_t \cdot x}{||\mathbf{w}_t||} \in (-\gamma/2, \gamma/2)$.
3. On a mistake (incorrect prediction or margin mistake), update as in the standard Perceptron algorithm: $\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t + \ell(\mathbf{x})\mathbf{x}$; $t \leftarrow t+1$.

Theorem 2 *Let $\mathcal{S}$ be a sequence of labeled examples consistent with a linear threshold function $\mathbf{w}^* \cdot \mathbf{x} > 0$, where $\mathbf{w}^*$ is a unit-length vector, and let*

$$R = \max_{\mathbf{x} \in \mathcal{S}} ||\mathbf{x}||, \text{ and } \gamma = \min_{\mathbf{x} \in \mathcal{S}} |\mathbf{w}^* \cdot \mathbf{x}|.$$

Then the number of mistakes (including margin mistakes) made by Margin Perceptron(γ) on $\mathcal{S}$ is at most $8(R/\gamma)^2 + 4(R/\gamma)$.

Proof: The argument for this new algorithm follows the same lines as the argument for the original Perceptron algorithm.

As before, each update increases $\mathbf{w}_t \cdot \mathbf{w}^*$ by at least γ. What is now a little more complicated is to bound the increase in $||\mathbf{w}_t||$. For the original algorithm, we had: $||\mathbf{w}_{t+1}||^2 \leq ||\mathbf{w}_t||^2 + R^2$, which implies $||\mathbf{w}_{t+1}|| \leq ||\mathbf{w}_t|| + \frac{R^2}{2||\mathbf{w}_t||}$.

For the new algorithm, we can show instead:

$$||\mathbf{w}_{t+1}|| \leq ||\mathbf{w}_t|| + \frac{R^2}{2||\mathbf{w}_t||} + \frac{\gamma}{2}. \tag{1}$$

To see this note that:

$$||\mathbf{w}_{t+1}||^2 = ||\mathbf{w}_t||^2 + 2l(x)\mathbf{w}_t \cdot \mathbf{x} + ||\mathbf{x}||^2 = ||\mathbf{w}_t||^2 \left(1 + \frac{2l(x)}{||\mathbf{w}_t||} \frac{\mathbf{w}_t \cdot \mathbf{x}}{||\mathbf{w}_t||} + \frac{||\mathbf{x}||^2}{||\mathbf{w}_t||^2}\right)$$

Taking the square-root of both sides and using the inequality $\sqrt{1+\alpha} \leq 1 + \frac{\alpha}{2}$ and $||x||^2 \leq R^2$ we get:

$$||\mathbf{w}_{t+1}|| \leq ||\mathbf{w}_t|| \left(1 + \frac{l(x)}{||\mathbf{w}_t||} \frac{\mathbf{w}_t \cdot \mathbf{x}}{||\mathbf{w}_t||} + \frac{R^2}{2||\mathbf{w}_t||^2}\right).$$

Combining this with the fact $\frac{l(\mathbf{x})\mathbf{w}_t \cdot \mathbf{x}}{||\mathbf{w}_t||} \leq \frac{\gamma}{2}$ (since w_t made a mistake or margin mistake on $\mathbf{x}$) we get the desired upper bound on $||\mathbf{w}_{t+1}||$, namely: $||\mathbf{w}_{t+1}|| \leq ||\mathbf{w}_t|| + \frac{\gamma}{2} + \frac{R^2}{2||\mathbf{w}_t||}$.

Note that (1) implies that if $||\mathbf{w}_t|| \geq 2R^2/\gamma$ then $||\mathbf{w}_{t+1}|| \leq ||\mathbf{w}_t|| + 3\gamma/4$. Note also that $||\mathbf{w}_{t+1}|| \leq ||\mathbf{w}_t|| + ||\mathbf{x}||$ (by triangle inequality), so $||\mathbf{w}_{t+1}|| \leq ||\mathbf{w}_t|| + R$. These two facts imply that after M updates we have:

$$||\mathbf{w}_{M+1}|| \leq (2R^2/\gamma + R) + 3M\gamma/4.$$

Solving $M\gamma \leq (2R^2/\gamma + R) + 3M\gamma/4$ we get $M \leq 8R^2/\gamma^2 + 4R/\gamma$, as desired. ■