CS269: Machine Learning Theory
Lecture 14: Generalization Error of Adaboost
November 10, 2010

Lecturer: Jennifer Wortman Vaughan
Scribe: David Lei, Daniel Quach, Alex Lee

In this lecture we will continue our discussion of the Adaboost algorithm and derive a bound on the generalization error. We saw last time that the training error decreases exponentially with respect to the number of rounds T. However, we also want to see the performance of this algorithm on new test data. Today we will show why the Adaboost algorithm generalizes so well and why it avoids overfitting.

The first part of the lecture will recap the Adaboost algorithm, and introduce an upper bound on the generalization error based on VC dimension. However, it turns out that this bound does not capture the fact that Adaboost avoids overfitting in practice.
To capture this, we will turn to the notion of large margin. The second part of the lecture will show the existence of a large margin classifier of the form: $h(\vec{x}) = \text{sign}(\sum_{t=1}^{T} \alpha_t h_t(\vec{x}))$
Finally, we will bound the generalization error of Adaboost by making use of the following two facts:
1. Adaboost will increase the margin as T grows.
2. A large margin on the training data implies a smaller generalization error $err(h)$.

1 Adaboost Algorithm

We first review the Adaboost algorithm.

Input: m labeled data points $(\vec{x}_1, y_1)...(\vec{x}_m, y_m)$ and a weak learning algorithm A that is guaranteed to output a label with error less than $\frac{1}{2} - \gamma$.

Initialize $D_1(i) = \frac{1}{m}\ \forall i$

For $t = 1 : T$
Run A on a sufficiently large sample drawn i.i.d. from D_t to produce some function h_t
Set $\epsilon_t = \sum_{i=1}^{m} D_t(i)\mathbb{1}(h_t(\vec{x}_i) \neq y_i)$
Set $\alpha_t = \frac{1}{2} ln(\frac{1-\epsilon_t}{\epsilon_t})$
$\forall i$ update $D_{t+1}(i) = \frac{D_t(i)}{Z_t} exp(-\alpha_t y_i h_t(\vec{x}_i))$

Output $h(\vec{x}) = \text{sign}(\sum_{t=1}^{T} \alpha_t h_t(\vec{x}))$

Here $\mathbb{1}(\cdot)$ is an indicator function that is 1 if its input is true and 0 otherwise, and Z_t is the normalization factor needed to ensure that D_{t+1} is a probability distribution.

Last time we derived the bounds for the empirical error, showing that

$$\widehat{err}(h) \leq exp(-2\gamma^2 T) \quad if \quad \epsilon_t \leq \frac{1}{2} - \gamma \quad \forall t$$

But how well does this function h perform for new samples? In other words, what is the upper bound for the generalization error of h?

1.1 An upper bound based on VC Dimension

To bound the generalization error of Adaboost, we will look at the class of functions from which it chooses its output, which we will denote as H'. From there, we could potentially find a bound on the VC dimension or growth function for this class, and then apply the generalization error bounds that were discussed in the first half of the course.

Let $H' = \{h : h(\vec{x}) = \text{sign}(\sum_{t=1}^{T} \alpha_t h_t(\vec{x})) \;\; for \;\; h_1, h_2, ..., h_T \in H\}$

Let $d = VCdim(H)$. It is possible to show (though we will not do it here) that with high probability,

$$err(h) \leq \widehat{err}(h) + \tilde{O}\left(\sqrt{\frac{Td}{m}}\right)$$

where $\tilde{O}()$ denotes that we are hiding log terms in the Big-O notation.

As T grows, $\widehat{err}(h)$ decreases, while the class of functions we are choosing from becomes more complex and $\sqrt{\frac{Td}{m}}$ increases. According to the Occam's Razor principle, we would expect to find that there is an optimal value of T that optimizes this trade-off and minimizes $err(h)$.

However in real test situations, people have found that $err(h)$ continues to decrease even after $\widehat{err}(h)$ reaches 0. This is good news in the sense that overfitting does not occur as T increases, but why does it contradict the Occam's Razor principle above? (The bound is still *valid*, it just doesn't convey the right intuition.)

The reason is that not only do we have to know whether our hypothesis h is right or wrong, but also with how much "confidence" it labels our data. Even as $\widehat{err}(h)$ reaches 0, the confidence (as measured by the margin) continues to increase with the number of rounds, which decreases the generalization error $err(h)$.

2 Existence of Large Margin Classifiers

Recall that for a linear separator $h(\vec{x}) = \text{sign}(\vec{w} \cdot \vec{x})$, if $||\vec{w}|| = 1$, then we define the margin of a point $(\vec{x}_i, y_i)$ as:

$$margin = (\vec{w} \cdot \vec{x}_i) y_i$$

This is simply the distance between the linear separator and the point $\vec{x}_i$.

Similarly, when we have $h(\vec{x}) = \text{sign}(\sum_{t=1}^{T} \alpha_t h_t(\vec{x}))$, we can define the margin as

$$margin = \frac{\sum_{t=1}^{T} \alpha_t h_t(\vec{x}_i) y_i}{\sum_{t=1}^{T} \alpha_t}$$

In Theorem 2, we show that if the weak learning assumption holds, then there must exist a large margin classifier of the form $h(\vec{x}) = \text{sign}(\sum_{t=1}^{T} \alpha_t h_t(\vec{x}))$. The proof is based on the Minimax Theorem.

Recall the Minimax Theorem:

Theorem 1. *For any matrix $M \in \mathbb{R}^{m \times n}$*

$$\min_{p \in \Delta_m} \max_{q \in \Delta_n} \vec{p}^T M \vec{q} = \max_{q \in \Delta_n} \min_{p \in \Delta_m} \vec{p}^T M \vec{q}.$$

This immediately implies that for all M we also have

$$\max_{p \in \Delta_m} \min_{q \in \Delta_n} \vec{p}^T M \vec{q} = \min_{q \in \Delta_n} \max_{p \in \Delta_m} \vec{p}^T M \vec{q}.$$

Using the above, we can define the following theorem, using a finite H for simplicity:

Theorem 2. *Consider a finite class $H = \{h_1, \cdots, h_n\}$ of size n and a set of points $(\vec{x}_1, y_1), ..., (\vec{x}_m, y_m)$. Suppose $\forall$ distributions $\vec{p} \in \Delta_m, \exists h \in H$ such that*

$$\sum_{i=1}^{m} p_i \mathbb{1}(h(\vec{x}_i) \neq y_i) \leq \frac{1}{2} - \gamma$$

Then $\exists \vec{q} \in \Delta_n$ such that $\min_{i \in \{1, \cdots, m\}} \sum_{j=1}^{n} q_j h_j(\vec{x}_i) y_i \geq 2\gamma$.

Here, $\vec{p}$ corresponds to a distribution over input points and $\vec{q}$ corresponds to a weighting over the functions in H (like $\vec{\alpha}$ will be). The result then says that if the weak learning assumption holds then there exists a function $h(\vec{x}) = \text{sign}(\sum_{j=1}^{n} q_j h_j(x_i))$ that has $margin \geq 2\gamma$. This is kind of a sanity check – if it wasn't the case that a large margin classifier of this form always existed, then we couldn't hope to say that Adaboost is increasing the margin.

Proof: Let us define a matrix M as follows:

$$M_{ij} = \mathbb{1}(h_j(\vec{x}_i) \neq y_i)$$

Fixing $\vec{p} \in \Delta_m$, we have:

$$\begin{aligned}
\min_{\vec{q} \in \Delta_n} \vec{p}^T M \vec{q} &= \min_{\vec{q} \in \Delta_n} \sum_{i=1}^{m} \sum_{j=1}^{n} p_i q_j \mathbb{1}(h_j(\vec{x}_i) \neq y_i) \\
&= \min_{\vec{q} \in \Delta_n} \sum_{j=1}^{n} q_j \sum_{i=1}^{m} p_i \mathbb{1}(h_j(\vec{x}_i) \neq y_i) \\
&= \min_{j \in \{1, \cdots, n\}} \sum_{i=1}^{m} p_i \mathbb{1}(h_j(\vec{x}_i) \neq y_i) \\
&\leq \frac{1}{2} - \gamma
\end{aligned}$$

Since this holds for any distribution $\vec{p}$, we have

$$\max_{\vec{p}\in\Delta_m} \min_{\vec{q}\in\Delta_n} \vec{p}^T M\vec{q} \leq \frac{1}{2} - \gamma$$

By the Minimax Theorem, this implies that

$$\min_{\vec{q}\in\Delta_n} \max_{\vec{p}\in\Delta_m} \vec{p}^T M\vec{q} \leq \frac{1}{2} - \gamma$$

Fixing $\vec{q} \in \Delta_n$, leaves us with:

$$\begin{aligned}\max_{\vec{p}\in\Delta_m} \vec{p}^T M\vec{q} &= \max_{\vec{p}\in\Delta_m} \sum_{i=1}^{m} p_i \sum_{j=1}^{n} q_j \mathbb{1}(h_j(\vec{x}_i) \neq y_i) \\ &= \max_{i\in\{1,\cdots,m\}} \sum_{j=1}^{n} q_j \mathbb{1}(h_j(\vec{x}_i) \neq y_i)\end{aligned}$$

The expression we derived above implies that there exists a value of $\vec{q}$ such that

$$\max_{i\in\{1,\cdots,m\}} \sum_{j=1}^{n} q_j \mathbb{1}(h_j(\vec{x}_i) \neq y_i) \leq \frac{1}{2} - \gamma$$

Note that this does not necessarily hold for $\forall \vec{q}$. We are only guaranteed it holds for the value of $\vec{q}$ that minimizes the expression above.

This implies that for this particular value of $\vec{q}, \forall i$

$$\begin{aligned}&\sum_{j=1}^{n} q_j h_j(\vec{x}_i) y_i \\ &= \sum_{j=1}^{n} q_j (1 - \mathbb{1}(h_j(\vec{x}_i) \neq y_i)) - q_j \mathbb{1}(h_j(\vec{x}_i) \neq y_i) \\ &= \sum_{j=1}^{n} q_j (1 - 2 * \mathbb{1}(h_j(\vec{x}_i) \neq y_i)) \\ &= 1 - 2\sum_{j=1}^{n} q_j \mathbb{1}(h_j(\vec{x}_i) \neq y_i) \geq 2\gamma\end{aligned}$$

□

3 Bounding the Generalization Error of Adaboost

Let's now turn our attention back to Adaboost. We can prove the following theorem.

Theorem 3. *Let h be the function output by Adaboost after T rounds. Then for any θ:*

$$\frac{1}{m}\sum_{i=1}^{m}\mathbb{1}(y_i h(\vec{x}_i) \leq \theta) \leq \prod_{t=1}^{T} 2\sqrt{\epsilon_t^{1-\theta}(1-\epsilon_t^{1+\theta})}$$

We see that when $\theta = 0$, this is precisely the bound on the training error of h that we derived in the previous lecture. Because the proof of this theorem is extremely similar to that proof, we will not go through it in detail.
We can see that if $\theta \leq \gamma$, then the right hand side goes to 0 as $T \to \infty$.
Now let us define CO(H), the convex hull, as follows:

$$CO(H) = \left\{ f : f(\vec{x}) = \sum_{t=1}^{T} \alpha_t h_t(\vec{x}), T \geq 1, \vec{\alpha} \in \Delta_T, h_1, h_2, ...h_T \in H \right\}$$

We state the following without proving it.

Theorem 4. *For a sample of size m, for any δ with probability $1-\delta, \forall f \in CO(H), \forall \theta \geq 0$:*

$$Pr_{\vec{x}\sim D}[yf(\vec{x}) \leq 0] \leq \frac{1}{m}\sum_{i=1}^{m}\mathbb{1}(y_i f(\vec{x}_i) \leq \theta) + O\left(\frac{1}{\sqrt{m}}\sqrt{\frac{\log m \log|H|}{\theta^2}} + \log\frac{1}{\delta}\right)$$

Let's examine this theorem. We see that if $h(\vec{x}) = \text{sign}(f(\vec{x}))$, then the left hand side of this expression is precisely $err(h)$. The first term on the right hand side is precisely the fraction of points in our sample that have margin less than θ (which we know from the previous theorem goes to 0 as T gets big). Therefore, this theorem provides a bound on $err(h)$ with one term that goes to 0 as T gets big, and one term that doesn't depend on T at all. This confirms the empirical observations that the generalization error of Adaboost tends to decrease as T gets big.