COS 511: Foundations of Machine Learning

Rob Schapire
Scribe: Jie Chen

Lecture #8
February 27, 2003

1 Introduction to Boosting

We will focus on the idea of boosting for the next two lectures. The algorithms in PAC learning models can produce hypothesis with arbitrary error rate, as long as sample complexity is satisfied. However, suppose a learning algorithm can do a little bit better than random, that is, its error rate is less than 50%, can we take this error rate and drive it down to zero?

We introduce some definitions which will be used to understand the idea of boosting.
Definition 1
$\mathcal{C}$ is learnable if
$\exists$ algorithm A
$\forall c \in \mathcal{C}$
$\forall D$
$\forall \epsilon > 0$
$\forall \delta > 0$
A given $m = poly(\frac{1}{\epsilon}, \frac{1}{\delta})$ examples $(x_1, c(x_1)), \ldots (x_m, c(x_m))$
computes h, $Pr[err(h) > \epsilon] \leq \delta$

Definition 2
$\mathcal{C}$ is efficiently learnable if
A runs in time polynomial in $\frac{1}{\epsilon}, \frac{1}{\delta}, s, n$, where s is the size of the target concept, and n is the size of the instances. For example, n is the instance size when the domain X_n is $\{0,1\}^n$ or $\Re^n$.

Definition 3
$\mathcal{C}$ is weakly learnable if
$\exists \gamma > 0$
$\exists$ algorithm A
$\forall c \in \mathcal{C}$
$\forall D$
$\forall \delta > 0$
A given $m = poly(\frac{1}{\delta})$ examples $(x_1, c(x_1)), \ldots (x_m, c(x_m))$
computes h, $Pr[err(h) > \frac{1}{2} - \gamma] \leq \delta$

A weak learner can be trivial. For example if given a sample set of more than 75% positive examples, a learner can output a hypothesis that always predicts positive. However that is not weakly learnable since the latter is defined as having an error rate slightly smaller than 50% on *any* distribution of examples.
On the other hand, weakly learnable does not necessarily mean strong learnable given a *fixed* distribution. For example, let $\mathcal{C}$ be all boolean functions on $\{0,1\}^n \cup \{Z_0\}$, let the

distribution be $\frac{1}{4}$ on the point Z_0 and uniform on all the remaining points. Take a sample of size m, Z_0 is likely to be included. Also included are a tiny fraction of other points because m is small compared to the 2^n possibilities of 0/1 string. Assuming Z_0 got its label correct, the total error rate in this case is roughly $\frac{3}{4} \cdot \frac{1}{2} = \frac{3}{8} < \frac{1}{2}$ because we are getting so few samples on the other points that we are merely guessing. On the other hand, for this fixed distribution, there is really no way of driving the error rate significantly below $\frac{3}{8}$ with a polynomial size sample.

Boosting is the idea of converting weak learning to strong learning. In the following section we will see algorithms for boosting do exist.

2 Boosting Algorithms

2.1 General Algorithm for boosting

Given $S =< (x_1, y_1), \ldots, (x_m, y_m) >, x_i \in \mathrm{X}, y_i \in \{-1, 1\}$,
and access to a weak learning algorithm A,

$\forall D$, compute h, such that $Pr[err_D(h) \leq \frac{1}{2} - \gamma] \geq 1 - \delta$,
we would like to have a learning algorithm with arbitrary low error rate.
What if we run the same algorithm several times? Then we have to change D. Otherwise the learner A could output the same h every time.
Therefore for each running of A, we will change D to force A to learn something new every time. Each time A will output a hypothesis $h_t, t \in [1 \ldots T]$. In the end we combine all the hypotheses h_t into a final resulting H.
Let D_t constructed on given examples while $D_t(i)$ is the weight on example (x_i, y_i).

The general algorithm of boosting goes like this:

for $t = 1 \ldots T$
construct D_t on training examples
run A on D_t
get $h_t : \mathrm{X} \to \{-1, 1\}$
$\epsilon_t = Pr_{D_t}[h_t(x_i) \neq y_i] = \epsilon_t = \frac{1}{2} - \gamma_t \leq \frac{1}{2} - \gamma$
end
output H

Previously the algorithm A has access to all the examples and their labels. To incorporate the distribution D_t, two approaches exist in practice: (1) the algorithm picks a set of examples according to D_t; (2) the algorithm A tries to directly minimize weighted training error $\sum_{i:h_t(x_i) \neq y_i} D_t(i)$.

2.2 AdaBoost

AdaBoost is the first practical boosting algorithm. In this algorithm we have to answer two questions: (1) how to construct D_t; (2) how to combine h_t to H.

2.2.1 Construction of D_t

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

$$D_{t+1}(i) = \frac{D_t(i)}{Z_t} \cdot \begin{cases} e^{-\alpha_t} & \text{if } h_t(x_i) = y_i \\ e^{\alpha_t} & \text{if } h_t(x_i) \neq y_i \end{cases}$$

The intuition is we want to focus on the hard example, which is the example that a weak classifier misclassified. So we increase weight on a previous wrong example and lower weight on a previous correct one. Z_t in the formula is a normalization factor.

2.2.2 Combining h_t

We will take a weighted majority vote of the individual hypotheses

$$H(x) = \text{sign}(\sum_{t=1}^{T} h_t(x) \cdot \alpha_t)$$

where

$$\text{sign}(x) = \begin{cases} -1 & \text{if } x < 0 \\ 0 & \text{if } x = 0 \\ +1 & \text{if } x > 0. \end{cases}$$

2.3 Empirical Error Rate of AdaBoost

Theorem

$$\begin{aligned} \hat{err}(H) &\leq \prod_{t=1}^{T} [2\sqrt{\epsilon_t(1-\epsilon_t)}] \\ &= \exp\left(-\sum_t \text{RE}(\frac{1}{2}||\epsilon_t)\right) \\ &= \prod_t \sqrt{1-4\gamma_t^2} \\ &\leq \exp\left(-2\sum_t \gamma_t^2\right). \end{aligned}$$

If we assume $\gamma_t \geq \gamma$ then

$$\hat{err}(H) \leq e^{-2\gamma^2 T}.$$

The theorem shows the training error would decrease exponentially as the number of times to repeat increases. Specificly, if $T > \frac{1}{2\gamma^2} \ln m$, then $\hat{err}(H) < \frac{1}{m}$, which implies the training error will be zero. Note the theorem does not have any assumptions on h_t and where samples are from. We are going to prove the theorem in 3 steps.

1. Show that

$$D_{T+1}(x_i) = \frac{\exp(-y_i f(x_i))}{m \prod_t Z_t}$$

where $f(x_i) = \sum_{t=1}^{T} \alpha_t \cdot h_t(x_i)$

Proof:

$$\begin{aligned} D_{T+1}(i) &= \frac{D_T(i) \cdot \exp(-\alpha_T y_i h_T(x_i))}{Z_T} \\ &= \frac{D_{T-1}(i) \cdot \exp(-\alpha_{T-1} y_i h_{T-1}(x_i)) \cdot \exp(-\alpha_T y_i h_T(x_i)}{Z_{T-1} \cdot Z_T} \\ &\vdots \\ &= \frac{1}{m} \cdot \frac{\exp(-y_i \cdot \sum_t \alpha_t h_t(x_i))}{\prod_t Z_t} \end{aligned}$$

2. Show that

$$\hat{err}(H) \leq \prod_t Z_t$$

Proof:

$$\begin{aligned} \hat{err}(H) &= \frac{1}{m} \cdot \sum_{i=1}^{m} [\![y_i \neq H(x_i)]\!] \\ &= \frac{1}{m} \cdot \sum_{i=1}^{m} [\![y_i f(x_i) \leq 0]\!] \\ &\leq \frac{1}{m} \sum_{i=1}^{m} e^{-y_i f(x_i)} \\ &= \frac{1}{m} \cdot \sum_{i=1}^{m} D_{T+1}(i) \cdot m \cdot \prod_t Z_t \\ &= \frac{1}{m} \prod_t Z_t \cdot \sum_{i=1}^{m} D_{T+1}(i) \cdot m \\ &= \prod_t Z_t. \end{aligned}$$

Here, $[\![\pi]\!]$ is 1 if π is true, and 0 otherwise. Note the third line follows because for each term that $y_i f(x_i) \leq 0$, the corresponding $e^{-y_i f(x_i)}$ is greater than 1 while for each term $y_i f(x_i) > 0$ the term $e^{-y_i f(x_i)}$ is greater than zero. The fourth line comes from step 1 above.

3. Show that

$$Z_t \leq 2\sqrt{\epsilon_t(1-\epsilon_t)}.$$

Proof:

$$
\begin{aligned}
Z_t &= \sum_{i=1}^{m} D_t(i) \cdot e^{-\alpha_t y_i h_t(x_i)} \\
&= \sum_{i:h_t(x_i) \neq y_i} D_t(i) \cdot e^{\alpha_t} + \sum_{i:h_t(x_i)=y_i} D_t(i) \cdot e^{-\alpha_t} \\
&= \epsilon_t \cdot e^{\alpha_t} + (1-\epsilon_t) \cdot e^{-\alpha_t}.
\end{aligned}
$$

We choose α_t so that Z_t is minimized. In this case,

$$\alpha_t = \frac{1}{2} \ln \left(\frac{1-\epsilon_t}{\epsilon_t} \right)$$

and

$$Z_t = 2\sqrt{\epsilon_t (1-\epsilon_t)^2}.$$

Note the third line follows the definition of weighted training error.