Lecture 8

Instructor: Haipeng Luo

1 Boosting and AdaBoost

In this lecture we discuss the connection between boosting and online learning. Boosting is not only one of the most fundamental theories in machine learning, but also one of the most widely used machine learning algorithms in practice. It was originally proposed in a statistical learning setting for binary classification, which will also be the focus here. Informally, the key question that boosting tries to answer is that if we have a learning algorithm with some nontrivial (but also not great) prediction accuracy (say 51%), is it possible to boost the accuracy to something arbitrarily high (say 99.9%), in a blackbox manner?

Formally we consider the following binary classification task. Given a set of training examples $S = \{(x_1, y_1), \ldots, (x_N, y_N)\}$ where each $(x_i, y_i) \in \mathcal{X} \times \{-1, +1\}$ (for some feature space $\mathcal{X}$) is an i.i.d. sample of an unknown distribution $\mathcal{D}$, our goal is to output a binary classifier $H : \mathcal{X} \to \{-1, +1\}$ with small training error $\frac{1}{N}\sum_{i=1}^{N} \mathbf{1}\{H(x_i) \neq y_i\}$ and more importantly small generalization error $\mathbb{E}_{(x,y)\sim\mathcal{D}}[\mathbf{1}\{H(x) \neq y\}]$.

Now suppose we are given a *weak learning oracle* $\mathcal{A}$ which takes a training set S and a distribution over the training examples $p \in \Delta(N)$ as input (in other words, a weighted training set), and outputs a classifier $h = \mathcal{A}(S, p) \in \mathcal{H}$ for some hypothesis space $\mathcal{H}$ such that the weighted average error according to p is always bounded as

$$\mathbb{E}_{i\sim p}[\mathbf{1}\{h(x_i) \neq y_i\}] = \sum_{i:h(x_i)\neq y_i} p(i) \leq \frac{1}{2} - \gamma \tag{1}$$

for some constant $\gamma > 0$. This is called the *weak learning assumption* and h is also called a weak classifer. One should think about γ as something very small such as 1%, so the weak learning assumption is very mild and is just saying that the oracle $\mathcal{A}$ is doing something slightly nontrivial since the trivial random guessing has expected error rate $1/2$. We called γ the *edge* of the oracle.

In practice, such oracle can be any existing off-the-shelf learning algorithms, such as SVM, decision tree algorithms (e.g. C4.5), neural nets algorithms, or even something much simpler such as decision stump algorithms (a stump is a tree with depth 1). Many of these algorithms support taking weighted training set as input. However, even if weighted training set is not supported, one can simply generate a new (and large enough) training set by sampling from S according to p with replacement and use it as the input for the oracle instead, so that the (unweighted) training error on this new training set is close to the weighted error rate on S.

A boosting algorithm is then a master algorithm that uses the oracle as a subroutine and outputs a strong classifier with very high accuracy. The fact that this is even possible is highly nontrivial and was not clear until the seminal work of [Schapire, 1990]. The idea is to repeatedly apply the oracle with a distribution that focuses on "hard" examples and to combine all the weak classifiers outputted by the oracle in some smart way.

The most successful boosting algorithm is AdaBoost [Freund and Schapire, 1997] (short for Adaptive Boosting), outlined in Algorithm 1. At each iteration AdaBoost maintains a distribution p_t and invoke the oracle with a training set weighted by p_t to get a weak classifier h_t. Afterwards, based on the weighted error rate of h_t, p_t is updated in a multiplicative way so that misclassified examples get more weights in the next iteration. The final output of AdaBoost is a weighted majority vote of

Algorithm 1: AdaBoost

Input: training set $S = \{(x_1, y_1), \ldots, (x_N, y_N)\}$, weak learning oracle $\mathcal{A}$
Initialize: $p_1(i) = 1/N$ for $i = 1, \ldots, N$
for $t = 1, \ldots, T$ **do**
  invoke the oracle to get $h_t = \mathcal{A}(S, p_t)$
  calculate weighted error $\epsilon_t = \mathbb{E}_{i\sim p}[\mathbf{1}\{h(x_i) \neq y_i\}]$, and $\alpha_t = \frac{1}{2}\ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right)$
  update distribution: $\forall i \in [N]$

$$p_{t+1}(i) \propto p_t(i) \times \begin{cases} e^{-\alpha_t} & \text{if } h_t(x_i) = y_i \\ e^{\alpha_t} & \text{else} \end{cases}$$

output the final classifier: $H(x) = \textsc{sgn}\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$.

all the weak classifiers ($\textsc{sgn}(z)$ is 1 if $z > 0$ and -1 otherwise), where the weight (α_t) for each h_t is calculated based on the error rate of h_t so that more accurate classifier has more weights in the final vote. (Note that the algorithm makes sense even when $\epsilon_t \geq 1/2$. Think about why.)

The first result about AdaBoost is that it drives the training error to zero exponentially fast.

Theorem 1. *After T rounds, the error rate of the final output of AdaBoost is bounded as*

$$\frac{1}{N}\sum_{i=1}^{N} \mathbf{1}\{H(x_i) \neq y_i\} \leq \exp\left(-2\sum_{t=1}^{T}\gamma_t^2\right) \tag{2}$$

where $\gamma_t = \frac{1}{2} - \epsilon_t$ is the edge of classifier h_t. Under the weak learning assumption, the error rate is then bounded by $\exp\left(-2T\gamma^2\right)$ and is 0 as long as $T > \frac{\ln N}{2\gamma^2}$.

Proof. Let $F(x) = \sum_{t=1}^{T} \alpha_t h_t(x)$ so that $H(x) = \textsc{sgn}(F(x))$. The first step is to realize 0-1 loss is bounded by the exponential loss

$$\frac{1}{N}\sum_{i=1}^{N} \mathbf{1}\{H(x_i) \neq y_i\} = \sum_{i=1}^{N} p_1(i)\mathbf{1}\left\{y_i F(x_i) \leq 0\right\} \leq \sum_{i=1}^{N} p_1(i)\exp\left(-y_i F(x_i)\right).$$

Now notice that the update rule of p_t can be written as $p_{t+1}(i) = p_t(i)\exp(-y_i\alpha_t h_t(x_i))/Z_t$ where Z_t is the normalization factor. We then have

$$p_{T+1}(i) = p_1(i)\prod_{t=1}^{T}\frac{\exp(-y_i\alpha_t h_t(x_i))}{Z_t} = \frac{p_1(i)\exp\left(-y_i F(x_i)\right)}{\prod_{t=1}^{T} Z_t}$$

and therefore the error rate is bounded by $\sum_{i=1}^{N}\left(p_{T+1}(i)\prod_{t=1}^{T} Z_t\right) = \prod_{t=1}^{T} Z_t$. It remains to bound each Z_t:

$$\begin{aligned} Z_t &= \sum_{i=1}^{N} p_t(i)\exp(-y_i\alpha_t h_t(x_i)) \\ &= \sum_{i:h_t(x_i)=y_i} p_t(i)\exp(-\alpha_t) + \sum_{i:h_t(x_i)\neq y_i} p_t(i)\exp(\alpha_t) \\ &= (1-\epsilon_t)\exp(-\alpha_t) + \epsilon_t\exp(\alpha_t) \\ &= \sqrt{1-4\gamma_t^2} \leq \exp(-2\gamma_t^2). \end{aligned}$$

where the last equality is by the definition of α_t (which is in fact chosen to minimize the expression), and the last step is by the fact $1 + z \leq e^z$. This finishes the proof for Eq. (2). Under weak learning assumption, we further have $\gamma_t \geq \gamma$ and thus the stated bound $\exp\left(-2T\gamma^2\right)$. Finally, note that as soon as the error rate drops below $1/N$, it must become zero. Therefore, as long as $\exp\left(-2T\gamma^2\right) < 1/N$, which means $T > \frac{\ln N}{2\gamma^2}$, AdaBoost ensures zero training error. □

Of course, at the end of the day one cares about the generalization error instead of the training error. Standard VC theory says that the difference between the generalization error and the training error is bounded by something like $\widetilde{\mathcal{O}}(\sqrt{C/N})$ where C is some complexity measure of the hypothesis space. For boosting, it is not so surprising that this complexity is growing as $O(T)$, since combining more weak classifiers leads to a more complicated final classifier H. Since we just proved that after $T > \frac{\ln N}{2\gamma^2}$ rounds the training error of AdaBoost is zero, it means that if we stop the algorithm at the right time, AdaBoost will have generalization error of order $\widetilde{\mathcal{O}}\left(\sqrt{\frac{\ln N}{\gamma^2 N}}\right)$, which can be arbitrarily small when N is large enough. In other words, under the weak learning assumption AdaBoost does ensure arbitrarily small generalization error given enough examples, implying that "boosting" is indeed possible, or in more technical terms, weak learnability is equivalent to strong learnability.

Preventing Overfitting. Like all other machine learning algorithms, AdaBoost could also overfit the training set. This is consistent with the VC theory: if we keep running the algorithm even after the training error has dropped to zero, then the generalization error is of order $\sqrt{T/N}$, which is increasing in the number of rounds T.

However, what usually happens in practice is that AdaBoost tends to prevent overfitting, in the sense that even if one keeps running the algorithm for many more rounds after the training error has dropped to zero, the generalization error still keeps decreasing. How should we explain this?

It turns out that the concept of *margin* is the key for understanding this phenomenon. The margin of an example (x, y) (with respect to the classifier H) is defined as $yf(x)$ where

$$f(x) = \frac{F(x)}{\sum_{t=1}^{T} \alpha_t} = \sum_{t=1}^{T} \left(\frac{\alpha_t}{\sum_{\tau=1}^{T} \alpha_\tau} \right) h_t(x).$$

It is clear that the margin is always in $[-1, 1]$, and the sign of the margin indicates whether the final classifier H makes a mistake on x or not. Specifically we want the margin to be positive in order to have low error rate. However, intuitively we also want the margin to be a large positive (that is, close to 1), since in this case there is a huge difference between the vote for $+1$ and -1 (recall H is just doing a weighted majority vote), and the decisive win of one label makes us feel more confident about the final prediction.

Indeed, margin theory says that for any θ, the generalization error of H and the fraction of training examples with margin at most θ are related as follows:

$$\mathbb{E}_{(x,y)\sim\mathcal{D}}[\mathbf{1}\{H(x) \neq y\}] \leq \frac{1}{N} \sum_{i=1}^{N} \mathbf{1}\{y_i f(x_i) \leq \theta\} + \widetilde{\mathcal{O}}\left(\frac{1}{\theta} \sqrt{\frac{C_{\mathcal{H}}}{N}} \right)$$

where $C_{\mathcal{H}}$ is the complexity of the hypothesis space $\mathcal{H}$ used by the oracle (recall $h_t \in \mathcal{H}$), which is independent of the number of weak classifiers combined by H (that is, T)! Therefore, if keep running AdaBoost has the effect of increasing margin even after the training error drops to zero, then this explains why overfitting does not happen. This is indeed true: under the weak learning assumption AdaBoost guarantees

$$\frac{1}{N} \sum_{i=1}^{N} \mathbf{1}\{y_i f(x_i) \leq \theta\} \leq \left(\sqrt{(1-2\gamma)^{1-\theta}(1+2\gamma)^{1+\theta}} \right)^T.$$

One way to interpret this bound is to note that as long as θ is such that $(1-2\gamma)^{1-\theta}(1+2\gamma)^{1+\theta} < 1$, which translates to $\theta \leq \Gamma(\gamma) \stackrel{\text{def}}{=} \frac{-\ln(1-4\gamma^2)}{\ln\left(\frac{1+2\gamma}{1-2\gamma}\right)} \leq 2\gamma$, then the fraction of examples with margin at most θ is eventually zero when T is large enough. In other words, if we keep running AdaBoost, eventually the smallest margin is $\Gamma(\gamma)$ and the generalization error is thus $\widetilde{\mathcal{O}}\left(\frac{1}{\Gamma(\gamma)} \sqrt{\frac{C_{\mathcal{H}}}{N}} \right)$.

As a final remark, interestingly AdaBoost is not doing the best possible job in maximizing the smallest margin. The best possible smallest margin under the weak learning assumption can be shown to be exactly 2γ.

2 Boosting via Online Learning

In this section we show a simple reduction from boosting to an expert problem. In other words, given an expert algorithm as a blackbox, we can turn it into a boosting algorithm, as shown below.

Algorithm 2: Boosting to Expert Problem

Input: training set $S = \{(x_1, y_1), \ldots, (x_N, y_N)\}$, weak learning oracle $\mathcal{A}$, expert algorithm $\mathcal{E}$
Initialize the expert algorithm $\mathcal{E}$ with N experts
for $t = 1, \ldots, T$ **do**
  let p_t be the prediction of $\mathcal{E}$ at round t
  invoke the oracle to get $h_t = \mathcal{A}(S, p_t)$
  feed the loss vector ℓ_t to $\mathcal{E}$ where $\ell_t(i) = \mathbf{1}\{h_t(x_i) = y_i\}$
output the final classifier: $H(x) = \textsc{sgn}\left(\sum_{t=1}^{T} h_t(x)\right)$.

While one might imagine that it is natural to treat each possible weak classifier $h \in \mathcal{H}$ as an expert, the reduction above actually treats each training example as an expert. Moreover, the reduction punishes an expert/example by assigning loss 1 if it is *correctly* classified, which seems counter-intuitive at first glance but is in fact consistent with the key idea of boosting, that is, more focus should be put on "hard" examples.

Now suppose the regret of $\mathcal{E}$ is $\mathcal{R}_T$, we have by construction

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{i=1}^{N} p_t(i)\mathbf{1}\{h_t(x_i) = y_i\} \le \frac{1}{T}\sum_{t=1}^{T}\mathbf{1}\{h_t(x_j) = y_j\} + \frac{\mathcal{R}_T}{T}$$

for any $j \in [N]$. On the other hand, reusing the notation $\epsilon_t = \mathbb{E}_{i\sim p}[\mathbf{1}\{h(x_i) \neq y_i\}]$ and $\gamma_t = 1/2 - \epsilon_t$, we have under the weak learning assumption

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{i=1}^{N} p_t(i)\mathbf{1}\{h_t(x_i) = y_i\} = \frac{1}{T}\sum_{t=1}^{T}(1-\epsilon_t) = \frac{1}{2} + \frac{1}{T}\sum_{t=1}^{T}\gamma_t \ge \frac{1}{2} + \gamma.$$

Combining leads to

$$\frac{1}{2} + \gamma \le \frac{1}{T}\sum_{t=1}^{T}\mathbf{1}\{h_t(x_j) = y_j\} + \frac{\mathcal{R}_T}{T}. \tag{3}$$

Therefore, as long as $\gamma > \frac{\mathcal{R}_T}{T}$, we have $\frac{1}{2} < \frac{1}{T}\sum_{t=1}^{T}\mathbf{1}\{h_t(x_j) = y_j\}$, implying that more than half of the weak classifiers predicts correctly. Since the final classifier H is a simple majority vote, this also implies that H has zero training error. Plugging the optimal regret $\mathcal{R}_T = \mathcal{O}(\sqrt{T\ln N})$ (for example by using Hedge), we thus only need $T > \mathcal{O}(\frac{\ln N}{\gamma^2})$ rounds to achieve zero training error, same as AdaBoost.

Moreover, since $\mathbf{1}\{h(x) = y\} = \frac{1}{2}(yh(x) + 1)$, plugging this fact into Eq. (3) and simplifying lead to (with $f(x) = \frac{1}{T}\sum_{t=1}^{T} h_t(x)$)

$$2\gamma - \frac{2\mathcal{R}_T}{T} \le y_j f(x_j).$$

Note that $y_j f(x_j)$ is exactly the margin for example x_j. The above is therefore saying that if $\mathcal{R}_T$ is sublinear and we keep running the algorithm, eventually every example will have margin above something arbitrarily close to 2γ, the optimal margin as mentioned in the last section. In other words, this expert-algorithm-turned boosting is doing an even better job in maximizing margin compared to AdaBoost.

Finally, we point out how adaptive quantile regret bound can say something more meaningful in this context. For any margin threshold θ, what does the above derivation tell us about the fraction of examples with margin at most θ, that is, $\frac{1}{N}\sum_{i=1}^{N}\mathbf{1}\{y_i f(x_i) \le \theta\}$? In fact, it only says that if $\theta \le 2\gamma - \frac{2\mathcal{R}_T}{T}$ then the fraction is zero, otherwise it could be as large as 1, a trivial bound. However suppose the expert algorithm admits quantile regret bounds (such as Squint), then assuming

$y_1 f(x_1) \leq \cdots \leq y_N f(x_N)$ without loss of generality, we have

$$2\gamma - \mathcal{O}\left(\sqrt{\frac{\ln(N/j)}{T}}\right) \leq y_j f(x_j).$$

Therefore for any $\theta < 2\gamma$, we can just find the largest j_θ such that $2\gamma - \mathcal{O}\left(\sqrt{\frac{\ln(N/j_\theta)}{T}}\right) \leq \theta$, and assert that the fraction of examples with margin at most θ will be j_θ/N, which is of order $\exp(-\Omega(T(2\gamma-\theta)^2))$.

References

Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, August 1997.

Robert E Schapire. The strength of weak learnability. *Machine learning*, 5(2):197–227, 1990.