COS 511: Theoretical Machine Learning

Lecturer: Rob Schapire
Scribe: Chong Wang

Lecture #14
March 26, 2008

1 Review of Last Lecture

Recall the *online learning model* we discussed in the previous lecture:

$N = \#$ experts

For $t = 1, 2, \ldots, T$ rounds:

1) each expert i, $1 \le i \le N$, makes a prediction $\xi_i \in \{0, 1\}$

2) learner makes a prediction $\hat{y} \in \{0, 1\}$

3) observe outcome $y \in \{0, 1\}$ (a mistake happens if $\hat{y} \neq y$)

With this framework in hand, we investigated a particular algorithm, *Weighted Majority Algorithm* (WMA), as follows:

N = # experts

Initially $w_i = 1$, $1 \le i \le N$

For $t = 1, 2, \ldots, T$ rounds:

1) each expert i, $1 \le i \le N$, makes a prediction $\xi_i \in \{0, 1\}$

2) calculate $q_0 = \sum_{i:\xi_i=0} w_i$ and $q_1 = \sum_{i:\xi_i=1} w_i$

3) learner makes a prediction $\hat{y} = \begin{cases} 1 & \text{if } q_1 > q_0 \\ 0 & \text{else} \end{cases}$

4) observe outcome $y \in \{0, 1\}$ (a mistake happens if $\hat{y} \neq y$)

5) $\forall i$, if $\xi_i \neq y$, then $w_i \leftarrow w_i \beta$, where $\beta \in [0, 1]$.

For WMA, we have the following theorem:

Theorem 1 *For WMA, we have*

$$(\#\textit{mistakes of learner}) \le a_\beta (\#\textit{mistakes of the best expert}) + c_\beta \lg N,$$

where

$$a_\beta = \frac{\lg(1/\beta)}{\lg(2/(1+\beta))}, \quad c_\beta = \frac{1}{\lg(2/(1+\beta))}.$$

Now, let's take a deep look at a_β. It is not difficult to see that $a_\beta \ge 2$. This means, if the best expert makes more than 25% mistakes, this bound becomes trivial, since random guessing has 50% chance to be correct. So we really want a_β to be close to 1 and introducing randomness is one of the ways to go.

2 Randomized Weighted Majority Algorithm (RWMA)

Different from WMA, the *Randomized Weighted Majority Algorithm* (RWMA) predicts the outcome in a *random* way. The predictions made by the learner are randomized. Let $W = \sum_i w_i = q_0 + q_1$, RWMA predicts as

$$\hat{y} = \begin{cases} 1 & \text{with probability } \frac{q_1}{W} \\ 0 & \text{with probability } \frac{q_0}{W}, \end{cases}$$

where RWMA computes the fraction of the experts predicting positive or negative, and predicts randomly according to that fraction. This is also equivalent to choosing expert i with probability w_i/W, and predicting what that expert says.

The following theorem states the upper bound of the expected number of mistakes of RWMA.

Theorem 2 *For RWMA, we have*

$$E[\#\textit{mistakes of learner}] \leq a_\beta(\#\textit{mistakes of the best expert}) + c_\beta \ln N,$$

where

$$a_\beta = \frac{\ln(1/\beta)}{1-\beta}, \quad c_\beta = \frac{1}{1-\beta}.$$

We note that the expectation is taken over the randomization of the learning algorithm. All the others are the same as WMA and not random. The good thing here is that $a_\beta \to 1$ when $\beta \to 1$. This means the expected number of mistakes will not be much larger than the best expert. We also note $c_\beta \to \infty$ when $\beta \to 1$. We will discuss how to select β for tradeoff.

Proof: For a particular round $t, 1 \leq t \leq T$, let

$$\ell = \text{probability of the learner making mistakes} = \frac{\sum_{i:\xi_i \neq y} w_i}{W}.$$

Then,

$$W_{new} = \sum_{i:\xi_i \neq y} w_i\beta + \sum_{i:\xi_i = y} w_i = \ell W\beta + W(1-\ell) = W(1-\ell(1-\beta)).$$

Let ℓ_t be the probability of the learner making mistakes on round t. Considering all T rounds, we obtain an upper bound for W_{final},

$$\begin{aligned} W_{final} &= N\prod_{t=1}^{T}(1-\ell_t(1-\beta)) \\ &\leq N\prod_{t=1}^{T}\exp\left(-\ell_t(1-\beta)\right) \text{ (according to } 1-x \leq e^{-x}) \\ &= N\exp\left(-(1-\beta)\sum_{t=1}^{T}\ell_t\right). \end{aligned}$$

Let $L_A = \sum_{t=1}^{T}\ell_t = E[\#\text{mistakes of the learner}]$. Now let L_i be the number of mistakes made by expert i. We have

$$\beta^{L_i} \leq W_{final} \leq N\exp\left(-(1-\beta)L_A\right).$$

Solving for L_A, we have, $\forall i$

$$L_A \leq \frac{L_i\ln(1/\beta) + \ln N}{1-\beta}.$$

This means

$$L_A \leq \frac{\min_i L_i\ln(1/\beta) + \ln N}{1-\beta} = a_\beta \min_i L_i + c_\beta \ln N. \blacksquare$$

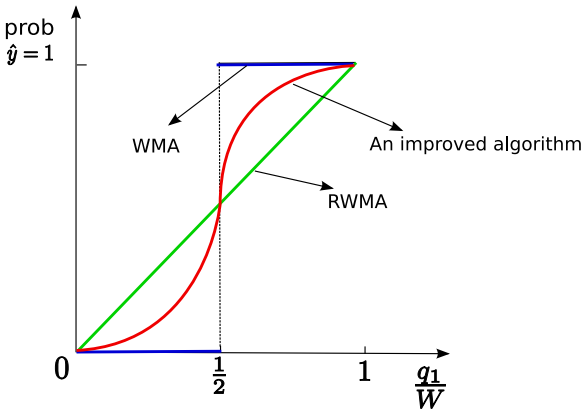

Figure 1: Comparisons of WMA, RWMA and an improved algorithm.

The choice of β: suppose we know the best expert makes no more than K mistakes, that is $\min_i L_i \leq K$. Without the proof, we set

$$\beta = \frac{1}{1+\sqrt{\frac{2\ln N}{K}}}.$$

Then, we have

$$L_A \leq \min_i L_i + \sqrt{2K\ln N} + \ln N.$$

This bound can be further improved if we select a better prediction strategy. Figure 1 shows how we can improve the algorithm. The horizontal axis is the fraction of the experts predicting positive, q_1/W, and the vertical axis is the probability of $\hat{y}=1$. So the blue curve describes the WMA, where, if $q_1/W > 1/2$, $\text{Prob}[\hat{y}=1]=1$, otherwise $\text{Prob}[\hat{y}=1]=0$. The green curve describes the RWMA, where $\text{Prob}[\hat{y}=1]=q_1/W$. If we choose the red curve between WMA and RWMA, we can potentially improve the bound.

After carefully designing the prediction strategy, we can have

$$L_A \leq \min_i L_i + \sqrt{2K\ln N} + \frac{\lg N}{2}.$$

So if we have an perfect expert ($K=0$), then the expected mistakes will be less than $(\lg N)/2$ (see homework).

Now suppose $\min_i L_i \leq K = rT$. Here r can be thought of as the rate at which the best expert makes mistakes. We have

$$\frac{L_A}{T} \leq \min_i \frac{L_i}{T} + \sqrt{\frac{r\ln N}{T}} + \frac{\lg N}{2T}.$$

Here are some observations and intuitions. As $T\to\infty$,

$$\sqrt{\frac{r\ln N}{T}} \to 0 \text{ and } \frac{\lg N}{2T} \to 0.$$

This means the learner will eventually behave like the best expert as more and more rounds happen. Furthermore,

$$\text{convergence rate} \sim \begin{cases} O(1/T) & \text{if } r = o(1) \\ O(1/\sqrt{T}) & \text{otherwise.} \end{cases}$$

We can almost always choose r to be $1/2$, since we can have one expert always predict 1, another expert always predict 0. The number of mistakes of one of them must be no more than $T/2$. Thus, setting $r = 1/2$, we have

$$\frac{L_A}{T} \leq \min_i \frac{L_i}{T} + \sqrt{\frac{\ln N}{2T}} + \frac{\lg N}{2T}.$$

Finally, let's see why we do not lose anything by allowing the data to be non-random. Assume that all the experts predict at random and the outcomes are also random:

$$\xi_i = \begin{cases} 1 & \text{with probability } 1/2 \\ 0 & \text{with probability } 1/2 \end{cases}$$

$$y = \begin{cases} 1 & \text{with probability } 1/2 \\ 0 & \text{with probability } 1/2. \end{cases}$$

For any algorithm A, it is easy to see $E[L_A] = T/2$, since no matter what the algorithm predicts, the probability of being right is $1/2$. Similarly, we have $E[L_i] = T/2$. Now let's look at $\min_i L_i$. We note that it is a random variable, the expectation of $\min_i L_i$ can be shown to be

$$E\left[\min_i L_i\right] \approx \frac{T}{2} - \sqrt{\frac{T \ln N}{2}}.$$

This means, for any learning algorithm,

$$E\left[L_A\right] \gtrsim E\left[\min_i L_i\right] + \sqrt{\frac{T \ln N}{2}}.$$

Note this is the *lower bound* of $E\left[L_A\right]$. If T is large, this lower bound is quite close to the upper bound of unrandomized experts and outcomes even up to constants. Thus, RWMA is very close to the best possible and what's more, the case of random experts is actually the worst case.

3 Perceptron Algorithm

We are going to discuss the Perceptron algorithm. Although it is a very old algorithm, it is still very effective and useful. Consider the online learning algorithm we discussed before. Can we use a combined result from several experts, instead of a single best expert?

To better describe the Perceptron algorithm, we change our notations as follows:
$N = \#$ experts
For $t = 1, 2, \ldots, T$ rounds:
get $\mathbf{x}_t \in \{-1, +1\}^N$
learner predicts $\hat{y}_t \in \{-1, +1\}$
observe the outcome $y_t \in \{-1, +1\}$.

We note that each component of $\mathbf{x}_t$ can be viewed as the prediction of an expert. However, in the Perceptron algorithm, we allow these to be any real value, i.e. $\mathbf{x}_t \in \mathbb{R}^N$.

We assume there is some weighted combination of experts that gives perfect predictions. That is we assume that $\exists \mathbf{u} \in \mathbb{R}^N$, $\forall t, y_t = \text{sign}(\mathbf{u} \cdot \mathbf{x}_t)$ (i.e. $y_t(\mathbf{u} \cdot \mathbf{x}_t) > 0$). This means the examples $(\mathbf{x}_t, y_t)$ are linearly separable. In fact, we allow some of the experts to have negative weights. Figure 2 shows the idea.

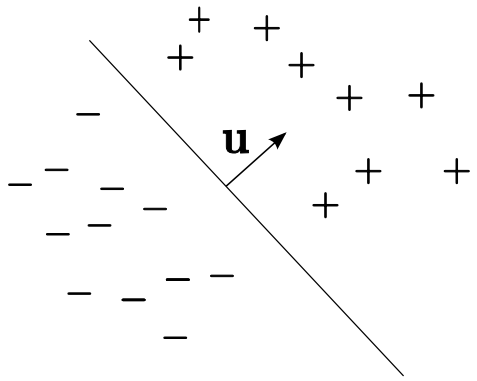

Figure 2: The separating hyperplane

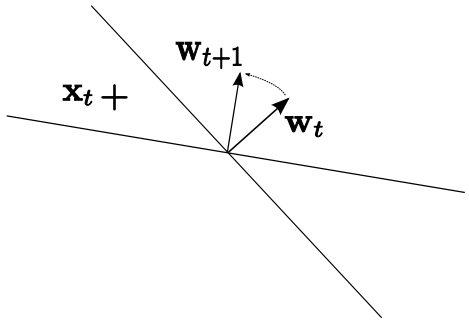

Figure 3: The Perceptron algorithm

Now we introduce the framework of learning the hyperplane in Figure 2. The algorithm works by maintaining its own weight vector $\mathbf{w}_t$. In general, we will consider algorithms with the following structure:

Initialize $\mathbf{w}_1$

for $t = 1, 2, \cdots, T$

predict $\hat{y}_t = \text{sign}(\mathbf{w}_t \cdot \mathbf{x}_t)$

update $\mathbf{w}_{t+1} = F(\mathbf{w}_t, \mathbf{x}_t, y_t)$.

The key idea of the framework is how we choose the update function F. Now we begin to introduce the *Perceptron algorithm*.

Initialize $\mathbf{w}_1 = \mathbf{0}$

update:

if $\hat{y}_t \neq y_t$, where $\hat{y}_t = \text{sign}(\mathbf{w}_t \cdot \mathbf{x}_t)$

$\mathbf{w}_{t+1} = \mathbf{w}_t + y_t \mathbf{x}_t$

else $\mathbf{w}_{t+1} = \mathbf{w}_t$.

Figure 3 gives a geometrical explanation of the Perceptron algorithm. If the $(\mathbf{x}_t, y_t)$ (in Figure 3, $y_t = 1$) is misclassified, adding $y_t \mathbf{x}_t$ to $\mathbf{w}_t$ moves $\mathbf{w}_t$ to the direction which is likely to classify $(\mathbf{x}_t, y_t)$ correctly next time.

Without loss of generality, we assume

- There is a mistake in every round so that $T = \#$ mistakes. This is because if on a certain round there is no mistake, then the Perceptron algorithm does nothing.

- $||\mathbf{x}_t||_2 \leq 1$. This is because the value of sign$(\cdot)$ will not be affected by this normalization.

Furthermore, *with* loss of generality, we assume that there exists

$$\begin{aligned} &\mathbf{u}, \delta > 0 \\ &\text{s.t. } ||\mathbf{u}|| = 1, y_t(\mathbf{u} \cdot \mathbf{x}_t) \geq \delta > 0. \end{aligned}$$

In other words, we assume the data are linearly separable with margin at least δ.

Theorem 3 *Under the assumptions above, T = # mistakes, we have*

$$T \leq \frac{1}{\delta^2}.$$

Proof: First, we define

$$\Phi_t = \cos(\text{angle between } \mathbf{u} \text{ and } \mathbf{w}_t) = \frac{\mathbf{w}_t \cdot \mathbf{u}}{||\mathbf{w}_t||_2} \leq 1.$$

Step 1: Prove $\mathbf{w}_{T+1} \cdot \mathbf{u} \geq T\delta$.

Proof:

$$\begin{aligned} \mathbf{w}_{t+1} \cdot \mathbf{u} &= (\mathbf{w}_t + y_t \mathbf{x}_t) \cdot \mathbf{u} \\ &= \mathbf{w}_t \cdot \mathbf{u} + y_t(\mathbf{u} \cdot \mathbf{x}_t). \end{aligned}$$

According to our assumption, $y_t(\mathbf{u} \cdot \mathbf{x}_t) \geq \delta$. Applying repeatedly, we have $\mathbf{w}_{T+1} \cdot \mathbf{u} \geq \mathbf{w}_1 + T\delta$. Since $\mathbf{w}_1 = \mathbf{0}$, we obtain $\mathbf{w}_{T+1} \cdot \mathbf{u} \geq T\delta$.

Step 2: Prove $||\mathbf{w}_{T+1}||_2^2 \leq T$.

Proof:

$$\begin{aligned} ||\mathbf{w}_{t+1}||_2^2 &= ||\mathbf{w}_t + y_t \mathbf{x}_t||_2^2 \\ &= (\mathbf{w}_t + y_t \mathbf{x}_t) \cdot (\mathbf{w}_t + y_t \mathbf{x}_t) \\ &= ||\mathbf{w}_t||_2^2 + 2y_t \mathbf{w}_t \cdot \mathbf{x}_t + y_t^2 ||\mathbf{x}_t||_2^2. \end{aligned}$$

Since the algorithm makes a mistake at each round, we have $y_t \mathbf{w}_t \cdot \mathbf{x}_t \leq 0$. Also, we assume $||\mathbf{x}_t||_2 \leq 1$, then $y_t^2 ||\mathbf{x}_t||_2^2 \leq 1$. Thus

$$||\mathbf{w}_{t+1}||_2^2 = ||\mathbf{w}_t||_2^2 + 2y_t \mathbf{w}_t \cdot \mathbf{x}_t + y_t^2 ||\mathbf{x}_t||_2^2 \leq ||\mathbf{w}_t||_2^2 + 1.$$

Applying repeatedly, we have $||\mathbf{w}_{T+1}||_2^2 \leq ||\mathbf{w}_1||_2^2 + T$. Since $\mathbf{w}_1 = \mathbf{0}$, we obtain $||\mathbf{w}_{T+1}||_2^2 \leq T$.

Now considering both results from steps 1 and 2, we know

$$1 \geq \Phi_{T+1} = \frac{\mathbf{w}_t \cdot \mathbf{u}}{||\mathbf{w}_t||_2} \geq \frac{T\delta}{\sqrt{T}} \Rightarrow T \leq \frac{1}{\delta^2}. \blacksquare$$

Finally we talk a little about the VC-dimension of hyperplanes with margin at least δ. Let $\mathcal{H}$ be the concept space and $M_A(\mathcal{H})$ be the number of mistakes made by A when learning. In the previous lecture, we proved that, for a deterministic online learning algorithm A,

$$\text{VC-dim}(\mathcal{H}) \leq \min_A M_A(\mathcal{H}).$$

Since this is for any A, this also applies to the Perceptron algorithm. This means,

$$\begin{aligned} \text{VC-dim}(\mathcal{H}) &\leq \text{\# mistakes by the Perceptron algorithm} \\ &\leq \frac{1}{\delta^2}. \end{aligned}$$

Thus, the VC-dimension of hyperplanes with margin at least δ is at most $1/\delta^2$.