Lecture 16

The Multiplicative Weights Algorithm*

In the next couple lectures, we will devise modular, iterative algorithms for solving LPs and SDPs. "Multiplicative weights" is a retronym for the simple iterative rule underlying these algorithms; it is known by different names in the various fields where it was (re)discovered. Check out the survey by Arora, Hazan and Kale [AHK05]; our discussion will be based on their treatment. Due to its broad appeal, we will consider multiplicative weights in more generality than we need for solving LPs and SDPs. In this lecture, we'll introduce some strategies for playing a prediction game. We'll tweak the game to suit our optimization needs. Finally, we'll play the tweaked game with a strategy called Hedge.

16.1 Warmup: prediction with expert advice

The following sequential game is played between an omniscient Adversary and an Aggregator who is advised by N experts. Special cases of this game include predicting if it will rain tomorrow, or if the stock market will go up or down.

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

1. Each expert $i \in [N]$ advises either yes or no.
2. Aggregator predicts either yes or no.
3. Adversary, with knowledge of the expert advice and Aggregator's prediction, decides the yes/no outcome.
4. Aggregator observes the outcome and suffers if his prediction was incorrect.

Naturally, Aggregator wants to make as few mistakes as possible. Since the experts may be unhelpful and the outcomes may be capricious, Aggregator can hope only for a relative

*Lecturer: Anupam Gupta. Scribe: Shiva Kaul.

performance guarantee. In particular, Aggregator hopes to do as well as the best single expert in hindsight [1]. In order to do so, Aggregator must track which experts are helpful. We will consider a few tracking strategies. Almost every other aspect of the game - that advice is aggregated into a single value, that this value is binary, and even that the game is sequential - is not relevant; we will generalize or eliminate these aspects.

If there is a perfect expert, then an obvious strategy is to dismiss experts who aren't perfect. With the remaining experts, take a majority vote. Every time Aggregator makes a mistake, at least half of the remaining experts are dismissed, so Aggregator makes at most $\log_2 N$ mistakes. We can use the same strategy even when there isn't a perfect expert, if we restart after every expert has been eliminated. If the best expert has made M mistakes by time T, then Aggregator has restarted at most $M+1$ times, so it has made at most $(M+1)\log_2 N$ mistakes. This bound is rather poor since it depends multiplicatively on M.

16.1.1 Fewer mistakes with Weighted Majority

We may obtain an additive mistake bound by softening our strategy: instead of dismissing experts who erred, discount their advice. This leads to the Weighted Majority algorithm of Littlestone and Warmuth [LW89]. Assign each expert i a weight $w_i^{(1)}$ initialized to 1. Thereafter, for every t:

- Predict yes/no based on a weighted majority vote per $\vec{w}^{(t)} = (w_1^{(t)}, \ldots, w_N^{(t)})$
- After observing the outcome, for every mistaken expert i, set $w_i^{(t+1)} = w_i^{(t)}/2$

Theorem 16.1. *For any sequence of outcomes, duration T and expert i,*

$$\textit{\# of WM mistakes} \leq 2.41 \cdot (\textit{\# of i's mistakes} + \log_2 N)$$

Proof. Let

$$\Phi^{(t)} = \sum_{i \in [N]} w_i^{(t)}$$

be a 'potential' function. Observe the following facts:

- By definition, $\Phi^{(1)} = N$
- Also by definition, $\frac{1}{2}^{\text{\# of } i\text{'s mistakes}} \leq \Phi^{(T+1)}$
- At any τ when WM errs, at least half of the weight gets halved:

$$\Phi^{(\tau+1)} \leq \frac{3}{4}\Phi^{(\tau)}$$

This implies

$$\Phi^{(T+1)} \leq \frac{3}{4}^{\text{\# of WM mistakes}} \cdot \Phi^{(1)}$$

[1] The excess number of mistakes is called (external) regret.

Combining these facts yields

$$\frac{1}{2}^{\text{\# of } i\text{'s mistakes}} \leq \frac{3}{4}^{\text{\# of WM mistakes}} \cdot N$$

Taking logarithms of both sides,

$$-(\text{\# of } i\text{'s mistakes}) \leq \log_2 N + \log_2(3/4) \cdot \text{\# of WM mistakes}$$

so finally

$$\text{\# of WM mistakes} \leq \frac{1}{\log_2(4/3)} \cdot \Big(\text{\# of } i\text{'s mistakes} + \log_2 N\Big)$$

□

The unseemly leading constant is a consequence of our arbitrary choice to halve the weights. If we optimize ϵ in the update rule

$$w_i^{(t+1)} = w_i^{(t)}/(1+\epsilon)$$

then we may achieve

$$\text{\# of WM mistakes} \leq 2(1+\epsilon) \cdot (\text{\# of } i\text{'s mistakes}) + O(\log N/\epsilon).$$

16.2 Tweaking the game

We now modify the game with a view to solving LPs and SDPs. We perform these modifications individually in order to dispel some seductive misconceptions about the new game's meaning. The impervious (or impatient) reader may skip to the game description at the end of the section.

The first modification bakes weighting into the game.

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

1. Each expert $i \in [N]$ advises either yes or no.
2. Allocator picks some distribution $\vec{p}^{(t)} = (p_1^{(t)}, \ldots, p_N^{(t)})$ over the experts.
3. Adversary, with knowledge of the expert advice and $\vec{p}^{(t)}$, decides the yes/no outcome.
4. Allocator observes the outcome.
5. A single expert is sampled from $\vec{p}^{(t)}$ but isn't revealed to either Allocator or Adversary. Allocator suffers if this expert errs.

Let $m_i^{(t)}$ be 1 if expert i erred at t, and 0 otherwise. The new goal is to bound his total expected number of mistakes

$$\sum_t \vec{p}^{(t)} \cdot \vec{m}^{(t)} \tag{16.1}$$

in terms of the total number of mistakes made by any single expert

$$\sum_t m_i^{(t)}$$

Note the sampled expert isn't revealed to either party. By arguments posted on the blog [Gup11], Adversary may declare the entire sequence of outcomes in advance without losing any power. Eliminating the sequential nature of the game was on our agenda.

The attentive reader recalls that eliminating the aggregation step was also on our agenda. Yet this section has introduced a new aggregation step: randomly choosing a single expert rather than taking a deterministic weighted-majority vote. The truly important change was not randomized aggregation, but rather Allocator's new goal of minimizing 16.1. This quantity may be interpreted as the expected number of mistakes of a randomized Aggregator, but it is still well-defined even if there's no aggregation [2]. We consider $\vec{p}^{(t)}$ to be chosen deterministically; randomized aggregation may be layered on top.

Finally, we replace binary mistakes with continuous costs. Rather than returning a yes/no outcome which induces a mistake vector in $\{0,1\}^N$, Adversary returns a cost vector in $[-1,1]^N$. Negative cost may be interpreted as benefit. As we will see, $[-\rho,\rho]$ could work as well. Congruently, each expert advises some value in $[-1,1]$ rather than yes/no.

In summary, the game proceeds as follows.

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

1. Each expert $i \in [N]$ advises some value in $[-1,1]$.
2. Allocator picks some distribution $\vec{p}^{(t)} = (p_1^{(t)}, \ldots, p_N^{(t)})$ over the experts.
3. Adversary, with knowledge of the expert advice and $\vec{p}^{(t)}$, determines a cost vector $\vec{m}^{(t)} = (m_1^{(t)}, \ldots, m_N^{(t)}) \in [-1,1]^N$.
4. Allocator observes the cost vector and suffers $\vec{p}^{(t)} \cdot \vec{m}^{(t)}$.

16.3 Hedge and a useful corollary

We play the new game with the Hedge strategy of Freund and Schapire [FS97]. Its exponential update rule distinguishes it from Weighted Majority. Assign each expert i a weight $w_i^{(1)}$ initialized to 1. At each time t:

[2] It's also still useful. In the next lecture, the 'experts' correspond to individual constraints of an LP or SDP. Higher weight is given to constraints satisfied by thin margins. The convex combination of constraints is a single 'summary' constraint which emphasizes the difficult constraints. Reducing many constraints to a single summary constraint will be algorithmically useful.

- Pick the distribution $p_j^{(t)} = w_j^{(t)}/\Phi^{(t)}$
- After observing the cost vector, set $w_i^{(t+1)} = w_i^{(t)} \cdot \exp(-\epsilon \cdot m_i^{(t)})$

The following theorem may be interpreted as "the total expected cost of Hedge is not much worse than the total cost of any individual expert."

Theorem 16.2. *Suppose $\epsilon \leq 1$ and for $t \in [T]$, $\vec{p}^{(t)}$ is picked by Hedge. Then for any expert i,*

$$\sum_{t \leq T} \vec{p}^{(t)} \cdot \vec{m}^{(t)} \leq \sum_{t \leq T} m_i^{(t)} + \frac{\ln N}{\epsilon} + \epsilon T$$

Proof. This proof also involves the potential function Φ. By definition,

- $\Phi^{(1)} = N$.
- $\Phi^{(T+1)} \geq w_i^{(T+1)} = \exp(-\epsilon \sum_{t \leq T} m_i^{(t)})$

Again by definition,

$$\begin{aligned}\Phi^{(t+1)} =& \sum_j w_j^{(t+1)} \\ =& \sum_j w_j^{(t)} \cdot \exp(-\epsilon m_j^{(t)})\end{aligned}$$

The exponentiated term is in $[-1, 1]$. Since $e^x \leq 1 + x + x^2$ for $x \in [-1, 1]$,

$$\begin{aligned}\leq& \sum_j w_j^{(t)}(1 - \epsilon m_j^{(t)} + \epsilon^2 (m_j^{(t)})^2) \\ \leq& \sum_j w_j^{(t)}(1 - \epsilon m_j^{(t)} + \epsilon^2) \\ =& \sum_j w_j^{(t)}(1 + \epsilon^2) - \sum_j w_j^{(t)} \cdot \epsilon \cdot m_j^{(t)} \\ =& \Phi^{(t)}(1 + \epsilon^2) - \epsilon \sum_j \Phi^{(t)} \cdot p_j^{(t)} \cdot m_j^{(t)} \\ =& \Phi^{(t)}(1 + \epsilon^2 - \epsilon(\vec{p}^{(t)} \cdot \vec{m}^{(t)})) \\ \leq& \Phi^{(t)} \cdot \exp(\epsilon^2 - \epsilon \vec{p}^{(t)} \cdot \vec{m}^{(t)})\end{aligned}$$

Combining these statements yields

$$\exp(-\epsilon \sum_t m_i^{(t)}) \leq \Phi^{(T+1)} \leq \Phi^{(1)} \cdot \exp(\epsilon^2 T - \epsilon \sum_t \vec{p}^{(t)} \cdot \vec{m}^{(t)})$$

Taking (natural) logarithms,

$$-\epsilon \sum_t m_i^{(t)} \leq \ln \Phi^{(1)} + \epsilon^2 T - \epsilon \sum_t \vec{p}^{(t)} \cdot \vec{m}^{(t)})$$

The final result follows after some rearranging. □

In the next lecture, we will use an 'average cost' corollary of the previous result.

Corollary 16.3. *Suppose $\epsilon \leq 1$ and for $t \in [T]$, $p^{(t)}$ is picked by Hedge in response to cost vectors $\vec{m}^{(t)} \in [-\rho, \rho]^N$. If $T \geq (4\rho^2 \ln N)/\epsilon^2$, then for any expert i:*

$$\frac{1}{T} \sum_t \vec{p}^{(t)} \cdot \vec{m}^{(t)} \leq \frac{1}{T} \sum_t m_i^{(t)} + 2\epsilon$$

Its extension to cost vectors in $[-\rho, \rho]^N$ is simple: run Hedge on cost vectors normalized within $[-1, 1]$, and then scale up the bound.

16.3.1 Multiplicative weights

For completeness, we will mention the update rule which is most closely associated with the term 'multiplicative weights':

$$w_i^{(t+1)} = w_i^{(t)}(1 - \epsilon m_i^{(t)})$$

This update rule achieves a mistake bound of:

$$\sum_{t \leq T} \vec{p}^{(t)} \cdot \vec{m}^{(t)} \leq \sum_{t \leq T} m_i^{(t)} + \frac{\ln N}{\epsilon} + \epsilon \sum_t |m_i^{(t)}|$$

Since $\sum_t |m_i^{(t)}|$ may be smaller than T, this improves upon Hedge for benign cost vectors.

Bibliography

[AHK05] Sanjeev Arora, Elad Hazan, and Satyen Kale. The multiplicative weights update method: a meta algorithm and applications. Technical report, Princeton University, 2005. 16

[FS97] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *J. Comput. Syst. Sci.*, 55:119–139, August 1997. 16.3

[Gup11] Anupam Gupta. Lecture #17: Multiplicative weights, discussion. `http://lpsdp.wordpress.com/2011/11/09/lecture-17-multiplicative-weights-discussion/`, 2011. 16.2

[LW89] Nick Littlestone and Manfred K. Warmuth. The weighted majority algorithm. In *FOCS*, pages 256–261, 1989. 16.1.1