15-451/651: Design & Analysis of Algorithms April 4, 2019
Lecture #19: Prediction from Expert Advice last changed: April 1, 2019

1 Prediction with Expert Advice

Today we'll study the problem of making predictions based on expert advice. There are n "experts" (who are just people with opinions, the quotes suggesting that they may or may not have any expertise in the matter). Each day the following sequence of events happens:

- We see the n experts' predictions of the outcome.
- We make our own prediction about the outcome.
- In parallel, the actual outcome is revealed.
- We are correct if we made the right prediction, and make a mistake otherwise.

This process goes on indefinitely. Our goal: at any time (say after some T days), if the best of the experts so far has made only few mistakes, we want to have made "not too many more" mistakes. We want to bound the number of mistakes, and hence this is called the "mistake-bound" model.

2 Warmup: Simple Strategies

To start off, say we have just "Up/Down" predictions. (If the market will go up or down, or if the weather will be fair or not.)

Suppose we know the best expert makes no mistakes. Can we hope to make only a few mistakes? Here's a strategy that makes only $\lfloor \log_2 n \rfloor$ mistakes. Just predict what the majority of the remaining experts predicts. (In case of a tie, choose arbitrarily.) When an expert is incorrect, discard him. So each time we make a mistake, we reduce by the number of experts by at least $1/2$. This means after $\log_2 n$ mistakes we will be left with the perfect expert.

> **Exercise:** Argue that you cannot get an algorithm that makes fewer than $\log_2 n$ mistakes in this setting.

Suppose the best expert makes at most M mistakes on some sequence. Can we hope to make only a few mistakes? Here's a strategy that makes only $(M + 1)(\log_2 n + 1)$ mistakes.[1] Run the above majority-and-halving strategy, but when you have discarded all the experts, bring them all back (call this the beginning of a new "phase"), and continue. Note that in each phase each expert makes at least one mistake, and you made $\log_2 n + 1$ mistakes. Hence, if the best expert makes only M mistakes, there would be at most M finished phases (plus the last unfinished one), and hence at most $(M + 1)(\log_2 n + 1)$ mistakes in all.

> **Exercise:** Suppose the predictions belonged to some set of K items. (E.g., you could say "cloudy", "sunny", "rain", "snow".) Give algorithms showing the bound of $O(M \log n)$ mistakes still holds.

[1] Let's assume n is a power of 2.

3 The Weighted Majority/Multiplicative Weights Algorithm

Throwing away an expert when he makes a mistake seems too drastic. Suppose we instead assign weights w_j to the experts, sum the weights of the expert saying `Up`, sum the weights of the of the expert saying `Down`, and predict the outcome with greater weight. (This is called the *weighted majority* rule, since we are following the advice of the experts that form the weighted majority.)

Then once we see the outcome, we can reduce the weight of the experts who were wrong. In the above algorithm, we were zeroing out the weight, but suppose we are gentler?

The *(basic) deterministic weighted majority* algorithm does the following:

> Start with each expert having weight 1. Each time an expert makes a mistake, half his weight. Output the prediction of the experts who form the weighted majority.

Remarkably, we can get a much stronger result now.

Theorem 1 *If on some sequence of days, the best expert makes M mistakes. The basic deterministic weighted majority algorithm makes $\leq 2.41(M + \log_2 n)$ mistakes.*

Proof: Let $\Phi := \sum_{i=1}^{n} w_i$ be the sum of weights of the n experts. Note that initially $\Phi = n$. Moreover, we claim that each time we make a mistake

$$\Phi_{new} \leq \frac{3}{4}\Phi_{old}.$$

Indeed, at least half the weight (which was making the majority prediction) gets halved (because it made a mistake), so we lose at least a quarter of the weight with each mistake. (Also if we don't make a mistake, Φ does not increase.) So if we've made m mistakes at some point, the total weight is at most

$$\Phi_{final} \leq (3/4)^m \cdot \Phi_{init} = (3/4)^m \cdot n.$$

Moreover, if the best expert i^* has made M mistakes, then $\Phi_{final} \geq w_{i^*} = (1/2)^M$. So

$$(1/2)^M \leq (3/4)^m \cdot n \quad \Rightarrow \quad (4/3)^m \leq n2^M.$$

Taking logs (base 2) and noting that $\frac{1}{\log_2(4/3)} = 2.41\ldots$ completes the proof. ∎

This is pretty cool! If the best expert makes mistakes 10% of the time we err about 24% of the time (plus $O(\log n)$, but since this depends only on the number of experts, it will be a negligible fraction as $M \gg \log n$). We can improve the 2.41 factor down to as close to 2 as we want, as the next exercise shows.

> **Exercise*:** Show that if we reduce the weight not by 1/2 but by $(1-\epsilon)$ for some $\epsilon \leq 1/2$, then the number of mistakes is at most $2(1+\epsilon)M + O(\frac{\log n}{\epsilon})$. (Hint: check out the approximations we use in the proof of Theorem 2.)

However, you can show that no deterministic prediction algorithm can make fewer than $2M$ mistakes. How? Two experts: one says "Up" all the time, the other "Down" all the time. Fix any prediction algorithm. Since this algorithm is deterministic, you (as the adversary) know what prediction it will make on any day, if you know what happened on all previous days. So as the adversary, you can now make the "real" outcome on this day be the opposite of the algorithm's prediction for this day. This means the algorithm makes a mistake on all days. But one of the experts must be right on at least 50% of the days.

4 Randomized Weighted Majority

Given the lower bound above for deterministic algorithms, randomization seems like a natural source of help. (At least the example above would fail.) Let's define the *Randomized Weighted Majority* algorithm as follows:

> Start with unit weights. At each time, predict `Up` with probability
>
> $$\frac{\sum_{j \text{ says Up}} w_j}{\sum_j w_j},$$
>
> and `Down` otherwise. Whenever an expert makes a mistake, multiply its weight by $(1-\epsilon)$.

Note that the weights are pretty much like in the basic MW algorithm, just reduced more gently. (Moreover, note that the prediction we make does not alter the weight reduction step.) A slightly different, but equivalent view is the following:

> Start with unit weights. At each time, pick a random expert, where expert i is picked with probability $\frac{w_i}{\sum_j w_j}$, and predict that picked expert's prediction. And each time an expert makes a mistake, multiply its weight by $(1-\epsilon)$.

The analysis very similar to Theorem 1, we just need to handle expectations.

Theorem 2 *Let $\epsilon \leq 1/2$. If on some sequence of days, the best expert makes M mistakes. The expected number of mistakes the randomized weighted majority algorithm makes is at most*

$$(1+\epsilon)M + \frac{\ln n}{\epsilon}.$$

Proof: Again, let us look at the potential $\Phi = \sum_j w_j$, the total weight in the system. Having fixed the outcome on all days, this potential varies deterministically.

Let F_t be the fraction of the total weight on the t^{th} day that is on the experts who make a mistake on that day. On day t our probability of making a mistake is F_t. Hence

- The expected number of mistakes we make is $\sum_t F_t$.
- On the t^{th} day, $\Phi_{new} = \Phi_{old} \cdot (1 - \epsilon F_t)$. So,

  $$\Phi_{final} = n \cdot \prod_t (1 - \epsilon F_t) \leq n \cdot e^{-\epsilon \sum_t F_t},$$

  where we used that $(1+x) \leq e^x$ for all $x \in \mathbb{R}$.

Again, $\Phi_{final} \geq (1-\epsilon)^M$, so

$$(1-\epsilon)^M \leq n \cdot e^{-\epsilon \sum_t F_t} \quad \Rightarrow \quad \epsilon \sum_t F_t \leq M \ln \frac{1}{(1-\epsilon)} + \ln n.$$

We can now use that $\ln \frac{1}{(1-\epsilon)} = -\ln(1-\epsilon) \leq \epsilon + \epsilon^2$ for $\epsilon \in [0, \frac{1}{2}]$ to get

$$\text{the expected number of mistakes } = \sum_t F_t \leq M(1+\epsilon) + \frac{\ln n}{\epsilon}.$$

■

Pretty sweet, right? The number of mistakes we make is almost the same as the best expert, plus this additive term that just depends on n and ϵ, but is independent of the input sequence. Since the optimal number of mistakes M is at most T, the number of days, we can say

$$\Rightarrow \text{our \# of errors} \ \leq\ \text{optimal \# of errors} \ + \epsilon T + \frac{\ln n}{\epsilon} \tag{1}$$

So dividing both sides by T, we get the error "rate" (i.e., errors per unit time):

$$\Rightarrow \text{our error rate} \ \leq\ \text{optimal error rate} \ + \epsilon + \frac{\ln n}{\epsilon T} \tag{2}$$

And now setting $\epsilon = \sqrt{\frac{\ln n}{T}}$:

$$\Rightarrow \text{our error rate} \ \leq\ \text{optimal error rate} \ + 2\sqrt{\frac{\ln n}{T}}. \tag{3}$$

This last term is called the "regret". So as we do the prediction for longer and longer (i.e., $T \to \infty$), our error rate gets as close as we want to the optimal error rate. (We have "vanishing regret".)

5 Some Extensions (optional material)

Repeated Zero-Sum Game. The interpretation of choosing a random expert (according to the probability distribution $\frac{w_i}{\sum_j w_j}$) allows us to view the process as playing a two-player zero-sum game repeatedly. Let the cost matrix contain only 0s and 1s. Each column is an expert. Each day the adversary plays a row (which defines the costs for us), we play a column (which is like choosing an expert). Theorem 2 says that we can play almost as best as the best column in hindsight.

Fractional Costs. Suppose each day t, instead of costs that are zero (no mistake) or one (mistake), we have costs c_i^t in $[0,1]$. We can extend the result to that setting with a very slight change: update the weight of an expert i by $w_i \leftarrow w_i(1-\epsilon c_i^t)$. With small changes in the analysis, the guarantee of Theorem 2 goes through unchanged! (Exercise: Show this!)

5.1 A Proof of the Minimax Theorem (optional material)

Recall the minimax theorem: it says that in any zero-sum game (given by the row player's payoff matrix M), if we define the row player's guaranteed payoff (which is a lower bound on what she's guaranteed to earn regardless of what the column player does):

$$V_R = \max_{\mathbf{p}} \min_{j} \mathbf{p}^T M e_j = \max_{\mathbf{p}} \min_{\mathbf{q}} \mathbf{p}^T M \mathbf{q}$$

and the column player's guaranteed payoff (which is an upper bound on what the row player can earn, regardless of what he does):

$$V_C = \min_{\mathbf{q}} \max_{i} e_i^T M \mathbf{q} = \min_{\mathbf{q}} \max_{\mathbf{p}} \mathbf{p}^T M \mathbf{q},$$

then $V_C = V_R$. Clearly, $V_R \leq V_C$. The tricky part is the other direction, which we now use the experts theorem to prove.

Suppose not. There is some zero-sum game where $V_R = V_C - \delta$ for $\delta > 0$, with a strict inequality. Remember,

- If the column player commits to some strategy first, there is some row that the row player can play to get payoff at least V_C.
- On the other hand, if the row player commits first, then the column the column player can play so that row player get at most $V_R = V_C - \delta$.

By scaling the payoffs, imagine they are in $[0, 1]$.

Now imagine we use Randomized Weighted Majority (RWM) to play the columns. Since the distribution from which RWM draws is updated in a deterministic fashion, let the row player play optimally against this distribution at each step.

- In T steps, RWM has cost at most that of the best column (expert) in hindsight $+\epsilon T + \frac{\ln n}{\epsilon}$.
- The best column in hindsight can ensure cost at most $V_R \cdot T$, since it's like the row player has played first.
- But at each time, the row player knows your strategy, so his expected payoff is at least $V_C \cdot T$.

Putting these together

$$V_C \cdot T \leq \text{ row player's payoff } \leq V_R \cdot T + \epsilon T + \frac{\ln n}{\epsilon},$$

or

$$\delta T \leq \epsilon T + \frac{\ln n}{\epsilon}.$$

But we were free to choose ϵ and T as we want, so if we choose $\epsilon = \delta/2$ and $T \geq \frac{\ln n}{\epsilon^2}$, then we get a contradiction. Hence, there cannot be any gap δ, and $V_R = V_C$.