CS 170 HW 12

Due on 2019-04-22, at 11:59 pm

1 Study Group

List the names and SIDs of the members in your study group.

2 Experts Alternatives

Recall the experts problem. Every day you must take the advice of one of n experts. At the end of each day t, if you take advice from expert i, the advice costs you some c_i^t in $[0,1]$. You want to minimize the regret R, defined as:

$$R = \frac{1}{T}(\sum_{t=1}^{T} c_{i(t)}^t - \min_{1\leq i\leq n} \sum_{t=1}^{T} c_i^t)$$

where $i(t)$ is the expert you choose on day t. Your strategy will be probabilities where p_i^t denotes the probability with which you choose expert i on day t. Assume an all powerful adversary can look at your strategy ahead of time and decide the costs associated with each expert on each day. Give the maximum possible (expected) regret that the adversary can guarantee if your strategy is:

(a) Choose expert 1 at every step. That is, if $\forall t\ p_1^t = 1$ and C_i^t is the set of costs for all experts and all days, what is $\max_{C_i^t} R$?

(b) Any deterministic strategy. Note that a "deterministic strategy" can be thought of as a probability distribution that satisfies the following: $\forall t\ \exists i\ p_i^t = 1$.

(c) Always choose an expert according to some fixed probability distribution at every time step. That is, if for some $p_1 \dots p_n$, $\forall t, p_i^t = p_i$, what is $\max_{C_i^t}(\mathbb{E}[R])$?

What distribution minimizes the regret of this strategy? In other words, what is $\text{argmin}_{p_1\dots p_n} \max_{C_i^t}(\mathbb{E}[R])$?

This analysis should conclude that a good strategy for the problem must necessarily be randomized and adaptive.

Solution:

(a) 1. Consider the case where the cost of expert 1 is always 1 and the cost of expert 2 is always 0. Thus $C = \sum_{t=1}^{T}\sum_{i=1}^{n} p_i^t c_i^t = \sum_{t=1}^{T} c_1^t = T$, we have $C^* = \sum_{i=1}^{T} c_2^t = 0$, so regret is $\frac{1}{T}(T-0) = 1$.

(b) $\frac{n-1}{n}$. Consider the case where the cost of the chosen expert is always 1, and the cost of each other expert is 0. Let k be the least-frequently chosen expert, and let m_k be the number of times that expert is chosen. This will result in a regret of $\frac{1}{T}(T - m_k)$

Since the best expert is the one that is chosen least often, the best strategy will try to maximize the number of times we choose the expert that is chosen least often. This means we want to choose all the experts equally many times, so expert k is chosen in at most T/n of the rounds. Therefore, $m_k \leq \frac{T}{n}$, thus the regret is at least $\frac{1}{T}(T - \frac{T}{n}) = \frac{n-1}{n}$

(c) $(1 - \min_i p_i)$. Like in part 1, the distribution is fixed across all days, so we know ahead of time which expert will be chosen least often in expectation. Let $k = \arg\min_i p_i$ be the expert with least cost. Let $c_k^t = 0$ for all t, and let $c_i^t = 1$ for all $i \neq k$ and for all t. This way, $C = T(1 - p_k)$, as this is a binomial random variable distributed as $C \sim Bin(T, 1-p_k)$. $C^* = 0$, so we end up with a regret of $\frac{1}{T}(C - C^*) = \frac{1}{T}(T(1-p_k) - 0) = 1 - p_k$.

To minimize the expectation of R is the same as maximizing $\min_i p_i$. This is maximized by the uniform distribution, obtaining regret $\frac{n-1}{n}$ (this is the same worst case regret as in part 2).

3 Follow the regularized leader

(a) **Follow the leader.** You are playing T rounds of the following game: At round t you pick one of n strategies; your payoff for picking strategy i is $A(t,i) \in [0,1]$. You try the following algorithm: at each iteration pick the strategy which gave the highest average payoff so far (on the first iteration, you pick strategy 1).

Give an example of payoffs for $T = 100$ and $n = 2$, where your algorithm obtains a payoff of 0, but sticking to either $i = 1$ or $i = 2$ would have given you a payoff of almost 50.

Solution: Let the payoffs be at $t = 1$, $(0, 1 - \varepsilon)$ and then for every other odd t be $(0, 1)$ and for each even t be $(1, 0)$.

Prior to every even numbered round, the strategy with the higher average payoff is strategy 2. Similarly, prior to ever odd numbered round, the strategy with the higher average payoff is strategy 1. But, by construction, this will yield an overall payoff of 0 as the strategies alternate in success in dissonance with the alterantion of average payoff.

If you stuck to strategy 1, you would obtain a payoff of 50 and if you stuck to strategy 2, you would obtain a payoff of $50 - \varepsilon$.

(b) **Follow the randomized leader.** The reason the algorithm above didn't do so well, is because when we deterministically jump from one strategy to another, an adversarially chosen set of strategies can be designed to thwart the algorithm.

To trick such adversaries, we want to use a *randomized* strategy; at time t we pick our strategy i at random from distribution D_t (These distributions are chosen upfront, i.e., before the algorithm is run and not chosen adaptively based on the loss functions). Let $p_t(i) \geq 0$ denote the probability that we assign to strategy i (i.e. $\sum_{i=1}^n p_t(i) = 1$).

The previous algorithm ("Follow the leader") corresponds to setting D_t that maximizes

$$\sum_{i=1}^{n} \left(p_t(i) \cdot \sum_{\tau \in \{1,\ldots,t-1\}} [A(\tau, i)] \right).$$

Why is this no better?.

Solution: Since the distributions are chosen upfront, the randomized solution over all time is just a linear combination of any deterministic solution (this is based on the averaging the probabilities for every decision over D_t upto some timestep). Hence, the adversary can set up losses based on the distributions so far such that the choices which are more likely suffer more loss, like in the deterministic case in part (a).

(c) **Follow the regularized leader.** Instead, it is common to add a /regularized term that favors smoother distributions. A commonly used regularizer is the entropy function, i.e. we want to use pick i from the distribution that maximizes

$$\sum_{i=1}^{n} \left(p_t(i) \cdot \sum_{\tau \in \{1,\ldots,t-1\}} [A(\tau, i)] - \eta p_t(i) \ln p_t(i) \right). \tag{1}$$

(Here, $\eta > 0$ is a parameter that we can tweak to balance exploration and exploitation. Notice also that $\ln p_t(i) \leq 0$.)

In this exercise you will show that following the regularized leader with the entropy regularizer is the same as Multiplicative Weights Update!

Show that for any distribution p_t, (1) is at most

$$\eta \cdot \ln \left(\sum_{i=1}^{n} e^{\sum_{\tau \in \{1,\ldots,t-1\}} [A(\tau,i)]/\eta} \right) \tag{2}$$

(Hint: you may use the inequality $\sum_{i=1}^n p_t(i) \cdot \ln(y_i) \leq \ln\left(\sum_{i=1}^n p_t(i) \cdot y_i\right)$ for any vector $\vec{y}$.)

Solution:

$$\begin{aligned}
(1) &= \eta \sum_{i=1}^{n} p_t(i) \cdot \left(\sum_{\tau \in \{1,\ldots,t-1\}} [A(\tau,i)]/\eta - \ln p_t(i) \right) \\
&= \eta \sum_{i=1}^{n} p_t(i) \cdot \ln \left(e^{\sum_{\tau \in \{1,\ldots,t-1\}}[A(\tau,i)]/\eta} / p_t(i) \right) \\
&\leq \eta \cdot \ln \left(\sum_{i=1}^{n} \left(p_t(i) \cdot e^{\sum_{\tau \in \{1,\ldots,t-1\}}[A(\tau,i)]/\eta} / p_t(i) \right) \right) \\
&= \eta \cdot \ln \left(\sum_{i=1}^{n} e^{\sum_{\tau \in \{1,\ldots,t-1\}}[A(\tau,i)]/\eta} \right).
\end{aligned}$$

(d) Show that for some choice of ϵ (which depends on η), when computing p_t using Multiplicative Weight Update, (1) is equal to (2). What is the dependence of ϵ on η?

Solution: When using the MWU algorithm, we have:

$$\begin{aligned}
p_t(i) &= \frac{w_t(i)}{\sum_{j=1}^{n} w_t(j)} \\
&= \frac{(1-\epsilon)^{-\sum_{\tau \in \{1,\ldots,t-1\}}[A(\tau,i)]}}{\sum_{j=1}^{n} (1-\epsilon)^{-\sum_{\tau \in \{1,\ldots,t-1\}}[A(\tau,j)]}}.
\end{aligned}$$

If we set ϵ such that $(1-\epsilon) = e^{-1/\eta}$, and plugin into the last equation, we have:

$$p_t(i) = \frac{e^{\sum_{\tau \in \{1,\ldots,t-1\}}[A(\tau,i)]/\eta}}{\sum_{j=1}^{n} e^{\sum_{\tau \in \{1,\ldots,t-1\}}[A(\tau,j)]/\eta}}.$$

Therefore,

$$\begin{aligned}
(1) &= \eta \sum_{i=1}^{n} p_t(i) \cdot \ln \left(e^{\sum_{\tau \in \{1,\ldots,t-1\}}[A(\tau,i)]/\eta} / p_t(i) \right) \\
&= \eta \sum_{i=1}^{n} p_t(i) \cdot \ln \left(\sum_{i=1}^{n} e^{\sum_{\tau \in \{1,\ldots,t-1\}}[A(\tau,i)]/\eta} \right) \\
&= \eta \cdot \ln \left(\sum_{i=1}^{n} e^{\sum_{\tau \in \{1,\ldots,t-1\}}[A(\tau,i)]/\eta} \right).
\end{aligned}$$

4 Linear Classifiers using Multiplicative Weights

In this problem, we will learn a basic algorithm in machine learning used to learn linear classifiers. The setup is as follows : We are given m labelled examples $(\mathbf{a}_1, y_1), \ldots, (\mathbf{a}_m, y_m)$

where $\mathbf{a}_j \in \mathbb{R}^n$ are m *data points* where each data point is a list of n *features* (we assume that all the co-ordinates/feature of all the data points $\mathbf{a}_j$ are either +1 or -1), and $y_j \in \{-1, +1\}$ are their labels. This can be modelled (**after some transformations to the problem which lets us get rid of the labels $\mathbf{y}_j$**) as the following LP which we will aim to solve:

$$\begin{aligned} \text{for all} \quad j = 1, \dots, m: \quad & \langle \mathbf{a}_j, \mathbf{x} \rangle \geq 0 \\ & \langle \mathbf{1}, \mathbf{x} \rangle = 1 \\ \text{for all} \quad i = 1, \dots, n: \quad & \mathbf{x}_i \geq 0 \end{aligned}$$

Here $\langle \mathbf{x}, \mathbf{y} \rangle = \sum_{i=1}^{n} x[i] \cdot y[i]$ and $\mathbf{v}[j]$ refers to the j^{th} component of the vector.
Assume that there is a *large-margin* solution, i.e., there exists a $\eta > 0$ and a distribution $\mathbf{x}^*$ so that for all j, we have $\langle \mathbf{a}_j, \mathbf{x}^* \rangle \geq \eta$. Consider the following bound on the regret of the multiplicative weights algorithm (which is a modification of the one showed in class; See Section 3 in the notes for the Multiplicative Weights lecture for the general theorem).

THEOREM The multiplicative weights algorithm start with an iterate $\mathbf{x}_1 \in \mathbb{R}^n$ and then suffer a sequence of loss vectors $\ell_1, \dots, \ell_T \in \mathbb{R}^n$ such that $\ell_t[i] \in [-1, 1]$ and $\varepsilon < 1/2$ and we produce a sequence of iterates $\mathbf{x}_2, \dots, \mathbf{x}_T$ such that for any $i \in \{1, \dots, n\}$, we have

$$\sum_{t=1}^{T} \langle \ell_t, x_t \rangle \leq \frac{\log(n)}{\varepsilon} + \min_{i=1}^{n} \sum_{t=1}^{T} (\ell_t[i] + \varepsilon |\ell_t[i]|)$$

(a) Using the bounds in the regret of the multiplicative weights algorithm mentioned above, prove that for any probability distribution $\mathbf{p}^*$,

$$\sum_{t=1}^{T} \langle \ell_t, x_t \rangle \leq \frac{\log(n)}{\varepsilon} + \sum_{i=1}^{n} \mathbf{p}^*[i] \Big(\sum_{t=1}^{T} (\ell_t[i] + \varepsilon |\ell_t[i]|) \Big)$$

(b) We want to use Multiplicative weights to find an $\mathbf{x}$ which satisfies the above LP. We will think of being an adversary who is providing the loss functions based on the constraints. What are the experts for running the algorithm?

(c) What are the loss functions for running the Multiplicative weights algorithm? Note that your loss functions should satisfy the conditions for applying the regret bound (assumptions required for the above theorem). (Hint : Consider what happens when there is some constraint not satisfied by your current iterate $\mathbf{x}_t$. Can you use that to find a loss vector to improve your solution?)

(d) Clearly specify the condition when you will terminate the algorithm?

(e) Combine parts (a) - (d) to devise a multiplicative weights algorithm to find a $\widetilde{\mathbf{x}}$ such that $\langle \mathbf{a}_j, \widetilde{\mathbf{x}} \rangle \geq 0$ for all j. Clearly specify what $\widetilde{x}$ should be. Also state the number of iterations T it takes to find this solution. You will have to take $\varepsilon = \eta/2$ for the algorithm. (You should think of what should $\mathbf{p}^*$ supposed to be and how should we plug in what you have derived in part (b)-(d) into the regret bound in part (a)).

Solution:

(a) Note that the regret bound has a minimum over a set of numbers in the RHS. This in particular implies that for any $i \in \{1, \ldots, n\}$

$$\sum_{t=1}^{T} \langle \ell_t, x_t \rangle \le \frac{\log(n)}{\varepsilon} + \sum_{t=1}^{T} (\ell_t[i] + \varepsilon|\ell_t[i]|)$$

So given any probability vector $\mathbf{p}^* \in \mathbb{R}^n$, i.e, $\mathbf{p}^*[i] \ge 0$ for all i and $\sum_{i=1}^{n} \mathbf{p}^*[i] = 1$, we can multiply the inequality given by the i^{th} expert by $\mathbf{p}^*[i]$ on both sides and sum the inequalities up to get

$$\sum_{t=1}^{T} \langle \ell_t, x_t \rangle \le \frac{\log(n)}{\varepsilon} + \sum_{i=1}^{n} \mathbf{p}^*[i] \Big(\sum_{t=1}^{T} (\ell_t[i] + \varepsilon|\ell_t[i]|) \Big)$$

(b) We consider n experts, one for each feature, i.e., one for each co-ordinate of $\mathbf{x}$.

(c) The losses are specified by the m examples. The loss for feature i for example j is $-a_{i,j}$. Note that these losses lie in the range $[-1, 1]$ as required. In each round t, let x_t be the vector generated by the MW algorithm (note that this is a probability vector). Now, we look for a misclassified example, i.e., an example j such that $\langle \mathbf{a}_j, \mathbf{x}_t \rangle < 0$. If no such constraint exists, we are done and we can stop. Otherwise, if j is a misclassified example, then it specifies the loss for round t.

(d) We now keep running the MW algorithm until we find a good solution, i.e, one that classifies all examples correctly.

(e) We will run the multiplicative weights algorithm with $\varepsilon = \eta/2$. Note that the loss in round t is

$$\langle \ell_t, x_t \rangle = \langle -\mathbf{a}_j/, \mathbf{x}_t \rangle > 0$$

, whereas for the solution $\mathbf{x}^*$, we have

$$\langle \ell_t, \mathbf{x}^* \rangle = \langle -\mathbf{a}_j/, \mathbf{x}_* \rangle < -\eta$$

We now keep running the MW algorithm until we find a good solution, i.e, one that classifies all examples correctly. To get a bound on the number of iterations until we find a good solution, we use the equation shown above(with $\mathbf{p}^* = \mathbf{x}^*$. Use the trivial bound that $\varepsilon \sum_{i=1}^{n} |\ell_t[i]|\mathbf{x}^*[i] \leq \varepsilon$, we get,

$$\begin{aligned}
&\sum_{t=1}^{T} \langle \ell_t, x_t \rangle \leq \frac{\log(n)}{\varepsilon} + \varepsilon T + \sum_{t=1}^{T}\sum_{i=1}^{n} x^*[i]\ell_t[i] \\
\implies &\sum_{t=1}^{T} \langle \ell_t, x_t \rangle \leq \frac{2\log(n)}{\eta} + \eta T/2 - T\eta \\
\implies &\sum_{t=1}^{T} \langle \ell_t, x_t \rangle \leq \frac{2\log(n)}{\eta} - \eta T/2
\end{aligned}$$

Notice that while the algorithm keeps going, the LHS is greater than zero. Hence we get, $T \leq 4\log(n)/\eta^2$, and after that we can't find a loss vector which is positive which implies

$$\sum_{t=1}^{T} \langle \mathbf{a}_j, x_T \rangle > 0$$

Taking $\widetilde{x} = \frac{1}{T}\sum_{t=1}^{T} x_t$ or even $\widetilde{x} = x_T$, which in both cases implies that $\widetilde{x}$ is a probability distribution as each of the x_t are probability distributions and we are taking an average of them, we find a satisfiable solution after at most $T = 4\log(n)/\eta^2$ iterations.