# COS 511: Theoretical Machine Learning

Homework #8 — Due: May 14, 2019
Log-loss and games

## Problem 1

Consider the problem of online learning with log-loss in which the goal is to do almost as well as the best fixed distribution (rather than the best expert). Thus, at each time step $t = 1, \ldots, T$, the learner chooses a distribution $q_t$ over some finite space $X$, where $|X| = k \geq 2$, and then observes an outcome $x_t \in X$. As usual, the resulting loss is $-\ln(q_t(x_t))$, but now the goal is to do almost as well as the best fixed distribution $p$ over $X$.

If the data were actually randomly generated i.i.d. by some distribution $D$ over $X$ (for instance, if we were observing the outcomes of repeated flips of a biased coin so that $k = 2$), then we could think of $q_t$ as an estimate of $D$ based on what has been observed so far. But of course, since we are working in the online learning model, the outcomes $x_t$ are *not* assumed to be random, and might even be chosen by an adversary.

Let $n_t(x) = \sum_{s=1}^{t-1} \mathbf{1}\{x_s = x\}$ denote the number of times $x$ has been observed up to, but not including, round $t$. Consider an algorithm that chooses $q_t$ as follows, for all $x$:

$$q_t(x) = \frac{n_t(x) + 1}{t + k - 1}.$$

a. [15] Prove that, for all sequences $x_1, \ldots, x_T$:

$$-\sum_{t=1}^{T} \ln q_t(x_t) \leq \min_p \left[ -\sum_{t=1}^{T} \ln p(x_t) \right] + O(\ln T),$$

where the minimum is over all distributions $p$ over $X$, and where, for purposes of big-Oh notation, we here treat $k$ as a constant. Give an exact bound with all constants filled in.

b. [5] We have seen repeatedly that there is a strong connection between learning with log-loss and sending a message using a minimal number of bits. Explain what the result in part (a) means in the context of coding, including what the regret bound says about coding efficiency, and specifically, in comparison to what kind of coding methods.

## Problem 2

[15] Suppose, as in class, that we are working over a finite space $X$, and that we are given examples $x_1, \ldots, x_m \in X$, as well as features $f_1, \ldots, f_n$ with $f_j : X \to [0, 1]$. In class, we studied a coordinate-descent algorithm for maximizing likelihood among Gibbs distributions of the form

$$q_{\boldsymbol{\lambda}}(x) = \frac{\exp\left(\sum_{j=1}^{n} \lambda_j f_j(x)\right)}{Z_{\boldsymbol{\lambda}}},$$

or equivalently, for finding a maximum-entropy distribution $\mathbf{p}$ subject to linear constraints:

$$\hat{\mathrm{E}}\left[f_j\right] = \mathrm{E}_{\mathbf{p}}\left[f_j\right] \quad \text{for } j = 1, \ldots, n.$$

(As in class, $\hat{\mathrm{E}}\left[f\right] = (1/m)\sum_{i=1}^{m} f(x_i)$, and $\mathrm{E}_{\mathbf{p}}\left[f\right] = \mathrm{E}_{x\sim\mathbf{p}}\left[f(x)\right]$.) Let $\mathbf{q}^*$ denote the desired solution to both of these problems.

The algorithm we studied in class is "greedy" in the sense that, on each iteration, it selects the one coordinate $\lambda_j$ to update that gives the largest improvement in (our bound on) the log-likelihood. Alternatively, we could simply cycle systematically through the $\lambda_j$'s, updating each one in turn. This leads to an algorithm like the following:

for $t = 1, 2, \ldots$:
  $j = (t \bmod n)$
  $\alpha = \ln\left(\dfrac{\hat{\mathrm{E}}\left[f_j\right]}{1 - \hat{\mathrm{E}}\left[f_j\right]} \cdot \dfrac{1 - \mathrm{E}_{\mathbf{p}_t}\left[f_j\right]}{\mathrm{E}_{\mathbf{p}_t}\left[f_j\right]}\right)$
  for $k = 1, \ldots, n$: $\lambda_{t+1,k} = \begin{cases} \lambda_{t,j} + \alpha & \text{if } k = j \\ \lambda_{t,k} & \text{else.} \end{cases}$

Here, $\mathbf{p}_t = \mathbf{q}_{\boldsymbol{\lambda}_t}$, the Gibbs distribution defined by $\boldsymbol{\lambda}_t = \langle \lambda_{t,1}, \ldots, \lambda_{t,n} \rangle$. And $(t \bmod n)$ returns that $j \in \{1, \ldots, n\}$ which gives the same remainder as $t$ when divided by $n$. Thus, the algorithm is the same as in class, except for the $\lambda_j$'s being updated cyclically.

Prove that $\mathbf{p}_t \to \mathbf{q}^*$, that is, that the distributions $\mathbf{p}_t$ converge to $\mathbf{q}^*$, the desired solution to the optimization problems being solved. (Be sure to show that the entire sequence of $\mathbf{p}_t$'s converges to $\mathbf{q}^*$, not just a subsequence.)

Note: To avoid annoying issues related to the possibility of dividing by zero, taking the logarithm of zero, etc., for this problem, you can assume that the features are bounded away from 0 or 1, that is, that for some $\varepsilon > 0$, it is the case that $f_j(x) \in [\varepsilon, 1 - \varepsilon]$ for every feature $f_j$ and for all $x \in X$.

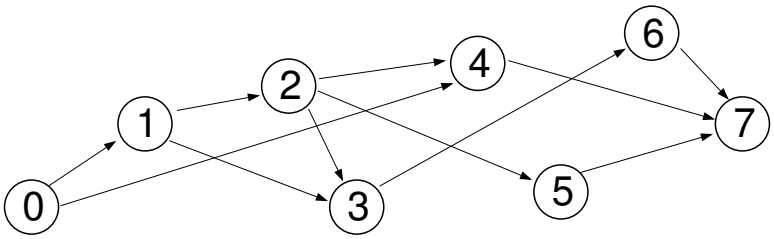


Figure 1: A sample graph $G$.

## Problem 3

Let $G = (V, E)$ be a directed, acyclic graph (DAG), with exactly one source (vertex with no incoming edges), and exactly one sink (vertex with no outgoing edges). See, for instance, the graph shown in Fig. 1. The vertices of the graph are the $n+2$ integers $V = \{0, 1, \ldots, n+1\}$, and $E$ is its set of directed edges (ordered pairs of vertices). Without loss of generality, we assume that the vertices have been numbered in such a way that every edge passes from a lower-numbered vertex to a higher-numbered vertex; in other words, if $(i, j) \in E$, then $i < j$. This implies that 0 is the unique source, and $n + 1$ is the unique sink. Throughout this problem, we only consider graphs with these properties.

A *path* $\pi$ is formally a set of edges (i.e., $\pi \subseteq E$) of the form

$$\pi = \{(i_0, i_1), (i_1, i_2), \ldots, (i_{k-1}, i_k)\}.$$

For shorthand, we write such a path as $i_0 \to i_1 \to \cdots \to i_k$. For $i \leq n$, let $\mathcal{P}_i$ denote the set of all paths beginning at vertex $i$ and ending at the sink, $n + 1$. In particular, $\mathcal{P}_0$ denotes the set of all paths from source to sink. Let $D$ be the length of the longest path in $\mathcal{P}_0$ (and therefore the longest path in the entire graph): $D = \max_{\pi \in \mathcal{P}_0} |\pi|$.

For instance, the graph in the figure, where $n = 6$, includes, as an example, the path $1 \to 2 \to 5 \to 7$, which formally is the set of edges $\{(1, 2), (2, 5), (5, 7)\}$. Also, $\mathcal{P}_2$ consists of the three paths $2 \to 4 \to 7$; $2 \to 5 \to 7$; and $2 \to 3 \to 6 \to 7$. The longest path in this graph has length $D = 5$.

a. [5] Prove that $|\mathcal{P}_0| \leq 2^n$ for all graphs $G$. Also, prove that this bound is tight (up to a constant in the exponent) in the sense that there exists a constant $c > 0$ and there exist graphs $G$ (of the form above) for infinitely many values of $n$ for which $|\mathcal{P}_0| \geq 2^{cn}$. (It is possible to prove this with $c = 1$, but that is not required.)

Consider a game in which Mindy chooses a path $\pi \in \mathcal{P}_0$, and Max simultaneously chooses a path $\rho \in \mathcal{P}_0$. Mindy's resulting loss is then the number of edges in her path that overlap with Max's (divided by $D$ so the loss will be in $[0, 1]$); that is, Mindy's loss is

$$M(\pi, \rho) = \frac{|\pi \cap \rho|}{D}.$$

Thus, Mindy is trying to choose a path that avoids Max, while Max is trying to do the opposite. (For instance, in the graph in the figure, if Mindy picks $0 \to 1 \to 3 \to 6 \to 7$, and Max picks $0 \to 1 \to 2 \to 3 \to 6 \to 7$ then Mindy's loss would be $3/5 = 0.6$ since these paths overlap in three edges, and $D = 5$.)

The loss given above effectively defines a $|\mathcal{P}_0| \times |\mathcal{P}_0|$ game matrix $\mathbf{M}$ with one row and one column for every path in $\mathcal{P}_0$. Suppose that Max and Mindy play this game repeatedly so that, on each round $t = 1, \ldots, T$, they choose a pair of paths $\pi_t$ and $\rho_t$ with resulting loss (to Mindy) of $M(\pi_t, \rho_t)$. Suppose further that Mindy, on each round $t$, uses the MW

algorithm applied to the matrix $\mathbf{M}$ to first compute a distribution $P_t$ over $\mathcal{P}_0$, and then selects a random path $\pi_t \sim P_t$. Max, on the other hand, chooses $\rho_t$ in any way that he pleases (with $Q_t$, for the purposes of the MW algorithm, defined to be the mixed strategy that assigns probability 1 to $\rho_t$).

Our analysis of the MW algorithm (for an appropriate setting of $\beta$) immediately implies that, for any sequence of plays $\rho_t$ by Max,

$$\frac{1}{T}\sum_{t=1}^{T} \mathrm{E}_{\pi_t \sim P_t}\left[M(\pi_t, \rho_t)\right] = \frac{1}{T}\sum_{t=1}^{T} M(P_t, \rho_t) \leq \min_{\pi \in \mathcal{P}_0} \frac{1}{T}\sum_{t=1}^{T} M(\pi, \rho_t) + O\left(\sqrt{\frac{\ln|\mathcal{P}_0|}{T}}\right)$$

where expectation is over Mindy's random choice of $\pi_t$ (which, technically, is conditional on the history of play up to that round). Thus, Mindy's per-round expected loss will quickly approach what she would have attained by playing the best fixed path $\pi$ (or also by playing the best fixed distribution over paths) against the sequence of plays $\rho_1, \ldots, \rho_T$ that were actually played by Max. However, the running time of a direct implementation of MW will require time $O(|\mathcal{P}_0|)$ per round. So, by the results of part (a), compared to the size of the graph $G$, the regret will be very reasonable, but the running time will be exponential. We will see, nevertheless, that this same algorithm can be implemented far more efficiently.

To see how, let $w : E \to \mathbb{R}$ be any real-valued function defined on the edges of $G$. For a path $\pi$, we define $\overline{w}(\pi)$ to be the product of the $w$-values along the edges in $\pi$:

$$\overline{w}(\pi) = \prod_{e \in \pi} w(e).$$

Further let $S_w : V \to \mathbb{R}$ be a function in which $S_w(i)$ is defined to be the sum of these values over all paths from $i$ to the sink; that is, for any vertex $i \leq n$ other than the sink,

$$S_w(i) = \sum_{\pi \in \mathcal{P}_i} \overline{w}(\pi).$$

For the sink, we define $S_w(n+1) = 1$.

b. [10] Give an algorithm that, for any provided function $w$ as above, computes *all* of the values $S_w(0), \ldots, S_w(n+1)$. Show that the total running time of the algorithm (to compute all $n+2$ values of $S_w$) is $O(|E|)$.

c. [15] Show how Mindy's application of MW as described above can be implemented in time $O(|E|)$ per round, and using space $O(|E|)$. Specifically, describe:

   (i) an appropriate data structure for (implicitly) representing $P_t$;

   (ii) methods for initializing and updating your data structure on each round; and

   (iii) a method for sampling a single $\pi_t \sim P_t$ using your data structure.

   Be sure to explain and justify your answers, including: how your data structure represents $P_t$ and why your proposed method for maintaining it is correct; why your sampling method for choosing $\pi_t$ is correct, in the sense of $\pi_t$ having the correct distribution; and why the overall space and time (per round) are $O(|E|)$.