E1 245 - Online Prediction and Learning, Aug-Dec 2015
Final Exam
December 2, 2015

Instructions:

- There are 5 main questions with a maximum score of 65 points. There is also a 6th (challenging) question that is worth 25 bonus points. The total time allotted is 3 hours.
- No cellphones or electronic aids are allowed. You may use notes made on one A4 size sheet of paper for reference.

---

1. *Fenchel dual and Bregman divergence* (10 points)
Let $F : (0, +\infty)^d \to \mathbb{R}$, $F(x) := \sum_{i=1}^d x_i \log x_i - \sum_{i=1}^d x_i$. This is called the generalized negative entropy function.

   (a) Find the Bregman divergence D_F induced by F. (2 points)

   (b) Find the Fenchel dual F^* of F. Mention its domain clearly. (4 points)

   (c) Find the Bregman divergence D_{F^*} induced by F^*. (2 points)

   (d) Show the following 'three-point inequality' for the Bregman divergence $D_R(x, y)$ induced by a differentiable convex function $R : \mathbb{R}^d \to \mathbb{R}$: (2 points)

$$\forall u, v, w \in \mathbb{R}^d : \quad D_R(u, v) + D_R(v, w) = D_R(u, w) + \langle u - v, \nabla R(w) - \nabla R(v) \rangle.$$

Solution.

(a) $\forall x, y \in (0, +\infty)^d \quad D_F(x, y) = \sum_{i=1}^d x_i \log \frac{x_i}{y_i} - \sum_{i=1}^d (x_i - y_i)$.

(b) Domain of $F^* = \left\{\nabla F(x) : x \in (0, +\infty)^d\right\} = \{(\log x_1, \ldots, \log x_d) : x_i > 0 \; \forall 1 \le i \le d\} = \mathbb{R}^d$.

For any $\theta \in \mathbb{R}^d$, we have $F^*(\theta) = \sup_{x \in (0,+\infty)} \langle x, \theta \rangle - F(x)$. At the optimizing point x^*, the gradient must vanish; thus $\forall i \; \theta_i = \log x_i^* \Rightarrow x_i^* = \exp(\theta_i) \Rightarrow F^*(\theta) = \sum_{i=1}^d \exp(\theta_i)$.

(c) $\forall \theta, \mu \in \mathbb{R}^d \quad D_{F^*}(\theta, \mu) = \sum_{i=1}^d (\exp(\theta_i) - \exp(\mu_i) - \exp(\mu_i)(\theta_i - \mu_i))$.

(d) We have,

$$\begin{aligned} &D_R(u, v) + D_R(v, w) = R(u) - R(v) - \langle \nabla R(v), u - v \rangle + R(v) - R(w) - \langle \nabla R(w), v - w \rangle \\ &= R(u) - R(w) - \langle \nabla R(w), u - w \rangle - \langle \nabla R(w), v - u \rangle - \langle \nabla R(v), u - v \rangle \\ &= D_R(u, w) + \langle u - v, \nabla R(w) - \nabla R(v) \rangle. \end{aligned}$$

2. *Online vs. Classical Optimization* (10 points)
Suppose you have a good online convex optimization algorithm $\mathcal{A}$ with the following property. For any number of rounds $T \ge 1$ and any sequence of T loss functions from $\mathcal{L}$: a family of convex functions defined over a convex set $\mathcal{K}$, the algorithm's regret is at most $f(T)$ in T rounds, with respect to any single point of $\mathcal{K}$. Assume that $f(T) = o(T)$ is a known sublinear function.

Now consider the standard (or 'batch') convex optimization problem: you want to find a minimum of the convex function $f \in \mathcal{L}$ over $\mathcal{K}$ to within an accuracy $\epsilon > 0$. In other words, you must output $x \in \mathcal{K}$ satisfying $f(x) \le \min_{y \in \mathcal{K}} f(y) + \epsilon$. How you would accomplish this using the algorithm $\mathcal{A}$?

[Hint: Run $\mathcal{A}$ with a particular sequence of functions and a particular number of rounds, and output a single point in the end. Jensen's inequality may be useful.]

Solution.

Feed the function f repeatedly for n rounds to the algorithm $\mathcal{A}$, i.e., $f_1 = f_2 = \cdots = f_n = f$ (2 points). If the algorithm plays points $w_1, \ldots, w_n \in \mathcal{K}$, then output the mean of these points $w := \frac{1}{n} \sum_{t=1}^n w_t$ which is guaranteed to lie in $\mathcal{K}$ since $\mathcal{K}$ is convex (3 points).

To determine n, observe that the regret property gives $\sum_{t=1}^{n} f(w_t) - n \cdot \min_{y \in \mathcal{K}} f(y) \leq f(n)$. Thus,

$$\begin{aligned} f(w) = f\left(\frac{1}{n}\sum_{t=1}^{n} w_t\right) &\leq \frac{1}{n}\sum_{t=1}^{n} f(w_t) \quad \text{(by Jensen's inequality – 2 points)} \\ &\leq \min_{y \in \mathcal{K}} f(y) + \frac{f(n)}{n} \quad \text{(by the regret property; 1 point)} \\ &\leq \epsilon, \end{aligned}$$

provided we take n to be any number satisfying $\frac{f(n)}{n} \leq \epsilon$ (this is possible since $f(n)$ is sublinear in n; 2 points).

3. *Stochastic Bandit Algorithms* (3×6 points; $+2$ points for getting all $= 20$ points)
Consider the general iid[1] stochastic bandit with N arms and all arms' rewards being Bernoulli-distributed. If μ_i denotes the expected reward of the ith arm, then the regret of a bandit algorithm, that plays an arm $I_t \in [N]$ at each time $1 \leq t \leq T$ and observes only the random reward from the chosen arm is defined to be $R(T) := T \cdot \max_i \mu_i - \sum_{t=1}^{T} \mathbb{E}[\mu_{I_t}]$.

Explain briefly which of the following algorithms will/will not always achieve sublinear regret with time horizon T ($R(T)$ is sublinear $\Leftrightarrow \lim_{T\to\infty} \frac{R(T)}{T} = 0$).

(a) Initialize $s_i := 0 \; \forall \; i \in [N]$. At each time $t \leq T$, play $I_t := \arg\max_i s_i$ (break ties in any fixed manner), get (stochastic) reward R_t and update $s_{I_t} \leftarrow s_{I_t} + R_t$.

(b) Play all arms exactly once. For each arm i, initialize s_i to be its observed reward and $n_i := 1$. At each time $t \leq T$, play $I_t := \arg\max_i s_i/n_i$ (break ties in any fixed manner), get (stochastic) reward R_t and update $s_{I_t} \leftarrow s_{I_t} + R_t$, $n_{I_t} \leftarrow n_{I_t} + 1$.

(c) Play all arms exactly once. For each arm i, initialize s_i to be its observed reward and $n_i := 1$. At each time $t \leq T$, play $I_t := \arg\max_i \left(s_i/n_i + \sqrt{2\log t/n_i}\right)$ (break ties in any fixed manner), get (stochastic) reward R_t and update $s_{I_t} \leftarrow s_{I_t} + R_t$, $n_{I_t} \leftarrow n_{I_t} + 1$.

(d) For each arm $i \in [N]$, initialize $u_i = 1, v_i = 1$. At each time $t \leq T$, sample independent random variables $\theta_i(t) \sim \text{Beta}(u_i, v_i)$, and play $I_t := \arg\max_i \theta_i(t)$ (break ties in any fixed manner). Get (stochastic) reward R_t and update $u_{I_t} \leftarrow u_{I_t} + R_t$, $v_{I_t} \leftarrow v_{I_t} + (1 - R_t)$.

(e) For each arm $i \in [N]$, initialize $u_i = 1, v_i = 1$. At each time $t \leq T$, for each arm i let $\theta_i(t) = u_i/(u_i + v_i)$, i.e., the expected value of the Beta(u_i, v_i) distribution. Play $I_t := \arg\max_i \theta_i(t)$ (break ties in any fixed manner). Get (stochastic) reward R_t and update $u_{I_t} \leftarrow u_{I_t} + R_t$, $v_{I_t} \leftarrow v_{I_t} + (1 - R_t)$.

(f) For each arm $i \in [N]$, initialize $p_i := 1/N$. At each time $t \leq T$, play a random arm I_t drawn independently at that time according to the probability distribution $(p_1, \ldots, p_N)$. Get (stochastic reward) R_t, let $\tilde{p}_i := p_i \exp\left(\frac{R_t \mathbb{1}[I_t = i]}{p_i\sqrt{T}}\right)$ for each $i \in [N]$, and update $p_i \leftarrow \tilde{p}_i / \sum_{j\in[N]} \tilde{p}_j$ for all arms i. (Note: $\mathbb{1}[A]$ is the indicator random variable of event A.)

Solution.

(a) Will not always achieve sublinear regret. The algorithm greedily plays the arm with the best current empirical mean. With non-zero probability the algorithm will get trapped into playing a bad arm at the start, leading to linear (expected) regret.

(b) Will not always achieve sublinear regret. The algorithm is essentially the same as in (a).

(c) Will always achieve sublinear regret. This is the Upper Confidence Bound (UCB) algorithm.

(d) Will always achieve sublinear regret. This is the Thompson sampling algorithm with a Uniform (i.e., Beta(1,1)) prior.

[1] independent and identically distributed

(e) Will not always achieve sublinear regret. The algorithm is essentiallt the same as in (b) and (a).

(f) Will always achieve sublinear regret. The algorithm is EXP3 with learning rate $\eta = 1/\sqrt{T}$, for which a sublinear regret guarantee holds even if rewards are arbitrary in $[0, 1]$.

4. *'Worst-case' UCB regret* (10 points)

Consider the same Bernoulli stochastic N-armed bandit setting as in Question 3. We proved in class the following bound for the Upper Confidence Bound (UCB) algorithm:

$$\mathbb{E}\left[T_i(T)\right] \leq \frac{8\log T}{\Delta_i^2} + \frac{\pi^2}{3}, \tag{1}$$

for any suboptimal arm i (i.e., $\mu_i < \max_j \mu_j$), with $T_i(T)$ denoting the total number of times that arm i was chosen up to time T, and $\Delta_i := \max_j \mu_j - \mu_i$ denoting its gap from an optimal arm. This directly gave us the regret bound $R(T) := \sum_{i=1}^N \Delta_i \, \mathbb{E}[T_i(T)] \leq 8 \sum_{i:\Delta_i>0} \frac{\log T}{\Delta_i} + \frac{N\pi^2}{3}$. The bound, however, is of no use when Δ_i is very small as the regret can never be more than T.

Show instead that the following *gap-independent* (or problem-independent, or 'worst-case') regret bound for UCB holds.

$$R(T) \leq \sqrt{NT\left(8\log T + \frac{\pi^2}{3}\right)} \quad \forall N, T.$$

[Hint: Bound the regret using a well-known inequality, use the bound (1), and note that the $T_i(T)$ sum to T across all i.]

Solution.

$$\begin{aligned}
R(T) &\overset{(2)}{:=} \sum_{i=1}^N \Delta_i \, \mathbb{E}[T_i(T)] \overset{(4)}{=} \sum_{i=1}^N \Delta_i \, \sqrt{\mathbb{E}[T_i(T)]}\sqrt{\mathbb{E}[T_i(T)]} \\
&\overset{(4)}{\leq} \sqrt{\sum_{i=1}^N \Delta_i^2 \, \mathbb{E}[T_i(T)]}\sqrt{\sum_{i=1}^N \mathbb{E}[T_i(T)]} \\
&\text{(by the Cauchy-Schwarz inequality, i.e., } \textstyle\sum_i x_i y_i \leq \sqrt{\sum_i x_i^2}\sqrt{\sum_i y_i^2}\text{)} \\
&\overset{(2)}{\leq} \sqrt{\sum_{i=1}^N \left(8\log T + \frac{\Delta_i^2\pi^2}{3}\right)}\sqrt{\mathbb{E}\left[\sum_{i=1}^N T_i(T)\right]} \quad \text{(by (1))} \\
&\overset{(2)}{\leq} \sqrt{NT\left(8\log T + \frac{\pi^2}{3}\right)} \quad \text{(since } \Delta_i^2 \leq 1 \text{ and } \textstyle\sum_i T_i(T) = T\text{).}
\end{aligned}$$

5. *Sequential probability estimation* (15 points)

Suppose you are observing an arbitrary stream of bits $y_1, y_2, \ldots$ with $y_i \in \{0, 1\}$, generated from some source (e.g., a digital voice signal). The following occurs at each round $t \geq 1$: You are asked to guess a probability distribution $\hat{p}_t \equiv (\hat{p}_t(0), \hat{p}_t(1)) \in \{(p, 1-p) : 0 \leq p \leq 1\}$ for the next bit y_t, Following your guess, y_t is revealed and you suffer a loss of $\log \frac{1}{\hat{p}_t(y_t)}$.

Consider competing in this game with the class of all 'constant experts'. A constant expert is a rule parameterized by $p \in [0, 1]$ that always guesses the probability distribution $(p, 1-p)$ (this is the analog of Constantly Rebalancing Portfolios in finance).

Write down the natural version of the Exponential Weights prediction algorithm with uniform initial weights and learning rate $\eta = 1$ [Note: There are infinitely many experts!]. Can you express its

prediction at each time t in the *simplest* possible form[2]? You may use the identity

$$\int_0^1 q^{n_1}(1-q)^{n_2}\,dq = \frac{1}{(n_1+n_2+1)\binom{n_1+n_2}{n_1}},$$

for any integers $n_1, n_2 \geq 0$, and where $\binom{a}{b}$ is the standard binomial coefficient $\frac{(a+b)!}{a!b!}$.

Solution.

The decision space is $\mathcal{D} = \{(p, 1-p) : 0 \leq p \leq 1\}$, the outcome space is $\mathcal{Y} = \{0, 1\}$, the loss function is $l((p(0), p(1)), y) = \log 1/p(y)$, and the set of experts is $\mathcal{E} = \{(p, 1-p) : 0 \leq p \leq 1\}$. (4 points)

The Exponential Weights algorithm with learning rate $\eta = 1$ is as follows: Initialize $w_1((p, 1-p)) = 1\ \forall (p, 1-p) \in \mathcal{E}$. At each round $t \geq 1$, guess the distribution $\hat{p}_t = (\hat{p}_t(0), \hat{p}_t(1))$ where

$$\hat{p}_t(0) = \frac{\int_0^1 w_t((q, 1-q))\ q\ dq}{\int_0^1 w_t((q, 1-q))\ dq}.$$

Following this, observe y_t and update weights $\forall (p(0), p(1)) \in \mathcal{E}$:

$$\begin{aligned}
w_{t+1}((p(0), p(1))) &= w_t((p(0), p(1))) \exp\left(-\eta \log \frac{1}{p(y_t)}\right) \\
&= w_t((p(0), p(1)))\ p(y_t) \\
&= w_t((p(0), p(1)))\ p(0)^{1-y_t}\ p(1)^{y_t} \\
&= \cdots \\
&= w_1((p(0), p(1)))\ p(0)^{t-\sum_{s=1}^t y_s}\ p(1)^{\sum_{s=1}^t y_s} \\
&= p(0)^{n_0(t)}\ p(1)^{n_1(t)}, \quad \text{(6 points)}
\end{aligned}$$

where $n_0^t := (t - \sum_{s=1}^t y_s)$ and $n_1^t := \sum_{s=1}^t y_s$ are simply the number of 0s and 1s in the first t bits, respectively. Hence we can simplify the expression for $\hat{p}_t(0)$ as

$$\begin{aligned}
\hat{p}_t(0) &= \frac{\int_0^1 q^{n_0^{t-1}}(1-q)^{n_1^{t-1}}\ q\ dq}{\int_0^1 q^{n_0^{t-1}}(1-q)^{n_1^{t-1}}\ dq} = \frac{\int_0^1 q^{1+n_0^{t-1}}(1-q)^{n_1^{t-1}}\ dq}{\int_0^1 q^{n_0^{t-1}}(1-q)^{n_1^{t-1}}\ dq} \\
&= \frac{(t-1+1)\binom{t-1}{n_0^{t-1}}}{(t+1)\binom{t}{1+n_0^{t-1}}} = \frac{n_0^{t-1}+1}{t+1}. \quad \text{(5 points)}
\end{aligned}$$

With the above, the Exponential Weights algorithm becomes the very simple 'add-1' rule: At each time t, predict the distribution $\hat{p}_t = (\hat{p}_t(0), \hat{p}_t(1))$ where

$$\hat{p}_t(0) = \frac{\text{number of 0s seen so far} + 1}{t+1} \quad \text{and} \quad \hat{p}_t(1) = 1 - \hat{p}_t(0).$$

6. $\star$ **Bonus question** - *Sequential probability estimation* (25 points)
With regard to the previous question. show that the regret of the Exponential Weights algorithm you wrote down (with $\eta = 1$) in T rounds with respect to all the constant experts, for any sequence of bits $y_1, \ldots, y_T$, is no more than $\log(1+T)$.

Solution.

[2] implementable using finitely many arithmetic operations

The regret incurred in T rounds by the algorithm, for the sequence $y_1, \ldots, y_T$, is

$$
\begin{aligned}
R(T) &= \sum_{t=1}^{T} \log \frac{1}{\hat{p}_t(y_t)} - \min_{(p(0),p(1))\in\mathcal{E}} \sum_{t=1}^{T} \log \frac{1}{p(y_t)} \\
&= \max_{(p(0),p(1))\in\mathcal{E}} \left(\sum_{t=1}^{T} \log \frac{1}{\hat{p}_t(y_t)} - \sum_{t=1}^{T} \log \frac{1}{p(y_t)} \right) \\
&= \max_{(p(0),p(1))\in\mathcal{E}} \left(\sum_{t=1}^{T} \log \frac{p(y_t)}{\hat{p}_t(y_t)} \right) = \max_{(p(0),p(1))\in\mathcal{E}} \log \prod_{t=1}^{T} \frac{p(y_t)}{\hat{p}_t(y_t)} \\
&= \log \frac{\max_{(p(0),p(1))\in\mathcal{E}} \prod_{t=1}^{T} p(y_t)}{\prod_{t=1}^{T} \hat{p}_t(y_t)} \\
&= \log \frac{\max_{(p(0),p(1))\in\mathcal{E}} \; p(0)^{n_0^T} \; p(1)^{n_1^T}}{\prod_{t=1}^{T} \hat{p}_t(y_t)} \\
&= \log \frac{\left(\frac{n_0^T}{T}\right)^{n_0^T} \; \left(\frac{n_1^T}{T}\right)^{n_1^T}}{\prod_{t=1}^{T} \hat{p}_t(y_t)} = \log \frac{\left(\frac{n_0^T}{T}\right)^{n_0^T} \; \left(\frac{n_1^T}{T}\right)^{n_1^T}}{\frac{n_0(T)!\, n_1(T)!}{(T+1)!}} \\
&= \log(T+1) + \log \underbrace{\binom{T}{n_0(T)} \left(\frac{n_0^T}{T}\right)^{n_0^T} \; \left(\frac{n_1^T}{T}\right)^{n_1^T}}_{\leq 1} \\
&\leq \log(T+1).
\end{aligned}
$$