Fall 2022 *Instructors: Jacob Abernethy, Bhuvesh Kumar, Guanghui Wang*

ECE 8803: Online Decision Making in Machine Learning

Homework 2

Released: Oct 2 *Due: Oct 19, 11:59 pm ET*

Objective. To help build a bridge from prediction to decision-making, and help students interpret online prediction algorithms through the framework of online optimization.

Problem 1 (Multiplicative weights = Follow-the-Regularized-Leader). 15 points In this problem, we will show that multiplicative weights can be expressed in the online convex optimization framework, by showing that it is equivalent to a Follow-the-Regularized-Leader problem with a particular type of regularization that encourages randomization. This regularization is called the *entropy* function, and originated in the study of information theory.

(a) For the case of binary prediction, recall that we defined $\widehat{P}_t := \mathbb{P}[\widehat{X}_t = 1]$. For any $p \in (0,1)$ define the *binary entropy function*

$$H(p) := -p\log(p) - (1-p)\log(1-p).$$

Plot $H(p)$ as a function of p (NOTE: you do not need to consider the cases $p = 0$ and $p = 1$). When is it maximized, and when is it minimized? (You do not need to prove this: the correct answer suffices.)

(b) Recall that the FTL algorithm was given by

$$\widehat{X}_t := \arg\min_{x\in\{0,1\}} L_{t-1,x}, \tag{1}$$

where we defined $L_{t-1,x} := \sum_{s=1}^{t-1} l_{s,x}$ and $l_{s,x} = \mathbb{I}[X_s \neq x]$. Show that this update can be written as the minimum to an optimization problem in the following form:

$$\widehat{P}_t := \arg\min_{p\in[0,1]} \left[f(p, L_{t-1,0}, L_{t-1,1})\right]$$

where $f(\cdot)$ is linear in each of $p, L_{t-1,0}$ and $L_{t-1,1}$.

Hint: Note that since FTL is a deterministic algorithm, it induces either $\widehat{P}_t = 0$ (when $\widehat{X}_t = 0$) or $\widehat{P}_t = 1$ (when $\widehat{X}_t = 1$).

(c) Using the function that you defined above, consider the *regularized* update

$$\widehat{P}_t := \arg\min_{p\in(0,1)} f(p, L_{t-1,0}, L_{t-1,1}) - \frac{1}{\eta}H(p).$$

Show that this is equivalent to the multiplicative weights algorithm by explicitly calculating $\widehat{P}_t$.

Hint: Use your observation from part (a) to argue that the optimization problem above is convex. Can you use this to calculate the minimum?

(d) Use your observation in part (a) to argue, in your own words, why a smaller value of η encourages more randomization.

(e) (BONUS – Prediction with expert advice (5 points)) In the more general setting of prediction with K experts, we can define losses $l_{t,i}$ at time t for each expert $i \in \{1, \ldots, K\}$. Accordingly, we can define a total loss vector $\boldsymbol{L}_t := \begin{bmatrix} L_{t,1} & \ldots & L_{t,K} \end{bmatrix}$ and a probability vector $\boldsymbol{p}_t := \begin{bmatrix} p_{t,1} & \ldots p_{t,K} \end{bmatrix}$. Finally, we can also define a binary entropy function $H(\boldsymbol{p}) := \sum_{i=1}^{K} -p_i \log(p_i)$.

Repeat parts (a) – (d) to show that the multiplicative weights algorithm for K experts can be written as a *regularized update*, i.e. a solution to the optimization objective $f(\boldsymbol{p}, \boldsymbol{L}_t) - \frac{1}{\eta} H(\boldsymbol{p})$.

Hint: Use your intuition developed from parts (a) – (d) to propose an optimization objective. Then, solve the minimum of this objective subject to $p_{t,i} \geq 0$, but without the probability constraint. Do you notice a pattern?

Problem 2 (New algorithms through regularization). 20 points See attached iPython notebook for details on this problem. Here, we only provide a brief description of the problem and its subparts.

In Problem 1, you showed that multiplicative weights can be written as the solution to the following optimization problem:

$$\widehat{P}_t = \arg\min_{p \in [0,1]} \left[f(L_{t-1,1}, L_{t-1,0}, p) - \frac{1}{\eta} H(p) \right],$$

where $H(p)$ was defined as the binary entropy function.

Now, we will explore a different type of regularizer, i.e. instead consider the update

$$\widetilde{P}_t = \arg\min_{p \in [0,1]} \left[f(L_{t-1,1}, L_{t-1,0}, p) + \frac{1}{\eta} R(p) \right],$$

where we now define

$$R(p) := \log\left(\frac{1}{p}\right) + \log\left(\frac{1}{1-p}\right).$$

This is often called the "log-barrier" regularizer and was actually first proposed by Nemirovski and Yudin.

(a) Plot $R(p)$ versus p. Where is it minimized? Where is it maximized? Use this plot to argue that minimizing $R(p)$ encourages randomization.

(b) The optimization problem defined for the log-barrier regularizer turns out to have a closed form solution, which is given by:

$$\widetilde{P}_t := f_2(D_t) := \frac{2}{\eta D_t + \sqrt{\eta^2 D_t^2 + 4} + 2},$$

where $D_t := L_{t-1,1} - L_{t-1,0}$, i.e. the difference in the cumulative losses.

Express the multiplicative weight update at time step t as a different function $\widehat{P}_t = f_1(D_t)$. Then, plot both functions $f_1(\cdot)$ and $f_2(\cdot)$ as a function of d ranging between -10 and 10 for the case $\eta = 1$. Which function decreases faster?

(c) Implement the multiplicative weights update and the log-barrier update here as a function of the losses $L_{t-1,0}, L_{t-1,1}$. You are free to use the starter code provided in the attached iPython notebook, or code provided in the earlier resource "MultiplicativeWeights.ipynb". Please make the output of your algorithm a numpy array "prob" with 2 entries, given by $[P_{t,0}, P_{t,1}]$.

Hint for log-barrier: recall that we defined $D_t = L_{t-1,1} - L_{t-1,0}$.

(d) Now, we compare the performance of both algorithms on the two running examples that we have been considering in class. We set $T = 600$, and consider two types of sequences:

a) the case where $X_t = 1$ for all $t = 1, \ldots, T$. b) the case of the 1-periodic sequence.

For both algorithms, we will set $\eta = 1$ in order to more dramatically visualize differences in performance.

Plot the total *loss* of each algorithm (what we defined as H_t in class) versus the number of time steps t on sequences (a) and (b). Please make separate figures for the evaluation of (a) and (b).

You are welcome to directly use the starter code provided below to evaluate the performance of an arbitrary algorithm (parameterized by eta) on an arbitrary binary sequence. This starter code will return a "total loss vector" given by $\begin{bmatrix} H_1 & \ldots H_T \end{bmatrix}$ for plotting convenience.

(e) Report the superior algorithm for sequence (a) and sequence (b) respectively. Based on this report, which algorithm do you think tends to randomize more? (You are also welcome to use the answer to part (b) as a hint.)

(f) (BONUS - theory (10 points)) Prove that the log-barrier regularizer leads to the explicit update given in part (b).

Hint: the quadratic formula may be useful.

Problem 3 (Lower bound for online linear optimization (OLO)). 15 points We have discussed several algorithms in class that achieve the optimal regret guarantee $R_T = \mathcal{O}(\sqrt{T})$. We have also mentioned that this guarantee is optimal (i.e. there exists an instance on which *any* algorithm would incur at least $\sqrt{T}$ regret). In the setting of binary prediction, or decision-making using expert advice, this instance is *randomly constructed* and somewhat complicated to describe. However, in the setting of online linear optimization, it turns out that we can create a remarkably simple adversary that forces *any* OLO algorithm to incur exactly $\sqrt{T}$ regret. In this problem, we will show precisely this. All norms $\|\cdot\|$ are, by default, the Euclidean (ℓ_2) norm.

(a) Let $\{\boldsymbol{\ell}_t\}_{t=1}^T$ be a set of vectors in $\mathbb{R}^d$. Assume that $\|\boldsymbol{\ell}_i\| = 1$ for all $t = 1, \ldots, T$, and furthermore, $\boldsymbol{\ell}_t^\top (\sum_{s=1}^{t-1} \boldsymbol{\ell}_s) = 0$ for all $t > 1$. Prove that $\|\sum_{t=1}^T \boldsymbol{\ell}_t\| = \sqrt{T}$.

Hint: You may find it useful to start from the fact that

$$\left\|\sum_{t=1}^{T} \boldsymbol{\ell}_t\right\|^2 = \left\|\sum_{t=1}^{T-1} \boldsymbol{\ell}_t + \boldsymbol{\ell}_T\right\|^2 = \left\|\sum_{t=1}^{T-1} \boldsymbol{\ell}_t\right\|^2 + \|\boldsymbol{\ell}_T\|^2 + 2\boldsymbol{\ell}_T^\top \left(\sum_{t=1}^{T-1} \boldsymbol{\ell}_t\right)$$

where the above uses the algebraic identity on $||\boldsymbol{a} + \boldsymbol{b}||^2$ for two vectors $\boldsymbol{a}$ and $\boldsymbol{b}$. After simplifying the above, apply the same procedure recursively, i.e. to the term $\left\|\sum_{t=1}^{T-1} \boldsymbol{\ell}_t\right\|^2$.

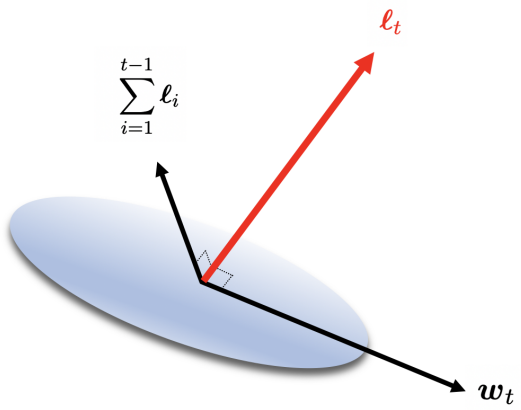

Figure 1: Example when $d = 3$.

(b) Consider the case where $d \geq 3$, and let the decision set $\mathcal{B}$ be a ball of radius 1 and centered at the origin. In other words, $\mathcal{B} := \{\boldsymbol{w} : \|\boldsymbol{w}\| \leq 1\}$. Prove that for *any* set of loss vectors $\{\boldsymbol{\ell}_t\}_{t=1}^T$, the optimal decision in hindsight is given by:

$$\boldsymbol{w}_* = -\frac{\sum_{t=1}^T \boldsymbol{\ell}_t}{\|\sum_{t=1}^T \boldsymbol{\ell}_t\|}.$$

(c) Now, we define our the adversary's policy is defined as follows: When $t = 1$, after observing the learner's decision $\boldsymbol{w}_1$, the adversary chooses a vector $\boldsymbol{\ell}_1$ such that $\|\boldsymbol{\ell}_1\| = 1$ and $\boldsymbol{\ell}_1^\top \boldsymbol{w}_1 = 0$. For round $t \geq 2$, the adversary will choose a $\boldsymbol{\ell}_t$ such that $\|\boldsymbol{\ell}_t\| = 1$, $\boldsymbol{\ell}_t^\top \boldsymbol{w}_t = 0$ and $\boldsymbol{\ell}_t^\top (\sum_{i=1}^{t-1} \boldsymbol{\ell}_i) = 0$. Note that such a $\boldsymbol{\ell}_t$ can always be found for $d \geq 3$ (as $\boldsymbol{\ell}_t$ is the normal vector of the plane spanned by $\boldsymbol{w}_t$ and $\sum_{i=1}^{t-1} \boldsymbol{\ell}_i$). Figure 1 shows an example for the 3-dimensional case.

Prove that the regret of *any* OLO algorithm *is equal to* $\sqrt{T}$ under this policy.

(d) (BONUS – 10 points) For $d = 2$, can we still design a policy for the adversary to ensure the $\Omega(\sqrt{T})$ lower bound?

Hint: In fact, $\boldsymbol{\ell}_t$ does not need to be exactly orthogonal to $\boldsymbol{w}_t$ and $\sum_{i=1}^{t-1} \boldsymbol{\ell}_i$ for the proof approach to work.

Problem 4 (Online gradient descent with a time-varying, data-dependent learning rate) 30 points In this problem, we consider solving online convex optimization by a slightly different algorithm, i.e., online gradient descent (OGD). The basic procedure of OGD is as follows: We begin with $\boldsymbol{w}_1 = 0$. For round $t = 1, \ldots, T$, OGD firstly observes the gradient of $\ell_t(\cdot)$ at $\boldsymbol{w}_t$, i.e., $\nabla \ell_t(\boldsymbol{w}_t)$, then preforms a descent step towards the gradient direction with learning rate $\eta_t > 0$,

$$\boldsymbol{z}_{t+1} = \boldsymbol{w}_t - \eta_t \nabla \ell_t(\boldsymbol{w}_t),$$

and next projects the result into the decision set to get $\boldsymbol{w}_{t+1}$:

$$\boldsymbol{w}_{t+1} = \Pi_{\mathcal{B}}[\boldsymbol{z}_{t+1}],$$

where

$$\Pi_{\mathcal{B}}[\boldsymbol{z}] = \arg\min_{\boldsymbol{w}\in\mathcal{B}} \|\boldsymbol{w} - \boldsymbol{z}\|_2^2$$

denotes projecting $\boldsymbol{z}$ onto a convex set $\mathcal{B}$.

This algorithm is related to FTRL, but slightly different. In FTRL, we ran the update from the *unconstrained* decision vector $\boldsymbol{z}_t$ as:

$$\boldsymbol{z}_{t+1} = \boldsymbol{z}_t - \eta\nabla\ell_t(\boldsymbol{w}_t) \tag{2}$$

and then projected $\boldsymbol{z}_{t+1}$ onto the convex set $\mathcal{B}$. We also considered a fixed learning rate η. However, we will now show that these algorithms have similar performance, and OGD allows us to consider a time-varying, data-dependent learning rate. You will find it useful to review the notes of Lectures 8 and 9 for answering this question.

(a) If we set the learning rate of OGD as $\eta_1 = \eta_2 = \cdots = \eta > 0$, and also ignore the projection steps in OGD and FTRL (review the Equation above Equation 10.10 for the latter), then what is the relationship between the two algorithms?

(b) Let $\boldsymbol{w}^*$ be the optimal decision in hindsight, defined as $\boldsymbol{w}_* = \arg\min_{\boldsymbol{w}\in\mathcal{B}} \sum_{t=1}^T \ell_t(\boldsymbol{w}_t)$. Prove that for every value of $t \geq 1$, we have:

$$\|\boldsymbol{w}_{t+1} - \boldsymbol{w}^*\|_2^2 \leq \|\boldsymbol{w}_t - \boldsymbol{w}_*\|_2^2 + \eta_t^2\|\nabla\ell_t(\boldsymbol{w}_t)\|_2^2 - 2\eta_t(\boldsymbol{w}_t - \boldsymbol{w}_*)^\top\nabla\ell_t(\boldsymbol{w}_t). \tag{3}$$

Hint: (1) First prove that the right hand side of the above equation is ***equal to*** $\|\boldsymbol{z}_{t+1} - \boldsymbol{w}^*\|_2^2$ *using an appropriate quadratic identity on* $\|\boldsymbol{a} - \boldsymbol{b}\|_2^2$*. Then review Lemma 2 in the Appendix of Lecture 11 to recall what projection does to the distance.*

(c) Based on (3), prove that $\forall t \geq 1$,

$$\ell_t(\boldsymbol{w}_t) - \ell_t(\boldsymbol{w}_*) \leq \frac{\|\boldsymbol{w}_t - \boldsymbol{w}^*\|_2^2 - \|\boldsymbol{w}_{t+1} - \boldsymbol{w}^*\|_2^2}{2\eta_t} + \frac{\eta_t}{2}\|\nabla\ell_t(\boldsymbol{w}_t)\|_2^2.$$

Hint: Review Section 9.2.1, and make use of the first-order condition for convex functions to start the proof. More specifically, if $f(\cdot) : \mathcal{B} \mapsto \mathbb{R}$ *is convex, then for any* $\boldsymbol{z}_1, \boldsymbol{z}_2 \in \mathcal{B}$*, we have*

$$f(\boldsymbol{z}_1) \geq f(\boldsymbol{z}_2) + (\boldsymbol{z}_1 - \boldsymbol{z}_2)^\top\nabla f(\boldsymbol{z}_2).$$

Then, combine your observation with the inequality that was derived in part (b).

(d) Based on (c), prove that the regret of OGD can be upper bounded by

$$\sum_{t=1}^T \ell_t(\boldsymbol{w}_t) - \ell_t(\boldsymbol{w}^*) \leq \frac{\|\boldsymbol{w}_1 - \boldsymbol{w}^*\|_2^2}{2\eta_1} + \frac{1}{2}\sum_{t=2}^T \left(\frac{1}{\eta_t} - \frac{1}{\eta_{t-1}}\right)\|\boldsymbol{w}_t - \boldsymbol{w}^*\|_2^2 + \sum_{t=1}^T \frac{\eta_t}{2}\|\nabla\ell_t(\boldsymbol{w}_t)\|_2^2.$$

Hint: Sum up the upper bounds derived in part (c). Then, relate the quantities $\sum_{t=1}^T \frac{\|\boldsymbol{w}_t - \boldsymbol{w}^*\|_2^2 - \|\boldsymbol{w}_{t+1} - \boldsymbol{w}^*\|_2^2}{\eta_t}$ *and* $\sum_{t=2}^T \left(\frac{1}{\eta_t} - \frac{1}{\eta_{t-1}}\right)\|\boldsymbol{w}_t - \boldsymbol{w}^*\|_2^2$*. You may find it helpful to do this for the case* $T = 3$ *first, to get intuition.*

(e) Let $\eta_t = \frac{D}{G\sqrt{t}}$. See Assumptions 1 and 2 in Lecture 11 for the defination of D and G. Prove that the regret bound of OGD is $O(DG\sqrt{T})$,
Hint:
(1) note that $\frac{1}{\eta_t} - \frac{1}{\eta_{t-1}} \geq 0$, *so* $\left(\frac{1}{\eta_t} - \frac{1}{\eta_{t-1}}\right) \|\boldsymbol{w}_t - \boldsymbol{w}^*\|_2^2 \leq 4D^2 \left(\frac{1}{\eta_t} - \frac{1}{\eta_{t-1}}\right)$.
(2)We have $\sum_{t=1}^{T} \frac{1}{\sqrt{t}} \leq 2\sqrt{T} - 1$.

(f) (BONUS – 5 points) Let $\eta_t = \frac{D}{\sqrt{\sum_{i=1}^{t} \|\nabla \ell_i(\boldsymbol{w}_i)\|^2}}$. Prove that the regret bound of OGD is on the order of $O(D\sqrt{\sum_{t=1}^{T} \|\nabla \ell_t(\boldsymbol{w}_t)\|_2^2})$. Is this bound better or worse than the bound that you derived in part (e)?
Hint: For nonnegative $a_1, \ldots, a_t$, *we have* $\sum_{i=1}^{t} \frac{a_i}{\sqrt{\sum_{j=1}^{i} a_j}} \leq 2\sqrt{\sum_{i=1}^{t} a_i}$. *Also note that the learning rate, as set, is decreasing with* t.