CS 506, HW2

Prof. Jared Saia, University of New Mexico

Due: March 7th

You are encouraged to work on the homework in groups of about 2 or 3. You may turn in one writeup per group, but please certify that all members worked on each problem.

1. In class, I claimed that in the expert learning problem, any *deterministic* algorithm must sometimes make a number of mistakes at least a factor of 2 times the smallest number of mistakes of any expert.[1] Prove this claim.

2. Prove that the type of linear program in System 1 in the Arora notes, Chapter 10, is as powerful as a general linear program. Hint: Use binary search, and use it not only on the objective function value!

3. Assume that there are n different lottery games, and that you can play exactly one each week. Your goal is to maximize your probability of winning at least once. For each of these games, there is a certain probability of losing, that you learn only at the end of the week (this probability may depend, for example, on the number of people who purchased tickets). For each game i, and each week t, let p_i^t be the probability that you would lose if you played game i in week t. Let p^* be the the minimum over all the lottery games of the probability of losing in all of the T weeks. Assume that $p_i^t \geq 1/2$ for all i and t.

   Show how to use MWU to play so that your probability of losing over all T weeks is at most $f(T, n) \cdot p^*$, for some function f of T and n that is asymptotically as small as possible. Analyze your algorithm. Hint 1: Review the analysis of Portfolio Management in Arora Notes chapter 10. Hint 2: Make use of the fact that the log function is concave.

[1] This fact motivates the use of randomness in the MWU algorithm in order to do better.

4. In this problem, you'll show that MWU "seeks to maximize both utility and entropy". In the MWU algorithm, let p_i^t be the probability of picking expert i, $1 \leq i \leq n$ at time t, and let m_i^t be the gain of expert i at time t. The expected gain of the algorithm through round $t-1$ is then $\sum_{j=1}^{t-1} \sum_{i=1}^{n} p_i^j m_i^j$. Let $\vec{p}$, be the vector of probabilities, and consider the following function (here η is the learning rate parameter in MWU):

$$f(\vec{p}) = \eta \sum_{j=1}^{t-1} \sum_{i=1}^{n} p_i^t m_i^j - \sum_{i=1}^{n} p_i^t \ln p_i^t$$

Note that $-\sum_{i=1}^{n} p_i^t \ln p_i^t$ is just the entropy of $\vec{p}$.

   (a) For what values of p_i^t is $f(\vec{p})$ maximized?

   (b) Now consider a function f defined for time $t+1$, and give the values of p_i^{t+1} for which that function is maximized.

   (c) Let p_i^t be your solution for part (a), and let p_i^{t+1} be your solution for part (b). Take logs of both expressions and subtract to show that $p_i^{t+1} = p_i^t e^{\eta m_i^t}$. Note that this is just the alternate MWU rule!!!

5. Recall the game of Rock, Paper Scissors. Assume that: 1) when scissors and paper are played, the scissors player wins 1 and the paper players loses 1; 2) when rock and scissors are played, the rock the winner wins 1 and the loser loses 1; 3) when rock and paper are played, the paper player wins .1 and the rock player loses .1;[2] and a draw results in both players getting 0. Suppose you make two copies of the MWU algorithm to play this game with each other over many iterations. Both copies start with a uniform random probability (i.e. play each of rock/paper/scissors with probability 1/3), and then learn from experience based on the MWU rule. Presumably, the players' probabilities will eventually converge to some sort of equilibrium. Predict, based on thought, what the probabilities will be in the equilibrium. Now, run this experiment on Matlab or any other programming environment, and report what you've discovered.

   Now, consider the same game, with the two players still using MWU, but assume that the second player has an η value much smaller than the first. For example, player 1 has $\eta = .1$, and player 2 has $\eta = .001$. Will the players still converge to an equilibrium? Will it take

[2] Really, how badly can a piece of paper hurt a rock anyway?

the second player 100 times as long to converge as the first? Make predictions, and then run an experiment on your favorite programming environment. Report what you've discovered and explain.

6. This problem concerns the Winnow algorithm discussed in Section 3.1 of the MWU survey paper by Arora, Hazan and Kale. Assume that you have the following data set, where the first two values are attributes, and the third value is the classification.

$$\begin{array}{ccc} -1/2 & +1/3 & +1 \\ +3/4 & -1/4 & +1 \\ +1/2 & -1/3 & -1 \\ -3/4 & +1/4 & -1 \end{array}$$

   (a) You guess that there is a large margin solution to this problem, x^*, such that for all j, $a_j \cdot x^* \geq .05$. So you set $\epsilon = .05$, and for simplicity, you also set $\rho = 1$. Give the values for n, T and η that you would use to run Winnow.

   (b) Now code up your Winnow algorithm in Matlab or your favorite programming language. What is the classifier your algorithm returns for the above data set?

   (c) **Challenge:** What if you want to modify Winnow so that it returns not just a classifier that classifies everything correctly, but a large-margin classifier with margin at least say $\epsilon/2$? Briefly sketch how you would modify the algorithm to do this. Next, modify your program, and run it on the above data. Using binary search, what is the largest margin classifier that you can find for the above data?

7. Write a brief proposal for your class project (2-3 paragraphs). Please talk to Prof. Saia about your ideas before you start writing this.