Assignment 1
CS289: Algorithmic Machine Learning, Fall 2016
Due: October 12, 10PM

Guidelines for submitting the solutions:

- The assignments need to be submitted on Gradescope. Make sure you follow all the instructions - they are simple enough that exceptions will not be accepted.
- The solutions need to be submitted by 10 PM on the due date. No late submissions will be accepted.
- Please adhere to the code of conduct outlined on the class page.

1. Consider the hypothesis class $\mathcal{H}$ consisting of functions $f : \{0,1\}^d \to \{0,1\}$ of the form $OR_S(x) = \vee_{i \in S} x_i$ for subsets $S \subseteq [d]$. Now, suppose you are given as input $(example, label)$ pairs $(x^1, y_1), \ldots, (x^m, y_m)$ where $x^\ell \in \{0,1\}^d$ and $y^\ell = OR_S(x^\ell)$ for some unknown S. Give a polynomial-time algorithm that given the data finds a hypothesis $h \in \mathcal{H}$ that is consistent with the data[1]; that is, give an algorithm to find a subset $T \subseteq [d]$ such that $OR_T(x^\ell) = y_\ell$ for all $\ell = 1, \ldots, m$. You need not prove your algorithm works. [2 points]
(Hint: You can give an iterative algorithm that goes through the examples in order and updates the candidate set T. Start by setting $T = [d]$. If you see a positive example, you need not do anything. How should you update T if you see a negative example (x^ℓ, y^ℓ) where $y^\ell = 0$?)

2. Show that the Boolean OR function, $f : \{0,1\}^n \to \{0,1\}$ with $f(x) = 0$ if and only if x is all-zeros can be written as a linear threshold function or halfspace. [1 point]

3. Write down a linear separator for the OR function as above whose margin is $\Omega(1/\sqrt{n})$. Prove the lower bound on the margin. [2 points]

4. Show that the weighted majority algorithm we did in class can be off from the best-expert by a factor of 2. That is, give a setting of the learning with experts framework (i.e., state what the experts do, what the truth is) where the loss of the best expert $L^*(T) \leq T/2 + 1$ for all time steps T, but the loss incurred by the weighted majority algorithm $L(T) \geq T - 1$ for all time steps T. [2 points]

[1] By Occam's razor and what I hinted in class, such an algorithm in fact would also give an efficient algorithm for PAC learning disjunctions.

5. The goal here is to solve a variant of the learning with experts setup where the experts incur gains instead of losses: in each round/day each expert gains 1 if her prediction is correct and gains 0 otherwise. Given $\varepsilon > 0$, design a randomized-algorithm whose net gain is as close to that of the best experts: The expected gain of the algorithm on day T is at least $(1-\varepsilon)G^*(T) - (\ln n)/\varepsilon$. Here n is the number of experts and G^* denotes the largest net gain until day T among the experts. You also have to prove that the algorithm achieves the above bound. [3 points]
   (Hint: You basically have to redo the analysis of the algorithm we did in class where you increase the weights of experts who are correct by a multiplicative $(1+\varepsilon)$ factor. That is, weights are updated as $w_i(t) = (1+\varepsilon g_i(t))w_i(t-1)$, where $g_i(t)$ denotes the amount gained by the i'th expert at the end of Day t. The prediction rule can be the same one we used in class.)

Bonus The following simple experiments will help understand and/or appreciate the material presented in class. You can try MATLAB (or Julia or Numpy). You DO NOT have to turn these in; these are meant for your own practice.

1. Implement the multiplicative-weights algorithm and check its performance in the following cases: Two experts, with the first one always saying UP, the second one always saying DOWN and the truth being completely random. What if the truth oscillates as in the example we saw in class?