CS 598 (CRN 62819) Topics in Algorithms, Spring 2015

Homework # 3 **Due: March 18, 2015**

Homework is due in class (on paper) on the due date. For this assignment you may work with others if you wish, but in that case please list the names of your collaborators.

1. **Piazza.** Make a Piazza comment related to Chapter 6 if you have not done so yet. (This problem should be done individually.)

2. **Pruning a Decision Tree.** Let S be a labeled sample drawn iid from some distribution $\mathcal{D}$ over $\{0,1\}^n$, and suppose we have used S to create some decision tree T. However, the tree T is large, and we are concerned we might be overfitting. Give a polynomial-time algorithm for *pruning* T that finds the pruning h of T that optimizes the right-hand-side of Corollary 6.8, i.e., that for a given $\delta > 0$ minimizes:

$$err_S(h) + \sqrt{\frac{\text{size}(h)\ln(4) + \ln(2/\delta)}{2|S|}}.$$

   To discuss this, we need to define what we mean by a "pruning" of T and what we mean by the "size" of h. A pruning h of T is a tree in which some internal nodes of T have been turned into leaves, labeled "+" or "−" depending on whether the majority of examples in S that reach that node are positive or negative. Let $\text{size}(h) = L(h)\log(n)$ where $L(h)$ is the number of leaves in h.

   Hint #1: it is sufficient, for each integer $L = 1, 2, \ldots, L(T)$, to find the pruning of T with L leaves of lowest empirical error on S, that is, $h_L = \text{argmin}_{h:L(h)=L} err_S(h)$. Then you can just plug them all into the displayed formula above and pick the best one.

   Hint #2: use dynamic programming.

3. **"Pruning" a Decision Tree Online via Sleeping Experts.** Suppose that, as in the above problem, we are given a decision tree T, but now we are faced with a sequence of examples that arrive online. One interesting way we can make predictions is as follows. For each node v of T (internal node or leaf) create two sleeping experts: one that predicts positive on any example that reaches v and one that predicts negative on any example that reaches v. So, the total number of sleeping experts is $O(L(T))$.

   (a) Say why any pruning h of T, and any assignment of $\{+,-\}$ labels to the leaves of h, corresponds to a subset of sleeping experts with the property that exactly one sleeping expert in the subset makes a prediction on any given example.

   (b) Prove that for any sequence S of examples, if we run the sleeping-experts algorithm using $\epsilon = \sqrt{\frac{L\log(L(T))}{|S|}}$, then the error rate of the algorithm on S (the total number of mistakes of the algorithm divided by $|S|$) will be at most $err_S(h_L) + O(\sqrt{\frac{L\log(L(T))}{|S|}})$, where $h_L = \mathrm{argmin}_{h:L(h)=L} err_S(h)$ is the pruning of T with L leaves of lowest error on S.

4. **Margin Perceptron.** In class we showed that the Perceptron algorithm makes at most $1/\gamma^2$ mistakes on any sequence of examples that is separable by margin γ (we assume all examples are normalized to have length 1). However, it need not find a separator of large margin. If we also want to find a separator of large margin, a natural alternative is to update on any example $\mathbf{x}$ such that $f^*(x)(\mathbf{w} \cdot \mathbf{x}) < 1$; this is called the *margin perceptron* algorithm.

   (a) Argue why margin perceptron is equivalent to running stochastic gradient descent on the class of linear predictors ($f_{\mathbf{w}}(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x}$) using hinge loss as the loss function.

   (b) Prove that on any sequence of examples that are separable by margin γ, this algorithm will make at most $3/\gamma^2$ updates.

   (c) In part (b) you probably proved that each update increases $||\mathbf{w}||^2$ by at most 3. Use this (and your result from part (b)) to conclude that if you have a dataset S that is separable by margin γ, and cycle through the data until the margin perceptron algorithm makes no more updates, that it will find a separator of margin at least $\gamma/3$.

5. **VC-dimension**. What is the VC-dimension V of the class $\mathcal{H}$ of axis-parallel boxes in R^d? That is, $\mathcal{H} = \{h_{\mathbf{a},\mathbf{b}} : \mathbf{a}, \mathbf{b} \in R^d\}$ where $h_{\mathbf{a},\mathbf{b}}(\mathbf{x}) = 1$ if $a_i \leq x_i \leq b_i$ for all $i = 1, \ldots, d$ and $h_{\mathbf{a},\mathbf{b}}(\mathbf{x}) = -1$ otherwise.

   (a) Prove that the VC-dimension is at least your chosen V by giving a set of V points that is shattered by the class (and explaining why it is shattered).

   (b) Prove that the VC-dimension is at most your chosen V by proving that no set of $V + 1$ points can be shattered.

6. **Boosting and linear separators**. Let S be a set of labeled examples in R^d that is linearly separable by some separator $\mathbf{w}^*$ of margin γ (assume $||\mathbf{x}|| = 1$ for all $\mathbf{x} \in S$).

   (a) Fix $\mathbf{x} \in S$ and let $\mathbf{w}$ be a *random* unit-length vector in R^d subject to $\mathbf{w} \cdot \mathbf{w}^* \geq 0$. Prove that
   $$\Pr(\mathrm{sgn}(\mathbf{x} \cdot \mathbf{w}) = \mathrm{sgn}(\mathbf{x} \cdot \mathbf{w}^*)) \geq 1/2 + \gamma/\pi.$$

   (b) Argue why (a) implies that for any weighting over S, the expected fraction of the weight that such a $\mathbf{w}$ gets correct is at least $1/2 + \gamma/\pi$.

   This implies that one way to learn a large margin separator is to run Boosting, at each stage repeatedly picking random vectors $\mathbf{w}$ until we find one such that either $\mathbf{w}$ or $-\mathbf{w}$ has accuracy $1/2 + \Omega(\gamma)$ under the current weighting, and use that as our h_t.