15-859(B) Machine Learning Theory

Homework # 2 **Due: February 8, 2010**

Groundrules: Same as before. You should work on the exercises by yourself but may work with a partner on the problems (just write down who you worked with). Also if you use material from outside sources, say where you got it.

Exercises:

1. **A bad modification to Winnow.** Suppose that we modify Winnow so that it doubles its weights on positive examples even when it did *not* make a mistake. Show how this can cause the algorithm to make an unbounded number of mistakes, even if all examples *are* consistent with some disjunction.

2. **Balanced Winnow.** Here is a variation on the Winnow algorithm, called *Balanced Winnow.* First of all, we introduce a fake variable x_0 which is set to 1 in every example. For each variable x_i ($0 \leq i \leq n$), and each output value y (as usual, $y \in \{-, +\}$, but you can also use this algorithm for multi-valued outputs) we have a weight w_{iy}. All weights are initialized to 1. In addition, we are given parameters $\alpha > 1$ and $\beta < 1$. The algorithm proceeds as follows:

   (a) Given example x, predict the label y such that $\sum_i x_i w_{iy}$ is largest.

   (b) If the algorithm makes a mistake, predicting y' when then correct answer is y, then for each $x_i = 1$, multiply the weight w_{iy} by α, and multiply $w_{iy'}$ by β.

   Using $\alpha = 3/2$ and $\beta = 1/2$, prove that as with the standard Winnow algorithm, this algorithm makes at most $O(r \log n)$ mistakes on any disjunction (OR-function) of r variables.

3. **About δ.** Suppose we changed the definition of PAC-learning to only require an algorithm succeed in finding a low-error hypothesis with probability at least 1/2 (rather than with probability $1 - \delta$). Show that this does not change what is learnable. Specifically, show that if we had an algorithm $\mathcal{A}$ with at least a $\frac{1}{2}$ chance of producing a hypothesis of error at most $\epsilon/2$, we could convert it into an algorithm $\mathcal{B}$ that has at least a $1 - \delta$ probability of producing a hypothesis of error at most ϵ. The reduction is that we first run $\mathcal{A}$ for $N = \lg \frac{2}{\delta}$ times (so with probability at least $1 - \delta/2$, at least one of the N hypotheses produced has error at most $\epsilon/2$), and we then test the N hypotheses produced on a new test set, choosing the one that performs best. Use Chernoff bounds to analyze this second step and finish the argument. That is, assuming that at least one of N hypotheses has error at most $\epsilon/2$, give an explicit bound (without O notation) on a size for the test set that is sufficient so that with probability at least $1 - \delta/2$, the hypothesis that performs best on the test set has error at most ϵ.

Problems:

4. **Winnow and L_1 margins.** Show that Winnow (the version discussed in class) with learning rate $\epsilon = \gamma/2$ makes at most $O((1/\gamma^2)\log n)$ mistakes on any sequence of examples in $\{0,1\}^n$ consistent with a linear separator of L_1-margin γ. In particular, we assume for some target vector $(w_1^*, \ldots, w_n^*)$ and threshold c, the positive examples satisfy $\sum_i w_i^* x_i \geq c$ and the negative examples satisfy $\sum_i w_i^* x_i \leq c - \gamma$, where we have normalized the target vector so that $\sum_i |w_i^*| = 1$. As mentioned in class, we can assume $w_i^* \geq 0$ for all i (otherwise we just define additional variables $y_i = 1 - x_i$).

   To do this, follow the analysis from class, where the quantity $\sum_i w_i^* \log_{1+\epsilon} w_i$ now plays the role of the "number of chips on relevant features" (and $\log_{1+\epsilon} w_i$ is the number of chips on feature i). At some point in your analysis you will use the fact that $\gamma - \epsilon(c - \gamma) \geq \gamma/2$, which follows from the definition of ϵ and the fact that $c \leq 1$ (because we have normalized the target to have L_1 norm of 1).

5. **Tracking a moving target.** Here is a variation on the deterministic Weighted-Majority algorithm, designed to make it more adaptive.

   (a) Each expert begins with weight 1 (as before).

   (b) We predict the result of a weighted-majority vote of the experts (as before).

   (c) If an expert makes a mistake, we penalize it by dividing its weight by 2, but *only* if its weight was at least 1/4 of the average weight of experts.

   Prove that in any contiguous block of trials (e.g., the 51st example through the 77th example), the number of mistakes made by the algorithm is at most $O(m + \log n)$, where m is the number of mistakes made by the best expert *in that block*, and n is the total number of experts.