15-859(B) Machine Learning Theory

Homework # 1 **Due: January 30, 2006**

Groundrules:

- Homeworks will generally consist of *exercises*, easier problems designed to give you practice, and *problems*, that may be harder, and/or somewhat open-ended. You should do the exercises by yourself, but you may work with a friend on the harder problems if you want. (Working together doesn't mean "splitting up the problems" though.) If you work with a friend, then write down who you are working with.
- If you've seen a problem before (sometimes I'll give problems that are "famous"), then say that in your solution. It won't affect your score, I just want to know. Also, if you use any sources other than the textbook, write that down too. It's fine to look up a complicated sum or inequality or whatever, but don't look up an entire solution.

Exercises:

1. **Expressivity of decision lists.** Show that conjunctions and disjunctions are both special cases of decision lists. That is, any function that can be expressed as a conjunction (or disjunction) can also be expressed as a decision list.

   Note: the example given in class shows that decision lists are strictly more general. That data set had a consistent decision list but no consistent conjunction or disjunction.

2. **Mistake-bound model.** Give an algorithm to learn 2-CNF formulas over n boolean features in the mistake-bound model. Your algorithm should run in polynomial-time per example (so the "halving algorithm" is not allowed). How many mistakes does it make at most?

Problems:

3. **Decision List mistake bound.** Give an algorithm that learns the class of decision lists in the mistake-bound model, with mistake bound $O(nL)$ where n is the number of variables and L is the length of the shortest decision list consistent with the data. The algorithm should run in polynomial time per example.

   Hint: think of using some kind of "lazy" version of decision lists as hypotheses.[1]

4. **Expressivity of LTFs.** Show that decisions lists are a special case of linear threshold functions. That is, any function that can be expressed as a decision list can also be expressed as a linear threshold function.

5. **Halving is not always optimal.** Describe a class C where the halving algorithm is not optimal: that is, where you would get a better worst-case mistake bound by *not* going with the majority vote of the available concepts.

[1] An alternative is to use your solution to problem 4, plus an appropriate online linear programming algorithm. But I'd rather you not solve this problem that way.