COS 511: Theoretical Machine Learning

Homework #1 — Due:
PAC Learning — February 20, 2019

General comments: Be sure to read the collaboration and late policies on the course website. In particular, you should attempt to solve all homework problems on your own before joining with a partner or in a small group to solve them together. Also, from time to time, especially if asked to do so by students, I may post hints for some of the problems on the course website which you can choose to look at, or not (but again, try to solve them first without the hints). And if you are stuck, you are always free to ask me or the TA's for help or hints.

It is a very good idea to start early on the problem sets, at least to read them over so that you can be thinking about them in the background.

Be sure to *show your work and justify your answers in a mathematically rigorous fashion.* For full credit, your solutions should be written up in a way that is clear, correct, complete, and convincing. Approximate point values are shown in brackets.

Information on how and when to turn in the homeworks is (or soon will be) available on the course website. If you turn your homework in late and in hard-copy, remember to write down the day and time when it was submitted. Also, in all cases, be sure to note anyone that you worked with.

Problem 1

[15] Consider the following learning problem: Let the domain be $X = \mathbb{R}$, and let $\mathcal{C} = \mathcal{C}_s$ be the class of concepts defined by unions of s intervals. That is, each concept c is defined by real numbers $a_1 \leq b_1 \leq \cdots \leq a_s \leq b_s$ where $c(x) = 1$ if and only if $x \in [a_1, b_1] \cup \cdots \cup [a_s, b_s]$.

Describe an efficient algorithm that learns the class $\mathcal{C}_s$ for every s, assuming that s is known ahead of time to the learner. You should describe a single algorithm that works for all $\mathcal{C}_s$, provided that s is known so that the learner can choose the number of examples needed as a function of ϵ, δ and s. You can use any hypothesis space you wish. Prove that your algorithm is PAC (i.e., produces a hypothesis with error at most ϵ with probability at least $1 - \delta$), and derive an exact expression for the number of examples needed. Also argue briefly that your algorithm runs in time polynomial in $1/\epsilon$, $1/\delta$ and s.

For this problem (only), it is okay to make simplifying assumptions about the target distribution, for instance, of the kind that were made in class. This also means that you do not need to handle all "corner cases." To the degree possible, the assumptions that you are making should be spelled out explicitly in your solution.

Problem 2

The Occam's razor result proved in class only applies to finite $\mathcal{H}$. Suppose now that $\mathcal{H}$ is discrete, i.e., either finite or countably infinite. Let $g : \mathcal{H} \to (0, 1]$ be any function such that

$$\sum_{h \in \mathcal{H}} g(h) \leq 1.$$

Although g may look a bit like a probability distribution, you should *not* think of it as one. It is just a function — any function — whose positive values happen to add up to a number not bigger than one.

Let m be the number of given examples (each chosen at random, as usual, from some unknown distribution D).

a. [10] Prove that, with probability at least $1 - \delta$,

$$\text{err}_D(h) \leq \frac{\ln(1/g(h)) + \ln(1/\delta)}{m}$$

for all $h \in \mathcal{H}$ that are consistent with the observed data. As usual, $\text{err}_D(h) = \Pr_{x \sim D}[h(x) \neq c(x)]$, and c is the target concept.

b. [10] Suppose hypotheses in $\mathcal{H}$ are represented by bit strings and that $|h|$ denotes the number of bits needed to represent h. Show how to choose g to prove that

$$\text{err}_D(h) \leq O\left(\frac{|h| + \ln(1/\delta)}{m}\right)$$

for all $h \in \mathcal{H}$ that are consistent with the observed data (with probability at least $1 - \delta$). Give explicit constants (in other words, give a bound that does not use $O(\cdot)$ notation).

c. [5] How does the bound in (b) reflect the intuition that "simpler" hypotheses should be prefered to more "complex" ones? How does the bound in (a) reflect the intuition that prior knowledge helps learning?

Problem 3

For this problem, you need not be concerned about computational efficiency. Throughout this problem, as usual, $\mathcal{C}$ and $\mathcal{H}$ are classes of concepts defined on the domain $\mathcal{X}$.

a. [10] Prove or disprove the following statement: For every *finite* domain $\mathcal{X}$, if $\mathcal{C}$ is PAC learnable by $\mathcal{H}$, then $\mathcal{C} \subseteq \mathcal{H}$. (To prove the statement, you of course need to give a proof showing that it is always true. To disprove the statement, you can simply provide a counterexample showing that it is not true in general.)

b. [10] Repeat part (a) *without* the assumption that $\mathcal{X}$ is finite. In other words, prove or disprove that: For every (not necessarily finite) domain $\mathcal{X}$, if $\mathcal{C}$ is PAC learnable by $\mathcal{H}$, then $\mathcal{C} \subseteq \mathcal{H}$.