Instructor: Rocco A. Servedio

Computer Science 4252: Introduction to Computational Learning Theory
Problem Set #3 Fall 2025

Due 11:59pm Tuesday, October 28, 2025

See the course Web page for instructions on how to submit homework.
Important: To make life easier for the TAs, **please start each problem on a new page.**

Remember to strive for both clarity and concision in your solutions;
solutions which are excessively long may be penalized.

Problem 1 In this problem you'll consider two weakenings of the notion of a "consistent hypothesis finder" and show that each of them is still strong enough for PAC learning. (As in the theorems about consistent hypothesis finders proved in class, you should assume throughout this problem that $\mathcal{H}$ is a finite hypothesis class.)

(i) Let us say that a (randomized) algorithm B is an "unreliable consistent hypothesis finder" for $\mathcal{C}$ using $\mathcal{H}$ if it has the following performance guarantee: Given any sample of m examples $(x^1, c(x^1)), \ldots (x^m, c(x^m))$ labeled according to some $c \in \mathcal{C}$, with probability $1/m$ (over B's own internal randomness) B outputs a hypothesis $h \in \mathcal{H}$ that is consistent with all the examples.

Show that an unreliable consistent hypothesis finder can be used to construct a PAC learning algorithm for $\mathcal{C}$. Justify your answer.

(ii) Now let us say that an algorithm B is an "almost consistent hypothesis finder" for $\mathcal{C}$ using $\mathcal{H}$ if it has the following performance guarantee: Given any sample of m examples $(x^1, c(x^1)), \ldots (x^m, c(x^m))$ labeled according to some $c \in \mathcal{C}$, B outputs a hypothesis $h \in \mathcal{H}$ that is incorrect on at most one of the m examples.

Show that an almost consistent hypothesis finder can be used to construct a PAC learning algorithm for $\mathcal{C}$. Justify your answer.

Problem 2 In this problem you'll be guided through a simplified derivation of a special case of the Chernoff bounds that we stated in class.

Suppose that a coin with heads probability $p = \frac{1}{2} - \varepsilon$ is tossed n times (so the expected number of times it comes up heads is $(\frac{1}{2} - \varepsilon)n$).

(a) Write down an expression which captures the exact probability that the coin comes up heads at least $n/2$ times. (Your expression should involve a sum; feel free to assume n is even.)

(b) By bounding the number of elements in the sum and the largest possible value of each summand, show that the above probability is at most $e^{-2\varepsilon^2 n}$.

Problem 3 In class we described a simple algorithm for PAC learning the concept class $\mathcal{C}$ of all closed intervals over the real line. However, there was a gap in our description of the algorithm, since we never specified how the algorithm should perform if the sample of draws it receives from $\mathsf{EX}(c, \mathcal{D})$ contains no positive examples.

Remedy this gap by explaining what the algorithm should do in this case and giving the necessary analysis to show that the resulting algorithm is indeed a PAC learning algorithm.

Problem 4 Recall that the conversion from an online algorithm with mistake bound m to a PAC algorithm given in class works as follows: "Run A on a sequence of examples each drawn independently from $\mathcal{D}$. If hypothesis h ever survives $\frac{1}{\varepsilon} \log \frac{m+1}{\delta}$ consecutive examples without making a mistake, stop and output h."

Now suppose that you have an online algorithm A with some finite mistake bound m, but you don't know what the value of m is. Explain how you can obtain a PAC algorithm from A with sample complexity $O(\frac{m}{\varepsilon} \log(\frac{m}{\delta}))$. (If you cannot achieve this exact sample complexity, give the best sample complexity that you can.)