Problem 1
(a) Consider the simple three-stage boosting scenario we analyzed in class, where the weak hypothesis always has accuracy exactly 0.6 with respect to the distribution used to generate it. Under this assumption, what is the smallest that $\Pr_{x \sim D_1}[h_1(x) \neq h_2(x)]$ could actually be?

(b) In our analysis of the three-stage boosting procedure, we described a way to simulate a draw from $EX(c, D_2)$ given access to $EX(c, D_1)$. What is the expected number of calls to $EX(c, D_1)$ that need to be performed in order to simulate a single draw from $EX(c, D_2)$?

(c) Describe another way to simulate a draw from $EX(c, D_2)$ given access to $EX(c, D_1)$, and analyze the efficiency of your approach (the expected number of calls to $EX(c, D_1)$ needed to simulate a single draw from $EX(c, D_2)$). (The new way should be more efficient than the way proposed in class and analyzed in part (b).)

Problem 2
In this problem you’ll explore how the AdaBoost algorithm (which, as we saw in class, works over a fixed sample of data points) can be used to efficiently PAC learn certain linear threshold functions.

(i) (easy) Suppose that $h$ and $f$ are both functions which take values in $\{-1, 1\}$. Show that for any distribution $D$, $h$ is a weak hypothesis for $f$ with advantage $\gamma$ if and only if $E_{x \sim D}[h(x)f(x)] \geq 2\gamma$.

(ii) Suppose that $f(x_1, \ldots, x_n) : \{-1, 1\}^n \to \{-1, 1\}$ is a linear threshold function $f(x) = \text{sign}(w \cdot x)$ where

1. each $x_i$ takes values in $\{-1, 1\}$;
2. $w = (w_1, \ldots, w_n)$ where each $w_i$ is an integer value and $W = \sum_{i=1}^n |w_i|$;
3. for all $x \in \{-1, 1\}^n$, we have $w_1x_1 + \cdots + w_nx_n \neq 0$.

Show that for any distribution $D$ over $\{-1, 1\}^n$, there must be some $x_i$ such that $|E_{x \sim D}[f(x) \cdot x_i]| \geq \frac{1}{W}$. (Hint: Use (and justify) the fact that $1 \leq E_{x \sim D}[|w \cdot x|].$)

(iii) Fix a polynomial $p(n)$ and let $C$ be the concept class of all linear threshold functions $f(x) = \text{sign}(w \cdot x)$ over $\{-1, 1\}^n$ as in (ii) where $\sum_{i=1}^n |w_i| \leq p(n)$. Show how AdaBoost can be used as a PAC learning algorithm for $C$. Analyze the running time and sample complexity of your algorithm. (Hint: Use AdaBoost as a consistent hypothesis finder.)
Problem 3: (a) Whoops! You ran home after lecture to code up the AdaBoost algorithm, but in your excitement you made a typo. Your finger slipped and you typed a “3” instead of a “2” in the definition of the \( \alpha_t \) parameter, i.e. your algorithm sets
\[
\alpha_t = \frac{1}{3} \ln \left( \frac{1 - \epsilon_t}{\epsilon_t} \right).
\]
State and prove an analogue of the \( \exp(-2 \sum_{t=1}^{T} \gamma_t^2) \) bound from class for your algorithm with the typo (the expression you obtain should be almost as simple as this). You don’t need to go through the entire analysis of the algorithm, only indicate where it is different and how. (Hint: Use the fact that
\[
(1/4 - \gamma^2)^{1/3} \left( (1/2 - \gamma)^{1/3} + (1/2 + \gamma)^{1/3} \right) \leq 1 - 16\gamma^2/9
\]
for all \(-1/2 \leq \gamma \leq 1/2\).)

(b) You fixed the typo from part (a), but unbeknownst to you there is a bug in the code for your weak learner as well. At each iteration \( t \) the weak learner constructs a weak hypothesis \( h_t \) which has error \( \epsilon_t = 1/2 - \gamma_t < 1/2 \), but instead of returning this weak hypothesis it returns the exact opposite of this hypothesis, \(-h_t\), instead. Will this bug affect the performance of your Adaboost implementation? Explain why or why not as clearly and precisely as you can.

Problem 4: Do Problem 5.4 from the Kearns and Vazirani textbook. That is, show that if there is an efficient algorithm for PAC learning in the presence of classification noise by an algorithm that is given a noise rate upper bound \( \eta_0 \) \((1/2 > \eta_0 \geq \eta \geq 0)\) and whose running time depends polynomially on \( \frac{1}{1-2\eta_0} \), then there is an efficient algorithm that is given no information about the noise rate \( \eta \) and whose running time depends polynomially on \( \frac{1}{1-2\eta} \).

Problem 5: Let \( H \) be a finite concept class on domain \( X \). We say that algorithm \( A \) minimizes disagreements if, given any sample \( S \) of labeled examples, it outputs the hypothesis \( h \) in \( H \) such that the number of examples in \( S \) on which \( h \) does not agree with the given label is minimum (over all hypotheses \( h \) in \( H \)).

Show that if \( A \) minimizes disagreements, then \( A \) can be used to obtain a (not necessarily computationally efficient) PAC learning algorithm for \( H \) in the presence of random classification noise at rate \( \eta < 1/2 \). You can assume that the noise rate \( \eta \) is given to the learning algorithm.

Problem 6: Consider the uniform distribution \( U \) over \( [N] = \{1, \ldots, N\} \). A single draw is guaranteed to return an element \( i \) that has \( U(i) = 1/N \), which is “typical” for draws from this distribution (since every element has weight 1/N).

Now consider the distribution \( D_i \) over \( [N] \) which puts all of its weight on the point \( i \). A single draw is guaranteed to return the element \( i \) that has \( D(i) = 1 \). Once again this is “typical” for draws from this distribution, since every draw from \( D_i \) will return an element (the same element) whose weight is 1.

In this problem you’ll show that the above examples are special cases of a general phenomenon: for any distribution, a small number of samples will “cover” most of the “typical” probability weights that the distribution assigns to elements.

Let \( D \) be a probability distribution on the set \( [N] = \{1, \ldots, N\} \). Given a value \( \epsilon > 0 \) and a point \( i \in [N] \), we say that a set \( R = \{r_1, \ldots, r_k\} \subseteq [N] \) \( \epsilon \)-covers \( i \) if there is some \( r_j \in R \) such that
\[
D(i) \in \left[ \frac{1}{1+\epsilon} \cdot D(r_j), (1+\epsilon) \cdot D(r_j) \right].
\]
(Here “$D(i)$” denotes the amount of probability weight that distribution $D$ puts on point $i$.)

Let $R$ be a sample of $m$ points drawn from $D$. Let $U$ (for “uncovered”) be the set of all points $i \in [N]$ such that $R$ does not $\epsilon$-cover $i$. Show that for a suitable choice of $m = \text{poly}(\log N, 1/\epsilon)$, with probability at least $99/100$ it is the case that $D(U) \leq \epsilon$. 