Instructor: Rocco A. Servedio

Computer Science 4252: Introduction to Computational Learning Theory
Problem Set #2 Fall 2025

Due 11:59pm Tuesday, October 7, 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 will analyze several variants of the Perceptron algorithm. For each variant, you should prove an analogue of the Perceptron Convergence Theorem we showed in class (under the same assumptions we used in class). How do the mistake bounds for the new algorithms compare with the original algorithm's mistake bound? (It's okay to do big-Oh style analysis in your bounds, you don't need to worry about exact constants.)

(i) Instead of adding or subtracting the example vector x that caused a mistake, on the k-th mistake the algorithm adds or subtracts $x/\sqrt{k}$ (so the changes to the current hypothesis vector grow smaller and smaller over time).

(ii) Instead of adding or subtracting the example vector x that caused a mistake, on the k-th mistake the algorithm adds or subtracts $\sqrt{k} \cdot x$ (so the changes to the current hypothesis vector grow larger and larger over time).

Problem 2 Recall that a "feature expansion" is a mapping $\Phi : \mathbb{R}^n \to \mathbb{R}^N$. The kernel function K corresponding to Φ is $K : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ defined as $K(x, y) = \Phi(x) \cdot \Phi(y)$.

Let $\Phi : \{0,1\}^n \to \{0,1\}^{2^n}$ be the feature expansion which has one feature for every possible monotone conjunction over the input variables $x_1, \dots, x_n$. For example, if $n = 3$ then $\Phi(x_1, x_2, x_3)$ equals

$$(1, x_1, x_2, x_3, x_1x_2, x_1x_3, x_2x_3, x_1x_2x_3).$$

(Note that the empty conjunction is equivalent to the always-true function, i.e. the constant-1 function).

(i) Show that the kernel function $K(x, y)$ for this Φ can be computed in time poly(n).

(ii) Now let Φ_ℓ be the feature expansion as in (i), but where the only output features are those corresponding to monotone conjunctions of length at most ℓ. Show that the kernel function $K_\ell(x, y)$ for this Φ_ℓ can be computed in poly(n) time.

(iii) Let us write $\{0,1\}^n_{\leq k}$ to denote the set of all n-bit strings that have at most k bits that are 1, i.e.

$$\{0,1\}^n_{\leq k} := \{x \in \{0,1\}^n : x_1 + \cdots + x_n \leq k\}.$$

It can be shown (you do **not** need to show this!) that the Perceptron algorithm can be used to learn an unknown monotone disjunction of length r over the domain $X = \{0,1\}^n_{\leq k}$ with a mistake bound of $O(rk)$.

Consider a learning scenario in which the unknown target concept is a monotone s-term DNF over n Boolean variables, but the domain is $X = \{0,1\}^n_{\leq \log n}$. (So in other words, you are promised you'll never receive an example with more than $\log n$ ones in it.) Explain how you could run the Perceptron algorithm to learn an unknown s-term monotone DNF in this setting. (Recall that a DNF is monotone if none of its terms contain any negations.) You may use the results of the earlier parts of this problem even if you do not succeed in solving them (and you may use the fact about the Perceptron algorithm over $\{0,1\}^n_{\leq k}$ stated above). What is the mistake bound of your algorithm? What is the worst-case running time of your algorithm on any trial?

Problem 3 Recall that in the setup for the basic Weighted Majority algorithm, there is a pool of N experts $E_1, E_2, \ldots, E_N$ each of whom makes a binary prediction at each trial. Suppose that expert E_i makes at most i mistakes over some sequence of trials.

(i) What bound can you prove for the Weighted Majority algorithm, run with update parameter β, on this sequence of trials?

(ii) What bound can you prove for the Randomized Weighted Majority algorithm, run with update parameter β, for the expected number of mistakes made on this sequence of trials? For $N = 1000$ and $\beta = .5$, which algorithm (Weighted Majority or Randomized Weighted Majority) has the better bound? If your life depended on making fewer than 25 mistakes, which algorithm would you use?

Problem 4 In this problem you'll think about an OLMB type learning scenario in a setting where there are *infinitely* many possible target concepts.

Let X be the infinite domain $\{1, 2, \ldots\}$ of all natural numbers. Suppose that c_1, c_2, $\ldots$ is an enumeration of some infinite set of computable 0/1-valued concepts over X. More precisely, this enumeration is computed by some program M; the output of M on input i is a program P_i that computes c_i.

Suppose you are learning an unknown function c, which is guaranteed to be one of the c_i's, in the online mistake-bound model.

(i) (easy) Give a learning algorithm which is guaranteed to make at most t prediction mistakes, where t is the least index such that $c = c_t$.

(ii) Now give a learning algorithm which is guaranteed to make at most $O(\log^2 t)$ prediction mistakes, where t is the least index such that $c = c_t$.

(iii) Now give a learning algorithm which is guaranteed to make at most $O(\log t)$ prediction mistakes, where t is the least index such that $c = c_t$.