CSE 417T: Homework 0

Due: 11:30am, January 25 (Tuesday), 2022

Notes:

- This is a special homework assignment for waitlisted students to complete. The instructor will check for correctness to make enrollment decisions. It will not be officially graded and will not factor in the final grades. However, the questions will appear again at homework 1. The submissions to homework 1 will be graded by TA and will impact the final grades as specified in the syllabus.
- **Enrolled students do not need to submit this homework assignment**. The same questions will appear in homework 1. Please submit your answers then.
- Please submit your homework via Gradescope. Please check the submission instructions for Gradescope provided on the course website. You must follow those instructions exactly.
- This special homework is due **by 11:30 AM on the due date. No late days are allowed**.
- The rule of academic integrity applies for this homework. **I intend to check for potential violations carefully**. If there are suspicions of cheating (for example, answers are too similar to other students' submissions or to other resources), it will be reported to the university even for students not enrolled. The university maintains **permanent record** if students are found guilty.
- Please keep in mind the collaboration policy as specified in the course syllabus. If you discuss questions with others you **must** write their names on your submission, and if you use any outside resources you **must** reference them. **Do not look at each others' writeups, including code.**

Problems:

1. Machine learning enables us to uncover the underlying patterns from data and builds on probabilistic inference. One of the key intuitions is that, with more data, we are more likely to more accurately estimate the underlying patterns. In this course, we will formalize this intuition using Hoeffding's inequality, one form of *the law of large numbers*. In this question, you are going to prove a simpler form of the law, Chebyshev's inequality.
   (a) If t is a non-negative random variable, prove that for any $\alpha > 0$, $\mathbb{P}[t \geq \alpha] \leq \mathbb{E}(t)/\alpha$. (Hint: Try to write down $\mathbb{E}(t)$ using the law of total expectation with the two partitions on t: $t \geq \alpha$ and $t < \alpha$.)
   (b) If u is any random variable with mean μ and variance σ^2, prove that for any $\alpha > 0$, $\mathbb{P}[(u-\mu)^2 \geq \alpha] \leq \frac{\sigma^2}{\alpha}$. (Hint: Use the result of (a)).

(c) if $u_1, \ldots, u_N$ are iid random variables, each with mean μ and variances σ^2, and $u = \frac{1}{N}\sum_{n=1}^{N} u_n$, prove that for any $\alpha > 0$,

$$\mathbb{P}[(u-\mu)^2 \geq \alpha] \leq \frac{\sigma^2}{N\alpha}.$$

The above result says that, if you have N data observations, when using the empirical mean as an estimation for the true mean, the probability for the estimation to be *bad* (if we define the estimation to be bad when the square error is larger than α) is in the order of $O(1/N)$. [1]

2. We have introduced perceptron learning algorithm (PLA) in the first lecture. In this question, you are going to first formally prove that PLA converges within finite steps for separable data. You then need to implement PLA using Python and conduct experiments to examine its empirical performance.

2.1. To prove the property of PLA, please follow the steps of LFD Problem 1.3.

- The photocopy of the problem is attached in the end in case you don't have access to the textbook yet.

2.2. Implement perceptron learning algorithm (PLA) and examine its performance. Please complete and submit the following python file for this problem. `http://chienjuho.com/courses/cse417t/hw0/hw0.py`

You need to submit both your code and the report of this problem. For the code submission, fill in the function implementations and submit `hw0.py`. You can write additional functions or additional code in the main function. You need to include the figures/answers **in the report**. Figures/answers in the code do not count.

Description: Consider the following experiment on running PLA for random training sets of size 100 and dimension 10 (i.e., $N = 100$ and $d = 10$.)

- Create a random optimal separator $\vec{w}^*$:
  Generate an 11-dimensional weight vector $\vec{w}^*$, where the first dimension (i.e., w_0^*) is 0 and the other 10 dimensions are sampled independently and uniformly at random between from $[0, 1]$ (we just set w_0^* to 0 for convenience).
- Generate a random training set with 100 data points, i.e., $D = \{(\vec{x}_1, y_1), ...(\vec{x}_{100}, y_{100})\}$, that are separable by $\vec{w}^*$:
  For each training data point $\vec{x}$, sample each of the 10 dimensions independently and uniformly at random from $[-1, 1]$ (Note that you need to insert $x_0 = 1$ for each data point $\vec{x}$). Calculate the label y of each data point $\vec{x}$ using the separator $\vec{w}^*$ (we assume separable data, so each point needs to be correctly classified by $\vec{w}^*$).
- Run the perceptron learning algorithm:
  Implement and run PLA on the training set you just generated, starting with the zero weight vector. Keep track of the number of iterations it takes to learn a hypothesis that correctly separates the training data.

Write code in Python to perform the above experiment and then repeat it 1000 times (note that you're generating a new $\vec{w}^*$ and a new training set D each time). We have provided two function headers (`perceptron_experiment` and `perceptron_learn`)

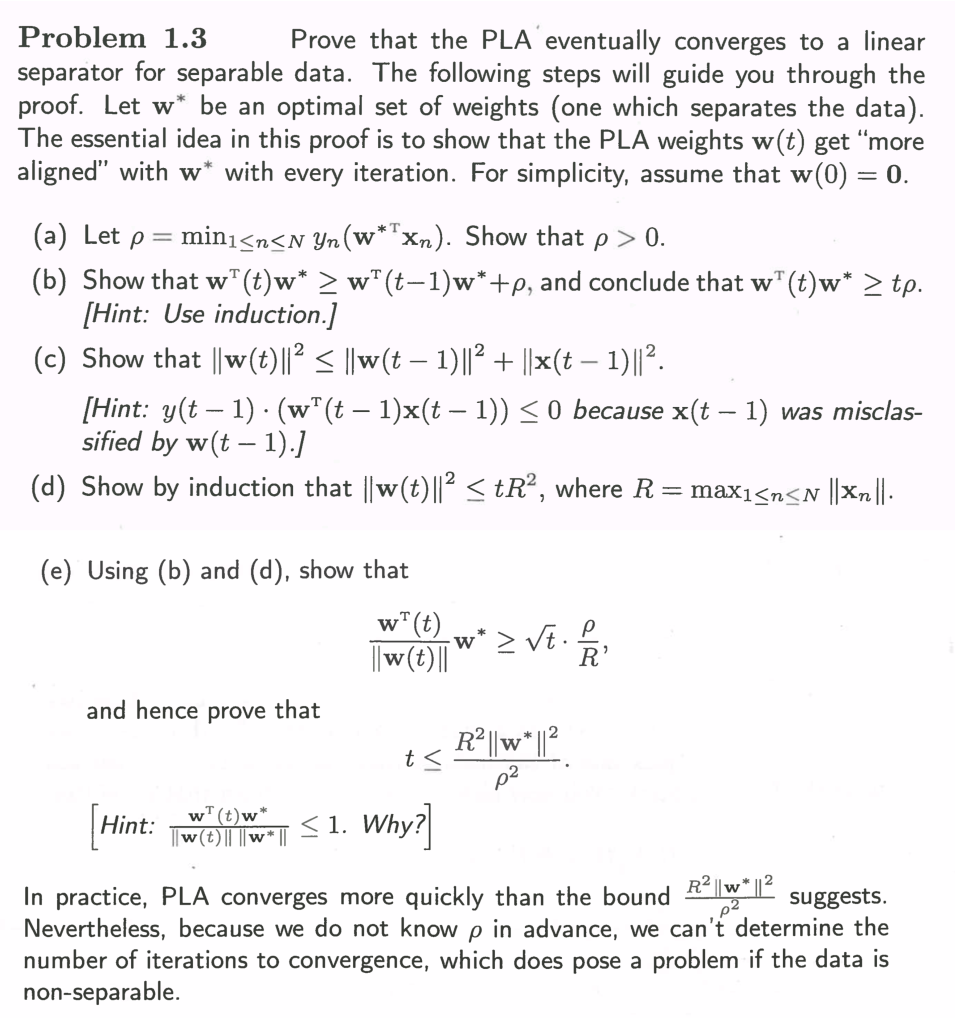
[1] This is the big-O notation you should know from CSE 247 or other algorithm courses. We'll also talk about a tighter bound in this course.

that you should complete for this purpose. The file has comments that explain their inputs and outputs.

Summarize your results in the report. Note that only the content included in the report will be graded. In particular, include the following in your report:

- Plot a histogram of the number of iterations PLA takes to learn a linear separator.
- Compare the number of iterations with the theoretical bound derived in Problem 2.1. Note that the bound will be different for each instantiation of $\vec{w}^*$ and the training set D. In order to answer this question, you should analyze the distribution of differences between the bound and the number of iterations. Plot a histogram of the **log** of this difference.
- Discuss your interpretation of these results.

3. Explain the reasons why you want/need to take this course in this semester. The enrollment priorities will be given to students who benefit the most by taking the course now.



Problem 1.3 Prove that the PLA eventually converges to a linear separator for separable data. The following steps will guide you through the proof. Let $\mathbf{w}^*$ be an optimal set of weights (one which separates the data). The essential idea in this proof is to show that the PLA weights $\mathbf{w}(t)$ get "more aligned" with $\mathbf{w}^*$ with every iteration. For simplicity, assume that $\mathbf{w}(0) = \mathbf{0}$.

(a) Let $\rho = \min_{1 \le n \le N} y_n(\mathbf{w}^{*\mathrm{T}}\mathbf{x}_n)$. Show that $\rho > 0$.

(b) Show that $\mathbf{w}^{\mathrm{T}}(t)\mathbf{w}^* \ge \mathbf{w}^{\mathrm{T}}(t-1)\mathbf{w}^* + \rho$, and conclude that $\mathbf{w}^{\mathrm{T}}(t)\mathbf{w}^* \ge t\rho$. *[Hint: Use induction.]*

(c) Show that $\|\mathbf{w}(t)\|^2 \le \|\mathbf{w}(t-1)\|^2 + \|\mathbf{x}(t-1)\|^2$.

[Hint: $y(t-1) \cdot (\mathbf{w}^{\mathrm{T}}(t-1)\mathbf{x}(t-1)) \le 0$ *because* $\mathbf{x}(t-1)$ *was misclassified by* $\mathbf{w}(t-1)$*.]*

(d) Show by induction that $\|\mathbf{w}(t)\|^2 \le tR^2$, where $R = \max_{1 \le n \le N} \|\mathbf{x}_n\|$.

(e) Using (b) and (d), show that

$$\frac{\mathbf{w}^{\mathrm{T}}(t)}{\|\mathbf{w}(t)\|}\mathbf{w}^* \ge \sqrt{t} \cdot \frac{\rho}{R},$$

and hence prove that

$$t \le \frac{R^2 \|\mathbf{w}^*\|^2}{\rho^2}.$$

$\left[\textit{Hint: } \frac{\mathbf{w}^{\mathrm{T}}(t)\mathbf{w}^*}{\|\mathbf{w}(t)\|\,\|\mathbf{w}^*\|} \le 1. \textit{ Why?}\right]$

In practice, PLA converges more quickly than the bound $\frac{R^2\|\mathbf{w}^*\|^2}{\rho^2}$ suggests. Nevertheless, because we do not know ρ in advance, we can't determine the number of iterations to convergence, which does pose a problem if the data is non-separable.