Homework #1

RELEASE DATE: 09/09/2024

RED CORRECTION: 09/16/2024 06:00

DUE DATE: 10/07/2024, BEFORE 13:00 on GRADESCOPE

QUESTIONS ARE WELCOMED ON DISCORD (INFORMALLY) OR VIA EMAILS (FORMALLY).

You will use Gradescope to upload your scanned/printed solutions. For problems marked with (*), please follow the guidelines on the course website and upload your source code to Gradescope as well. Any programming language/platform is allowed.

Any form of cheating, lying, or plagiarism will not be tolerated. Students can get zero scores and/or fail the class and/or be kicked out of school and/or receive other punishments for those kinds of misconducts.

Discussions on course materials and homework solutions are encouraged. But you should write the final solutions alone and understand them fully. Books, notes, and Internet resources can be consulted, but not copied from.

Since everyone needs to write the final solutions alone, there is absolutely no need to lend your homework solutions and/or source codes to your classmates at any time. In order to maximize the level of fairness in this class, lending and borrowing homework solutions are both regarded as dishonest behaviors and will be punished according to the honesty policy.

You should write your solutions in English ~~or Chinese~~ with the common math notations introduced in class or in the problems. We do not accept solutions written in any other languages.

This homework set comes with 200 points and 20 bonus points. In general, every homework set would come with a full credit of 200 points, with some possible bonus points.

1. (10 points, auto-graded) Which of the following tasks is best suited for machine learning? Choose the best answer.

[a] generate an image of Hercules that matches his actual facial look

[b] search for the shortest road path from Taipei to Taichung

[c] summarize any news article to 10 lines

[d] predict whether Schrödinger's cat is alive or dead inside the box

[e] none of the other choices

2. (10 points, auto-graded) Assume that a data set of an even size N, with $\frac{N}{2}$ being positive examples and $\frac{N}{2}$ being negative. If each example is used to update $\mathbf{w}_t$ in PLA exactly once. What is the resulting w_0 in $\mathbf{w}_{\text{PLA}}$? Please assume that the initial weight vector $\mathbf{w}_0$ is $\mathbf{0}$. Choose the correct answer.

[a] N

[b] $\frac{N}{2}$

[c] 0

[d] $-\frac{N}{2}$

[e] none of the other choices

3. (10 points, auto-graded) Dr. Norman thinks PLA will be highly influenced by very long examples, as $\mathbf{w}_t$ changes drastically if $\|\mathbf{x}_{n(t)}\|$ is large. Hence, ze decides to preprocess the training data by scaling down each input vector by 2 i.e., $\mathbf{z}_n \leftarrow \frac{\mathbf{x}_n}{2}$. How does PLA's upper bound on Page 19 of Lecture 2 change with this preprocessing procedure, with respect to the R and ρ that were calculated before scaling? Choose the correct answer.

[a] $\frac{2R^2}{\rho^2}$

[b] $\frac{R^2}{\rho^2}$

[c] $\frac{R^2}{2\rho^2}$

[d] $\frac{R^2}{4\rho^2}$

[e] none of the other choices

4. (10 points, auto-graded) Dr. Norman has another idea of scaling. Instead of scaling by a constant, ze decides to preprocess the training data by normalizing each input vector i.e., $\mathbf{z}_n \leftarrow \frac{\mathbf{x}_n}{\|\mathbf{x}_n\|}$. How does PLA's upper bound on Page 19 of Lecture 2 change with this preprocessing procedure in terms of $\rho_{\mathbf{z}} = \min_n \frac{y_n \mathbf{w}_f^T \mathbf{z}_n}{\|\mathbf{w}_f\|}$? Choose the correct answer.

[a] ∞ (i.e., PLA might never terminate)

[b] $\frac{1}{\rho_{\mathbf{z}}^2}$

[c] $\frac{1}{2\rho_{\mathbf{z}}}$

[d] $\frac{1}{\sqrt{\rho_{\mathbf{z}}}}$

[e] none of the other choices

5. (20 points, human-graded) Go ask any chatGPT-like agent the following question, "what is a possible application of active learning?", list the answer that you get, and argue with 10-20 English sentences on whether you agree with the agent or not, as if you are the "boss" of the agent. The TAs will grade based on the persuasiveness of your arguments—please note that our TAs are more used to being persuaded by humans than machines. So if your arguments do not look very human-written, the TAs may not be persuaded.

6. (20 points, human-graded) Go ask any chatGPT-like agent the following question, "can machine learning be used to predict earthquakes?", list the answer that you get, and argue with 10-20 English sentences on whether you agree with the agent or not, as if you are the "boss" of the agent. The TAs will grade based on the persuasiveness of your arguments—please note that our TAs are more used to being persuaded by humans than machines. So if your arguments do not look very human-written, the TAs may not be persuaded.

7. (20 points, human-graded) Before running PLA, our class convention adds $x_0 = 1$ to every $\mathbf{x}_n$ vector, forming $\mathbf{x}_n = (1, \mathbf{x}_n^{\text{orig}})$. Suppose that $x_0' = 2$ is added instead to form $\mathbf{x}_n' = (2, \mathbf{x}_n^{\text{orig}})$. Consider running PLA on $\{(\mathbf{x}_n, y_n)\}_{n=1}^N$ in a cyclic manner with the naïve cycle. That is, the algorithm keeps finding the next mistake in the order of $1, 2, \ldots, n, 1, 2, \ldots$. Assume that such a PLA with $\mathbf{w}_0 = \mathbf{0}$ returns $\mathbf{w}_{\text{PLA}}$, and running PLA on $\{(\mathbf{x}_n', y_n)\}_{n=1}^N$ with the same cyclic manner with $\mathbf{w}_0 = \mathbf{0}$ returns $\mathbf{w}_{\text{PLA}}'$. Prove or disprove that $\mathbf{w}_{\text{PLA}}$ and $\mathbf{w}_{\text{PLA}}'$ are equivalent. We define two weight vectors to be equivalent if they return the same binary classification output on every possible example in $\mathbb{R}^d$, the space that every $\mathbf{x}^{\text{orig}}$ belongs to. Please take any deterministic convention for sign(0), for example setting sign(0) = 1.

8. (20 points, human-graded) Before running PLA, our class convention adds $x_0 = 1$ to every $\mathbf{x}_n$ vector, forming $\mathbf{x}_n = (1, \mathbf{x}_n^{\text{orig}})$. Suppose that we scale every $\mathbf{x}_n$ by 3, and $x'_0 = 3$ is added instead to form $\mathbf{x}'_n = (3, 3\mathbf{x}_n^{\text{orig}})$. Consider running PLA on $\{(\mathbf{x}_n, y_n)\}_{n=1}^N$ in a cyclic manner with the naïve cycle. That is, the algorithm keeps finding the next mistake in the order of $1, 2, \ldots, n, 1, 2, \ldots$. Assume that such a PLA with $\mathbf{w}_0 = \mathbf{0}$ returns $\mathbf{w}_{\text{PLA}}$, and running PLA on $\{(\mathbf{x}'_n, y_n)\}_{n=1}^N$ with the same cyclic manner with $\mathbf{w}_0 = \mathbf{0}$ returns $\mathbf{w}'_{\text{PLA}}$. Prove or disprove that $\mathbf{w}_{\text{PLA}}$ and $\mathbf{w}'_{\text{PLA}}$ are equivalent. Similar to Problem 7, please take any deterministic convention for sign(0), for example setting sign(0) = 1.

9. (20 points, human-graded) Consider online hatred article detection with machine learning. We will represent each article $\mathbf{x}$ by the distinct words that it contains. In particular, assume that there are at most m distinct words in each article, and each word belongs to a big dictionary of size $d \geq m$. The i-th component x_i is defined as $[\![\text{word } i \text{ is in article } \mathbf{x}]\!]$ for $i = 1, 2, \ldots, d$, and $x_0 = 1$ as always. We will assume that d_+ of the words in the dictionary are more hatred-like, and $d_- = d - d_+$ of the words are less hatred-like. A simple function that classifies whether an artile is a hatred is to count $z_+(\mathbf{x})$, the number of more hatred-like words with the article (ignoring duplicates), and $z_-(\mathbf{x})$, the number of less hatred-like words in the article, and classify by

$$f(\mathbf{x}) = \text{sign}(z_+(\mathbf{x}) - z_-(\mathbf{x}) - 3.5).$$

That is, an article $\mathbf{x}$ is classified as a hatred iff the integer $z_+(\mathbf{x})$ is more than the integer $z_-(\mathbf{x})$ by 4.

Assume that f can perfectly classify any article into hatred/non-hatred, but is unknown to us. We now run an online version of Perceptron Learning Algorithm (PLA) to try to approximate f. That is, we maintain a weight vector $\mathbf{w}_t$ in the online PLA, initialized with $\mathbf{w}_0 = \mathbf{0}$. Then for every article $\mathbf{x}_t$ encountered at time t, the algorithm makes a prediction $\text{sign}(\mathbf{w}_t^T \mathbf{x}_t)$, and receives a true label y_t. If the prediction is not the same as the true label (i.e. a mistake), the algorithm updates $\mathbf{w}_t$ by

$$\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t + y_t \mathbf{x}_t.$$

Otherwise the algorithm keeps $\mathbf{w}_t$ without updating

$$\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t.$$

Derive an upper bound on the maximum number of mistakes that the online PLA can make for this hatred article classification problem. The tightness of your upper bound will be taken into account during grading.

Note: For those who know the bag-of-words representation for documents, the representation we use is a simplification that ignores duplicates of the same word.

10. (20 points, human-graded) Next, we use a real-world data set to study PLA. Please download the RCV1.binary (training) data set at

`https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/binary/rcv1_train.binary.bz2`

and takes the first $N = 200$ lines as our data set. Each line of the data set contains one $(\mathbf{x}_n, y_n)$ in the LIBSVM format, with $\mathbf{x}_n \in \mathbb{R}^{47205}$. The first number of the line is y_n, and the rest of the line is $\mathbf{x}_n$ represented in the sparse format that LIBSVM uses.

`https://www.csie.ntu.edu.tw/~cjlin/libsvm/faq.html#/Q03:_Data_preparation`

Please initialize your algorithm with $\mathbf{w} = \mathbf{0}$ and take sign(0) as -1.

Please first follow page 4 of Lecture 2, and add $x_0 = 1$ to every $\mathbf{x}_n$. Implement a version of PLA that randomly picks an example $(\mathbf{x}_n, y_n)$ in every iteration, and updates $\mathbf{w}_t$ if and only if $\mathbf{w}_t$ is incorrect on the example. Note that the random picking can be simply implemented *with replacement*—that is, the same example can be picked multiple times, even consecutively. Stop updating and return $\mathbf{w}_t$ as $\mathbf{w}_{\text{PLA}}$ if $\mathbf{w}_t$ is correct consecutively after checking $5N$ randomly-picked examples.

Hint: You can simply follow the algorithm above to solve this problem. But if you are interested in knowing why the algorithm above is somewhat equivalent to the PLA algorithm that you learned in class, here is some more information. (1) The update procedure described above is equivalent to the procedure of gathering all the incorrect examples first and then randomly picking an example among the incorrect ones. But the description above is usually much easier to implement. (2) The stopping criterion above is a randomized, more efficient implementation of checking whether $\mathbf{w}_t$ *makes no mistakes on the data set. Passing* $5N$ *times of correctness checking means that* $\mathbf{w}_t$ *is mistake-free with more than* 99% *of probability.*

Repeat your experiment for 1000 times, each with a different random seed. Plot a histogram to visualize the distribution of the number of updates needed before returning $\mathbf{w}_{\text{PLA}}$. Describe your findings. Then, provide the first page of the snapshot of your code as a proof that you have written the code.

(*Note: As a general principle, you can use any plotting software outside your data processing and machine learning code.*)

11. (20 points, human-graded) When running the 1000 experiments above, record $\|\mathbf{w}_t\|$ as a function of t. Plot the $\|\mathbf{w}_t\|$ in each experiment as a function of t for $t = 1, 2, \ldots, T_{\min}$, where $T_{\min}$ is the smallest number of updates in the previous problem. Superpose the 1000 functions on the same figure. Describe your findings. Then, provide the first page of the snapshot of your code as a proof that you have written the code.

12. (20 points, code needed, human-graded) Modify your PLA above to a variant that keeps correcting the same example until it is perfectly classified. That is, when selecting an incorrect example $(\mathbf{x}_{n(t)}, y_{n(t)})$ for updating, the algorithm keeps using that example (that is, $n(t+1) = n(t)$) to update until the weight vector perfectly classifies the example (and each update counts!). Repeat the 1000 experiments above. Plot a histogram to visualize the distribution of the number of updates needed before returning $\mathbf{w}_{\text{PLA}}$. What is the median number of updates? Compare your result to that of Problem 10. Describe your findings. Then, provide the first page of the snapshot of your code as a proof that you have written the code.

13. (Bonus 20 points, human-graded) When PLA makes an update on a misclassified example $(\mathbf{x}_{n(t)}, y_{n(t)})$, the new weight vector $\mathbf{w}_{t+1}$ does not always classify $(\mathbf{x}_{n(t)}, y_{n(t)})$ correctly. Consider a variant of PLA that makes an update by

$$\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t + \frac{1}{10} y_{n(t)} \mathbf{x}_{n(t)} \cdot \left\lfloor \frac{-10 y_{n(t)} \mathbf{w}_t^T \mathbf{x}_{n(t)}}{\|\mathbf{x}_{n(t)}\|^2} + 1 \right\rfloor .$$

First, prove that $\mathbf{w}_{t+1}$ always correctly classifies $(\mathbf{x}_{n(t)}, y_{n(t)})$ after the update. Second, prove that such a PLA halts with a perfect hyperplane if the data is linearly separable. (*Hint: Check Problem 12.*)