CIS5200: Machine Learning Spring 2025

Homework 4

Release Date: April 1, 2025 *Due Date: April 15, 2025*

- HW4 will count for 10% of the grade. This grade will be split between the written (30 points) and programming (18 points) parts.
- All written homework solutions are required to be formatted using LaTeX. Please use the template here. Do not modify the template. This is a good resource to get yourself more familiar with LaTeX, if you are still not comfortable.
- You will submit your solution for the written part of HW4 as a single PDF file via Gradescope. The deadline is **11:59 PM ET**. Contact TAs on Ed if you face any issues uploading your homeworks.
- Collaboration is permitted and encouraged for this homework, though each student must understand, write, and hand in their own submission. In particular, it is acceptable for students to discuss problems with each other; it is not acceptable for students to look at another student's written Solutions when writing their own. It is also not acceptable to publicly post your (partial) solution on Ed, but you are encouraged to ask public questions on Ed. If you choose to collaborate, you must indicate on each homework with whom you collaborated.
- Please refer to the notes and slides posted on the website if you need to recall the material discussed in the lectures.

1 Written Questions (30 points)

Problem 1: AdaBoost (18 points)

In this problem, we will learn more about the AdaBoost algorithm and prove that it can very quickly boost the performance of any weak learner.

Algorithm. Given a dataset $S = \{(x_1, y_1), \ldots, (x_m, y_m)\}$, at every iteration t, AdaBoost fits a weak learner h_t to a weighted dataset with weights w_t and then constructs a new weighted dataset with weights w_{t+1} by increasing the weights on *hard* examples that h_t makes mistakes on, and decreasing weights on *easy* examples that h_t gets right. In particular, for $i \in [m]$, the weights are recursively defined as:

$$\begin{aligned} w_{1,i} &= \frac{1}{m} \\ w_{t+1,i} &= \frac{w_{t,i}}{Z_t} \exp(-\alpha_t y_i h_t(x_i)), \end{aligned}$$

where α_t is the penalty for mistakes of h_t and $Z_t = \sum_{j=1}^{n} w_{t,j} \exp(-\alpha_t y_j h_t(x_j))$ is the normalizing constant that ensures w_{t+1} is a valid distribution. After running this procedure for T steps, AdaBoost returns the final predictor $H_T = \sum_{t=1}^{T} \alpha_t h_t$.

Note that even though each h_i is a binary predictor, H_T may not be a binary predictor, hence we will use the sign of H_T to get a prediction.

Guarantee. Assuming we have a weak learner, that is, for every weighted dataset, it gets error $\leq 1/2 - \gamma$, AdaBoost can boost it to high accuracy,

$$\mathsf{error}(H_T) := \frac{1}{m} \sum_{i=1}^{m} \mathbb{1}[\mathsf{sgn}(H_T(x_i)) \neq y_i] \leq \exp\left(-2\gamma^2 T\right).$$

1.1 (5 points) Show that the error of H_T is bounded by the exponential loss,

$$\mathsf{error}(H_T) \leq \frac{1}{m} \sum_{i=1}^{m} \exp(-y_i H_T(x_i)).$$

Hint: $x \leq 0$ if and only if $\exp(-x) \geq 1$

1.2 (5 points) Using the recursive definition of w_t, show that

$$\frac{1}{m} \sum_{i=1}^{m} \exp(-y_i H_T(x_i)) = \prod_{t=1}^{T} Z_t.$$

Hint: Compute $w_{T+1,i}$ in terms of $w_{1,i}$.

1.3 (5 points) Define the weighted error of the weak learner at time t as

$$\epsilon_t = \sum_{i=1}^{m} w_{t,i} \mathbb{1}[h_t(x_i) \neq y_i].$$

Show that,

$$Z_t = (1 - \epsilon_t) \exp(-\alpha_t) + \epsilon_t \exp(\alpha_t).$$

1.4 (3 points) Using 1.1 and 1.2, we have

$$\mathsf{error}(H_T) \leq \prod_{t=1}^{T} Z_t.$$

In order to make the error smallest possible, we can choose α_t that minimizes Z_t. Show that the optimal value is

$$\alpha_t = \frac{1}{2} \log \left(\frac{1 - \epsilon_t}{\epsilon_t} \right).$$

Completing the proof. Substituting the value of α_t in 1.3, we get

$$\begin{aligned} Z_t &= (1 - \epsilon_t) \exp(-\alpha_t) + \epsilon_t \exp(\alpha_t) \\ &= (1 - \epsilon_t) \sqrt{\frac{\epsilon_t}{1 - \epsilon_t}} + \epsilon_t \sqrt{\frac{1 - \epsilon_t}{\epsilon_t}} \\ &= 2\sqrt{\epsilon_t (1 - \epsilon_t)}. \end{aligned}$$

By the guarantee of the weak learner, we know that $\epsilon_t \leq 1/2 - \gamma$. We also know that $\epsilon_t(1 - \epsilon_t)$ is an increasing function of ϵ_t in $[0, 1/2)$. This gives us,

$$\begin{aligned} Z_t &\leq 2\sqrt{(1/2 - \gamma)(1/2 + \gamma)} \\ &= \sqrt{1 - 4\gamma^2} \\ &\leq \exp(-2\gamma^2). \end{aligned}$$

The last inequality used the fact that $1 - x \leq \exp(-x)$ for $x \in [0, 1]$. Now combining everything gives us,

$$\mathsf{error}(H_T) \leq \prod_{t=1}^{T} Z_t \leq \prod_{t=1}^{T} \exp(-2\gamma^2) = \exp(-2\gamma^2 T).$$

Optional (zero points) At iteration t, AdaBoost re-weighted the examples using the weak learner h_t, increasing the weights on points on which h_t was incorrect and decreasing the weight on which it is correct. Show that the error of h_t on the new distribution w_{t+1} is as good as random guessing, in particular,

$$\sum_{i=1}^{m} w_{t+1,i} \mathbb{1}[h_t(x_i) \neq y_i] = 0.5.$$

This is super cool because it ensures that h_t would not be selected again!

Problem 2: Auto-Differentiation (12 points)

Consider the function $y = f(x_1, x_2) = x_1 \cdot \ln(x_2) + x_1^2$.

2.1 (2 points) Draw the computational graph representing the evaluation of this function. Label all intermediate nodes (e.g., $v_1, v_2, \dots$).

2.2 (2 points) Let the input values be $x_1 = 2$ and $x_2 = e$. Perform the forward pass calculation. Show the computed value for each node in your computational graph. What is the final value of y?

2.3 (4 points) Perform the Forward-mode automatic differentiation trace to compute the partial derivative $\frac{\partial y}{\partial x_1}$. Show the derivative value ($\dot{v}_i = \frac{\partial v_i}{\partial x_1}$) computed at each node. Repeat the same for $\frac{\partial y}{\partial x_2}$.

2.4 (4 points) Perform the Reverse-mode automatic differentiation trace. Initialize the backward pass by setting $\overline{y} = \frac{\partial y}{\partial y} = 1$ at the output node. Show the derivative value ($\overline{v}_i = \frac{\partial y}{\partial v_i}$) computed at each node. What are the final values for $\frac{\partial y}{\partial x_1}$ and $\frac{\partial y}{\partial x_2}$?

2 Programming Questions (18 points)

Use the link here to access the Google Colaboratory (Colab) file for this homework. Be sure to make a copy by going to "File", and "Save a copy in Drive". As with the previous homeworks, this assignment uses the PennGrader system for students to receive immediate feedback. As noted on the notebook, please be sure to change the student ID from the default '99999999' to your 8-digit PennID.

Instructions for how to submit the programming component of HW 4 to Gradescope are included in the Colab notebook. You may find this PyTorch linear algebra reference and this general PyTorch reference to be helpful in perusing the documentation and finding useful functions for your implementation.