Written Assignment 1

Deadline: January 24th, 2024

Instruction: You may discuss these problems with classmates, but please complete the write-ups individually. Remember the collaboration guidelines set forth in class: you may meet to discuss problems with classmates, but you may not take any written notes (or electronic notes, or photos, etc.) away from the meeting. Your answers must be **typewritten**, except for figures or diagrams, which may be hand-drawn. Please submit your answers (pdf format only) on **Canvas**.

Q1. Least Square (35 points)

We have each data point $x = [x_1 \quad x_2]^T \in \mathbb{R}^2$, and we encounter the following data points:

i	$x_1^{(i)}$	$x_2^{(i)}$	$y^{(i)}$
1	1	1	-0.8
2	-1	-2	0.1
3	2	-1	-5.0

We are going to build a linear model (i.e., using least square with average error) to predict the label of each data point $x = (x_1, x_2)$ as follows: $\hat{y} = w_0 + w_1x_1 + w_2x_2$ where (w_0, w_1, w_2) are unknown weights that we will learn based on the above training dataset. Here w_0 is the intercept.

(a) Gradient descent. (20 points) Starting with $(w_0, w_1, w_2) = (0, 0, 0)$, we will run the gradient descent algorithm to learn these weights. Run for $T = 4$ iterations and report the value of the weights and the corresponding total error at the end of each iteration. The learning rate is $\alpha = 0.05$.
Note. For this question, you can compute each step manually or write a python program for it. If you write a python program, you will need to submit the program together with your answers.

(b) Closed-form solution. (15 points) Now we are going to use the closed-form solution to find (w_0, w_1, w_2). Provide the values of (w_0, w_1, w_2). What is the corresponding total error?
Note. For this question, you are allowed to use any tool to perform matrix inverse or multiplication.

Q2. Perceptrons (30 points)

Recall that a perceptron learns a linear classifier with weight vector w. It predicts:

$$y^{(i)} = sign(w \cdot x^{(i)})$$

(assuming that $y^{(i)} \in \{0, 1\}$). Also note that we are not using a bias weight w_0 for simplicity). When the perceptron makes a mistake, it updates the weights using the formula:

$$w = w + y^{(i)} x^{(i)}$$

Imagine that we have $x^{(i)} \in \mathbb{R}^2$, and we encounter the following data points:

i	$x_1^{(i)}$	$x_2^{(i)}$	$y^{(i)}$
1	1	1	1
2	2	-1	-1
3	-3	-1	1
4	-3	1	1

We are going to use the perceptron algorithm to learn the model weights. We will only train for one epoch (i.e., one complete round of going through each data point).

(a) (15 points) Starting with $w = [0 \quad 0]^T$, use the perceptron algorithm to learn on the data points in the order from top to bottom $i = 1, 2, 3, 4$. Provide values of the weights after each iteration of update using each data point. When the training process is done, does binary perceptron correctly classify all data points with the final values of the weights?

(b) (15 points) Starting with $w = [0 \quad 0]^T$, use the perceptron algorithm to learn on the data points in the order $i = 2, 1, 3, 4$. Provide values of the weights after each iteration of update using each data point. When the training process is done, does binary perceptron correctly classify all data points with the final values of the weights?

Q3. Logistic Regression (35 points)

We now consider the same training data set as in **Q2** but use the logistic regression instead. Recall that in logistic regression, we aim at maximizing the average log-likelihood of the entire dataset:

$$\max_w ll(w) = \max_w \frac{1}{n} \sum_{i=1}^{n} \log P(y^{(i)} \mid x^{(i)}; w)$$

where

$$P(y^{(i)} = +1 \mid x^{(i)}; w) = \frac{1}{1 + e^{-w \cdot x^{(i)}}}$$
$$P(y^{(i)} = -1 \mid x^{(i)}; w) = 1 - \frac{1}{1 + e^{-w \cdot x^{(i)}}}$$

In this question, we are not using a bias weight w_0 for simplicity and we start with $w = [0 \quad 0]^T$. We will run gradient ascent to iteratively update the weight:

$$w_1 = w_1 + \alpha \frac{\partial ll(w)}{\partial w_1}$$
$$w_2 = w_2 + \alpha \frac{\partial ll(w)}{\partial w_2}$$

(a) (15 points) Provide the gradient formulation for $\left[\frac{\partial ll(w)}{\partial w_1} \quad \frac{\partial ll(w)}{\partial w_2}\right]$ with respect to the training set in **Q2**.

(b) (15 points) What are values of the weight and the average log-likelihood objective value after four iterations of gradient ascent when the learning rate $\alpha = 0.01$ and when $\alpha = 0.2$, respectively?

(c) (5 points) Provide a brief comparison of the results between $\alpha = 0.01$ and $\alpha = 0.2$.

Note. For this question, you can either write a simple Python program to do all the computation steps or you can do these computation steps manually. You only have to provide values of the weight w. You don't need to submit your Python program (if you wrote one).