Name:

University of Illinois Spring 2020

CS 446/ECE 449 Machine Learning
Homework 2: Binary Logistic Regression

Due on Thursday February 13 2020, noon Central Time

1. **[22 points]** Binary Logistic Regression

   We are given a dataset $\mathcal{D} = \{(-1,-1), (1,1), (2,1)\}$ containing three pairs (x, y), where each $x \in \mathbb{R}$ denotes a real-valued point and $y \in \{-1, +1\}$ is the point's class label.

   We want to train the parameters $w \in \mathbb{R}^2$ (*i.e.*, weight w_1 and bias w_2) of a logistic regression model

   $$p(y|x) = \frac{1}{1 + \exp\left(-yw^\top \begin{bmatrix} x \\ 1 \end{bmatrix}\right)} \tag{1}$$

   using maximum likelihood while assuming the samples in the dataset $\mathcal{D}$ to be i.i.d.

   (a) (1 point) Instead of maximizing the likelihood we commonly minimize the negative log-likelihood. Specify the objective for the model given in Eq. (1). Don't use any regularizer or weight-decay.

   Your answer:

   (b) (3 points) Compute the derivative of the negative log-likelihood objective in general (the one specified in the previous question, *i.e.*, no regularizer or weight-decay). Sketch a simple gradient-descent algorithm using pseudo-code (use f for the function value, $g = \nabla_w f$ for the gradient, w for the parameters, and show the update rule).

   Your answer:

Name:

(c) (5 points) Implement the algorithm by completing A2_LogisticRegression.py. State the code that you implemented. What is the optimal solution w^* that your program found?

Your answer:

(d) (3 point) If the third datapoint $(2, 1)$ was instead $(10, 1)$, would this influence the bias w_2 much? How about if we had used linear regression to fit $\mathcal{D}$ as opposed to logistic regression? Provide a reason for your answer.

Your answer:

(e) (3 points) Instead of manually deriving and implementing the gradient we now want to take advantage of PyTorch auto-differentiation. Investigate A2_LogisticRegression2.py and complete the update step using the 'optimizer' instance. What code did you add? If you compare the result of A2_LogisticRegression.py with that of A2_LogisticRegression2.py after an equal number of iterations, what do you realize?

Your answer:

Name:

(f) (5 points) Instead of manually implementing the cost function we now also want to take advantage of available functions in PyTorch, specifically `torch.nn.BCEWithLogitsLoss`. Can we use the originally specified dataset $\mathcal{D}$ or do we need to modify it? How? What is the probability $p(y = 1|x)$, $p(y = 0|x)$ and $p(y|x)$ if we use `torch.nn.BCEWithLogitsLoss`, *i.e.*, how does it differ from Eq. (1)? (**Hint:** $w^\top \begin{bmatrix} x \\ 1 \end{bmatrix}$ still appears.)

Your answer:

(g) (2 points) Complete `A2_LogisticRegression3.py` and compare the obtained result after 100 iterations to the one obtained in previous functions. Does the result differ? Why? Why not?

Your answer: