Homework 1

Due: Sep 21, 2022 at 11:59pm EST

Instructions: This homework requires answering some open-ended questions, short proofs, and programming. This is an individual assignment, not group work. Though you may discuss the problems with your classmates, you must solve the problems and write the solutions independently. As stated in the syllabus, copying code from a classmate or the internet (even with minor changes) constitutes plagiarism. You are required to submit your answers in pdf form (use LaTeX) in a file called `hw1.pdf` to Gradescope under "HW1". Code should also be submitted to Gradescope in a file called `hw1.py` under "HW1 Programming". A LaTeX template for the pdf submission and a code skeleton for the code submission are available on Piazza. Please do **not** modify the base code in the coding skeleton; simply augment it with your solution. Late submissions do not receive full credit, except in extenuating circumstances such as medical or family emergency. Submissions submitted 0-24 hours late are eligible for only 90%, 24-48 hours late for 80%, 48-72 hours late for 70%, and later than 72 hours for 0% of the total credit for this assignment. Late days may be used (if available) to avoid these penalties.

The total assignment is worth 50 points.

Problem 1 (5 points)

This is an open-ended question to get you a head start on thinking about what you would like to do for your final project. Write a brief description of the kinds of data you have worked with (if you have not worked with data before, it is ok to say so) and would like to work with in this course. What kinds of predictive models would you like to build with such data, and for what purpose? There are no right or wrong answers; a paragraph or two is sufficient.

Problem 2 (7 points)

(i) Consider the absolute value function defined as,

$$f(x) = \begin{cases} x & \text{if } x > 0 \\ 0 & \text{if } x = 0 \\ -x & \text{if } x < 0. \end{cases}$$

Prove that the absolute value function is differentiable everywhere except at $x = 0$.

Hint: it is sufficient to state what the derivatives are for $x \neq 0$. For $x = 0$, show that the limit that defines the derivative at $x = 0$ does not exist. (5 points)

(ii) Comment (in a sentence or two) on how this affects the use of the mean absolute deviation as a loss function for machine learning. (2 points)

Problem 3 (5 points)

Provide 1 advantage and disadvantage for:

(i) Zero-order optimization methods in comparison to first-order methods
(ii) Second-order optimization methods in comparison to first-order methods

Problem 4 (13 points)

This question pertains to the convergence of gradient descent, and is meant to help develop better intuition about why it works. Recall the update rule for gradient descent at each step:

$$\theta^{(i+1)} = \theta^{(i)} - \alpha \nabla L(\theta^{(i)}).$$

Though gradient descent is a first-order method, most analyses of its convergence rely on twice differentiability and constraints on the smoothness of the loss function. For this proof, we assume $L(\theta)$ is twice differentiable and adopt a weak notion of smoothness known as Lipschitz continuity.

A function $L(\theta)$ is said to be Lipschitz continuous if the magnitude of its gradient does not vary too much with respect to its inputs. Formally, $L(\theta)$ is Lipschitz continuous with some constant $K > 0$ if the magnitude of the difference between the gradients for *any* two points $\theta^{(1)}$ and $\theta^{(2)}$ is less than or equal to K times the magnitude of the difference between the points themselves.[1]

Now consider the second-order Taylor approximation of $L(\theta)$ around $\theta^{(i)}$ (this is just a generalization of the Taylor expansion you may already know about to vector-valued inputs):

$$L(\theta) = L(\theta^{(i)}) + \nabla L(\theta^{(i)})^T(\theta - \theta^{(i)}) + \frac{1}{2}(\theta - \theta^{(i)})^T \nabla^2 L(\theta)(\theta - \theta^{(i)}).$$

Under our assumptions above, the loss function satisfies the following inequality:

$$L(\theta) \leq L(\theta^{(i)}) + \nabla L(\theta^{(i)})^T(\theta - \theta^{(i)}) + \frac{1}{2}(\theta - \theta^{(i)})^T K(\theta - \theta^{(i)}).$$

(i) Use the above relation (plugging in $\theta = \theta^{(i+1)}$ according to the gradient descent update rule) to prove that $L(\theta^{(i+1)}) \leq L(\theta^{(i)})$ if we use any constant learning rate $\alpha \leq 1/K$. That is, the Lipschitz constant K provides an intuitive bound on a learning rate α that will guarantee that the loss decreases (or at worst remains the same) at every step of gradient descent. (10 points)

(ii) Based on your analysis in (i), propose a learning rate α in relation to K that does not guarantee that the loss function will decrease at the next step. (3 points)

Problem 5 (20 points)

In this problem you will implement and apply linear regression based on the code skeleton provided in `hw1.py` and helper functions provided in `data_loader.py`. There are two datasets provided to help test your implementation: an artificially generated dataset in `simulated_data.csv` with features X_1, X_2, X_3 and outcome Y, and a real dataset that examines factors that might be

[1] In math, $||\nabla L(\theta)^{(1)} - \nabla L(\theta)^{(2)}|| \leq K||\theta^{(1)} - \theta^{(2)}||$, for any $\theta^{(1)}, \theta^{(2)}$. Here $||X||$ denotes the magnitude of a vector X defined as the square root of the dot product of X with itself: $\sqrt{X.X}$.

predictive of insurance charges – the features include continuous variables such as age and BMI, and categorical variables such as smoking status and sex. The outcome of interest in the insurance dataset is the charges incurred, i.e., medical costs billed by health insurance.[2]

The helper functions in data loader standardize continuous inputs in the data, converts categorical inputs to dummy 0/1 variables, and adds a column of ones to fit an intercept term. It also performs an 80-20 split of the insurance data into training and test sets.

(i) Implement linear regression using gradient descent.[3] (10 points)

(ii) Implement the R squared metric. Report and briefly comment on the R squared values obtained on the simulated data and the test set of the insurance data. (5 points)

(iii) Assume that you are building this model as a data scientist working for an insurance company. The administration is happy with the predictive accuracy of the model you've created, and are moving for its deployment to real-world settings, where they will use it to decide how much premium to charge to new clients. What are your thoughts on (a) the use of machine learning models (linear regression or otherwise) for this purpose, and (b) incorporating information on predictive but potentially sensitive attributes, e.g., sex, age, BMI, and race, in these models? Feel free to use the fitted weights of your model as a guide for how the outputs of the model would change to guide your discussion. Like question 1, there is not a single right way to answer this question, I am mostly looking for consistent arguments (for or against.) (5 points)

Important submission note:
Report the numbers you computed (just the Rsquared numbers, not the weights) and answers to the above questions in the PDF that you turn in. Hand in your Python code separately on Gradescope under HW 1 Programming. Please do **not** modify the existing code in the skeleton; simply augment it with your solution. Please make sure your variables and functions are aptly named, and your code is clear and readable, with comments where appropriate. I may try to execute your code on a new dataset as a test case, but don't worry too much about making your code robust in terms of error checking and adversarial inputs – this new dataset will look very similar to the one that's provided.

[2] All credit goes to Zach Stednick for scraping and cleaning the data from the public domain: `https://github.com/stedy/Machine-Learning-with-R-datasets`.

[3] Though there exists a closed form solution to gradient descent, we won't use it in this course as no other method we work with will have this property.