CSE514 – Fall 2022 Programming Assignment 1

This assignment is to enhance your understanding of objective functions, regression models, and the gradient descent algorithm for optimization. It consists of a programming assignment (with optional extensions for bonus points) and a report. This project is individual work, no code sharing please, but you may post bug questions to Piazza for help.

Topic

Design and implement a (stochastic) gradient descent algorithm or algorithms for regression.

Programming work

A) **Uni-variate linear regression**
In lecture, we discussed uni-variate linear regression y = f(x) = mx+b, where there is only a single independent variable x, using MSE as the loss function.
Your program must specify the objective function of mean squared error and be able to apply the gradient descent algorithm for optimizing a uni-variate linear regression model.

B) **Multi-variate linear regression**
In practice, we typically have multi-dimensional (or multi-variate) data, i.e., the input **x** is a vector of features with length p. Assigning a parameter to each of these features, plus the b parameter, results in p+1 model parameters. Multi-variate linear models can be succinctly represented as:

$$y = f(\mathbf{x}) = (\mathbf{m} \cdot \mathbf{x}) \qquad \text{(i.e., dot product between } \mathbf{m} \text{ and } \mathbf{x}\text{)},$$

where $\mathbf{m} = (m_0, m_1, \ldots, m_p)^T$ and $\mathbf{x} = (1, x_1, \ldots, x_p)^T$, with m_0 in place of b in the model.
Your program must be able to apply the gradient descent algorithm for optimizing a multi-variate linear regression model using the mean squared error objective function.

C) **Optional extension 1 – Mean Absolute Error as the loss function**
For bonus points, include the option of optimizing for the MAE instead of MSE. You'll get partial credit if this only works for the uni-variate model, and full credit if you make it work with the multi-variate model as well.

D) **Optional extension 2 – Data pre-processing**
For bonus points, get results after pre-processing the data by normalizing or standardizing each variable. This should be *in addition* to the results from not pre-processing, so you can analyze the effect that pre-processing the data has on the results.

IMPORTANT: Regression is basic, so there are many implementations available. But you MUST implement your method yourself. This means that you cannot an embedded function for regression or gradient descent within a software package. You may use other basic functions like matrix math, but the gradient descent and regression algorithm must be implemented by yourself.

Data to be used

We will use the Concrete Compressive Strength dataset in the UCI repository at

UCI Machine Learning Repository: Concrete Compressive Strength Data Set

(https://archive.ics.uci.edu/ml/datasets/Concrete+Compressive+Strength)

Note that the last column of the dataset is the response variable (i.e., y).

There are 1030 instances in this dataset.

Use 900 instances for training and 130 instances for testing, randomly selected. This means that you should learn parameter values for your regression models using the training data, and then use the trained models to predict the testing data's response values without ever training on the testing dataset.

What to submit – follow the instructions here to earn full points

- (80 pts total + 18 bonus points) The report
    - Introduction (15 pts + 6 bonus points)
        - (5 pts) Your description/formulation of the problem (what's the data and what practical application could there be for your work with it, beyond just "this is my homework" or "I want to optimize this equation"),
        - (5 pts) the details of your algorithm (e.g., stopping criterion, is this stochastic gradient descent or not, how you chose your learning rate, etc),
        - (5 pts) pseudo-code of your algorithm (see Canvas for an example)
        - (+3 bonus pts) if you include a description of how your algorithm calculates MAE
        - (+3 bonus pts) if you include a description of how you normalized or standardized your data. Include some histograms that illustrate how the distribution of feature values changed because of your pre-processing.
    - Results (52 pts + 8 bonus points)
        - To report the performance of your models, I would like you to calculate the variance explained (eg. R-squared) for the response variable, which is:
$$1 - \frac{MSE}{Variance(observed)}$$
        - (18 pts) Variance explained of your models on the training dataset when using only one of the predictor variables (uni-variate regression) and when using all eight (multi-variate regression). You should have a total of nine values.
        - (18 pts) Variance explained of your models on the testing data points. Again, you should have a total of nine values.
        - (16 pts) Plots of your trained uni-variate models on top of scatterplots of the training data used (please plot the data using the x-axis for the predictor variable and the y-axis for the response variable).
        - (+4 bonus points) if you include results from using the MAE loss function
        - (+4 bonus points) if you include results from using normalized or standardized feature values as input

- Discussion (13 pts + 4 bonus points)
    - (4 pts) Describe how the different models compared in accuracy on the training data. Did the same models that accurately predicted the training data also accurately predict the testing data?
    - (4 pts) Describe how the different models compared to train/test. Did different models take longer to train? Did you have to use different hyperparameter values for different models?
    - (5 pts) Draw some conclusions about what factors predict concrete compressive strength. What would you recommend for making the hardest possible concrete?
    - (+2 bonus points) if you include comparisons from using MAE
    - (+2 bonus points) if you include comparisons from normalized or standardized data
- **Note:** We won't be grading for good writing practices, but you may have points taken off if you don't write in full sentences and paragraphs, or if you fail to correct spelling and grammar that a simple spell-check tool would alert you of. Bullet point responses to each of these tasks is not sufficient for full credit. Results may be presented as a table, but you must include written descriptions of what can be found in the table, and where.

- (20 pts total + 7 bonus points) Your program (in a language you choose) including:
    - (15 pts) The code itself
    - (5 pts) Brief instructions on how to run your program (input/output plus execution environment and compilation if needed) – in a separate file
    - (+5 bonus points) if you include code for using MAE as the loss function
    - (+2 bonus points) if you include code for normalizing or standardizing the data
    - **Note:** We won't grade your program's code for good coding practices or documentation. However, if we find your code difficult to understand or run, we may ask you to run your program to show it works on a new dataset.

Due date

Wednesday, October 12 (midnight, STL time). Submission to Gradescope via course Canvas.

A one week late extension is available in exchange for a 20% penalty on your final score.

About the extra credit:

The bonus point values listed are the upper limit of extra credit you can earn for each extension. How many points you get will depend on how well you completed each task. Feel free to include partially completed extensions for partial extra credit!

In total, you can earn up to 25 bonus points on this assignment, which means you can get a 125% as your score if you submit it on time, or you can submit this assignment late with the 20% penalty and still get a 100% as your score. It's up to you how you would prefer to manage your time and effort.