CSE514 – Fall 2023 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 gradient descent algorithm or algorithms for regression.

Programming work

A) Data pre-processing

Pre-process the attribute values of your data by normalizing or standardizing each variable. Keep a copy that was not pre-processed, so you can analyze the effect that pre-processing the data has on the optimization.

B) Univariate linear regression

In lecture, we discussed univariate 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 univariate linear regression model.

C) Multivariate 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 multivariate linear regression model using the mean squared error objective function.

IMPORTANT: Regression is basic, so there are many implementations available, but you MUST implement your method yourself. This means that you cannot use an embedded function for regression or gradient descent from 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.

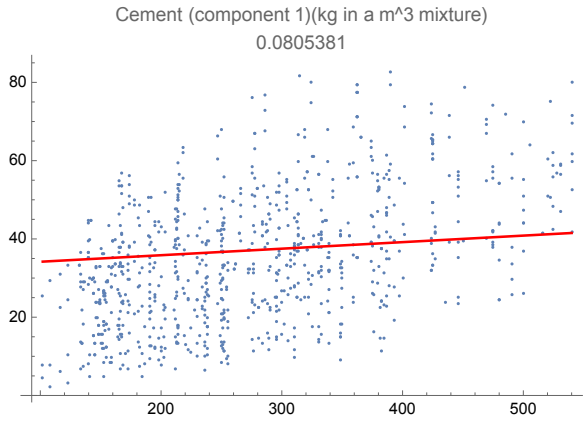
What to submit – follow the instructions here to earn full points

- (80 pts total) The report as a pdf
  - Introduction (15 pts)
    - (4 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"),
    - (3 pts) a description of how you normalized or standardized your data. Include some figures that illustrate how the distribution of feature values changed because of your pre-processing
    - (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),
    - (3 pts) pseudo-code of your algorithm (see Canvas for an example)
  - Results (52 pts)
    - To report the performance of your models, calculate the variance explained (eg. R-squared) for the response variable, which is:

      $$1 - \frac{MSE}{Variance(observed)}$$

      In other words, calculate the average squared error between predicted responses and actual responses (MSE). Then calculate the average squared difference between actual responses and mean actual response (Variance). Divide the former by the latter, then subtract from 1.
    - (26 pts) Variance explained of your models on the training dataset when using only one of the predictor variables (univariate regression) and when using all eight (multivariate regression).
      You should have a total of nine values from optimizing on the raw data, and nine values from optimizing on the pre-processed data.
      At least two of your models optimized on raw data must achieve a positive variance explained on the training data.
      At least two of your models optimized on pre-processed data must achieve a positive variance explained on the training data
    - (10 pts) Variance explained of your models on the testing data.
      You should have a total of nine values from optimizing on the raw data, and nine values from optimizing on the pre-processed data.

- (16 pts) Plots of your univariate 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.
  e.g.



- Discussion (13 pts)
  - (8 pts) Compare and contrast your models.
    - Did the same models that accurately predicted the training data also accurately predict the testing data?
    - Did different models take longer to train or require different hyperparameter values?
    - How did pre-processing change your results or optimization approach?
  - (5 pts) Draw some conclusions about what factors predict concrete compressive strength. What would you recommend for making the hardest possible concrete?

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. Results may be presented as a table, but you must label the rows/columns with enough detail for a reader to interpret it without searching your text, and the figures must be labeled as well.

- (20 pts total) Your program (in a language you choose) including:
  - (15 pts) The code itself
  - (5 pts) Sufficient instructions/documentation on how to run your program (input/output plus execution environment and compilation if needed)

  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 18 (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.

Extra credit opportunities:

Opportunities to submit sub-sections or side-goals of the project will be made available during the weeks leading up to the final submission date. In total, you can earn up to 20 bonus points on this assignment, with a cap of 110% as the maximum score.