BUAN 6341
APPLIED MACHINE LEARNING
ASSIGNMENT 1
Due date: February 13, 11:59 pm

In this assignment, we will be implementing linear and logistic regression on a given dataset. In addition, we will experiment with design and feature choices.

We will be using the SGEMM GPU kernel performance Data Set available for download at https://archive.ics.uci.edu/ml/datasets/SGEMM+GPU+kernel+performance

Goal:
Implement a linear regression model on the dataset to predict the GPU run time. Use the average of four runs as the target variable. You are **not allowed** to use any available implementation of the regression model. You should implement the gradient descent algorithm with batch update (all training examples used at once). Use the sum of squared error normalized by 2*number of samples [$J(\beta_0, \beta_1) = (1/2m)[\sum(y\hat{}(i) - y(i))^2]$ as your cost and error measures, where m is number of samples. You should use all 14 features.

Also implement a logistic regression model as described in Part 4. Again, you are **not allowed** to use any available implementation of the logistic regression model. You should implement the gradient descent algorithm with batch update (all training examples used at once). You should use the logistic regression cost/error function from the class. In addition you can also use accuracy/ROC/etc.

Tasks:
Part 1: Download the dataset and partition it randomly into train and test set using a good train/test split percentage.

Part 2: Design a linear regression model to model the average GPU run time. Include your regression model equation in the report.

Part 3: Implement the gradient descent algorithm with batch update rule. Use the same cost function as in the class (sum of squared error). Report your initial parameter values.

Part 4: Convert this problem into a binary classification problem. The target variable should have two categories. Implement logistic regression to carry out classification on this data set. Report accuracy/error metrics for train and test sets.

Experimentation:

1. Experiment with various parameters for linear and logistic regression (e.g. learning rate $\propto$) and report on your findings as how the error/accuracy varies for train and test sets with varying these parameters. Plot the results. Report the best values of the parameters.

2. Experiment with various thresholds for convergence for linear and logistic regression. Plot error results for train and test sets as a function of threshold and describe how varying the threshold affects error. Pick your best threshold and plot train and test error (in one figure) as a function of number of gradient descent iterations.

3. Pick eight features randomly and retrain your models only on these ten features. Compare train and test error results for the case of using your original set of features (14) and eight random features. Report the ten randomly selected features.

4. Now pick eight features that you think are best suited to predict the output, and retrain your models using these ten features. Compare to the case of using your original set of features and to the random features case. Did your choice of features provide better results than picking random features? Why? Did your choice of features provide better results than using all features? Why?

Deliverables:
You are required to turn in your code and a report. We should be able to run the code as is and get the results and plots that you have included in the report. You should include and describe results for all the experiments above. You should also mention how you constructed the classes for the classification problem (value of threshold and why you picked it). You can be creative and include other plots/results too. However, the report should not exceed 10 pages. Also describe your interpretation of the results. What do you think matters the most for predicting the value and category/class of GPU run time? What other steps you could have taken with regards to modeling to get better results?

Grading:
Total weightage: 12.5% of final grade

Breakdown:

Report: 100 points
If your code doesn't run or doesn't produce the same results then you get zero points.
Points will be awarded based not only on how good your results are, but also on how well you describe them as well as underlying experimentation.

Experiment 1: 20 points
Experiment 2: 20 points
Experiment 3: 20 points
Experiment 4: 20 points

Discussion: 20 points
Describe your interpretation of the results. What do you think matters the most for predicting the GPU run time? What other steps you could have taken with regards to modeling to get better results?