GPU KERNEL PERFORMANCE PREDICTION

OBJECTIVE

In this project, the objectives were to implement a gradient descent algorithm with batch update without using linear or logistic regression model and the goal is to predict the GPU run time. This report details the various experiments conducted using the dataset to understand the effect of tuning hyperparameters of the gradient descent algorithm. Also, the effect of the use of various independent variables/ features in prediction is evaluated and the best model was selected through experimentation.

DATASET DESCRIPTION AND SUMMARY STATISTICS

This data set measures the running time of a matrix-matrix product A*B = C, where all matrices have size 2048 x 2048, using a parameterizable SGEMM GPU kernel with 241600 possible parameter combinations.

A brief description of the data set is as following:

- Dataset consists of 241600 observations with 18 variables.
- For the training dataset, there are 14 parameters where the first 10 are ordinal and can only take up to 4 different powers of two values, and the 4 last variables are binary.
- The dependent variable for the linear regression model is Runs_Average: Average of 4 runs (the last 4 columns in the data set).
- No missing values are found in any column, so no missing value imputation is required.

PROJECT OUTLINE

The project is outlined to have 4 parts:

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

The dataset is downloaded from https://archive.ics.uci.edu/ml/datasets/SGEMM+GPU+kernel+performance
I partitioned the dataset in 70/30 split percentage.

```
x_train, x_test, y_train, y_test = train_test_split (X, Y, test_size=0.3)
```

Part 2,3: Design a linear regression model to model the average GPU run time. Include your regression model equation in the report.

The regression equation to predict the GPU run time is:

Runs_Average =B0 + B1(MWG)+ B2(NWG) + B3(KWG) + B4(MDIMC) +B5(NDIMC)+ B6(MDIMA) + B7(NDIMB) +B8(KWI) +B9(VWM) +B10(VWN) + B11(STRM) + B12(STRN) + B13(SA) +B14(SB)

For Linear Regression:

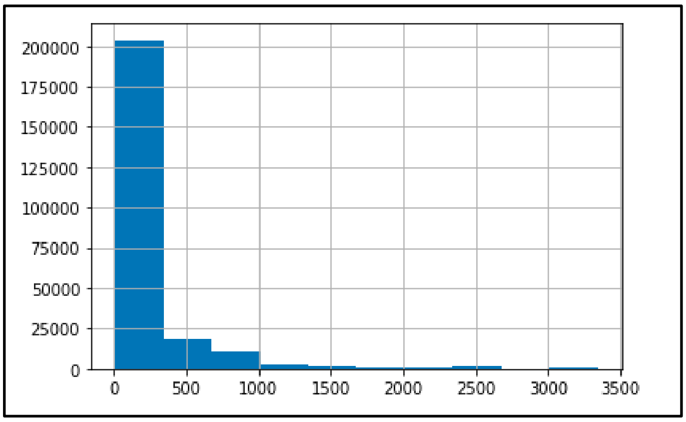
- The initial parameter values from the implementation of the linear regression model to the above model equation is taken as 0.
- The initial cost is 91552.314 for training set and 92683 for testing set with all beta values 0.

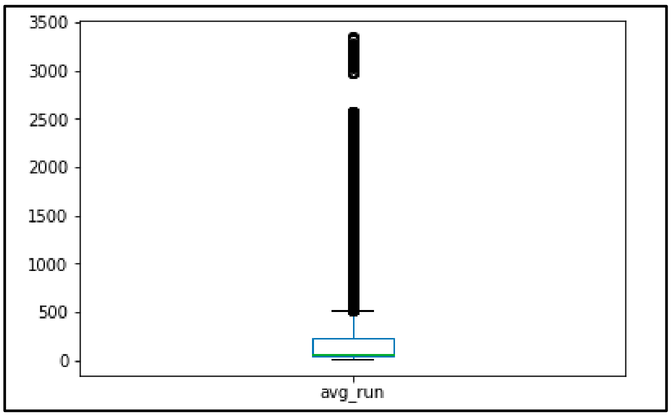
For logistic regression, the initial cost is:

- For training set = 0.693
- For testing set = 0.693

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.

To convert it into binary classification problem, runs_average was plotted as boxplot and histogram to determine the distribution that can help us dividing the variable into two groups.



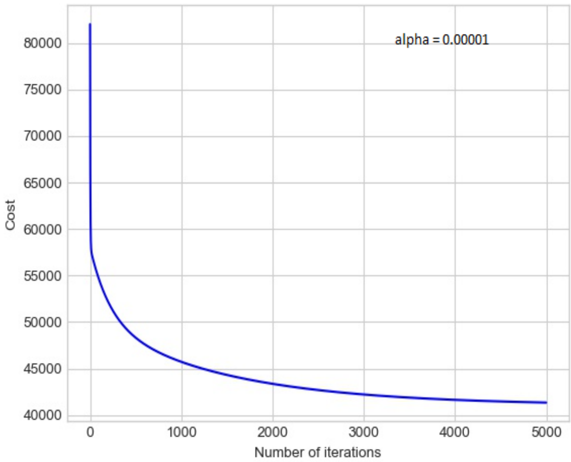


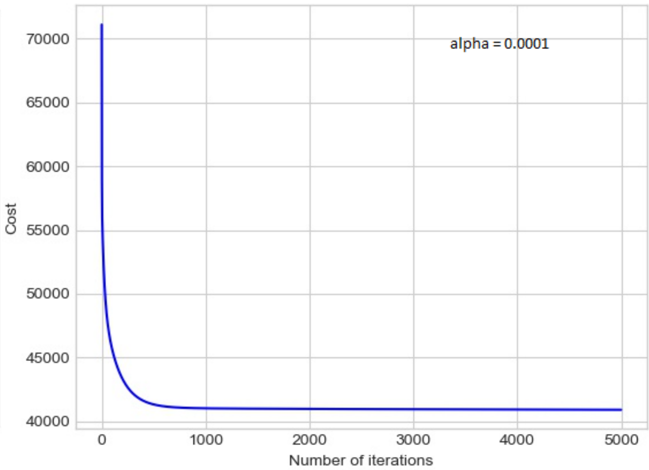
Based on the summary statistics I decided to divide the runs_average into two classes across the median. The summary statistics are displayed below:

Experiment 1: Experimenting with values of alpha

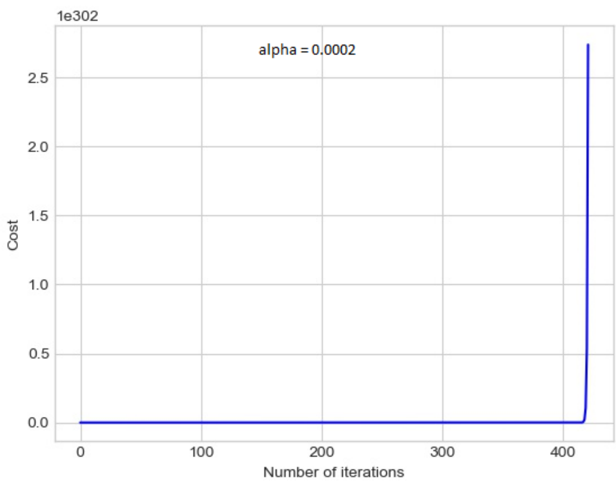
Linear Regression:

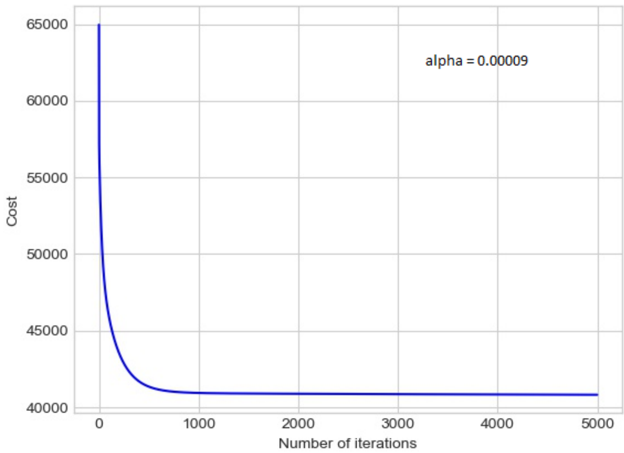
For this experiment, the maximum number of iterations has been set to 5000.
Tried: Alpha = .00001, Alpha = 0.0001, Alpha= 0.0002, Alpha= 0.00009
The variation of cost function with respect to number of iterations for the training set is shown in below figures.





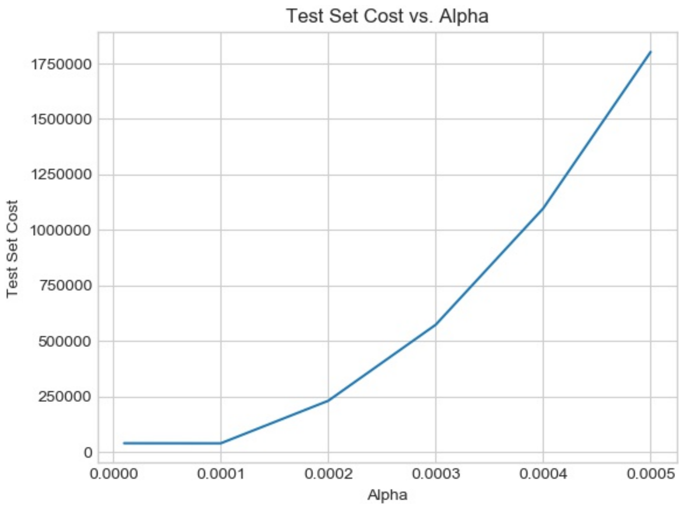
Alpha = 0.0001, Cost = $45109.55





Alpha	0.00001	0.0001	0.0002	0.00009
Iterations	5000	5000	5000	5000
Train	41401.45	40808.11		40817.56
Test	41248.43	40876.51	230144.55	40885.17

The variation of cost as a function of alpha values is shown in the below figure for both train & test set. By performing linear regression, it is found that the best value of alpha is **0.0001** as cost is minimum for both train & test.



Logistic Regression

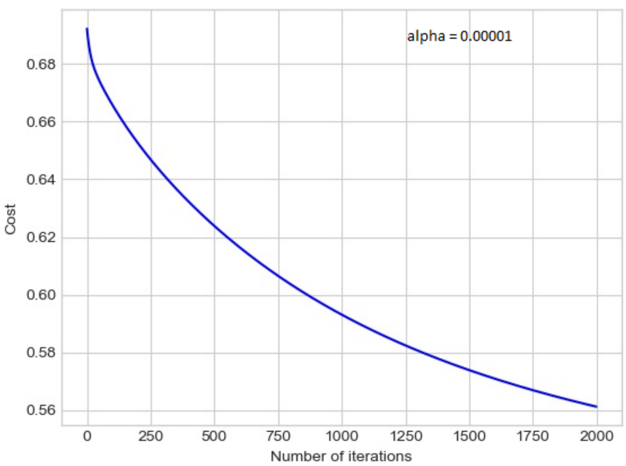
For this experiment, the maximum number of iterations has been set to 2000.

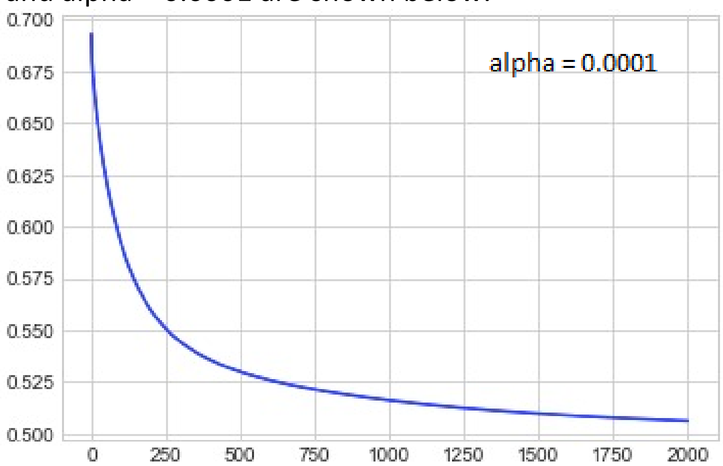
Tried: Alpha = .00001, Alpha = 0.00005, Alpha= 0.0001, Alpha= 0.001

The variation of cost function with respect to number of iterations for the training set is shown in below figures.

Alpha	0.00001	0.00005	0.0001	0.001
Iterations	2000	2000	2000	2000
Train	0.5612	0.5588	0.5075	0.61938
Test	0.5613	0.5591	0.5058	0.6224

The graphs for the training set at alpha = 0.00001 and alpha = 0.0001 are shown below.





It is again evident from the table that minimum cost is attained at alpha = **0.0001**. So, it is seen that for both linear and logistic regression, the minimum cost is obtained at alpha = **0.0001** which is considered as best.

Experiment 2: Experiment with various thresholds for convergence for linear and logistic regression.

The threshold values chosen for this experimentation are 0.1, 0.01, 0.001 at the fixed learning rate of alpha 0.0001. Between the thresholds of 0.001 to 0.1, the threshold of 0.001 made the most significant improvement to the cost function.

Linear Regression:
It is observed that by increasing the number of iterations, the train error and test error is decreasing in linear regression which shows that the model is highly performing when more iterations are tested. The number of iterations tested are 5000 and 10000 and the results of train error and test error are in the range of 40670 to 40870.

Logistic Regression:
By fixing the alpha at 0.0001 and tested on 2000 iterations, train and test errors are decreasing with the decrease in threshold value. The results of train and test error are in the range of 0.506 to 0.690.

Experiment 3: Pick eight features randomly and retrain your models only on these ten features.

By using random number generator, pick 8 random features out of 14 integers. Selected integers are [1, 3, 4, 7, 8, 9, 12] and their corresponding features are [**MWG, KWG, MDIMC, NDIMB, KWI, VWM, STRN**].

The comparison between the train and the test error between model containing 8 random features and the model containing all the features is given below:

Model Type	Train	Test
Iterations	1000	1000
Full Model	41162	39868
Random Feature Model	41824.31	42251.42

The model with 8 random selected features has higher cost for both training and testing set and the output is expected as the features were randomly selected without any statistical evaluation and the odds for randomly selected feature to perform well is very low.

Logistic Regression

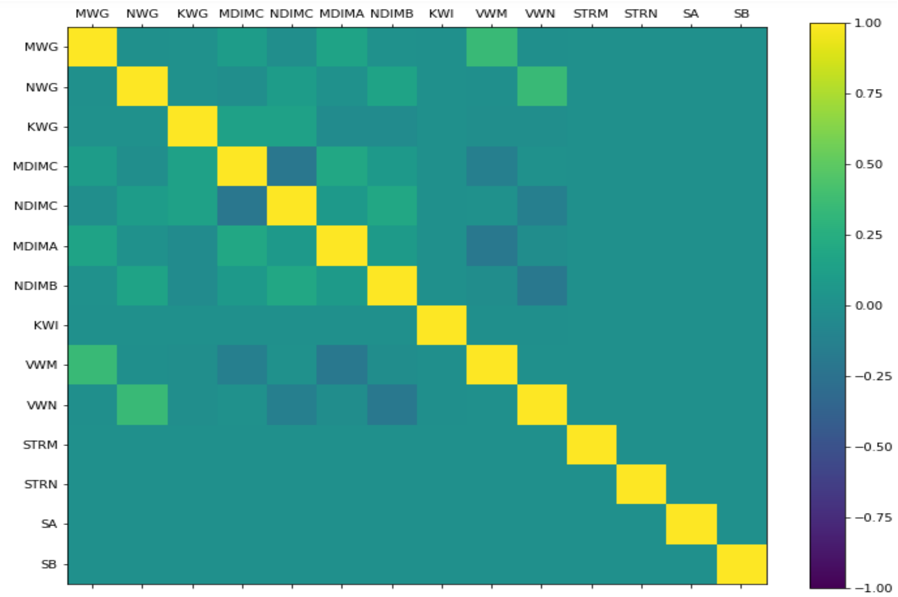
Similarly, for logistic regression, selected features are same i.e. [**MWG, KWG, MDIMC, NDIMB, KWI, VWM, STRN**]. The comparison between the full model and random selected feature model is shown in below table:

Model Type	Train	Test
Iterations	1000	1000
Full Model	0.5091	0.5076
Random Feature Model	0.5177	0.5178

Full model has lower cost value compared to random feature model.

Experiment4:
<u>Linear Regression</u>

From the heat map, the correlation between variables is very low is observed. So, selecting variables based on their correlation cannot be performed.



These are the 8 best features that I found after linear regression: -

	df	Sum of Square	RSS	AIC
SA	1	88746165	1.9568e+10	2730609
KWI	1	34851386	1.9622e+10	2731273
NDIMB	1	29993465	1.9627e+10	2731333
MDIMA	1	25935767	1.9631e+10	2731383
VWN	1	14899757	1.9642e+10	2731519
VWM	1	6300626	1.9650e+10	2731625
STRM	1	5203966	1.9652e+10	2731638
STRN	1	382	1.9657e+10	2731702

The comparison between the train and the test error between model containing 8 random features, 8 best feature results are below:

Model Type	Train	Test
Iterations	1000	1000
Full Model	41162	39868
Random Feature Model	41824.31	42251.42
Best Feature Model	41080	40786

All the features are important, but AIC value and RSS (the less the errors the better is the model) value of SA is lowest which suggests that it is the best feature among all other features.

Logistic Regression

Model Type	Train	Test
Iterations	1000	1000
Full Model	0.5091	0.5076
Random Feature Model	0.5177	0.5178
Best Feature Model	0.5167	0.5170

By looking at the table of linear and logistic regression, it is clear that best feature model did not perform well compared to full feature model. And the results were somewhat expected as all the variables apart from STRN were significant for the model even at alpha = 0.001% and removing significant variables from the model reduces model's ability to explain variations.

Discussions:

- The intercept on y axis is -00.1162
- Variable NWG, KWG, MDIMC, MDIMA, NDIMB, KWI, SA & SB affect the runtime positively, meaning that with a unit increase in these variables the run time average of GPU will also increase.
- Variable MDIMC, NDIMC, VWM, VWN, STRM, STRN affect the runtime negatively, meaning that with a unit increase in these variables the run time average of GPU will also decrease.
- Although all the features are important, but AIC value and RSS (the less the errors the better is the model) value of SA is lowest which suggests that it is the best feature among all other features.
- The dependent variable has positive skewness, which could have been corrected with Log transformation. The model would have given better fit, after the Y variable is log transformed.
- Interaction effects between the independent variables could be explored further, to factor in the joint effect.
- We can also explore the possibility of improving the model by adding polynomial terms.