Contents

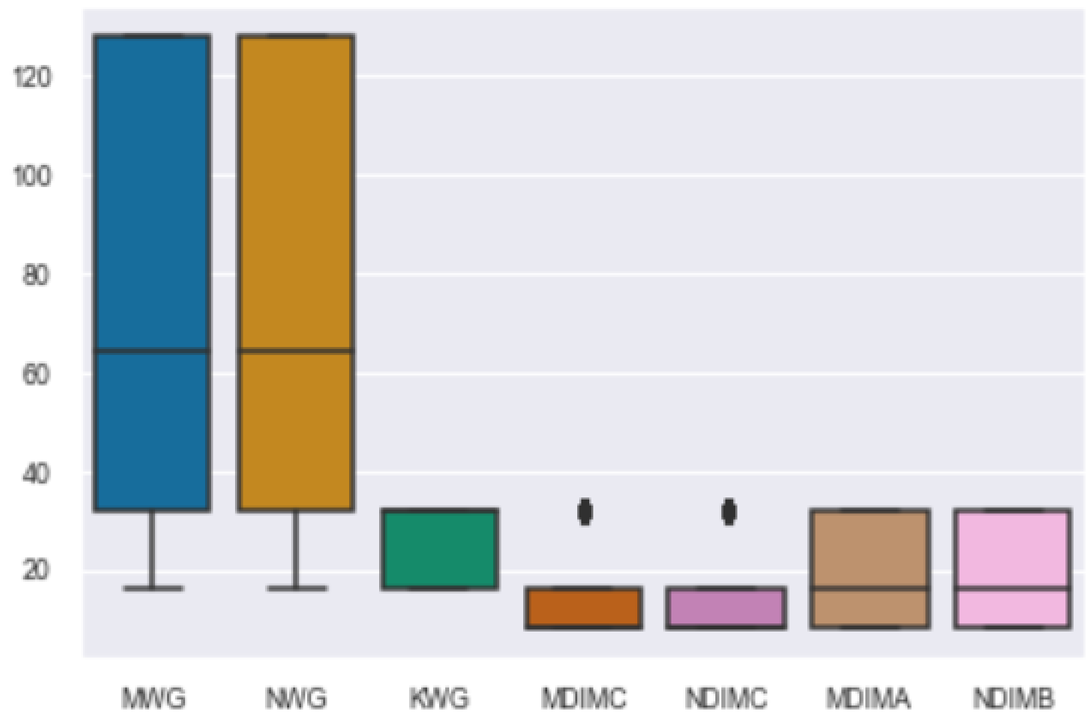
DESCRIPTIVE ANALYSIS 2
CORRELATION MATRIX 3
REGRESSION ANALYSIS 3
LINEAR REGRESSION EQUATION 3
LOGISTIC REGRESSION EQUATION 3
EXPERIMENT-1 4
Linear regression 4
Logistic regression 5
EXPERIMENT-2 6
Linear and Logistic Regression 6
EXPERIMENT-3 7
Linear Regression 7
Logistic Regression 8
EXPERIMENT-4 9
Linear Regression 9
Logistic Regression 10
DISCUSSION 11
NEXT STEPS 11

DESCRIPTIVE ANALYSIS

The dataset measures the running time of a matrix-matrix product with each matrix having size 2048 x 2048, using a parameterizable SGEMM GPU Kernel. Below are the specifications of the data:

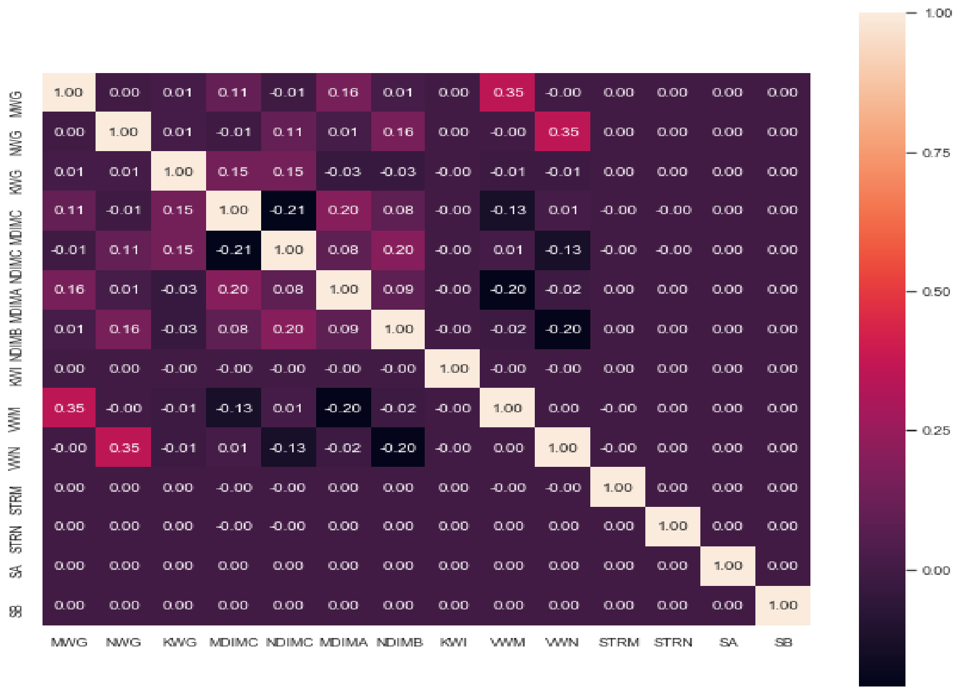
- The dataset is downloaded from UCI Machine learning repository which contains 241,600 rows and 18 attributes.
- The first 10 attributes take powers of two while last four attributes are binary
- There are no null values in the dataset
- The dependent variable () in the dataset is average of four runs in milliseconds
- The dataset is divided into train and test dataset in 70:30 ratio

The following boxplot and table show the unique values and the distribution of the features used in the model:



Features	Unique values	Features	Unique values
MWG	16, 32, 64,128	KWI	2,8
NWG	16, 32, 64,128	VWM	1,2,4,8
KWG	16,32	VWN	1,2,4,8
MDIMC	8,16,32	STRM	1,0
NDIMC	8,16,32	STRN	1,0
MDIMA	8,16,32	SA	1,0
NDIMB	8,16,32	SB	1,0

CORRELATION MATRIX



	MWG	NWG	KWG	MDIMC	NDIMC	MDIMA	NDIMB	KWI	VWM	VWN	STRM	STRN	SA	SB
MWG	1.00	0.00	0.01	0.11	-0.01	0.16	0.01	0.00	0.35	-0.00	0.00	0.00	0.00	0.00
NWG	0.00	1.00	0.01	-0.01	0.11	0.01	0.16	0.00	-0.00	0.35	0.00	0.00	0.00	0.00
KWG	0.01	0.01	1.00	0.15	0.15	-0.03	-0.03	-0.00	-0.01	-0.01	0.00	0.00	0.00	0.00
MDIMC	0.11	-0.01	0.15	1.00	-0.21	0.20	0.08	-0.00	-0.13	0.01	-0.00	-0.00	0.00	0.00
NDIMC	-0.01	0.11	0.15	-0.21	1.00	0.08	0.20	-0.00	0.01	-0.13	-0.00	-0.00	0.00	0.00
MDIMA	0.16	0.01	-0.03	0.20	0.08	1.00	0.09	-0.00	-0.20	-0.02	0.00	0.00	0.00	0.00
NDIMB	0.01	0.16	-0.03	0.08	0.20	0.09	1.00	-0.00	-0.02	-0.20	0.00	0.00	0.00	0.00
KWI	0.00	0.00	-0.00	-0.00	-0.00	-0.00	-0.00	1.00	-0.00	-0.00	0.00	0.00	0.00	0.00
VWM	0.35	-0.00	-0.01	-0.13	0.01	-0.20	-0.02	-0.00	1.00	0.00	-0.00	0.00	0.00	0.00
VWN	-0.00	0.35	-0.01	0.01	-0.13	-0.02	-0.20	-0.00	0.00	1.00	-0.00	0.00	0.00	0.00
STRM	0.00	0.00	0.00	-0.00	-0.00	0.00	0.00	0.00	-0.00	-0.00	1.00	0.00	0.00	0.00
STRN	0.00	0.00	0.00	-0.00	-0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00
SA	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00
SB	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00

- The correlation table on the left do not show any significant correlation between features
- We check the correlation between features to avoid the problem of multicollinearity which could create a bias in the model we performed on the dataset

REGRESSION ANALYSIS

- The features and dependent variable were normalized to cope up with the skewness of the data.
- The data was then split into train and test dataset with 70:30 ratio
- We can also do 75:25 or 80:20 ratio but this dataset is huge enough and training with 75% of the data might result in overfitting
- The aim of this assignment is to create the gradient descent function and model data b tuning the hyperparameters i.e. learning rate and number of iterations
- All 14 features were used to predict the GPU run time with various learning rate and iterations

LINEAR REGRESSION EQUATION

Run= 0.0006+ 0.38* MWG+ 0.35* NWG+0.11* KWG-0.35* MDIMC-0.35* NDIMC-0.028* MDIMA+0.027* NDIMB+0.034* KWI-0.006* VWM-0.015* VWN-0.011* STRM+0.0007* STRN+0.052* SA+0.064* SB

Where, Run= Average run time units

LOGISTIC REGRESSION EQUATION

Run= 0.05+1.32* MWG + 0.84* NWG + 0.15* KWG -0.90* MDIMC -0.76* NDIMC -0.05* MDIMA -0.06*NDIMB -0.01* KWI -0.08* VWM -0.14* VWN-0.32* STRM-0.02* STRN+0.40* SA-0.079* SB

Where, Run= 0, if average run is lesser than median average run

Run=1, if average run is higher than median average run

EXPERIMENT-1

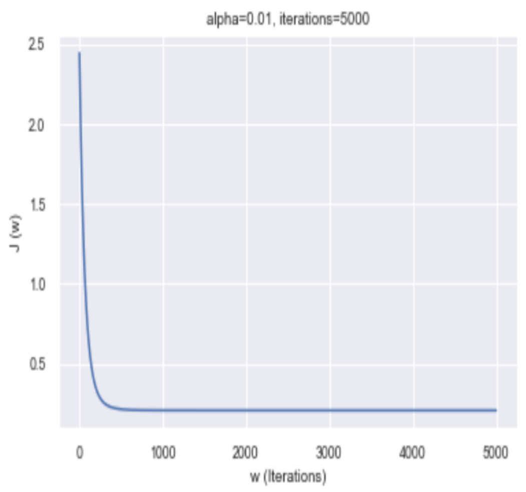
Linear regression

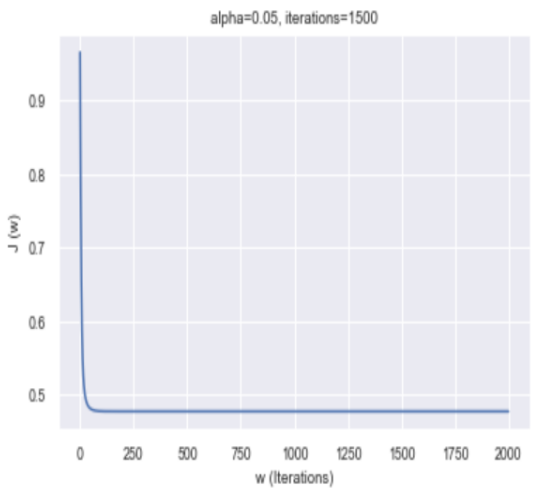
- The data was modelled to predict average GPU run time using all the 14 features.
- The data was normalized with various trials by tuning the hyperparameters i.e. learning rate (alpha) and iterations.
- Increasing the alpha leads to take bigger steps towards convergence but sometimes you can miss the minima and deflect.
- Lesser number of iterations will never let you reach the minima if the alpha is very small.
- Hence, the aim is to find the balance in the hyperparameters to make the process faster and efficient.

The table below compares the model statistics with respect to various hyperparameters:

Hyperparameters				Train		Test	
Learning rate	Iterations	Convergence point	R-square	RMSE	MAE	RMSE	MAE
0.01	3000	0.20746873	0.40779295	0.76991415	0.47887596	0.76943332	0.47821136
0.01	5000	0.20746873	0.40779295	0.76991415	0.47887865	0.76943332	0.47821404
0.01	2000	0.20746877	0.40779284	0.76991422	0.47880539	0.76943295	0.47814090
0.05	2000	0.20746873	0.40779295	0.76991415	0.47887865	0.76943332	0.47821404
0.05	1500	0.20746873	0.40779295	0.76991415	0.47887865	0.76943332	0.47821404
0.03	1500	0.20746873	0.40779295	0.76991415	0.47887584	0.76943340	0.47821124

Different cost plots for some of the models are as follows:





<u>Inferences:</u>

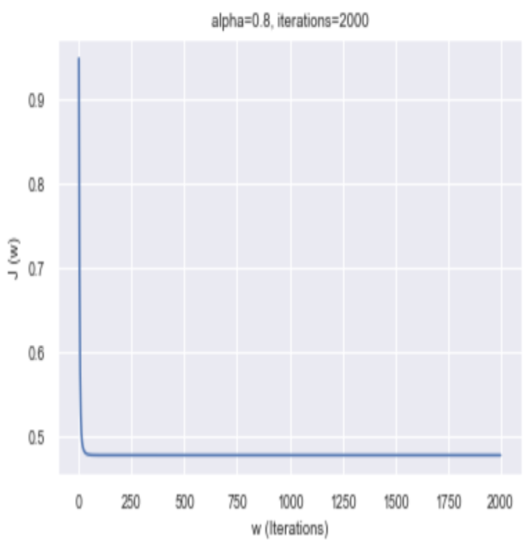
- The best hyperparameters were chosen based on beta values close to the inbuilt function and the error metrics in the above table.
- The best learning rate= 0.05 and iterations=1500 which is highlighted in yellow in the table.

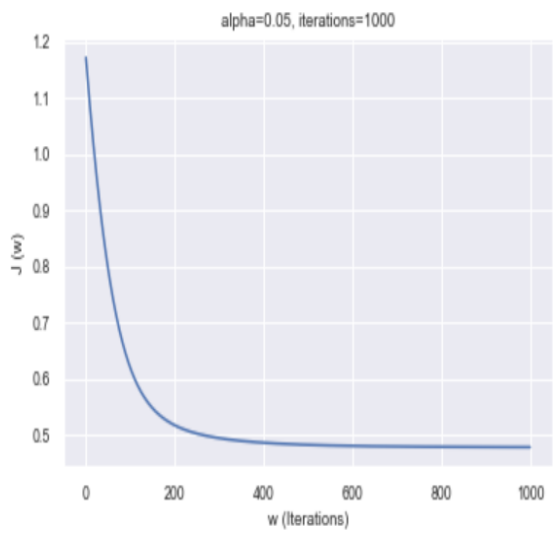
Logistic regression

- The data was modelled to predict whether the GPU run time is less than median (class=0) or greater than median (class= 1)
- The median was chosen over the mean because a few outliers can shift the mean which might affect the model predictions

The table below compares the model statistics with respect to various hyperparameters:

Hyperparameters			Accuracy	
Learning rate	Iterations	Convergence	Train	Test
0.05	1000	0.477963460	0.809277042	0.812334437
0.05	2000	0.477677540	0.809602649	0.812317881
0.1	2000	0.477676060	0.809569536	0.812251656
0.2	2000	0.477676060	0.809580574	0.812251656
0.5	2000	0.477676060	0.809580574	0.812251656
0.8	2000	0.477676060	0.809580574	0.812251656





Inferences:

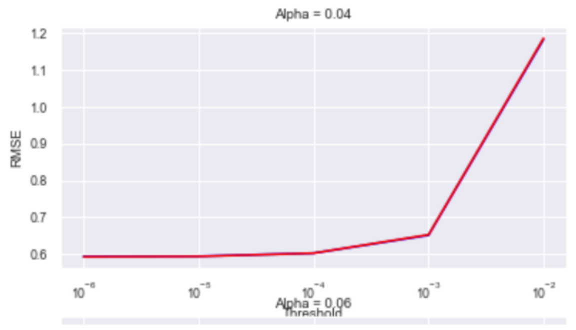
- The accuracy is similar for various pairs of hyperparameters as it differs only after fourth decimal point
- The key is to find the best accuracy with largest learning rate, lowest iterations and lowest convergence point
- Hence, the best learning rate=0.8 and iterations=2000 was decided based on convergence and accuracy which is highlighted in yellow in the table

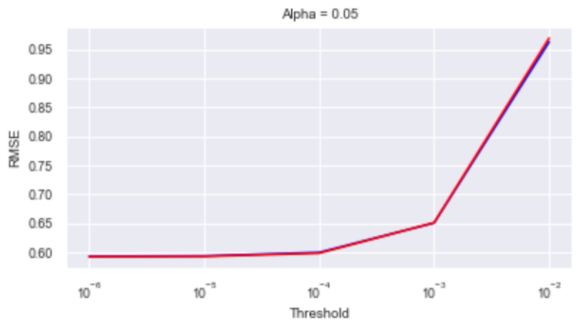
EXPERIMENT-2

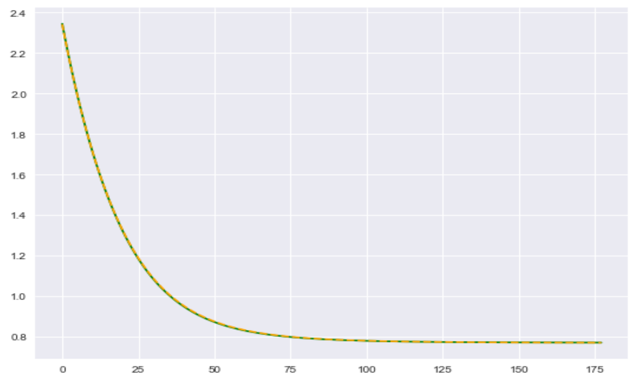
Linear and Logistic Regression

- In gradient descent, each step is determined by the value of learning rate
- If the learning rate is bigger, it will reach the minima earlier.
- After it reaches the minima, the slope decreases, and cost value converges or change in cost value is lesser than the threshold
- While the cost function converges, the variable coefficients are also approximately equal

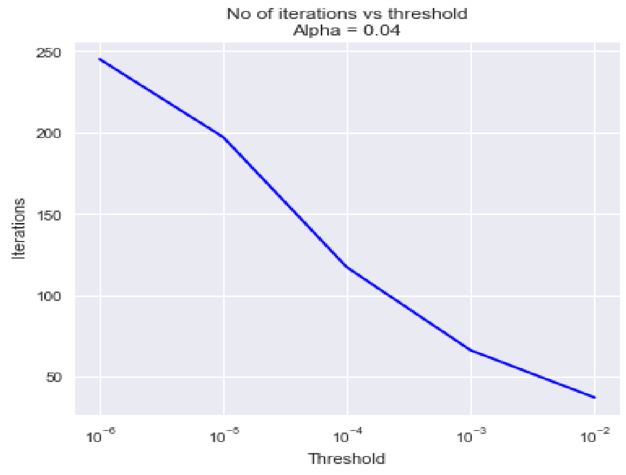
Hence, root mean square value of the train and test data also converges which is plotted below with respect to various thresholds i.e. 10^-2, 10^-3, 10^-4, 10^-5, 10^-6. The difference is very small between the train and test which can be seen by the overlapping blue and red lines.







- •The plot on the left is cost Vs iterations for learning rate =0.05 where the cost converges at 0.6 after 174 iterations
- •The number of iterations used were 3000
- •We can infer that if we increase the learning rate, the number of iterations it will take to converge will be lesser



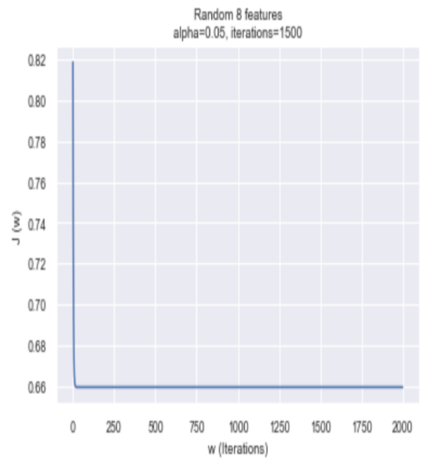
- • The plot on the left explains the number of iterations needed to reach the threshold for learning rate=0.04
- • We can infer that lower the threshold value between consecutive cost, higher is the number of iterations

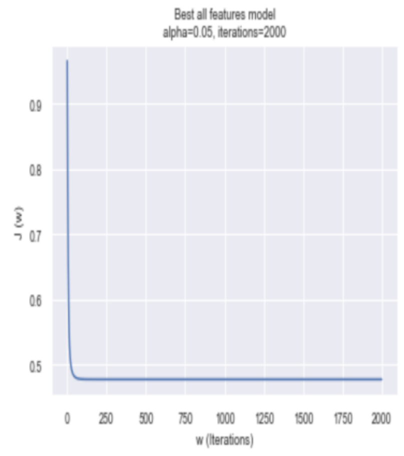
EXPERIMENT-3

Linear Regression

- The random features selected were NDIMB, KWI, VWM, VWN, STRM, STRN, SA, SB
- The model was run for the chosen hyperparameters i.e. alpha=0.05 and iterations=1500

Below is the comparison for the two models:





- The cost function plots do not say anything about the quality of the model
- The convergence point is slightly different which we can see in the table below

Comparing the error metrics and % variation explained by features:

Hyperparameters Alpha=0.05, Iterations=1500			Train		Test	
	Convergence point	R-square	RMSE	MAE	RMSE	MAE
All 14 Features	0.20746873	0.407793	0.769914	0.478879	0.769433	0.478214
Random 8 Features	0.33045385	0.056739	0.971676	0.568328	0.970610	0.568810

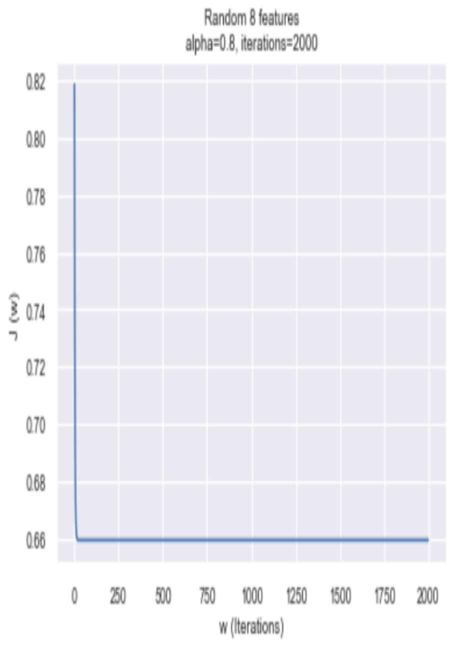
Inferences:

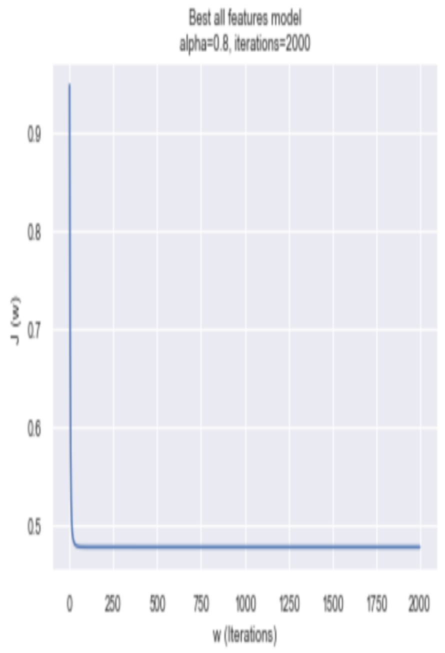
- When we compare the two models, we can see the random feature model do not explain much variation in the dependent variable (GPU run time) with R-square 0.056
- Also, the Root mean square error and Mean Absolute error are very high as compared to model with all 14 features
- The model with all 14 features explains more variability of the data and decreases error
- Also, the cost plot for both the models looks similar but the convergence point is lower in model with all features
- The lower the convergence point higher is the precision of the coefficients in model equation as it reaches very close to the minima
- The model with all 14 features is better as expected because it explains maximum variation in the dependent variable whereas random model has random features which were picked without any statistical inferences

Logistic Regression

- The random features selected were NDIMB, KWI, VWM, VWN, STRM, STRN, SA, SB
- The model was run for the chosen hyperparameters i.e. alpha=0.8 and iterations=2000

Below is the comparison for the two models:





- The cost function plots do not say anything about the quality of the model
- The convergence point is different which we can see in the table below

Comparing the accuracy and convergence point for the models:

Hyperparameters Alpha=0.8, Iterations=2000		Accuracy	
	Convergence	Train	Test
All 14 Features	0.47767606	0.809581	0.812252
Random 8 Features	0.65948038	0.610241	0.611562

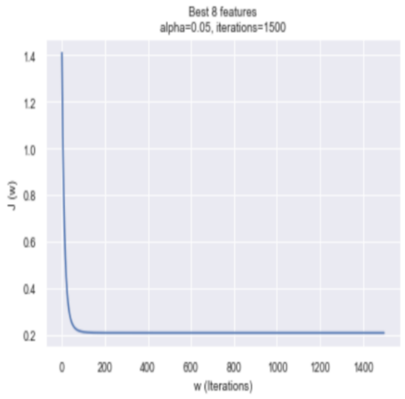
Inferences:

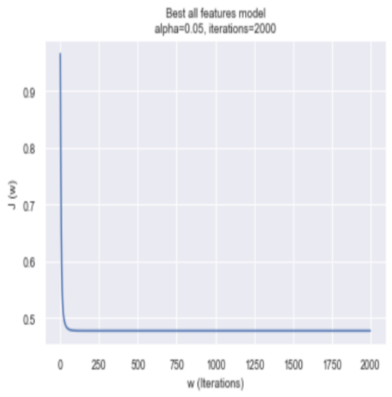
- The accuracy for random features model takes a hit as the variables which explain the variation to classify dependent variable are restricted
- The best model is the model which contains all 14 features with accuracy 80.95% and 81.22% as compared to random model with accuracy 61.02% and 61.15%. The accuracies are for train and test models respectively
- Usually, the test accuracy is lower than the train dataset accuracy, but sometimes due to sampling we can see small difference. As the difference is not that significant, it is considered a good model

EXPERIMENT-4

Linear Regression

- The best 8 features were selected based on the p-values and variables coefficients which aren't close to zero
- The p-value from the in-built regression implied that all the variables were significant except STRN with p-value 0.77
- Hence, coefficients close to 0 were dropped to find the best 8 features
- The best 8 features are MWG, NWG, KWG, MDIMC, NDIMC, MDIMA, NDIMB, VWM
- The model was run for best hyperparameters i.e. alpha=0.05 and iterations=1500 and the comparison is as follow:





- The cost function plots do not say anything about the quality of the model
- The convergence point is almost equal which we can see in the table below

Comparing the error metrics and % variation explained by features:

Hyperparameters Alpha=0.05, Iterations=1500				Train		Test	
	Convergence point	R-square	Adjusted R	RMSE	MAE	RMSE	MAE
All 14 Features	0.20746873	0.407793	0.407740418	0.769914	0.478879	0.769433	0.478214
Best 8 Features	0.21042274	0.399361	0.399328945	0.775376	0.47647	0.774399	0.474964

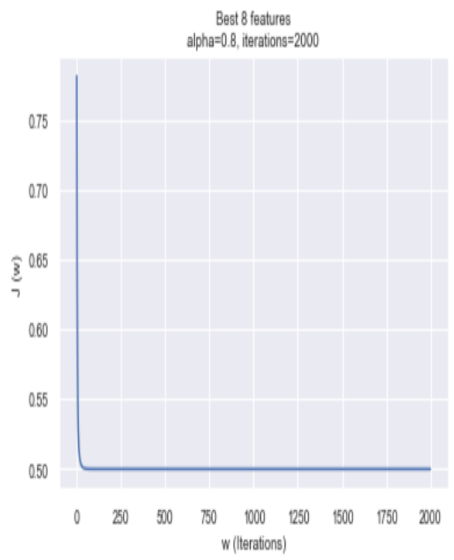
Inferences:

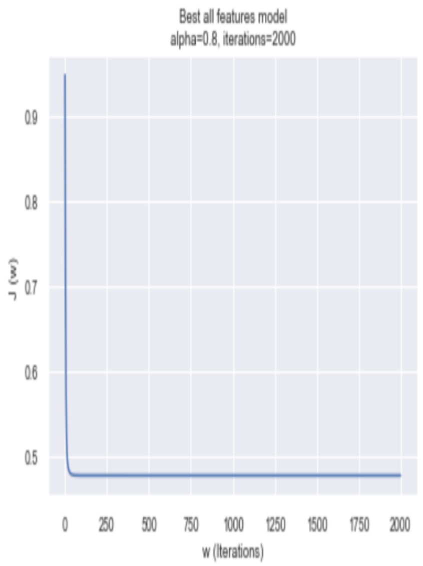
- The model with best 8 features has similar mean absolute error than the model with all 14 features
- The RMSE is almost similar in both the models
- The R-square of model with 14 features is slightly higher than the other model but Adjusted R-square has decreased a little
- This is because there is a penalty added with each feature being added in the model while we calculate adjusted R-square. The penalty is **(n-1/n-k-1)** where **n= number of rows and k=number of features**
- Also, the model with best 8 features has slightly higher convergence than the other model
- Analyzing all the above metrics, both the models are equally good but the model with lesser features is always recommended

Logistic Regression

- The 8 best features selected for linear regression will be same for the logistic model as well
- The model was run for the best hyperparameters for logistic model i.e. alpha=0.8 and iterations=2000

Below is the comparison between model with 8 best features and all 14 features:





•The cost function plots do not say anything about the quality of the model
•The convergence point is almost equal which we can see in the table below

Comparing the accuracy and convergence point for the models:

Hyperparameters Alpha=0.8, Iterations=2000	Convergence	Accuracy Train	Accuracy Test
All 14 Features	0.47767606	0.809581	0.812251656
Best 8 Features	0.49984318	0.82207	0.823311258

Inferences:

- The model with 8 features has higher accuracy than the other model
- Certainly, the best model is the one with higher accuracy and lesser features i.e. the model with best 8 features
- The convergence point for both the models is different but the difference is minute

DISCUSSION

The following are the key takeaways from the exercise:

1. Tuning the hyperparameters

   Learning rate: Increasing the alpha helps us reach the local minimum in shorter run time but it may deflect sometimes and never reach the minima. However, if we take very small learning rate it may never reach the minima if the number of iterations is small. Hence, the aim is to find the balance in the hyperparameters to achieve maximum accuracy in minimum time

   Threshold: Higher threshold may not let the cost function to converge at the minima and give spurious results and result in poor accuracy. However, very smaller threshold might just increase the computation time and not affect the accuracy at all. Hence, the aim is to find highest threshold to achieve maximum accuracy in the least time possible

2. Statistical Inference

   It is recommended that we pick variables which have high statistical inferences like t-statistics, p-value and F-statistics of the model

NEXT STEPS

- The domain knowledge of 2-D tiling can help us pick features based on their significance which can be explored
- Exploring more features which explain more variation in the dependent variable would help us get better results