Dataset: Seoul Bike Sharing Demand Data Set

Currently Rental bikes are introduced in many urban cities for the enhancement of mobility comfort. It is important to make the rental bike available and accessible to the public at the right time as it lessens the waiting time. Eventually, providing the city with a stable supply of rental bikes becomes a major concern. The crucial part is the prediction of bike count required at each hour for the stable supply of rental bikes.
The dataset contains weather information (Temperature, Humidity, Windspeed, Visibility, Dewpoint, Solar radiation, Snowfall, Rainfall), the number of bikes rented per hour and date information.

Attribute Information:

Date - year-month-day
Rented Bike count - Count of bikes rented at each hour
Hour - Hour of the day
Temperature-Temperature in Celsius
Humidity - %
Windspeed - m/s
Visibility - 10m
Dew point temperature - Celsius
Solar radiation - MJ/m2
Rainfall - mm
Snowfall - cm
Seasons - Winter, Spring, Summer, Autumn
Holiday - Holiday/No holiday
Functional Day - NoFunc(Non Functional Hours), Fun(Functional hours)

Objective:
The goal of this assignment is to implement a regression model on the dataset to predict the rented bike count using gradient descent algorithm with batch update. Later the problem is converted into logistic regression model with 2 classes and is implemented using gradient descent algorithm.

Data Pre-Processing:

- **For Linear Regression Data**

  1) Column Date has been removed as it has no significant role in the regression process.
  2) Columns with categorical values have been converted into dummy variables. Unique classes that are in the data are given below:

```
Unique Classes in Seasons column: ['Winter' 'Spring' 'Summer' 'Autumn']
Unique Classes in Holiday column: ['No Holiday' 'Holiday']
Unique Classes in Functioning Day column: ['Yes' 'No']
```

  3) The dummies are named as Seasons_Spring, Seasons_Summer, Seasons_Autumn, Holiday_Dummy, FunctioningDay and the data corresponding to them is intuitive to understand.
  4) There are no null values in the data.

- **For Logistic Regression Data**
  1) All the columns are same as above expect for one additional column DayLight. The column Hour has been removed as it is highly correlated with DayLight variable. Explanation is given in the future sections.

Tasks:

Part 1:

The dataset has been partitioned into train and test set with 80:20 ratio which I thought would be ideal for this analysis

Part 2:

All the variables in the data set have been chosen for the linear regression analysis. There are total 14 independent variables and 1 dependent variable.

Below is the regression equation:

Rented Bike Count = β_0 + β_1**Hour* + β_2**Temperature* + β_3**Humidity* + β_4**Wind Speed* + β_5**Visibility* + β_6**Dew point Temperature* + β_7**Solar Radiation* + β_8**Rainfall*+ β_9**Snowfall*+ β_{10}** Seasons_Spring* + β_{11}** Seasons_Summer* + β_{12}** Seasons_Autumn* + β_{13}**Holiday_Dummy*+ β_{14}**FunctioningDay*

Part 3:

Gradient descent algorithm with batch update rule has been implemented to find the beta parameter values in linear regression. The update rule is as given below:

$j = 0, 1, \ldots, n$

Simultaneous update

$$\beta_j := \beta_j - \alpha \frac{\partial}{\partial \beta_j} J(\beta)$$

$$\beta_j := \beta_j - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(\hat{y}^{(i)} - y^{(i)}\right) x_j^{(i)}$$

The Gradient Descent Cost Function is defined as:

$$J(\beta) = \frac{1}{2m} \sum_{i=1}^{m} \left(\hat{y}^{(i)} - y^{(i)}\right)^2$$

The initial parameters has been chosen randomly by a function in python called numpy.random.sample() with a seed value set to 10 and the parameters are:

Parameter	β_0	β_1	β_2	β_3	β_4	β_5	β_6	β_7	β_8	β_9	β_{10}	β_{11}	β_{12}	β_{13}	β_{14}
Value	7.71	0.21	6.34	7.49	4.99	2.25	1.98	7.61	1.69	0.88	6.85	9.53	0.04	5.12	8.13

Part 4:

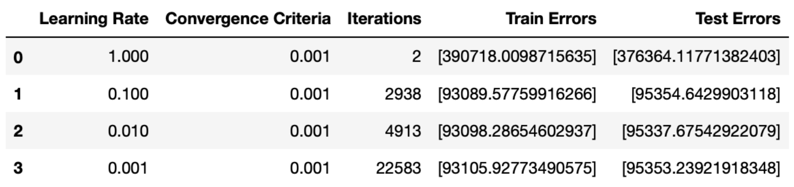
The problem has been converted to binary classification by taking the Hour column and converting it into a different column named DayLight where in all the hours from 6 am – 6pm have been marked 1 meaning there is presence of day light. From 6pm – 6am, the values in DayLight have been marked as 0, meaning there is no presence of day light in Seoul. This form of classification will help one to classify whether the bikes were rented during the day or the night. Column Hour has been deleted from the original data to get rid of multicollinearity problem.

<u>Experimentation:</u>

1. **Experimenting with different learning rates**

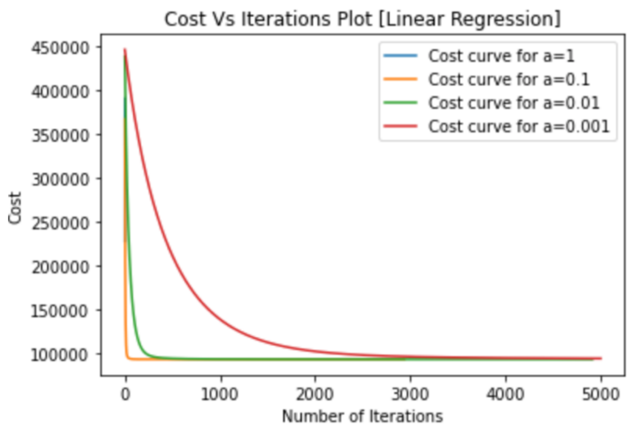
i) **Linear Regression**

Varying Learning Rate with constant convergence threshold



	Learning Rate	Convergence Criteria	Iterations	Train Errors	Test Errors
0	1.000	0.001	2	[390718.0098715635]	[376364.11771382403]
1	0.100	0.001	2938	[93089.57759916266]	[95354.6429903118]
2	0.010	0.001	4913	[93098.28654602937]	[95337.67542922079]
3	0.001	0.001	22583	[93105.92773490575]	[95353.23921918348]

You can see that with a constant Convergence threshold as the learning rate decreases, number of iterations increases. Also, if you look at the training and test errors, you can see that train errors are always less than the test errors. This is bound to happen because we are training the model on the train data and the parameters that are calculated from train data are used to calculate the test errors.



Going by the plot and the train errors we can observe that learning rate of 0.1 is doing a great job by reaching to the minimum squared error in lesser number of iterations compared to others and the training error is the least. But I would like to choose **learning rate of 0.01** as the best learning rate because it produces the least test error (see from table above). For predictive analysis, it is better to go with a parameter producing less test error since, train errors are pretty much in the similar levels and computation speed isn't our primary concern.

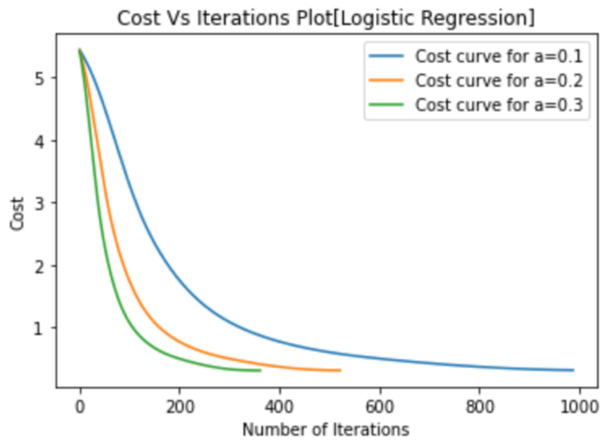
Therefore, the best learning rate for logistic regression for this data is 0.01

ii) Logistic Regression

Varying Learning Rate with constant convergence threshold

	Learning Rate	Convergence Threshold	Iterations	Train Errors	Test Errors
0	0.1	0.0001	988	0.315596	[0.3044293855706163]
1	0.2	0.0001	522	0.311633	[0.3013100373024061]
2	0.3	0.0001	362	0.309948	[0.300169029552984]

We can see that as the learning rate is increases, the train and test errors decrease along with number of iterations.



From the above plot we can clearly see that **learning rate of 0.3** converges to the minimum error faster than other learning rates.

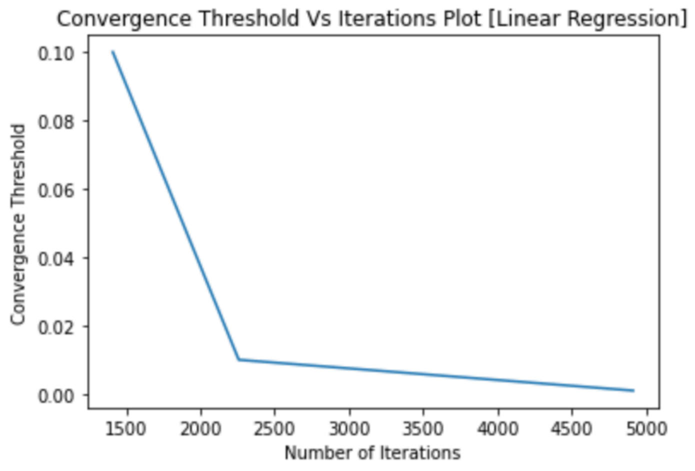
Therefore, the best learning rate for logistic regression for this data is 0.3

2. Experimenting with different convergence thresholds

i) Linear Regression

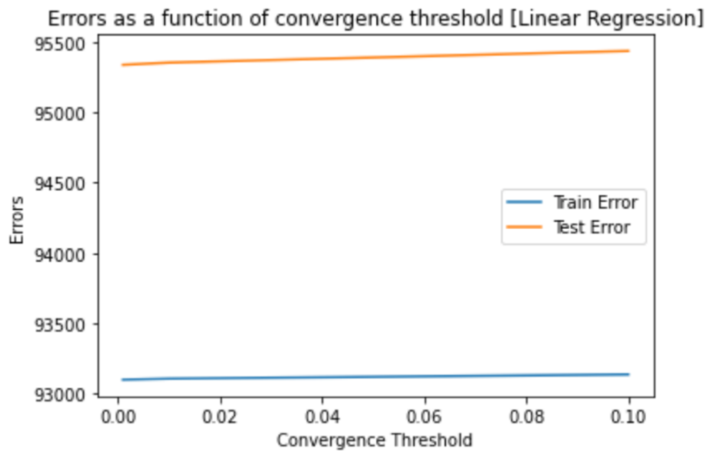
Varying Convergence threshold with Constant learning rate

	Learning Rate	Convergence Threshold	Iterations	Train Errors	Test Errors
0	0.01	0.100	1410	[93135.57196602893]	[95436.43371995896]
1	0.01	0.010	2258	[93105.92112689611]	[95353.19022505969]
2	0.01	0.001	4913	[93098.28654602937]	[95337.67542922078]



From the above table we can see that as we decrease the convergence threshold, the train error also decreases. Therefore, the **convergence threshold of 0.001** is a good metric to fix because it produces least train error and also almost least test error.

We can observe that at a constant learning rate, if we decrease the convergence threshold, the number of iterations increases. The plot below will give us a better idea.



If we plot the test and train errors as a function of convergence threshold, we can observe that errors increase with the increase in the convergence threshold. Also, the test error is always higher than that of the train errors.

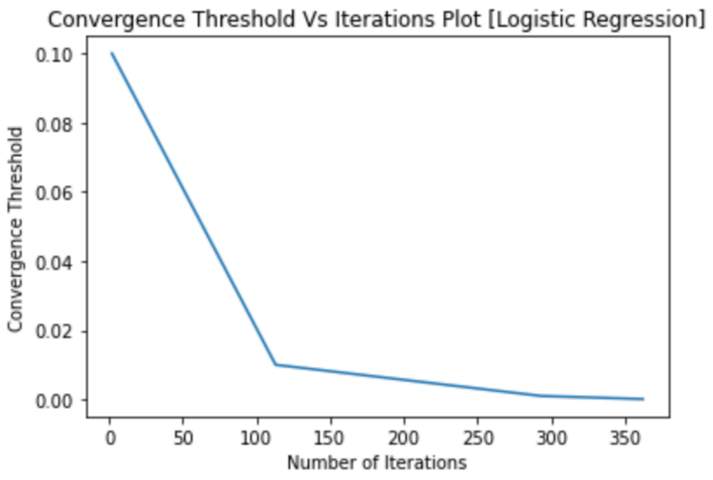
In conclusion, the best convergence threshold is 0.001 for this model.

ii) Logistic Regression

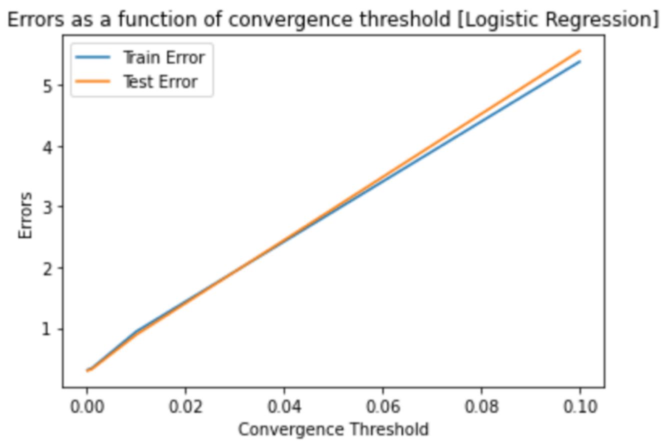
Varying Convergence threshold with Constant learning rate

	Learning Rate	Convergence Threshold	Iterations	Train Errors	Test Errors
0	0.3	0.1000	2	5.376862	[5.552838521408217]
1	0.3	0.0100	113	0.940512	[0.8876389133516024]
2	0.3	0.0010	293	0.337898	[0.32425823337794996]
3	0.3	0.0001	362	0.309948	[0.300169029552984]

We can observe that as the convergence threshold decreases the train errors and the test errors also decrease. You can also see that the number of iterations required to converge increases. You can better understand the effect of convergence threshold on iterations from below graph



When we plot the train and test errors as the function of convergence threshold, we can clearly observe that at lower convergence thresholds both the test and train errors are similar whereas at higher levels the test error is higher than that of the train errors.

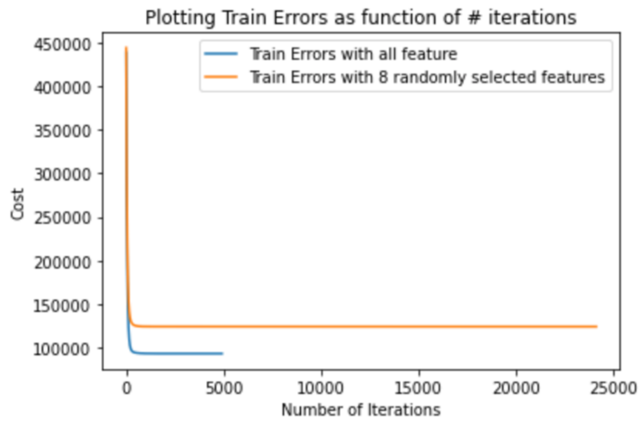


From the above information convergence threshold of 0.0001 is the best metric to be considered in this model

3. **Experimenting with 8 randomly picked features and retraining the models**

Randomly picked models are Temperature, Humidity, Dew point temperature, solar radiation, rainfall, snowfall, Seasons_Autumn, Holiday_Dummy

i) **Linear Regression Model**



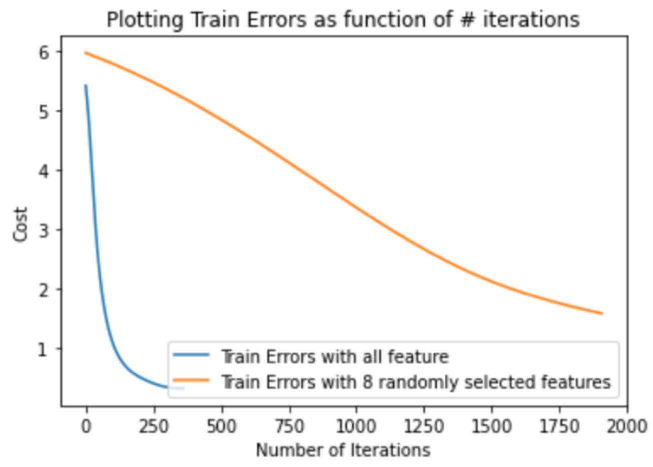
- It is observed that the model with 8 randomly picked model takes more iterations than the model with all the features
- Also, the training error for all feature is less than that of the randomly picked 8 features. Probably we have picked totally irrelevant features.
- Both the models converge almost at the same number of iterations

Summary

Model	Iterations	Train Error	Test Error
All Feature	4913	93098.29	95337.67
8 Random Features	24117	124011.77	128190.05

In linear regression the model with all features is performing way better than the random features model.

ii) **Logistic Regression Model**



- It is observed that the train error in all features model has converged faster than that of the 8 random features model.
- Random feature model takes more iterations to converge compared to all features model with a threshold of 0.0001.
- It looks like the train error of all features model is lesser than that of the random features model.

Summary

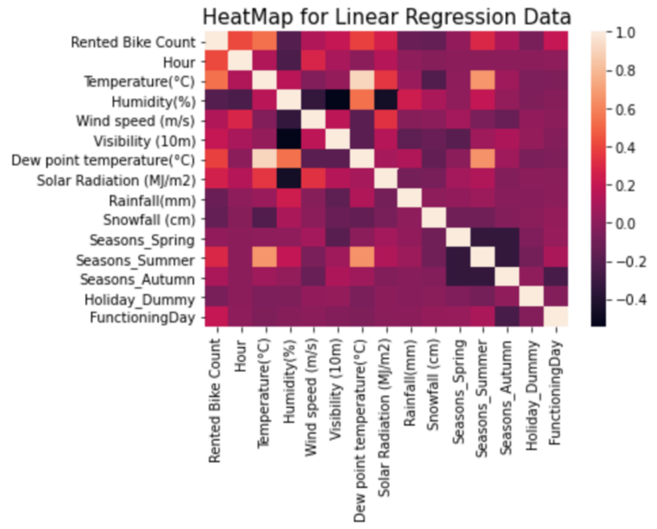
Model	Iterations	Train Error	Test Error
All Feature	362	0.3099	1.575
8 Random Features	1907	0.3	1.513

Clearly, all features model is performing better than the random features models.

4. **Experimenting with 8 best suited features and retraining the models**

i) Linear Regression Model

Best 8 Features:



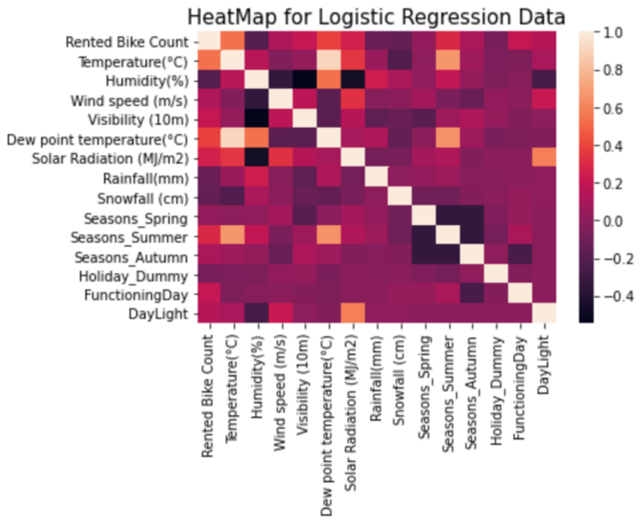
Based on the above heatmap we can have a brief idea about the important features based on their correlation. Based on the heat map and intuition, I would like to choose features: **Hour, Temperature, Humidity, Dew point temperature, Solar radiation, Rainfall, Snowfall, Seasons_Summer**

Model	Iterations	Train Error	Test Error
All Feature	4913	93098.29	95337.67
8 Random Features	24117	124011.77	128190.05
Best Features	2186	108978.49	113610.097

From the above table we can observe that all features model has the least train and test errors. For some reason, the number of iterations is very high in randomly selected features and the errors are high too. However, the best features model has least number of iterations and slightly higher errors than all features.

ii) Logistic Regression Model

Best 8 Features:



From the heatmap and intuition, I would like to choose 8 best features as **Bike count, temperature, Humidity, Wind Speed, Visibility, Dew Point Temperature, Solar Radiation, Functioning day**

Summary

Model	Iterations	Train Error	Test Error
All Feature	362	0.3099	0.3
8 Random Features	1907	1.575	1.513
Best Features	1779	0.825	0.829

From the above table we can observe that all features model has the least train and test errors followed by best features model and random features model. The number of iterations the model takes to converge is also in the same order as specified earlier. Train and test errors are the least in the all features model and highest in the random features model.

<u>Interpretations and Summary</u>

- We can clearly see that learning rate is very critical in this model as a lower learning rate can result in the number of iterations to increase to a large extent. Hence finding the correct learning rate and threshold value can work a long way in reducing the training and test error rate as much as possible.
- Experiment 1 proves us that in both the models as the learning rate increases, it takes lesser time to converge to the minimum cost function.
- Experiment 2 tells us the importance of convergence threshold, the lower the threshold is set, the greater number of iterations it is going to take.
- Perfect combination of both learning rate and convergence threshold will give us fruitful results. There were situations where the test error in logistic and linear regression models were lesser than train error. Although, it happened just by chance. In reality, the train errors are smaller than test errors. This has been observed through out my experimentations.
- Experiment 3 tells us, how important is it to select right features for the model. From my results we can see that in case of linear regression model with random models, the model took insane amount of iterations to converge to minimum cost function. Even, in logistic model it was the similar case but not as bad as linear model.
- Also as seen from the experiment 4, choosing the most relevant variables is the most important aspect of a model. Feature selection must be done so that it reduces the bias – variance tradeoff and does not cause underfitting or overfitting. The model also must not be made complex by adding too many features.
- In both linear and logistic models training and test errors in best features model are certainly lower than that of the randomly selected features, but still is not as good as the original model with all features. It performs better than the random model because in this model, the variables are carefully selected with common sense knowledge and the correlation plot, hence the error is a bit lower.
- In general, the errors are in the following fashion: 8 Random features model> Best 8 features Model > All features model. We can expect this trend because all the features give more information rather than randomly selected features.
- I would say that choosing the correct learning rate and feature selection are the most important aspects of prediction.
- In conclusion, the important factors to predict the rented bikes is the learning rate and convergence threshold of gradient descent algorithm. On top of that choosing the right features for the model makes a huge difference. Factors such as external weather conditions are used to predict better than the seasons in the data.

<u>Steps to be taken to improve the model</u>

- By looking at the nature of the data, it looks like a non-linear regression would've predicted better than the linear model. Probably, a polynomial model would have worked better than linear model.
- Also, we haven't dealt with the outliers yet. If that was done, we could've ensured better results.
- Cross validation can be performed to further fine tuning the model before testing the model with the test data so that the mean squared error can be decreased further.
- For feature selection, various techniques such as forward, backward and hybrid methods can be used to select the most important parameters for the model.