BIKE RENTAL PREDICTION REPORT

Created By: Prateek Rawat

Date: 05/19/2019

Contents of Report:

Link to Code:

Prediction Model :- https://tinyurl.com/y4vfstcz

Data Pipeline :- https://tinyurl.com/y3sedvfy

PROBLEM STATEMENTS

For this report three problem statements are given as the following statements :-

1. Build a Prediction Model to predict total count of bikes rented in an hour and coding assumptions.
2. Talk about the scaling properties of the Model built in statement 1.
3. Build a Data Pipeline using Python to ingest data from a webpage.

PROBLEM 1

Data Description :

This dataset is about Bike-Sharing rental process which is highly correlated to the environment and seasonal settings. Our aim is to predict the total count(cnt) of Bike booked per hour based on these characteristics.

The core data set is related to the two-year historical log corresponding to years 2011 and 2012 from Capital Bikeshare system, Washington D.C., USA which is publicly available at http://capitalbikeshare.com/system-data.

Dataset Consist of the following fields

- instant: record index
- dteday : date
- season : season (1:springer, 2:summer, 3:fall, 4:winter)
- yr : year (0: 2011, 1:2012)
- mnth : month (1 to 12)
- hr : hour (0 to 23)
- holiday : weather day is holiday or not (extracted from http://dchr.dc.gov/page/holiday-schedule)
- weekday : day of the week
- workingday : if day is neither weekend nor holiday is 1, otherwise is 0.
- weathersit : (weather Condition)
  1: Clear, Few clouds, Partly cloudy, Partly cloudy
  2: Mist + Cloudy, Mist + Broken clouds, Mist + Few clouds, Mist
  3: Light Snow, Light Rain + Thunderstorm + Scattered clouds, Light Rain + Scattered clouds
  4: Heavy Rain + Ice Pallets + Thunderstorm + Mist, Snow + Fog
- temp : Normalized temperature in Celsius. The values are divided to 41 (max)
- atemp: Normalized feeling temperature in Celsius. The values are divided to 50 (max)
- hum: Normalized humidity. The values are divided to 100 (max)
- windspeed: Normalized wind speed. The values are divided to 67 (max)
- casual: count of casual users
- registered: count of registered users
- cnt: count of total rental bikes including both casual and registered

Exploratory Data Analysis :

1. Data consist of both categorical and numerical variables.
2. Variable **‘cnt’** is our dependent variable which has to be predicted after building model.
3. Data Profiling of Data.
4. Checking of duplicate values in the Data.
5. Pearson and Spearman Correlation graphs plotted to check correlation between variables.
6. Distribution, kurtosis, and skewness of variables using Graphs.
7. Graphs to check correlation of categorical variables with target variable.
8. Considering important variables.

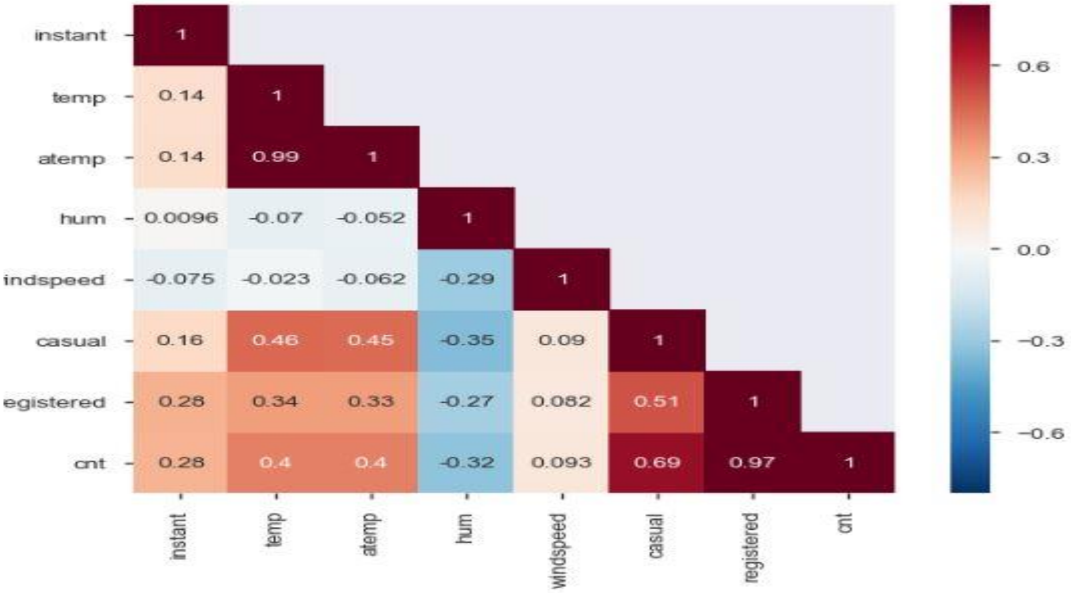
Actions Performed :

1. Data type of categorical variables changed from int/float/object to categorical.
2. Target variable **‘cnt’** is continuous numerical variable. Hence, we can build a Regression model.
3. After data profiling no missing values where found. However, some of the variables such as ‘cnt’ , ‘casual’, ‘new’ were rightly skewed. So, we performed log transformation for these variables.
4. No duplicate values where found.
5. From data profiling and correlation matrix it can be found that variable ‘temp’ and ‘atemp’ have high positive correlation.
6. Plots of categorical variables with target variable has shown a significant relationship.
7. Dropping variables ‘instant’, ’dteday’, ‘causal’, ’registered’, ’atemp’.
8. As there are categorical variables therefore apply one-hot encoding for those variables before running them in prediction model.

Visualization :

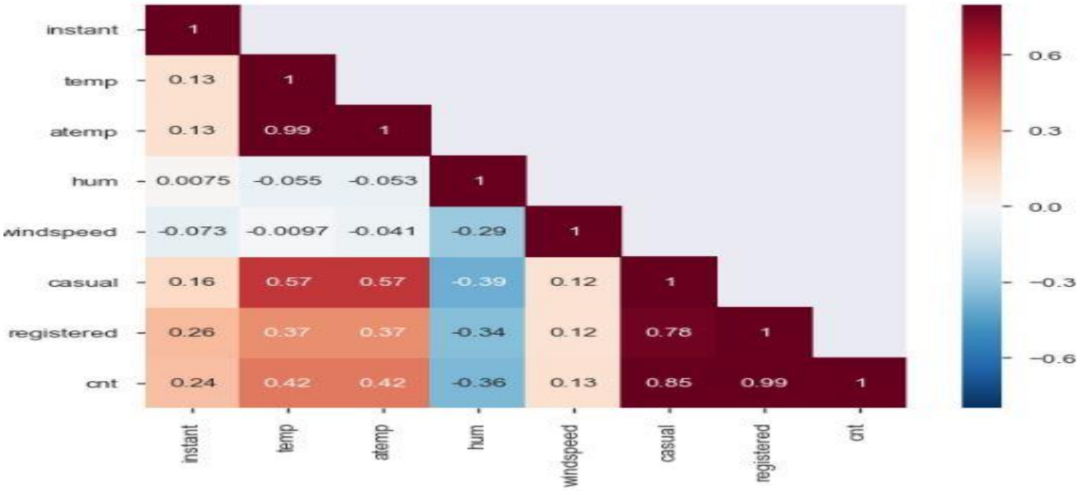
Pearson Correlation Graph:-

The Pearson correlation evaluates the linear relationship between two continuous variables. A relationship is linear when a change in one variable is associated with a proportional change in the other variable.



Spearman Correlation Graph:-

The Spearman correlation evaluates the monotonic relationship between two continuous or ordinal variables.

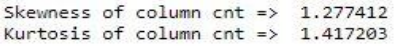


Though **casual** and **registered** variable has a strong correlation with target variable but we will not consider these variables in model because **casual** and **registered** are just count of bike which gives a total count of bikes i.e. **cnt**. So, we need further analysis to whether keep these variables or not with the help of business team. However, I have dropped them for this analysis.

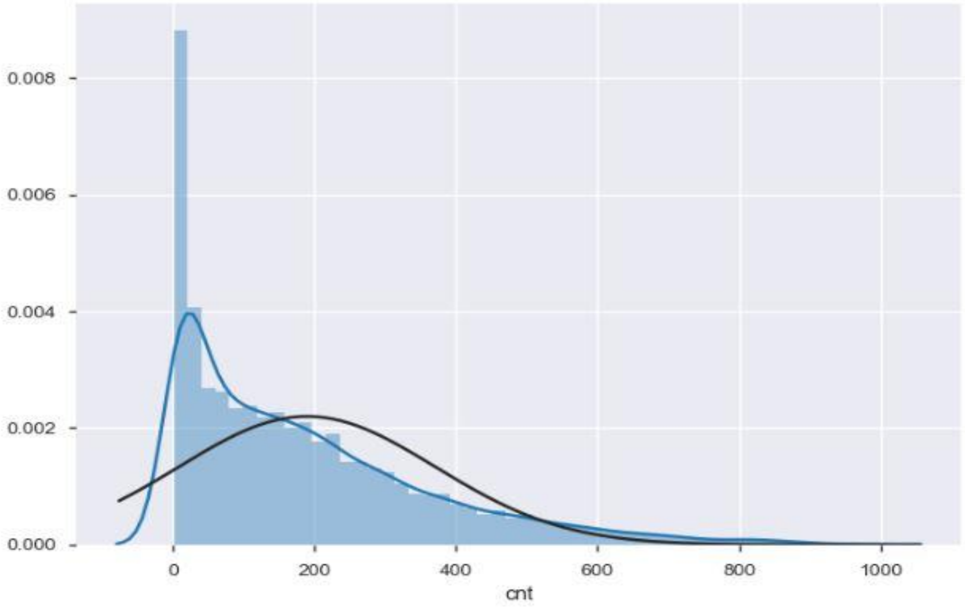
Skewness and Kurtosis:-

- Skewness is the degree of distortion from the symmetrical bell curve or the normal distribu tion so, if the skewness is between -0.5 and 0.5, the data are fairly symmetrical.
- If the skewness is between -1 and -0.5(negatively skewed) or between 0.5 and 1(positively skewed), the data are moderately skewed
- If the skewness is less than -1(negatively skewed) or greater than 1(positively skewed), the data are highly skewed.
- Kurtosis is all about the tails of the distribution — not the flatness. It is used to describe the extreme values in one versus the other tail. It is actually the measure of outliers present in the distribution
- (Kurtosis < 3): Distribution is shorter, tails are thinner than the normal distribution. The pea k is lower and broader than Mesokurtic, which means that data are light-tailed or lack of o utliers.

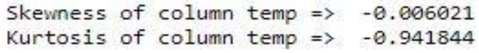
1. Distribution of variable 'cnt'.



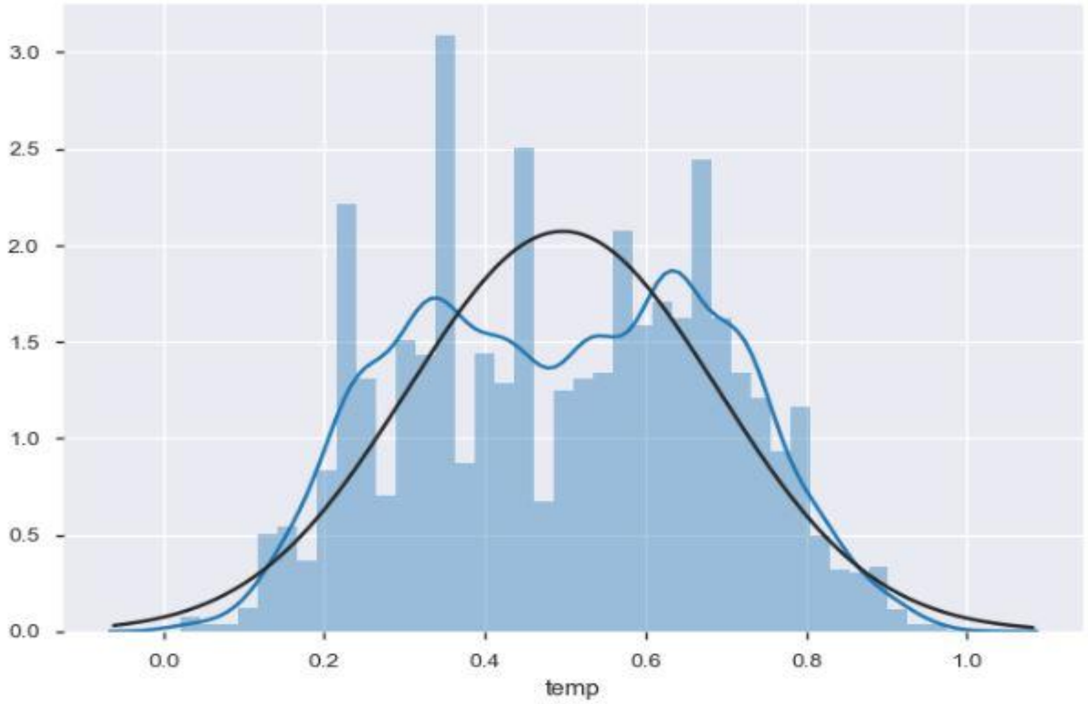
```
Skewness of column cnt =>  1.277412
Kurtosis of column cnt =>  1.417203
```



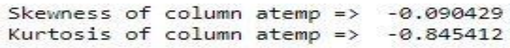
2. Distribution of ‘temp’ .



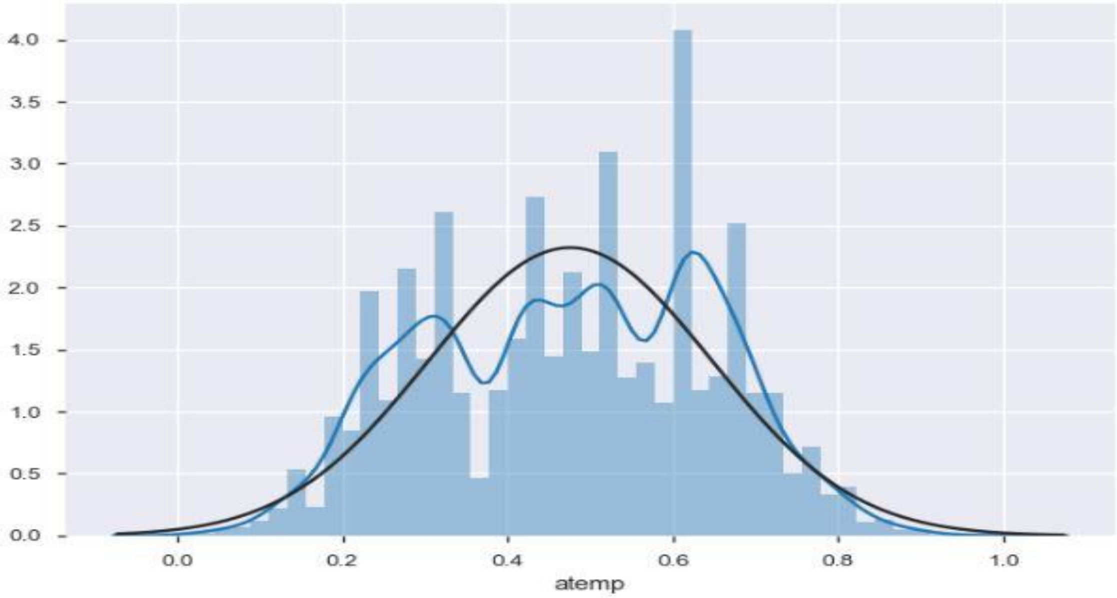
```
Skewness of column temp =>  -0.006021
Kurtosis of column temp =>  -0.941844
```



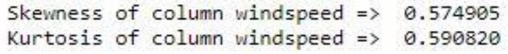
3. Distribution of ‘atemp’:



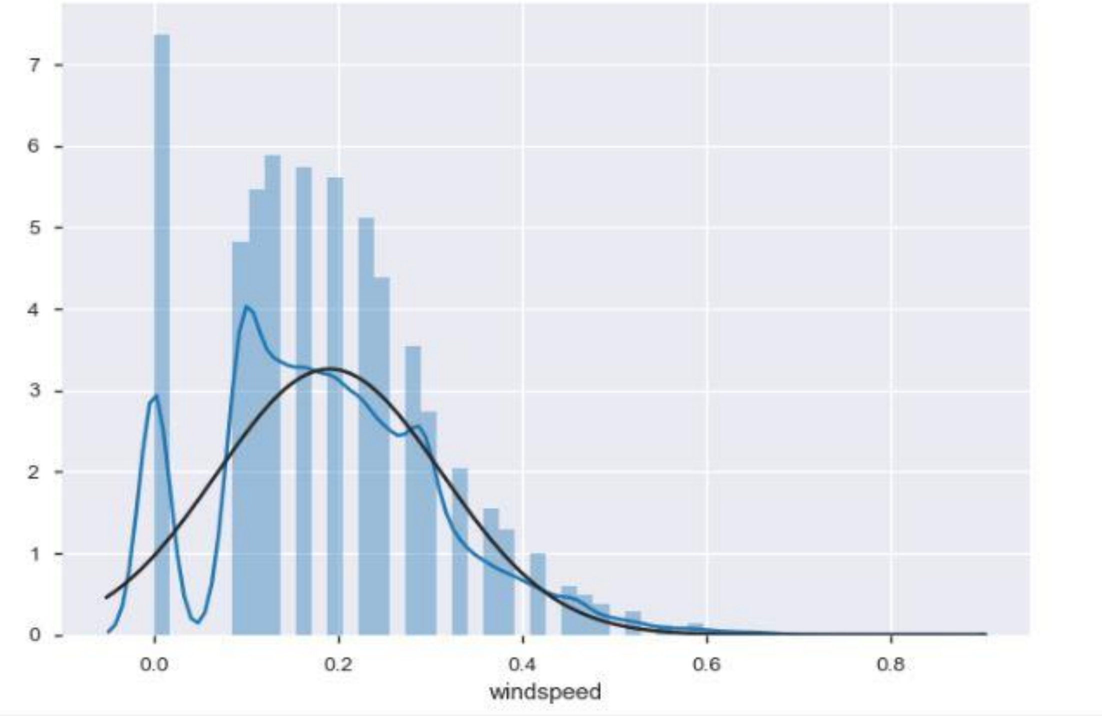
```
Skewness of column atemp =>  -0.090429
Kurtosis of column atemp =>  -0.845412
```



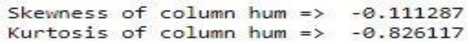
4. Distribution of ‘windspeed’



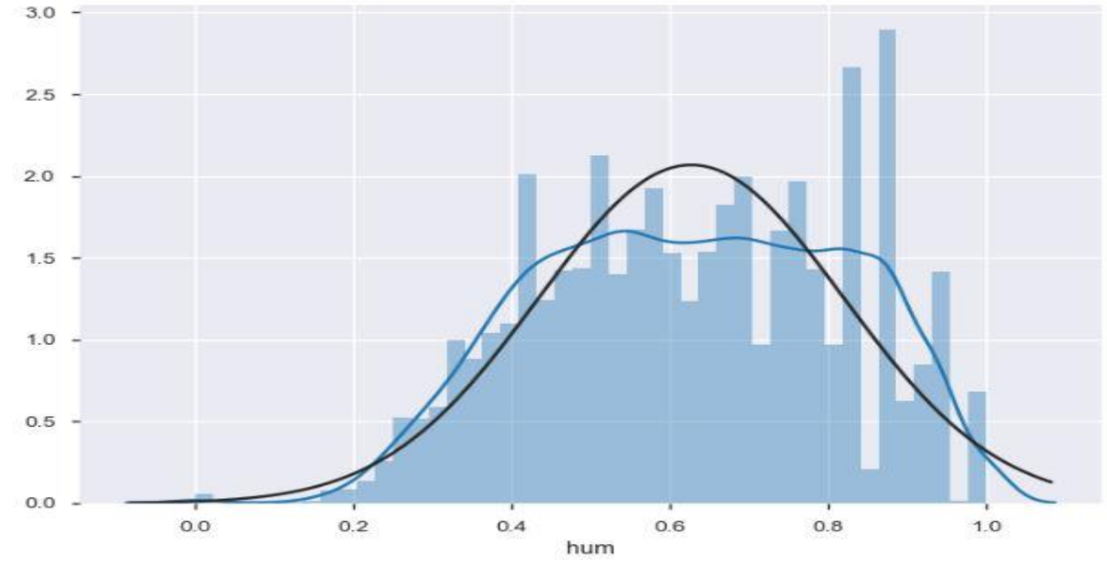
```
Skewness of column windspeed =>  0.574905
Kurtosis of column windspeed =>  0.590820
```



5. Distribution of ‘hum’:



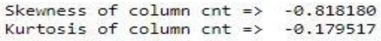
```
Skewness of column hum =>  -0.111287
Kurtosis of column hum =>  -0.826117
```



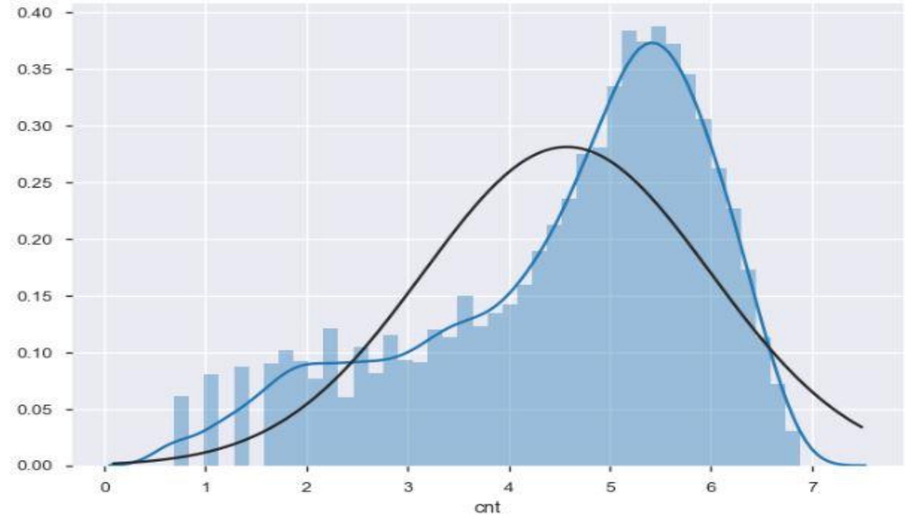
From the plots for Skewness and Kurtosis we found out that **‘cnt’** is rightly skewed and we need to log transform it. Rest all variables are almost normally distributed.

Based on correlation matrix we found out that ‘temp’ and ‘atemp’ are positively correlated. So, remove variable ‘atemp’ based on the finding that ‘temp’ has better skewness than ‘atemp’.

6. Log Transformed ‘cnt’:



```
Skewness of column cnt =>  -0.818180
Kurtosis of column cnt =>  -0.179517
```



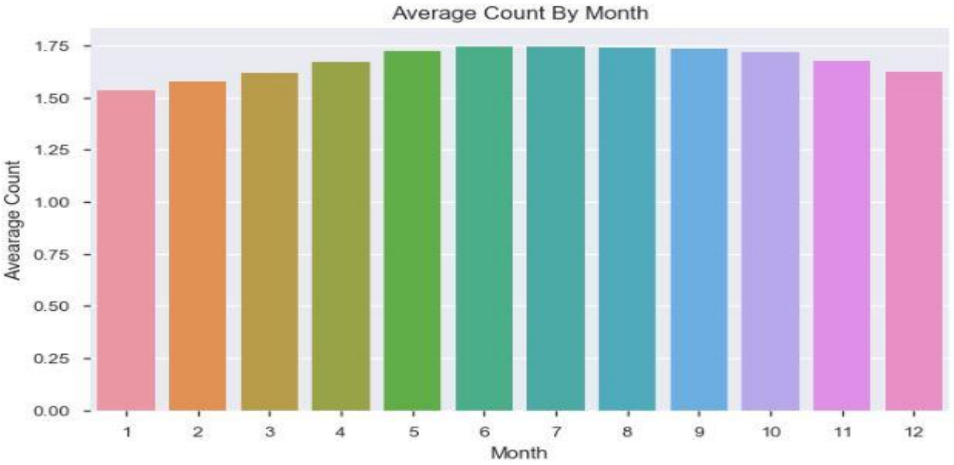
Now the distribution of ‘cnt’ looks better as compared to non-log transformed data.

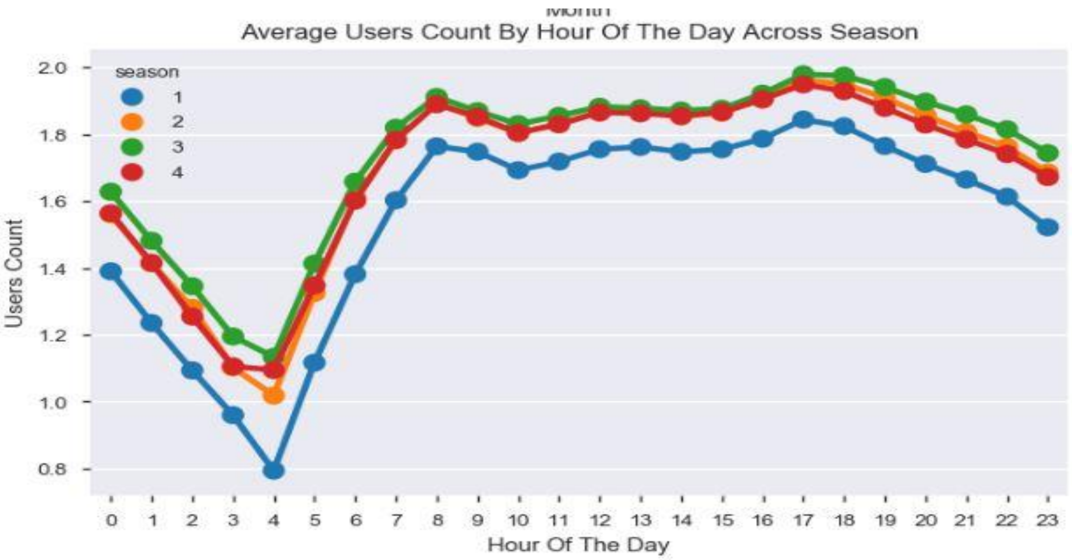
Reason to Check Correlation and Normal Distribution :

We need to remove correlated independent variables because during the model prediction it might happen that one correlated variable is affecting another variable in prediction of the target value and it can make our model biased.

We check for normal distribution of data because Linear Regression Model assume the data to be normally distributed.

Checking Correlation of Categorical Variables with Predictor Variable :





From the graphs above it can be seen that categorical variables do have correlation with predictor variable. Finding from these plots are as follows :-

1. Count of bikes increased from march(3) and keep almost constant till October.
2. It can be seen that during winter season the total count of bikes decreased.
3. We could also find that peak time to have maximum count of bikes is during 8 a.m. and 5-6 p.m.

Model Building :

- The final dataset consists of independent categorical variables ‘yr’ , ‘weathersit’, ‘mnth’ ,’hr’, ‘holiday’, ‘weekday’, ‘workingday’ and independent numerical variables ‘temp’ , ‘hum’, ‘windspeed’.
- As numeric variables are measured on different scales therefore perform standard scaling of the features before splitting our dataset.
- Perform a train- test split of data with 70% as train data and 30% as test data.

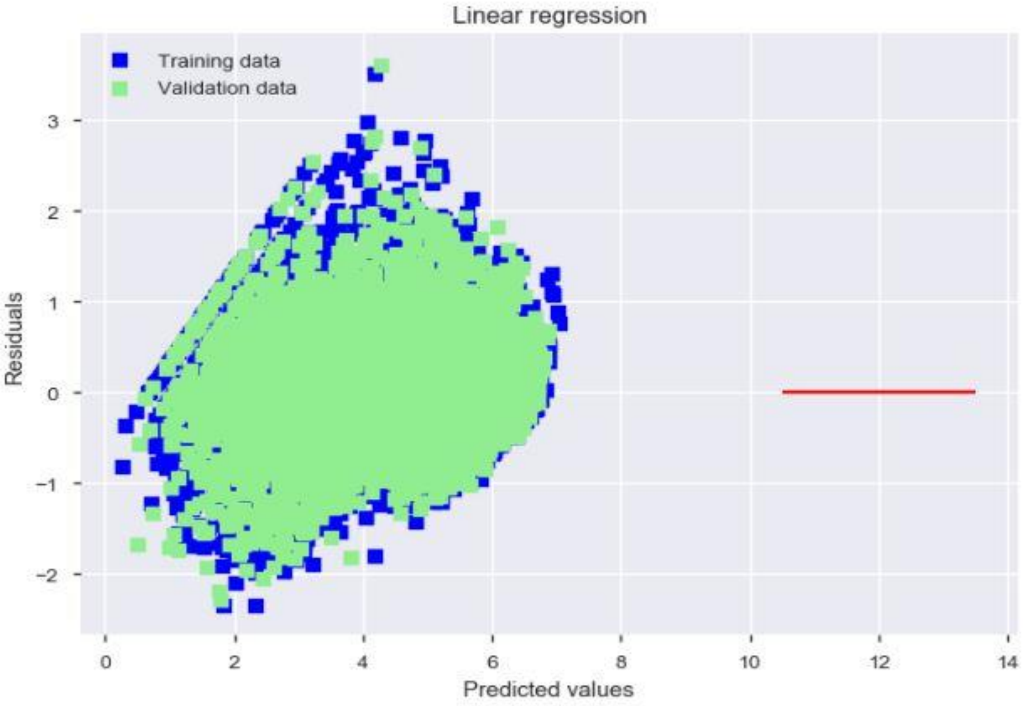
- **RMSE** is taken as official error measure scoring.
- Models applied for Regression : **Linear Regression without Regularization, Linear Regression with Ridge, and Linear Regression with Lasso.**

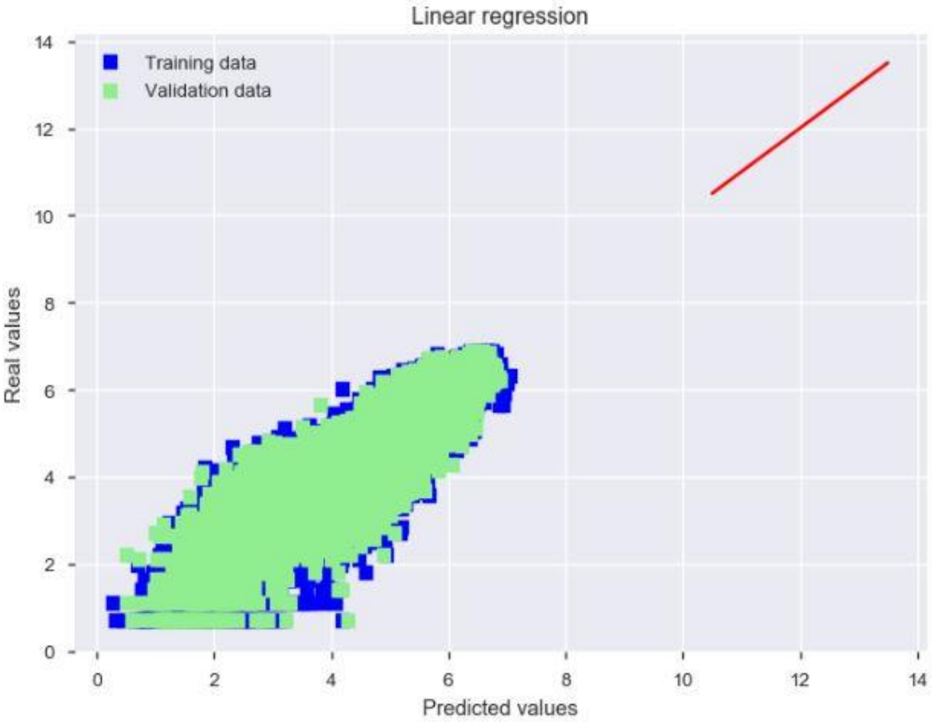
Model Selection :

- RMSE and R-Squared calculated from **Models:** Linear Regression without Regularization, Linear Regression with Ridge, and Linear Regression with Lasso are almost similar which means there is no multicollinearity, overfitting, and noise issues with Linear Regression without Regularization model, and it has handled well the sparse dataset which he got after one-hot encoding.
- As Linear Regression without Regularization is giving us good results we will go with this model as compared to Linear Regression with Ridge, and Linear Regression with Lasso because Linear Regression without Regularization is computationally less complex to run as compared to other models.

Linear Regression Without Regularization Results :

```
1. RMSE on Training set : 0.5889581289214753
2. RMSE on Test set : 0.5971279838150481
3. R^ 2 on Test  0.82
```





R- squared of **0.82** shows that model has done pretty well as the model has explained a variance of 82% from the total variance present in the model. RMSE of 0.59 on bikes count (target variable) explains that the model has performed well to maintain the bias. It can be seen the from the graphs above that Predicted values have covered almost every point of real values in the dataset which means it has covered most of the explanatory information.

Linear Regression without Regularization has performed well to maintain bias-variance trade off in the prediction the count of bikes used per hour.

Assumptions for Writing Code in Production

1. Before writing a production level code we should follow the coding standards followed across the industry so that it can be used by any one in the future with minimal efforts to understand the flow of code.
2. Camel casing should be followed.
3. Hard coding of the logic should be prevented as much as possible.
4. Functions should be written to reduce manual work.
5. All the variables and functions created should be described in comment section before writing the actual code.
6. One function should perform one function only.
7. Models built should be scalable.
8. Code should be platform independent.

PROBLEM 2

Above, you were asked to write a model for a small-to-medium data set. Can you outline how a solution would look like that is able to scale up?

- What are the scaling properties of your model, if you assume that the amount of data you need to handle go up to several terabytes? Do you see any problems?
- How would you address these problems? Are there technologies for data storage/predictive modelling you can build upon?

Describe how the technologies you mention solve the scaling problems you see with model.

- What are the limits and drawbacks for your new approach?
- Do you have hands-on experience with such technologies? Which ones? For how long?

Answer :-

Data Storage Issue : Terabytes and Petabytes of data cannot be directly fetched as done for building prediction model in Problem 1 . The solution to this is storing data onto HDFS (Hadoop File System) and Hive Data warehouse using Sqoop and Spark on Cloudera VM. The benefit of using this Big Data platform is that we can ingest data from various resources into single location. As Big Data(Cloudera) supports parallel computing it can scale up to large amount of data. Cloudera is platform independent as it supports coding in JAVA, SCALA, Python, R. We can perform data analysis as well as built machine learning models using Cloudera VM and Spark. Demo to work with Cloudera can be given during the interview.

Predictive Model Issue : Suppose 100 categorical variables are added to our dataset to predict target variable. In that case creating new features with one hot encoding will increase the dimensionality and make data sparse and Linear Regression might perform poorly. Solution to this can be applying Ridge and Lasso regularization with Linear Regression. We can also apply Random Forest on the dataset as Random forest does not require to do one hot encoding to categorical variables and it also manages to reduce the variance in the data. A sample code for Random Forest is provided in Jupyter Notebook.

Limitation with new approach : It might be expensive to set up Cloudera VM and storing data from various sources, other options might be available with Amazon Webservice, Microsoft Azure, and Google Cloud Platform. Another issue might be training with these new technologies.

Hands-on Experience : I have hands-on experience with Sqoop, Hive , Spark and HDFS for around 1.5 years in addition to Python. I have started learning Spark MLlib to build machine learning models.

PROBLEM 3

We would like you to show your approach in creating a simple data pipeline.

1. Take the data from http://www.nyc.gov/html/tlc/html/about/trip_record_data.shtml

2. Please write a Python program that calculates the average trip length of all Yellow Taxis for a month.

3. Extend this to a data pipeline that can ingest new data and calculates the 45 day rolling average trip length.

Answer :-

The data pipeline was built using Pandas in Python. Other libraries used to fetch the data from website were library requests and library io.

Functions were created to fetch data, update data, transform data types, sanity check on dataset and calculating mean.

These functions can be used of any data present of website mentioned in the problem, just pass the required values through the function.

Detailed explanation of the Data Pipe is provided in the notebook.

References

- https://scikit-learn.org/stable
- https://seaborn.pydata.org
- https://pandas.pydata.org/pandas-docs/stable
- https://stackoverflow.com
- An Introduction to Statistical Learning (By: Gareth James, Daniela Witten, Trevor Hastie and Robert Tibshirani)