**Predication of Bike Rentals**

Vanusha Baregal 23rd October 2019

**Table of Contents**

|  |  |  |
| --- | --- | --- |
| **S.no.** | **Contents** | **Page.No.** |
|  | **Introduction** |  |
| **1.1** | Problem Statement | **2** |
| **1.2** | Dataset | **2** |
|  | **Methodology** | **4** |
| 2.1 | Pre-Processing | **4** |
|  | **Exploratory Data Analysis (EDA)** | **4** |
| **2.2.1** | **Univariate &Bivariate Analysis** | **5** |
| **2.2.2** | **Missing Value analysis** | **5** |
| **2.2.3** | **Outlier Analysis** | **10** |
| **2.2.4** | **Correlation Analysis** | **11** |
| **2.2.5** | **Chi Square test** | **12** |
| **2.2.6** | **Normalisation** | **13** |
| **2.3** | **Modelling** | **13** |
| **2.3.1** | **Linear Regression** |  |
| **2.3.2** | **Random Forest Regression** | **18** |
| **3.1** | **Model Evaluation** | **22** |
| **3.2** | **Model Selection** | **22** |
|  | **Appendix** |  |
|  | R implementation |  |
|  | Python implementation |  |

**Chapter 1**

**Introduction**

* 1. **Problem Statement**

A bike rental business rents out bicycles for short periods of time provided mostly by bike shops as a side-line to their main businesses of sales and service and also by specialized shops in rentals. These rental shops primarily serve people who do not have access to a vehicle, typically travellers and particularly [tourists](https://en.wikipedia.org/wiki/Tourist). Specialized bicycle rental shops therefore typically operate at beaches, parks, or other locations that tourists frequent. These shops allow both registered and casual (‘walk in’) users to travel across cities, counties, and even to more remote destinations and one of the most important problem from a business point of view is to predict the bike demand on any particular day. While having excess bikes results in wastage of resource (both with respect to bike maintenance and the land/bike stand required for parking and security), having fewer bikes leads to revenue loss (ranging from a short term loss due to missing out on immediate customers to potential longer term loss due to loss in future customer base), Rental process is highly correlated to the environmental and seasonal settings. For instance, weather conditions, precipitation, day of week, season etc. can affect the rental behaviours. Data from the users is constantly being collected for analytics purposes to help prepare for a change in the demand of bike rental from their users. Thus, having an estimate on the demands would enable efficient functioning of these companies The objective of this Case is Predication of bike rental count on daily based on the environmental and seasonal settings.

* 1. **Data**

The data set consists of 731 observations recorded between the period of 2 Years, between 2011 and 2012. It has 15 variables or predictors and 1 target variable. Given below is a sample of the data set that we are using to predict the bike rental counts:

Table 1.1: Bike Renting Sample Data (Columns: 1-5)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Instant | Date | Season | | Year | | Month | | Holiday | | Weekday | | Workingday | |
| 1 | 01-01-2011 | 1 | 0 | | 1 | | 0 | | 6 | | 0 | |
| 2 | 02-01-2011 | 1 | 0 | | 1 | | 0 | | 0 | | 0 | |
| 3 | 03-01-2011 | 1 | 0 | | 1 | | 0 | | 1 | | 1 | |
| 4 | 04-01-2011 | 1 | 0 | | 1 | | 0 | | 2 | | 1 | |
| 5 | 05-01-2011 | 1 | 0 | | 1 | | 0 | | 3 | | 1 | |

Table 1.2: Bike Renting Sample Data (Columns: 7-12)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Weathersit | Temp | Atemp | Humidity | Windspeed | Casual | Registered | Count |
| 2 | 0.344 | 0.364 | 0.806 | 0.160 | 331 | 654 | 985 |
| 2 | 0.363 | 0.354 | 0.696 | 0.249 | 131 | 670 | 801 |
| 1 | 0.196 | 0.189 | 0.437 | 0.248 | 120 | 1229 | 1349 |
| 1 | 0.200 | 0.212 | 0.590 | 0.160 | 108 | 1454 | 1562 |
| 1 | 0.227 | 0.229 | 0.437 | 0.187 | 82 | 1518 | 1600 |

As you can see in the table below we have the following 16 variables, using which we have to predict the bike rental counts:

|  |  |
| --- | --- |
| Variable | Description |
| 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 | Whether day is holiday or not (extracted from holiday Schedule) |
| Weekday | Day of the week |
| Working day | If day is neither weekend nor holiday is 1, otherwise is 0. |
| Weathersit | (extracted from freemeteo)  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 derived via  (t-t\_min)/(t\_max-t\_min),  T\_min=-8, t\_max=+39 (only in hourly scale) |
| Atemp | Normalized feeling temperature in Celsius. The values are derived via  (t-t\_min)/(t\_maxt\_  Min), t\_min=-16, t\_max=+50 (only in hourly scale) |
| 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 |

The above data set consists of 8 Categorical, 7 Continuous and 1 Target Variable.

**Chapter 2**

**Methodology**

Before building any predictive model it is necessary to look at the raw data. Looking at data refers to exploring the data, cleaning the data as well as visualizing the data through graphs and plots. This is often called as Exploratory Data Analysis. As the target variable “Count” is continuous, our task is to build regression model to predict the count of bike rented depending on various environmental and seasonal settings. First step in EDA is to look at all the probability distributions of the variables. Most analysis like regression, require the data to be normally distributed. We can visualize that in a glance by looking at the probability distributions or probability density functions of the variable. Linear regression and Random forest regression were used for modelling and their performance comparison was performed. Both the algorithms were implemented in R and python.

**2.1 Pre Processing**

Pre-processing was performed in both R and python. The dataset consists of 731 observations,

and 16 predictors. The process of pre-processing techniques was used for cleaning and reorder the data set in a proper format by changing into categorical variables and Variable (columns) names.

**Exploratory Data Analysis**

In exploring the data, we have

* Rename variables
* Univariate analysis and variable consolidation
* Converted Season, Month, Working day, Weather into categorical variables
* Deleted instant variable as it is nothing but an index and Date variable as month and week are already included
* Omitted registered and casual variable as sum of registered and casual is the total count that is what we have to predict.

**Summary**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Temperature | Atemperature | Humidity | Windspeed | Casual | Registered | Count |
| Mean | 0.495385 | 0.474354 | 0.627894 | 0.190486 | 848.176471 | 3656.172367 | 4504.348837 |
| Std | 0.183051 | 0.162961 | 0.142429 | 0.077498 | 686.622488 | 1560.256377 | 1937.211452 |
| Min | 0.059130 | 0.079070 | 0.000000 | 0.022392 | 2.000000 | 20.000000 | 22.000000 |
| 25% | 0.337083 | 0.337842 | 0.520000 | 0.134950 | 315.500000 | 2497.000000 | 3152.000000 |
| 50% | 0.498333 | 0.486733 | 0.626667 | 0.180975 | 713.000000 | 3662.000000 | 4548.000000 |
| 75% | 0.655417 | 0.608602 | 0.730209 | 0.233214 | 1096.000000 | 4776.500000 | 5956.000000 |
| Max | 0.861667 | 0.840896 | 0.972500 | 0.507463 | 3410.000000 | 6946.000000 | 8714.000000 |

**2.1.1 Missing value Analysis**

In statistics, missing data, or missing values, occur when no data value is stored for the variable in an observation. Missing data are a common occurrence and can have a significant effect on the conclusions that can be drawn from the data. Missing value analysis was performed in both R and Python. It was found that there were no missing values in the data.

|  |  |
| --- | --- |
| Variable | Missing Values |
| Season | 0 |
| Year | 0 |
| Month | 0 |
| Holiday | 0 |
| Weekday | 0 |
| Workingday | 0 |
| Weather | 0 |
| Temperature | 0 |
| Atemperature | 0 |
| Humidity | 0 |
| Windspeed | 0 |
| Casual | 0 |
| Registered | 0 |
| Count | 0 |

**2.1.2 (a) Univariate Analysis**

In univariate analysis, we look at the probability density functions numeric variables present in the data including target variable

1. Target variable Count is normally distributed
2. Independent variables like ‘Temperature’,’Atemperature’, and ‘Registered’ data is distributed normally.
3. Independent variable ‘casual’ data is slightly skewed to the right so, there is chances of getting outliers.
4. Other Independent variable ‘Humidity’ data is slightly skewed to the left, here data is already in normalized form.

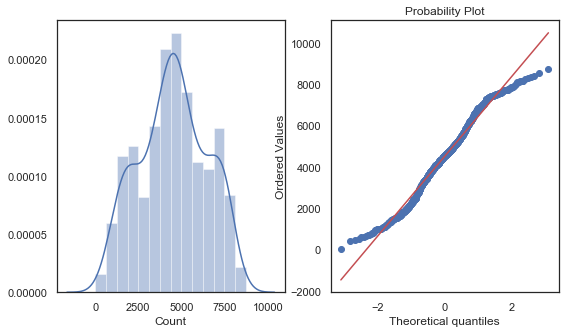
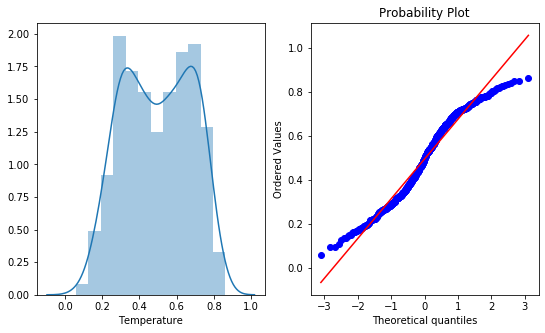
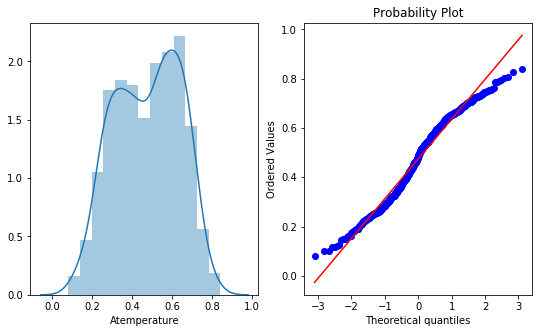
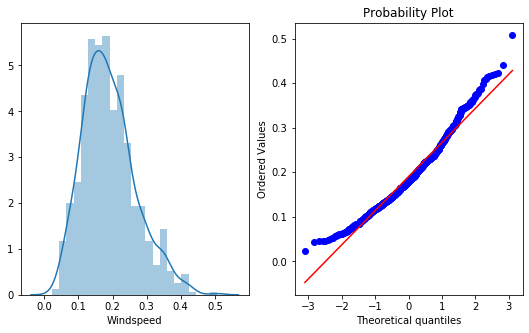
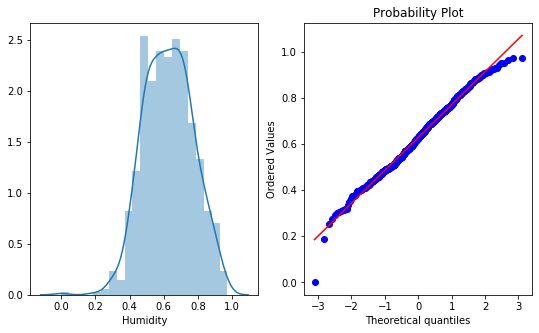
****

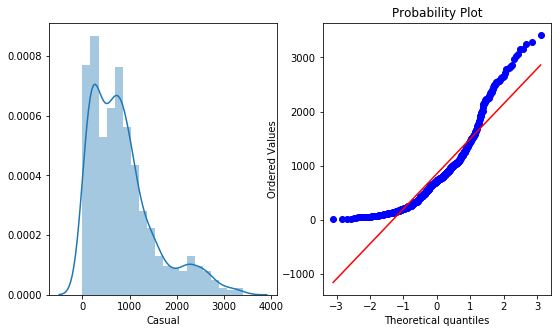
Figure 2.1 Distribution of target variable (Count)

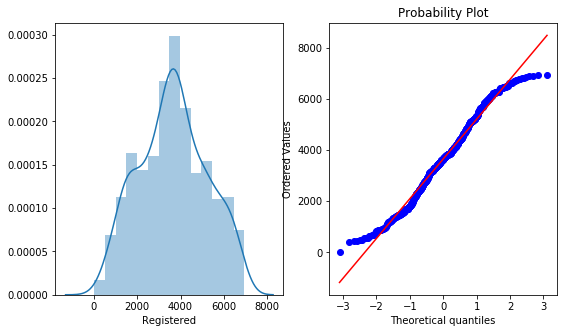
****

****

****

****

****

****

**Figure 2.2.** Distribution of temp, atemp, windspeed, humidity,casual,registered

**2.1.2 (b) Bivariate analysis**

In bivariate analysis, we will look at the relationship between target variable and predictor.

From the scatter plots, findings are:

* ‘Count’ and ‘Temperature’ have strong and positive relationship. It means that as the temperature rises, the bike demand also increase.
* ‘Atemperature’ and ‘Count’ have strong and positive relationship. It means that as the ambient temperature rise, demand for bikes also increases.
* Humidity’ has a negative linear relationship with ‘Count’. As humidity increases, count decreases.
* ‘Wind speed’ has negative linear relationship with ‘Count’. With an increase in wind speed, bike count decreases.

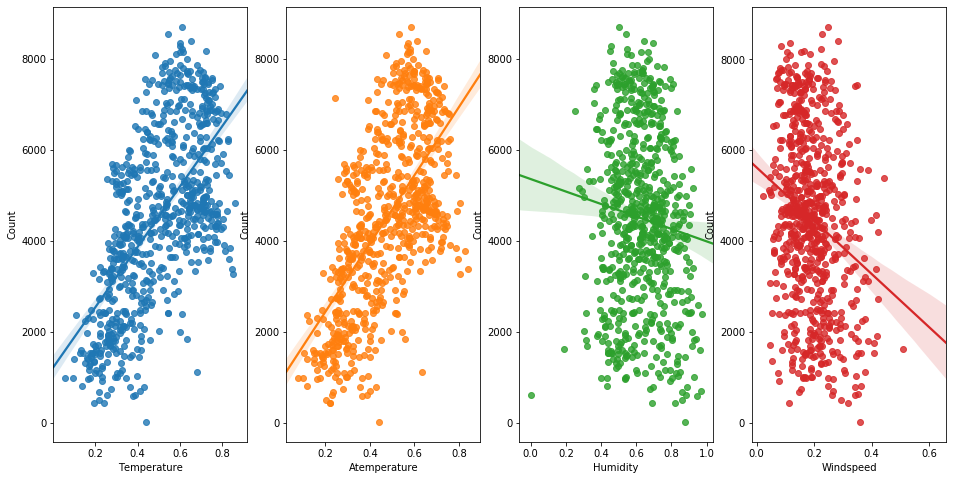
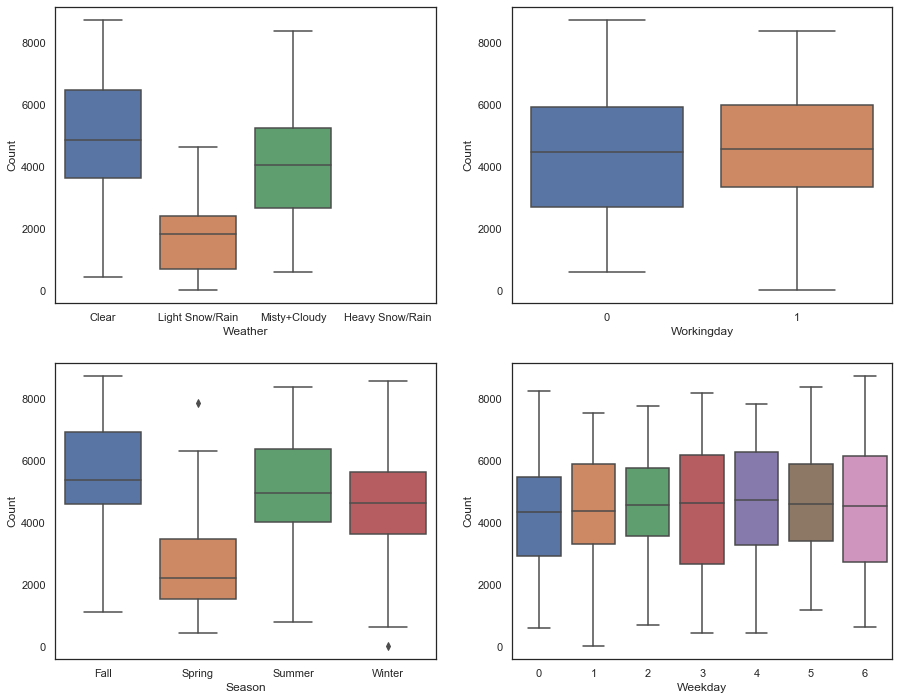
****

Fig 2.3. Relationship between target variable and continuous predictors

****

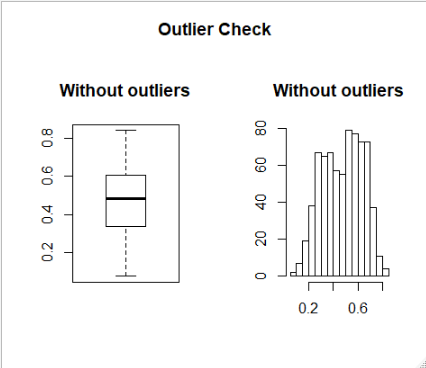
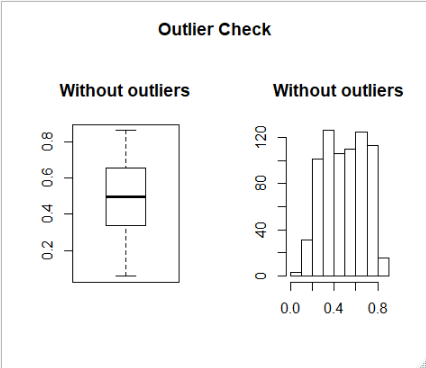
**Few Observations**

1. Count of bike rentals is higher during clear, few clouds, partly cloudy, cloudy weather and less during light and heavy rains
2. No significant effect of either holiday or working day on count of bike rentals
3. Season ‘Fall’ has seen good number of users renting bikes

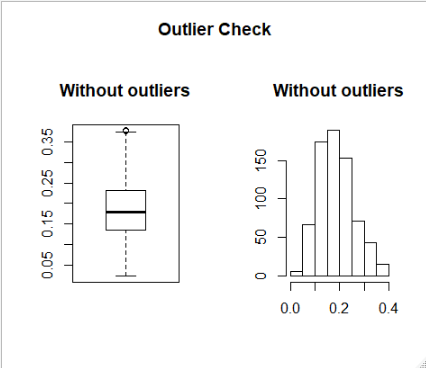
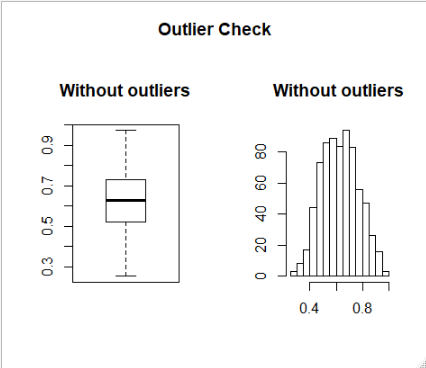
**2.1.4 Outlier Analysis**

After missing value analysis, we check for outliers in target variable and predictors. Outlier analysis is done to handle all inconsistent observations present in given dataset. As outlier analysis can only be done on continuous variable. There were no outliers present in the dataset. Some extreme values were present in the predictors but those seems to be logical. So no observations were removed and no imputation was performed on the dataset. Boxplot method was used to check for outliers. Below are the figures from the R implementation.

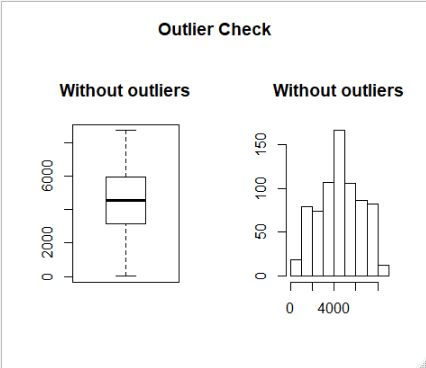
1. Temperature 2. Atemperature



3.Humidity 4. Wind speed



5. Count



**2.1.5 Correlation Analysis**

Correlation analysis is a method of statistical evaluation used to study the strength of a relationship between two, numerically measured, continuous variables (e.g. height and weight). This particular type of analysis is useful when a researcher wants to establish if there are possible connections between variables. It is also used to check for multicollinearity among predictors. Multicollinearity exists whenever two or more of the predictors in a regression model are moderately or highly correlated. The basic problem is multicollinearity results in

unstable estimation of coefficients which makes it difficult to access the effect of independent variable on dependent variable.

**Correlation Matrix**

|  | **Temperature** | **Atemperature** | **Humidity** | **Windspeed** | **Casual** | **Registered** | **Count** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Temperature** | 1 | 0.99 | 0.13 | -0.16 | 0.54 | 0.54 | 0.63 |
| **Atemperature** | 0.99 | 1 | 0.14 | -0.18 | 0.54 | 0.54 | 0.63 |
| **Humidity** | 0.13 | 0.14 | 1 | -0.25 | -0.077 | -0.091 | -0.1 |
| **Windspeed** | -0.16 | -0.18 | -0.25 | 1 | -0.17 | -0.22 | -0.23 |
| **Casual** | 0.54 | 0.54 | -0.077 | -0.17 | 1 | 0.4 | 0.67 |
| **Registered** | 0.54 | 0.54 | -0.091 | -0.22 | 0.4 | 1 | 0.95 |
| **Count** | 0.63 | 0.63 | -0.1 | -0.23 | 0.67 | 0.95 | 1 |

**Findings:**

1. Temperature and Atemperature are highly correlated, thereby one variable will be dropped
2. Sum of casual and registered users gives us the count which we are predicting thereby these two variables will be dropped.
3. Count’ have a strong and positive relationship with temperature and ambient temperature which is logical. People tend to rent bikes more when temperature is high
4. Relationship between Humidity, Wind speed and count is very weak.

**2.1.7 Chi-squared Test of Independence**

The Chi-squared test is used to determine whether an association (or relationship) between 2 categorical variables in a sample is likely to reflect a real association between these 2 variables in the population. The result from the analysis showed that there was association between the dependent and independent categorical variables.

Null hypothesis: Variable (Season, Year, Month, Holiday, Weekday, Weather) and Target variable (Count) are independent

Alternate hypothesis: Variable (Season, Year, Month, Holiday, Weekday, Weather) and Target variable (Count) are correlated

As all the variables had p-value higher than 0.05, we rejected null hypothesis and concluded that the variables have relation with the target variable.

|  |  |
| --- | --- |
| Variable | p-value |
| Season | 0.5441 |
| Year | 0.3677 |
| Month | 0.4918 |
| Holiday | 0.6781 |
| Weekday | 0.4102 |
| Working day | 0.4544 |
| Weather | 0.4678 |

**2.1.8 Feature Scaling and Normalization**

Data normalization is the process of rescaling one or more attributes to the range of [0, 1]. This means largest value of each attribute is 1 and smallest is 0. Normalization is a good technique to use when you know that your data distribution is not Gaussian.

As given in the problem statement Temperature, Atemperature, Humidity are already normalised. So here we use Normalisation technique on target variable “Count” for rescaling.

**Normalised Values of Numerical Variables (Top 6 rows)**

|  |  |  |  |
| --- | --- | --- | --- |
| Temperature | Humidity | Windspeed | Count |
| 0.3441670 | 0.805833 | 0.1604460 | 0.11079153 |
| 0.3634780 | 0.696087 | 0.2485390 | 0.08962264 |
| 0.1963640 | 0.437273 | 0.2483090 | 0.15266912 |
| 0.2000000 | 0.590435 | 0.1602960 | 0.17717441 |
| 0.2269570 | 0.436957 | 0.1869000 | 0.18154625 |
| 0.2043480 | 0.518261 | 0.0895652 | 0.18223654 |

**2.2 Modeling**

**2.2.1 Model Selection**

In our bike renting project the target variable is continuous in nature, hence the task of predicting the rentals is regression problem. Two Machine learning algorithms were used.

1. Multivariate linear regression

2. Random forest regressor – an ensemble tree based regression

After EDA and pre-processing steps, data was divided into training and test dataset with 80 % and 20 % ratio. Model was built using the above two machine learning algorithms and after that the diagnostic plots were used to check the assumptions of linear regression. For performance tuning of random forest, hyper parameter tuning was used.

**Linear Regression**

Linear regression is a technique in which we try to model a linear relationship with target and predictors.

First linear regression was used.

* Data was divided into train and test.
* Linear regression was trained on training data.
* Backward and Forward elimination method was used on model with all predictors to select the best model.
* MAP and RMSE was used to check the performance of the model
* Prediction were done on the test data.

**R Implementation:**

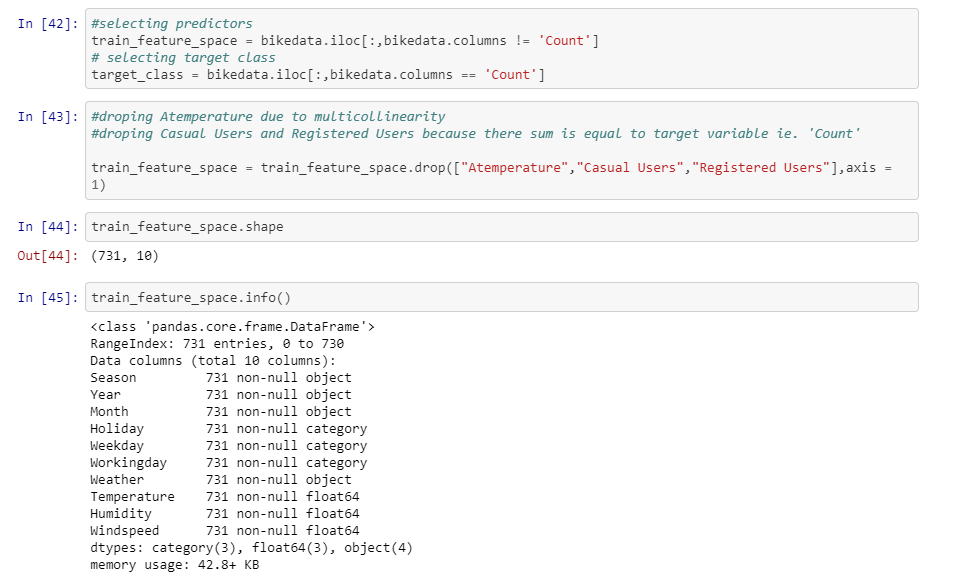
First a model will all the predictors was trained in R. I.e. model1. Below is summary of model1

|  |
| --- |
| # ----------------- Model 1 Linear Regression -----------------------------------------------------#  >  >  > set.seed(654)  > split <- sample.split(bikedata$Count, SplitRatio = 0.70)  > training\_set <- subset(bikedata, split == TRUE)  > test\_set <- subset(bikedata, split == FALSE)  >  >  > model1 <- lm(Count ~ ., data = training\_set)  >  > # step wise model selection  >  > modelAIC <- stepAIC(model1, direction = "both")  Start: AIC=6800.56  Count ~ Season + Year + Month + Holiday + Weekday + Workingday +  Weather + Temperature + Atemperature + Humidity + Windspeed  Step: AIC=6800.56  Count ~ Season + Year + Month + Holiday + Weekday + Weather +  Temperature + Atemperature + Humidity + Windspeed  Df Sum of Sq RSS AIC  - Atemperature 1 604568 275309771 6799.7  <none> 274705203 6800.6  - Temperature 1 1680528 276385731 6801.7  - Holiday 1 2541499 277246702 6803.3  - Weekday 6 8931824 283637028 6804.9  - Humidity 1 8547280 283252484 6814.2  - Windspeed 1 15775500 290480703 6827.1  - Month 11 38792108 313497311 6846.1  - Weather 2 43451067 318156270 6871.6  - Season 3 45689214 320394417 6873.2  - Year 1 507678731 782383934 7333.4  Step: AIC=6799.69  Count ~ Season + Year + Month + Holiday + Weekday + Weather +  Temperature + Humidity + Windspeed  Df Sum of Sq RSS AIC  <none> 275309771 6799.7  + Atemperature 1 604568 274705203 6800.6  - Holiday 1 2726121 278035892 6802.7  - Weekday 6 8679904 283989675 6803.5  - Humidity 1 8284810 283594581 6812.8  - Windspeed 1 17582336 292892107 6829.3  - Month 11 38214582 313524353 6844.1  - Temperature 1 35748724 311058495 6860.1  - Weather 2 44428926 319738697 6872.1  - Season 3 45830789 321140560 6872.4  - Year 1 507074640 782384411 7331.4  > summary(modelAIC)  Call:  lm(formula = Count ~ Season + Year + Month + Holiday + Weekday +  Weather + Temperature + Humidity + Windspeed, data = training\_set)  Residuals:  Min 1Q Median 3Q Max  -3479.9 -351.7 71.3 425.4 2418.5  Coefficients:  Estimate Std. Error t value Pr(>|t|)  (Intercept) 1514.61 293.24 5.165 3.52e-07 \*\*\*  SeasonSummer 1058.45 199.47 5.306 1.71e-07 \*\*\*  SeasonFall 1092.89 243.02 4.497 8.63e-06 \*\*\*  SeasonWinter 1740.57 209.33 8.315 9.41e-16 \*\*\*  Year2012 2054.43 68.88 29.826 < 2e-16 \*\*\*  MonthFeb 211.07 170.44 1.238 0.216161  MonthMar 505.08 195.72 2.581 0.010158 \*  MonthApr 471.39 284.54 1.657 0.098240 .  MonthMay 897.34 310.55 2.889 0.004032 \*\*  MonthJune 667.54 329.89 2.024 0.043568 \*  MonthJuly 53.63 371.28 0.144 0.885217  MonthAug 488.20 357.27 1.366 0.172427  MonthSep 928.93 309.64 3.000 0.002839 \*\*  MonthOct 612.68 285.72 2.144 0.032506 \*  MonthNov -71.15 268.27 -0.265 0.790960  MonthDec -144.34 210.37 -0.686 0.492969  Holiday1 -493.88 225.83 -2.187 0.029226 \*  Weekday1 84.97 132.12 0.643 0.520427  Weekday2 212.54 127.53 1.667 0.096254 .  Weekday3 344.98 126.02 2.738 0.006417 \*\*  Weekday4 302.66 125.41 2.413 0.016180 \*  Weekday5 365.09 125.24 2.915 0.003721 \*\*  Weekday6 339.40 124.79 2.720 0.006767 \*\*  WeatherCloudy -412.23 94.14 -4.379 1.46e-05 \*\*\*  WeatherLight Snow -2059.08 234.47 -8.782 < 2e-16 \*\*\*  Temperature 3986.92 503.44 7.919 1.65e-14 \*\*\*  Humidity -1398.37 366.79 -3.812 0.000155 \*\*\*  Windspeed -2708.02 487.59 -5.554 4.62e-08 \*\*\*  ---  Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1  Residual standard error: 755 on 483 degrees of freedom  Multiple R-squared: 0.8558, Adjusted R-squared: 0.8477  F-statistic: 106.1 on 27 and 483 DF, p-value: < 2.2e-16  >  > # Apply prediction on test set  > test\_prediction <- predict(modelAIC, newdata = test\_set)  >  > test\_rmse <- rmse(test\_set$Count, test\_prediction)  > print(paste("root-mean-square error for linear regression model is ", test\_rmse))  [1] "root-mean-square error for linear regression model is 821.372628075882"  > print(paste("Mean Absolute Error for linear regression model is ",MAE(test\_set$Count,test\_prediction)))  [1] "Mean Absolute Error for linear regression model is 575.459501759832"  > print("summary of predicted count values")  [1] "summary of predicted count values"  > summary(test\_prediction)  Min. 1st Qu. Median Mean 3rd Qu. Max.  -1334 3543 4716 4547 5903 7889  > print("summary of actual Count values")  [1] "summary of actual Count values"  > summary(test\_set$Count)  Min. 1st Qu. Median Mean 3rd Qu. Max.  506 3112 4650 4550 5949 8395  >  > # From the summary we can observe negative prediction values  > #We will perform log transformation of trarget variable  > model2 <- lm(log(Count)~., data = training\_set)  >  > stepwiseLogAICModel <- stepAIC(model2,direction = "both")  Start: AIC=-1172.01  log(Count) ~ Season + Year + Month + Holiday + Weekday + Workingday +  Weather + Temperature + Atemperature + Humidity + Windspeed  Step: AIC=-1172.01  log(Count) ~ Season + Year + Month + Holiday + Weekday + Weather +  Temperature + Atemperature + Humidity + Windspeed  Df Sum of Sq RSS AIC  - Weekday 6 0.6975 46.728 -1176.32  - Atemperature 1 0.0220 46.053 -1173.77  <none> 46.031 -1172.01  - Holiday 1 0.3205 46.351 -1170.46  - Temperature 1 0.4928 46.523 -1168.57  - Month 11 2.8682 48.899 -1163.12  - Humidity 1 1.3827 47.413 -1158.89  - Windspeed 1 2.0611 48.092 -1151.63  - Season 3 5.9065 51.937 -1116.32  - Weather 2 9.1973 55.228 -1082.92  - Year 1 24.7937 70.824 -953.82  Step: AIC=-1176.32  log(Count) ~ Season + Year + Month + Holiday + Weather + Temperature +  Atemperature + Humidity + Windspeed  Df Sum of Sq RSS AIC  - Atemperature 1 0.0075 46.736 -1178.24  <none> 46.728 -1176.32  + Workingday 1 0.1100 46.618 -1175.53  - Holiday 1 0.5013 47.229 -1172.87  + Weekday 6 0.6975 46.031 -1172.01  - Temperature 1 0.6271 47.355 -1171.51  - Month 11 2.8524 49.581 -1168.05  - Humidity 1 1.5565 48.285 -1161.58  - Windspeed 1 2.1192 48.847 -1155.66  - Season 3 5.9384 52.667 -1121.19  - Weather 2 9.1419 55.870 -1089.02  - Year 1 24.9092 71.637 -959.99  Step: AIC=-1178.24  log(Count) ~ Season + Year + Month + Holiday + Weather + Temperature +  Humidity + Windspeed  Df Sum of Sq RSS AIC  <none> 46.736 -1178.24  + Workingday 1 0.1082 46.627 -1177.43  + Atemperature 1 0.0075 46.728 -1176.32  - Holiday 1 0.5106 47.246 -1174.69  + Weekday 6 0.6830 46.053 -1173.77  - Month 11 2.8514 49.587 -1169.98  - Humidity 1 1.5490 48.285 -1163.58  - Windspeed 1 2.2438 48.979 -1156.28  - Season 3 5.9438 52.679 -1123.07  - Temperature 1 6.7043 53.440 -1111.74  - Weather 2 9.2252 55.961 -1090.19  - Year 1 24.9068 71.642 -961.95  > test\_prediction\_log<- predict(stepwiseLogAICModel, newdata = test\_set)  > predict\_test\_nonlog <- exp(test\_prediction\_log)  >  > test\_rmse2 <- rmse(test\_set$Count, predict\_test\_nonlog)  > print(paste("root-mean-square error between actual and predicted", test\_rmse))  [1] "root-mean-square error between actual and predicted 821.372628075882"  > print(paste("Mean Absolute Error for linear regression model is ",  + MAE(test\_set$Count,predict\_test\_nonlog)))  [1] "Mean Absolute Error for linear regression model is 696.180959982148"  >  > summary(predict\_test\_nonlog)  Min. 1st Qu. Median Mean 3rd Qu. Max.  486 3063 4381 4484 5822 10614  > summary(test\_set$Count)  Min. 1st Qu. Median Mean 3rd Qu. Max.  506 3112 4650 4550 5949 8395  >  >  >  > par(mfrow = c(1,1))  > plot(stepwiseLogAICModel) |

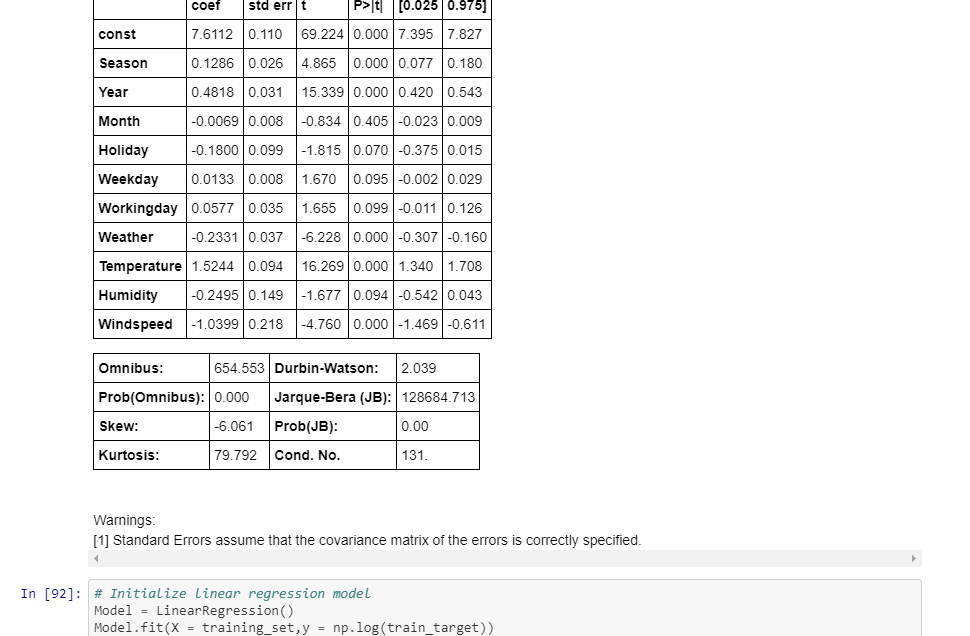
|  |
| --- |
| # ----------------- Model 2 Random forest -----------------------------------------------------#  model1 <- randomForest(Count ~.,  > data = training\_set,ntree = 500, mtry = 8, importance = TRUE)  > print(model1)  Call:  lm(formula = Count ~ ., data = training\_set)  Coefficients:  (Intercept) SeasonSummer SeasonFall SeasonWinter Year2012 MonthFeb  1465.98 1052.92 1090.08 1739.77 2056.81 207.30  MonthMar MonthApr MonthMay MonthJune MonthJuly MonthAug  505.95 468.06 914.46 695.86 74.15 537.96  MonthSep MonthOct MonthNov MonthDec Holiday1 Weekday1  954.10 611.19 -77.42 -149.70 -477.98 84.20  Weekday2 Weekday3 Weekday4 Weekday5 Weekday6 Workingday1  216.58 349.06 304.14 374.08 342.14 NA  WeatherCloudy WeatherLight Snow Temperature Atemperature Humidity Windspeed  -409.63 -2041.38 2548.79 1571.57 -1423.48 -2611.79  > par(mfrow = c(1,1))  > plot(model1)  Hit <Return> to see next plot:  Hit <Return> to see next plot:  Hit <Return> to see next plot: # 300 trees selected from the plot  Hit <Return> to see next plot:  > tumedmodel <- tuneRF(training\_set[,1:11], training\_set[,12], stepFactor = 0.5, plot = TRUE,  + ntreeTry = 250, trace = TRUE, improve = 0.05)  mtry = 3 OOB error = 482840.4  Searching left ...  mtry = 6 OOB error = 450199.4  0.06760194 0.05  mtry = 12 OOB error = 460451.7  -0.0227728 0.05  Searching right ...  mtry = 1 OOB error = 915072.8  -1.032594 0.05  > # selected mtry = 6 from the plot  >  > tuned\_randomForest <- randomForest(Count ~. - Atemperature,  + data = training\_set,ntree = 250, mtry = 6, importance = TRUE)  > tuned\_randomForest  Call:  randomForest(formula = Count ~ . - Atemperature, data = training\_set, ntree = 250, mtry = 6, importance = TRUE)  Type of random forest: regression  Number of trees: 250  No. of variables tried at each split: 6  Mean of squared residuals: 460970  % Var explained: 87.66  >  > # predicting using random forest model 1  > rf1\_prediction <- predict(tuned\_randomForest,test\_set[,-12])  > rmse(rf1\_prediction,test\_set$Count)  [1] 749.583  > print(paste("Mean Absolute Error for Random forest regressor is ",  + MAE(test\_set$Count,rf1\_prediction)))  [1] "Mean Absolute Error for Random forest regressor is 501.871369582841"  > |

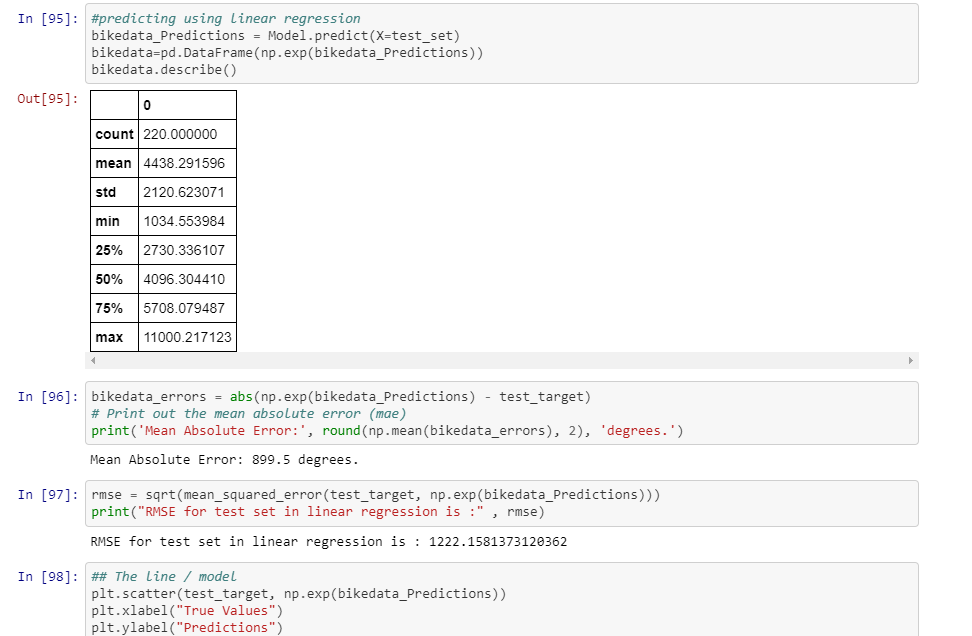
**Python Implementation**:

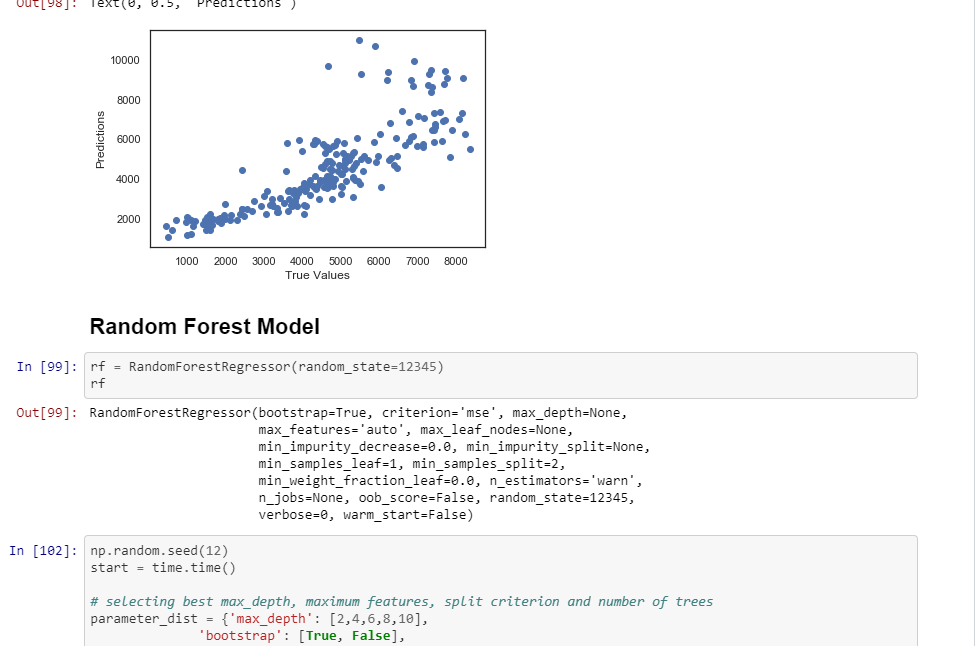
In python a single regression model was trained after all pre-processing. Python don’t have step wise regression implementation. Same log transformation was performed to avoid negative prediction.



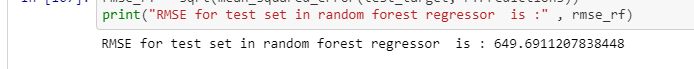




****

****

****

****

**3.1 Model Evaluation**

Now that we have a few models for predicting the target variable, we need to decide which one to choose. There are several criteria that exist for evaluating and comparing models.

**Performance Measure**

**R implementation**

For measuring rmse, Metric package was used. For measuring MAE, a function was written. The

values for both the metric for linear regression and random forest are as follow.

|  |  |  |
| --- | --- | --- |
| **Error metric** | **Linear Regression** | **Random Forest** |
| RMSE | 821.37 | 749.58 |
| MAE | 696.18 | 501.87 |

Error metric / Algorithm Linear Regression Random Forest

As from the table we can see that random forest performing better than linear regression on both the error metric.

**Python implementation**

In python, both the error metric was calculated using python functions. No pre-built package or

modules were used. The values for both metric are given below.

E

|  |  |  |
| --- | --- | --- |
| **Error metric** | **Linear Regression** | **Random Forest** |
| RMSE | 1222.15 | 649.69 |
| MAE | 899.5 | 495.28 |

/ Algorithm Linear Regression Random Forest

As we can see random forest performing better than linear regression.

**Result**

From the error metric we can see that random forest is performing better than linear regression

in both implementations. The result for random forest is similar in both R and python. But in case of linear regression, R’s implementation is performing better than python. The difference here is that data in R was normalized before regression.

**Model selection**

Selection of model depends on use case. If we want to study the effects of predictors in details, we will go for linear regression and look at the regression equation. If we are care about more precise prediction, we will opt for random forest.

R implementation- Codes

|  |
| --- |
| #Clear Environment-  rm(list=ls())  library(corrplot)  library(ggplot2)  library(dplyr)  library(rcompanion)  library(mlr)  library(caTools)  library(MASS)  library(Metrics)  library(randomForest)  #Set working directory-  setwd("F:/EdwisorVanusha/Project/Master files")  #Check working directory-  getwd()  #load data-  bikedata= read.csv("bike\_rental.csv")  #------------------------------Exploratory Data Analysis-------------------------------------------#  class(bikedata)  dim(bikedata)  head(bikedata)  names(bikedata)  str(bikedata)  summary(bikedata)  #Remove the instant variable, as it is index in dataset.  bikedata= subset(bikedata,select=-(instant))  #Remove date variable as we have to predict count on seasonal basis not date basis-  bikedata= subset(bikedata,select=-(dteday))  #check the remaining variables-  names(bikedata)  #Rename the variables-  names(bikedata)[1]="Season"  names(bikedata)[2]="Year"  names(bikedata)[3]="Month"  names(bikedata)[4]="Holiday"  names(bikedata)[5]="Weekday"  names(bikedata)[6]="Workingday"  names(bikedata)[7]="Weather"  names(bikedata)[8]="Temperature"  names(bikedata)[9]="Atemperature"  names(bikedata)[10]="Humidity"  names(bikedata)[11]="Windspeed"  names(bikedata)[12]="Casual"  names(bikedata)[13]="Registered"  names(bikedata)[14]="Count"  #Seperate categorical and numeric variables-  names(bikedata)  #numeric variables-  cnames= c("Temperature","Atemperature","Humidity","Windspeed","Count")  #categorical varibles-  cat\_cnames= c("Season","Year","Month","Holiday","Weekday","Workingday","Weather")  str(bikedata)  #=================================Data Pre-processing==========================================#  #--------------------------------Missing Vlaue Analysis----------------------------------------#  #Check missing values in dataset-  sum(is.na(bikedata))  #Missing value= 0  #No Missing values in data.  #convering categorical variables into factor  bikedata$Season <- as.factor(bikedata$Season)  levels(bikedata$Season) <- c("spring", "summer", "fall", "winter")  bikedata$Year <- as.factor(bikedata$Year)  levels(bikedata$Year) <- c(2011, 2012)  bikedata$Month <- as.factor(bikedata$Month)  levels(bikedata$Month) <- c("Jan", "Feb", "Mar", "Apr", "May", "Jun", "Jul", "Aug", "Sep", "Oct", "Nov", "Dec")  bikedata$Holiday <- as.factor(bikedata$Holiday)  levels(bikedata$Holiday) <- c("Not Holiday", "Holiday")  bikedata$Weekday <- as.factor(bikedata$Weekday)  levels(bikedata$Weekday) <- c("Sun", "Mon", "Tues", "Wed", "Thurs", "Fri", "Sat")  bikedata$Workingday <- as.factor(bikedata$Workingday)  levels(bikedata$Workingday) <- c("Holiday", "Workingday")  bikedata$Weather<- as.factor(bikedata$Weather)  levels(bikedata$Weather) <- c("Clear", "Cloudy", "Rainy","Heavy rain")  str(bikedata)  #-----------------------------------Outlier Analysis----------------------------------------------#  #create Box-Plot for outlier analysis-  outlierKD <- function(dt, var) {  var\_name <- eval(substitute(var), eval(dt))  na1 <- sum(is.na(var\_name))  m1 <- mean(var\_name, na.rm = T)  par(mfrow = c(1, 2), oma = c(0, 0, 3, 0))  boxplot(var\_name, main = "With outliers")  hist(var\_name,  main = "With outliers",  xlab = NA,  ylab = NA)  outlier <- boxplot.stats(var\_name)$out  mo <- mean(outlier)  var\_name <- ifelse(var\_name %in% outlier, NA, var\_name)  boxplot(var\_name, main = "Without outliers")  hist(var\_name,  main = "Without outliers",  xlab = NA,  ylab = NA)  title("Outlier Check", outer = TRUE)  na2 <- sum(is.na(var\_name))  cat("Outliers identified:", na2 - na1, "n")  cat("Propotion (%) of outliers:", round((na2 - na1) / sum(!is.na(var\_name)) \*  100, 1), "n")  cat("Mean of the outliers:", round(mo, 2), "n")  m2 <- mean(var\_name, na.rm = T)  cat("Mean without removing outliers:", round(m1, 2), "n")  cat("Mean if we remove outliers:", round(m2, 2), "n")    }  outlierKD(bikedata, Temperature) #no outliers  outlierKD(bikedata, Atemperature) #no outliers  outlierKD(bikedata, Humidity) # no extreme outlier detected  outlierKD(bikedata, Windspeed) #some extreme values are present but canot be considered as outlier  outlierKD(bikedata, Casual) # no logical outliers  outlierKD(bikedata, Registered)# no ouliers  outlierKD(bikedata, Count)# no ouliers  #--------------------------------------------------------------------------------------------------#  # #  # Correlation Analysis #  # #  #--------------------------------------------------------------------------------------------------#  par(mfrow = c(1, 1))  numeric\_predictors <- unlist(lapply(bikedata, is.numeric))  numVarDataset <- bikedata[, numeric\_predictors]  corr <- cor(numVarDataset)  corrplot(  corr,  method = "color",  outline = TRUE,  cl.pos = 'n',  rect.col = "black",  tl.col = "indianred4",  addCoef.col = "black",  number.digits = 2,  number.cex = 0.60,  tl.cex = 0.70,  cl.cex = 1,  col = colorRampPalette(c("green4", "white", "red"))(100)  )  # Findings :  # 1. temp and atemp are highly correlated  # Looking at target variable  ggplot(data = bikedata, aes(Count)) +  geom\_histogram(aes(  y = ..density..,  binwidth = .10,  colour = "black"  ))  # Target variable looks like normal distribution  #--------------------------------------------------------------------------------------------------#  # #  # Univariate Analysis #  # #  #--------------------------------------------------------------------------------------------------#  # 1. Continous predictors  univariate\_continuous <- function(dataset, variable, variableName) {  var\_name = eval(substitute(variable), eval(dataset))  print(summary(var\_name))  ggplot(data = dataset, aes(var\_name)) +  geom\_histogram(aes(binwidth = .8, colour = "black")) +  labs(x = variableName) +  ggtitle(paste("count of", variableName))  }  univariate\_continuous(bikedata, Count, "Count")  univariate\_continuous(bikedata, Temperature, "Temperature")  univariate\_continuous(bikedata, Atemperature, "Atemperature")  univariate\_continuous(bikedata, Humidity, "Humidity") # skwed towards left  univariate\_continuous(bikedata, Windspeed, "Windspeed") #skewed towards right  univariate\_continuous(bikedata, Casual, "Casual") # skwed towards right  univariate\_continuous(bikedata, Registered, "Registered")  #2. categorical variables  univariate\_categorical <- function(dataset, variable, variableName) {  variable <- enquo(variable)    percentage <- dataset %>%  dplyr::select(!!variable) %>%  group\_by(!!variable) %>%  summarise(n = n()) %>%  mutate(percantage = (n / sum(n)) \* 100)  print(percentage)    dataset %>%  count(!!variable) %>%  ggplot(mapping = aes\_(  x = rlang::quo\_expr(variable),  y = quote(n),  fill = rlang::quo\_expr(variable)  )) +  geom\_bar(stat = 'identity',  colour = 'white') +  labs(x = variableName, y = "count") +  ggtitle(paste("count of ", variableName)) +  theme(legend.position = "bottom") -> p  plot(p)  }  univariate\_categorical(bikedata, Season, 'Season')  univariate\_categorical(bikedata, Year, "Year")  univariate\_categorical(bikedata, Month, "Month")  univariate\_categorical(bikedata, Holiday, "Holiday")  univariate\_categorical(bikedata, Weekday, "Weekday")  univariate\_categorical(bikedata, Workingday, "Workingday")  univariate\_categorical(bikedata, Weather, "Weather")  # ------------------------------------------------------------------------------------------------ #  #  # bivariate Analysis  #  #------------------------------------------------------------------------------------------------- #  # bivariate analysis for categorical variables  bivariate\_categorical <-  function(dataset, variable, targetVariable) {  variable <- enquo(variable)  targetVariable <- enquo(targetVariable)    ggplot(  data = dataset,  mapping = aes\_(  x = rlang::quo\_expr(variable),  y = rlang::quo\_expr(targetVariable),  fill = rlang::quo\_expr(variable)  )  ) +  geom\_boxplot() +  theme(legend.position = "bottom") -> p  plot(p)    }  bivariate\_continous <-  function(dataset, variable, targetVariable) {  variable <- enquo(variable)  targetVariable <- enquo(targetVariable)  ggplot(data = dataset,  mapping = aes\_(  x = rlang::quo\_expr(variable),  y = rlang::quo\_expr(targetVariable)  )) +  geom\_point() +  geom\_smooth() -> q  plot(q)    }  bivariate\_categorical(bikedata, Season, Count)  bivariate\_categorical(bikedata, Year, Count)  bivariate\_categorical(bikedata, Month, Count)  bivariate\_categorical(bikedata, Holiday, Count)  bivariate\_categorical(bikedata, Weekday, Count)  bivariate\_categorical(bikedata, Workingday, Count)  bivariate\_categorical(bikedata, Weather, Count)  bivariate\_continous(bikedata, Temperature, Count)  bivariate\_continous(bikedata, Atemperature, Count)  bivariate\_continous(bikedata, Humidity, Count)  bivariate\_continous(bikedata, Windspeed, Count)  bivariate\_continous(bikedata, Casual, Count)  bivariate\_continous(bikedata, Registered, Count)  # removing instant and dteday  bikedata$instant <- NULL  bikedata$Date <- NULL  bikedata$Casual <- NULL  bikedata$Registered <- NULL  # ------------------------------------------------------------------------------------------------ #  #  # Feature scaling or Normalization #  #  #------------------------------------------------------------------------------------------------- #  scaledData <- normalizeFeatures(bikedata,'Count')  # Function for calculating Mean Absolute Error  MAE <- function(actual,predicted){  error = actual - predicted  mean(abs(error))  }  # ----------------- Model 1 Linear Regression -----------------------------------------------------#  set.seed(654)  split <- sample.split(bikedata$Count, SplitRatio = 0.70)  training\_set <- subset(bikedata, split == TRUE)  test\_set <- subset(bikedata, split == FALSE)  model1 <- lm(Count ~ ., data = training\_set)  # step wise model selection  modelAIC <- stepAIC(model1, direction = "both")  summary(modelAIC)  # Apply prediction on test set  test\_prediction <- predict(modelAIC, newdata = test\_set)  test\_rmse <- rmse(test\_set$Count, test\_prediction)  print(paste("root-mean-square error for linear regression model is ", test\_rmse))  print(paste("Mean Absolute Error for linear regression model is ",MAE(test\_set$Count,test\_prediction)))  print("summary of predicted count values")  summary(test\_prediction)  print("summary of actual Count values")  summary(test\_set$Count)  # From the summary we can observe negative prediction values  #We will perform log transformation of trarget variable  model2 <- lm(log(Count)~., data = training\_set)  stepwiseLogAICModel <- stepAIC(model2,direction = "both")  test\_prediction\_log<- predict(stepwiseLogAICModel, newdata = test\_set)  predict\_test\_nonlog <- exp(test\_prediction\_log)  test\_rmse2 <- rmse(test\_set$Count, predict\_test\_nonlog)  print(paste("root-mean-square error between actual and predicted", test\_rmse))  print(paste("Mean Absolute Error for linear regression model is ",  MAE(test\_set$Count,predict\_test\_nonlog)))  summary(predict\_test\_nonlog)  summary(test\_set$Count)  par(mfrow = c(1,1))  plot(stepwiseLogAICModel)  # ----------------- Model 2 Random forest -----------------------------------------------------#  model1 <- randomForest(Count ~.,  data = training\_set,ntree = 500, mtry = 8, importance = TRUE)  print(model1)  par(mfrow = c(1,1))  plot(model1)  # 300 trees selected from the plot  tumedmodel <- tuneRF(training\_set[,1:11], training\_set[,12], stepFactor = 0.5, plot = TRUE,  ntreeTry = 250, trace = TRUE, improve = 0.05)  # selected mtry = 6 from the plot  tuned\_randomForest <- randomForest(Count ~. - Atemperature,  data = training\_set,ntree = 250, mtry = 6, importance = TRUE)  tuned\_randomForest  # predicting using random forest model 1  rf1\_prediction <- predict(tuned\_randomForest,test\_set[,-12])  rmse(rf1\_prediction,test\_set$Count)  print(paste("Mean Absolute Error for Random forest regressor is ",  MAE(test\_set$Count,rf1\_prediction)))  # Tuned Random Forest  varImpPlot(tuned\_randomForest)  # Random forest is performing better than linear regression.  # Model input and output for linear regression and Random forest  write.csv(test\_set, file = "InputLinearRegressionR.csv")  write.csv(test\_set, file = "InputRandomForestR.csv")  write.csv(predict\_test\_nonlog, file="outputLogisticRegressionR.csv") |