**Bike Rental Prediction**

By Gunjan Panda

14th July, 2019

**Contents**

**1. Introduction**  3

1.1 Problem Statement . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 3

1.2 Data . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 4

**2. Methodology**  5

2.1 Pre Processing . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 5

2.1.1 Missing Value Analysis . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 5

2.1.2 Outlier Analysis . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 7

2.1.3 Feature Selection . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 9

2.2 Modeling . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 11

2.2.1 Model Selection . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 11

2.2.2 Multiple Linear Regression . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 12

2.2.3 Regression Trees . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 13

**3. Conclusion**  16

3.1 Model Evaluation . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 16

3.1.1 R Square . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 16

3.1.2 Mean Absolute Percentile Error (MAPE) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 16

3.2 Model Selection . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 17

16. Appendix A - Extra Figures 19

17. Appendix B – R code 21

18. References 28

**1. Introduction**

**1.1 Problem Statement**

The objective of this case is to predict bike rental count on daily based on the environmental and seasonal settings. The aim of this project is to understand all patterns and to apply analytics for rental count prediction.

This problem statement lies in the category of forecasting which deals with predicting continuous values for future(in our case the continuous value is the bike rental count.) This is Time Series Forecasting as it deals with the timestamp variable and on the basis of this and other variables we are going to predict the future rental count based on different hours, seasons and environment.

Given below is a sample of the data set that we are using to predict the bike rental count:

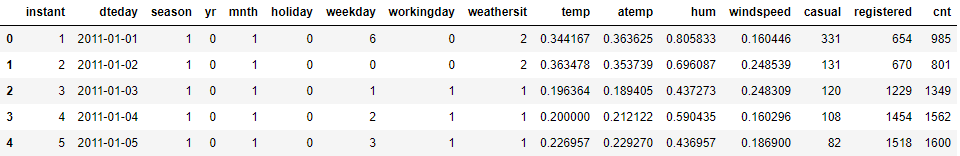


Figure 1.1(a)

As you can see, there are 15 predictor variables (i.e 15 independent variables) and 1 target variable (i.e dependent variable).

Predictors

1. instant: Record index
2. dteday: Date season: Season (1:spring, 2:summer, 3:fall, 4:winter)
3. yr: Year (0: 2011, 1:2012)
4. mnth: Month (1 to 12)
5. hr: Hour (0 to 23)
6. holiday: weather day is holiday or not
7. weekday: Day of the week
8. workingday: If day is neither weekend nor holiday is 1, otherwise is 0.
9. weathersit: 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
10. 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)
11. 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)
12. hum: Normalized humidity. The values are divided to 100 (max)
13. windspeed: Normalized wind speed. The values are divided to 67 (max)
14. casual: count of casual users
15. registered: count of registered users

Target

1. cnt: count of total rental bikes including both casual and registered

* Removed instant, holiday, casual, registered as all these variables are redundant variables and do not contribute to the target variable.

Data structure after proper data type conversion shown in figure 1.1(b):

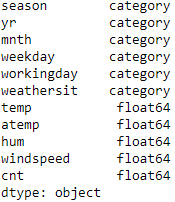


Figure 1.1(b)

**Methodology**

**2.1 Pre Processing**

To build any model, the first step is to understand the problem statement and choose the appropriate category that fits in. Since our statement fits into forecasting we use Regression. Regression is a Supervised Learning approach as we know the target variable beforehand which is count.

After identifying the approach, the next step is preprocessing the 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. Refer to Appendix A to see the plots.

To start this process we will first try and look at some of 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. In figure 2.1(a), figure 2.1(b) and figure 2.1(c) we have plotted the probability density functions of few variables available in the data as well as the dependent count variable. The distributions depict a mix of skewed and normally distributed data points. Skewed ones are indicating the presence of outliers.

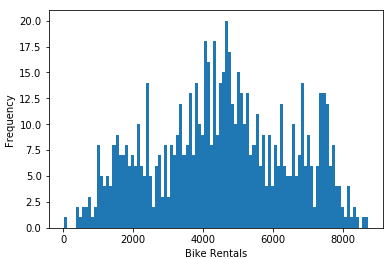


Figure 2.1(a)

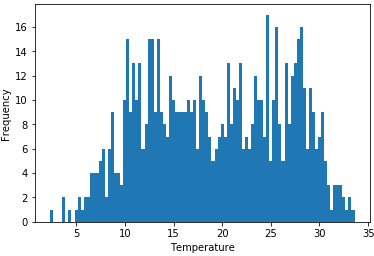


Figure 2.1(b)

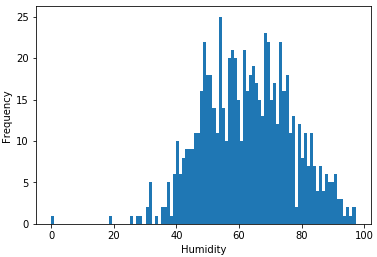


Figure 2.1(c)

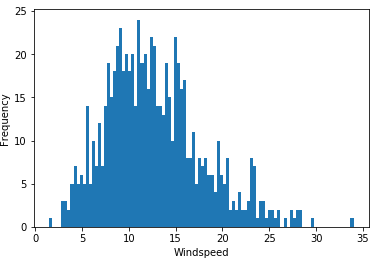


Figure 2.1(d)

**2.1.1 Missing Value Analysis**

Once proper data conversion is done next step is to analyze the missing values. According to our data, there are no missing values as you can see in figure 2.1.1(a)🡪

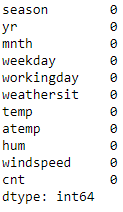


Figure 2.1.1(a)

**2.1.2 Outlier Analysis**

In this case we use a classic approach of removing outliers, Turkey’s method. We visualize the outliers using boxplots.

In figure 2.1.2(a) we have plotted the boxplots for the 4 predictor variables and in figure 2.1.2(b) we have plotted the target variable. A lot of useful inferences can be made from these plots. We have a lot of outliers and extreme values in each of the data set.

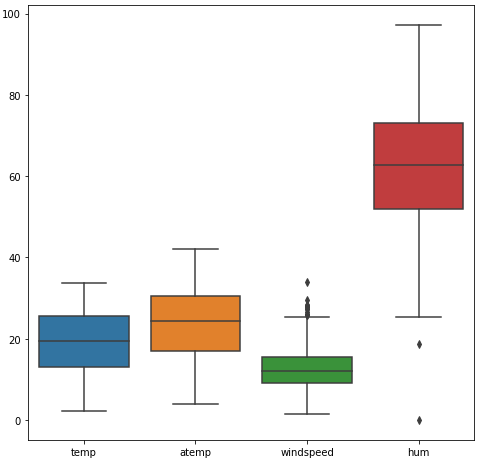


Figure 2.1.2(a)

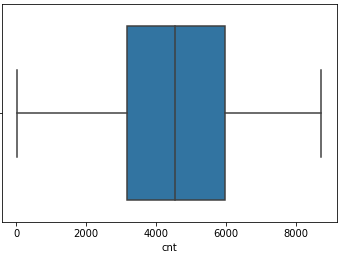


Figure 2.1.2(b)

After removing the outliers, our data is now clean and looks like shown in figure 2.1.2(c) and figure 2.1.2(d) shows the variables after removing outliers.

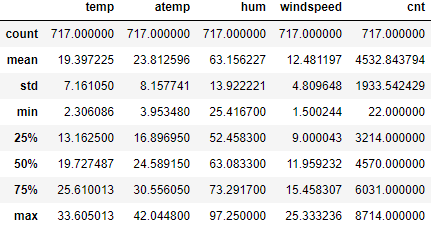


Figure 2.1.2(c)

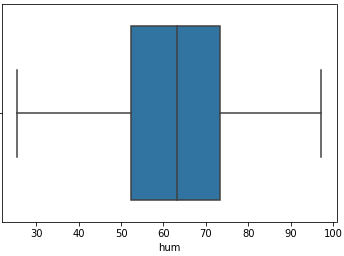
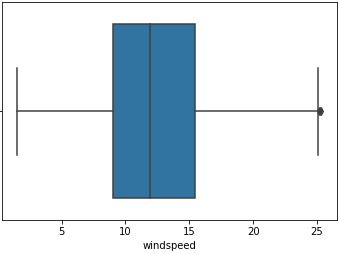


Figure 2.1.2(d)

**2.1.3 Feature Selection**

Since all our variables are numeric so we can extract the important features using the correlation matrix. As we can see from figure 2.1.3(a) variables temp and atemp are highly correlated with each other(r = 0.99), so we remove variable atemp and keep the rest of the features for model building.

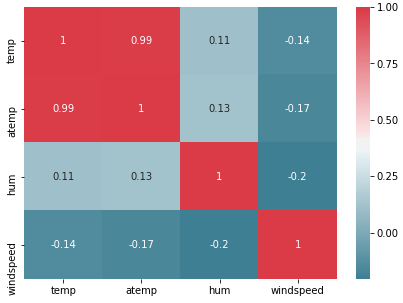
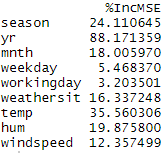


Figure 2.1.3(a)

> importance(model\_rf,type = 1)



We can see that yr has the highest prediction power for rental count whereas workingday and weekday have the least prediction power.

Also, removed variables hum and windspeed , as the distribution with target variable count had no pattern it was a random distribution. Therefore it didn’t contribute any information for the target variable. (Refer Appendix A)

**2.2 Modeling**

**2.2.1 Model Selection**

In our early stages of analysis during pre-processing we have come to understand that count is dependent on multiple aspects. Therefore, it’s important to build a model in such a way that it takes in all the required inputs and fits the model in such a way that it gives us the most accurate result amongst all the other models.

The dependent variable can fall in any of the four categories:

1. Nominal

2. Ordinal

3. Interval

4. Ratio

The dependent variable, in our case “count”, is Ratio so the only predictive analysis that we can perform is a Regression analysis.

We always start our model building from the simplest to more complex. Therefore we use Multiple Linear Regression at first.

1. **Multiple Linear Regression**

As this is a Regression problem and we have multiple variables as our parameters so the first model to be applied is Multiple Linear Regression model. We use the OLS(Ordinary Least Squares) model to fit the model and evaluate its performance.

The summary report gives some valuable information regarding the model built.

1. **Summary report in R**

Call:

lm(formula = cnt ~ ., data = train)

Residuals:

Min 1Q Median 3Q Max

-3884.3 -351.5 49.0 456.2 2950.4

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) 1563.61 274.41 5.698 1.99e-08 \*\*\*

season2 1029.15 209.97 4.901 1.26e-06 \*\*\*

season3 823.35 245.29 3.357 0.000844 \*\*\*

season4 1729.30 208.87 8.279 9.61e-16 \*\*\*

yr1 2005.14 67.14 29.864 < 2e-16 \*\*\*

mnth2 154.89 168.49 0.919 0.358372

mnth3 540.23 188.36 2.868 0.004288 \*\*

mnth4 327.36 286.26 1.144 0.253312

mnth5 563.99 305.79 1.844 0.065674 .

mnth6 390.90 321.35 1.216 0.224349

mnth7 -52.43 353.85 -0.148 0.882257

mnth8 423.80 343.58 1.234 0.217918

mnth9 999.37 299.10 3.341 0.000891 \*\*\*

mnth10 314.54 272.24 1.155 0.248434

mnth11 -230.87 264.03 -0.874 0.382281

mnth12 -222.52 211.79 -1.051 0.293869

weekday1 -566.47 209.70 -2.701 0.007121 \*\*

weekday2 -409.71 232.43 -1.763 0.078503 .

weekday3 -370.15 231.42 -1.599 0.110291

weekday4 -272.28 230.85 -1.179 0.238732

weekday5 -265.46 229.10 -1.159 0.247088

weekday6 454.21 124.46 3.650 0.000288 \*\*\*

workingday1 722.73 197.15 3.666 0.000271 \*\*\*

weathersit2 -488.58 90.13 -5.421 8.92e-08 \*\*\*

weathersit3 -1950.92 244.05 -7.994 7.86e-15 \*\*\*

temp 4487.17 471.48 9.517 < 2e-16 \*\*\*

hum -1581.20 351.78 -4.495 8.51e-06 \*\*\*

windspeed -2719.14 489.50 -5.555 4.35e-08 \*\*\*

---

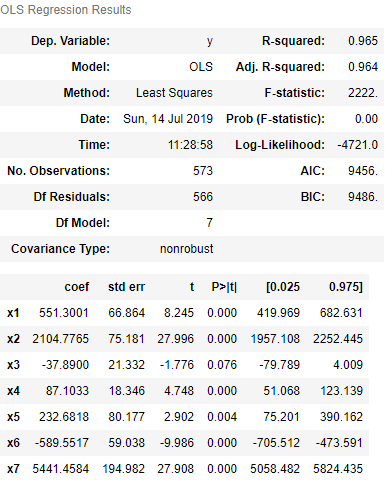
Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

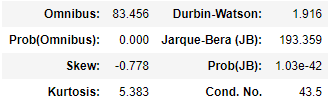
Residual standard error: 778.6 on 545 degrees of freedom

Multiple R-squared: 0.8439, Adjusted R-squared: 0.8362

F-statistic: 109.1 on 27 and 545 DF, p-value: < 2.2e-16

1. **Summary report in Python**





The summary provides several measures to give you an idea of the data distribution and behavior. From here we can see if the data has the correct characteristics to give us confidence in the resulting model. We aren't testing the data, we are just looking at the model's interpretation of the data. If the data is good for modeling, then our residuals will have certain characteristics.

* **Omnibus**– a test of the skewness and kurtosis of the residual.
* **Prob(Omnibus)-** performs a statistical test indicating the probability that the residuals are normally distributed.
* **Skew**– a measure of data symmetry.
* **Kurtosis**- Greater Kurtosis can be interpreted as a tighter clustering of residuals around zero, implying a better model with few outliers.
* **Durbin-Watson**– tests for homoscedasticity (characteristic #3). We hope to have a value between 1 and 2.
* **Jarque-Bera (JB)/Prob(JB)**– like the Omnibus test in that it tests both skew and kurtosis. We hope to see in this test a confirmation of the Omnibus test.
* **Condition Number**– This test measures the sensitivity of a function's output as compared to its input (characteristic #4).
* **R Squared Value**- It tells how well the model fits in the data and also explains the variance of the target variable.

1. **Decision Tree**

Decision trees are non linear. Unlike Linear regression there is no equation to express relationship between independent and dependent variables. So for a better result the next model that we can choose is Decision Tree Regression.

A decision tree is a tree-like graph with nodes representing the place where we pick an attribute and ask a question; edges represent the answers to the question; and the leaves represent the actual output or class label.

Decision Tree algorithms are referred to as CART or Classification and Regression Trees.

maxDepth : 5 larger the dataset harder to visualize so we have taken the maximum branching to be 5 as of now.

**train,test = train\_test\_split(bike\_df,test\_size = 0.2,random\_state=0)**

**fit = DecisionTreeRegressor(max\_depth=5).fit(train.iloc[:,:-1],train.iloc[:,-1])**

1. **Random Forest**

Random forest is a tree-based algorithm which involves building several trees (decision trees), then combining their output to improve generalization ability of the model. The method of combining trees is known as an ensemble method. Ensembling is nothing but a combination of weak learners (individual trees) to produce a strong learner.

Random Forest can be used to solve regression and classification problems. In regression problems, the dependent variable is continuous. In classification problems, the dependent variable is categorical.

**RandomForestRegressor(n\_estimators=500, n\_jobs = -1,random\_state = 0,max\_features = "auto").fit(train.iloc[:,:-1], train.iloc[:,-1])**

n\_estimators : The number of trees to be used in the forest.

n\_jobs : The number of jobs to run in parallel for both [**fit**](https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html#sklearn.ensemble.RandomForestRegressor.fit) and [**predict**](https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html#sklearn.ensemble.RandomForestRegressor.predict)

**Conclusion**

**3.1 Model Evaluation**

The quality of a regression model is how well its predictions match up against actual values, but how do we actually evaluate quality? Luckily, smart statisticians have developed **error metrics** to judge the quality of a model and enable us to compare regressions against other regressions with different parameters.

As our model deals with regression so we will choose amongst those error metrics that are used for regression.

If our collection of residuals is small, it implies that the model that produced them does a good job at predicting our output of interest. Conversely, if these residuals are generally large, it implies that model is a poor estimator.

Thus, statisticians have developed summary measurements that take our collection of residuals and condense them into a single value that represents the predictive ability of our model.

**For our problem we have used MAPE and R squared as we are dealing with forecasting.**

1. The mean absolute percentage error (MAPE) is a statistical measure of how accurate a forecast system is. It measures this accuracy as a percentage, and can be calculated as the average absolute percent error for each time period minus actual values divided by actual values. Where At is the actual value and Ft is the forecast value, this is given by:

[https://www.statisticshowto.datasciencecentral.com/wp-content/uploads/2017/09/mape.jpeg](https://www.statisticshowto.datasciencecentral.com/wp-content/uploads/2017/09/mape.jpeg)  
  
The mean absolute percentage error (MAPE) is the most common measure used to forecast error, and works best if there are no extremes to the data.

1. To understand how well the independent variables “explain” the variance in our model, the R-Squared formula is used.

For the R-Squared, the closer the value to 1, the better our model is performing.

According to our model, the following table describes its error metrics:

|  |  |  |
| --- | --- | --- |
| Model | MAPE Score | R Square |
| Multiple Linear Regression | 18.97%  Accuracy : 81.03% | 0.965 (Overfit)  Actual r2 : 0.8348 |
| Decision Tree | 18.17%  Accuracy : 81.83% | 0.8611 |
| Random Forest | 16.00%  Accuracy : 84% | 0.8873 |

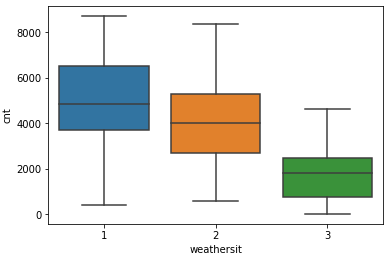
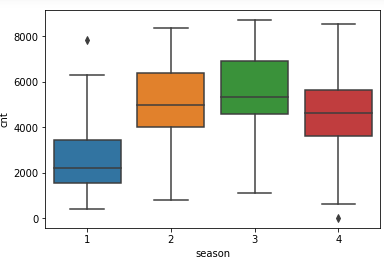
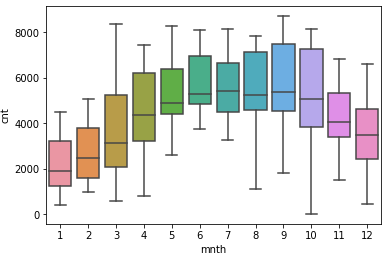
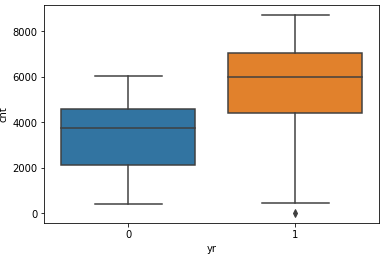
Amongst all the models, Multiple Linear Regression could have been the best model but its not as the R-square is due to overfitting of parameters which can fail in capturing real patterns. This could be happening because of less number of observations.

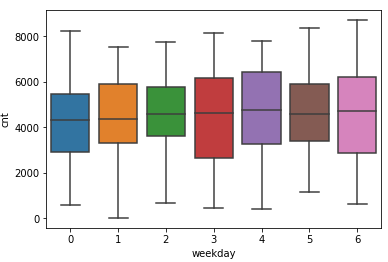
So the best model is Random Forest as it has the highest R-square as well as accuracy.

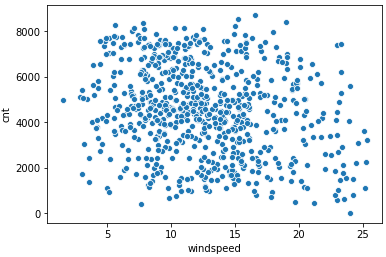
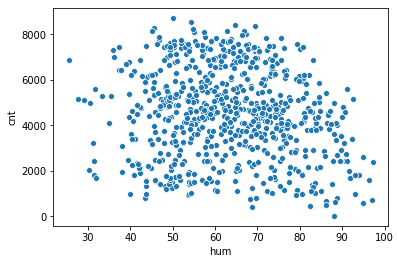
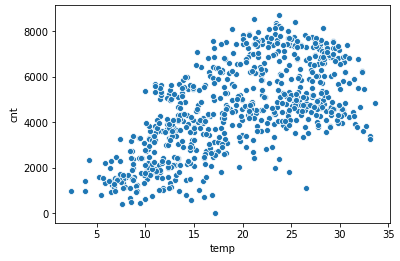
**3.2 Model Selection**

We select Random Forest as our final model because of highest accuracy score and highest R square.

**Appendix A - Extra Figures**







**Appendix B - R Code**

rm(list = ls())

setwd('C:/Users/admin/Desktop')

#Read Data

bike\_df = read.csv('day.csv',header = T)

x = c("ggplot2","corrgram","DMwR","caret","randomForest","unbalanced",

"C50","dummies","e1071","Information","MASS","rpart","gbm","ROSE","lubridate"

,"dplyr","rsq","usdm","gplots","scales","psych")

lapply(x,require,character.only = TRUE)

rm(x)

#Understand the structure of data

str(bike\_df)

#Removing unnecessary variables

bike\_df$instant = NULL

bike\_df$dteday = NULL

bike\_df$holiday = NULL

bike\_df$casual = NULL

bike\_df$registered = NULL

#Understanding the distribution of values

table(bike\_df$season)

table(bike\_df$yr)

table(bike\_df$mnth)

table(bike\_df$weekday)

table(bike\_df$workingday)

table(bike\_df$weathersit)

#Data type conversion

bike\_df$season = as.factor(bike\_df$season)

bike\_df$yr = as.factor(bike\_df$yr)

bike\_df$mnth = as.factor(bike\_df$mnth)

bike\_df$weekday = as.factor(bike\_df$weekday)

bike\_df$workingday = as.factor(bike\_df$workingday)

bike\_df$weathersit = as.factor(bike\_df$weathersit)

bike\_df$cnt = as.numeric(bike\_df$cnt)

#Missing Value Analysis : 0 missing values

colSums(is.na(bike\_df))

#Outlier Analysis

boxplot(bike\_df$temp,bike\_df$atemp,bike\_df$hum,bike\_df$windspeed)

boxplot(bike\_df$cnt)

#Remove outliers present in hum and windspeed

numeric\_index = sapply(bike\_df, is.numeric)

numeric\_data = bike\_df[,numeric\_index]

cnames = colnames(numeric\_data)

for (i in cnames){

print(i)

val = bike\_df[,i][bike\_df[,i]%in%boxplot.stats(bike\_df[,i])$out]

print(length(val))

bike\_df = bike\_df[which(!bike\_df[,i]%in%val),]

}

dim(bike\_df)

summary(bike\_df)

#Feature Selection : removing atemp as high correlation with temp~0.99

corrgram(bike\_df[,numeric\_index],order = F,

upper.panel = panel.pie,text.panel = panel.txt, main = "Correlation Plot")

bike\_df$atemp = NULL

#Distribution of data

# Boxplot of rental count by year

boxplot(cnt~yr,data=bike\_df, main="Bike Rental",

xlab="Years", ylab="Rental Count")

# Boxplot of rental count by season

boxplot(cnt~season,data=bike\_df, main="Bike Rental",

xlab="Season", ylab="Rental Count")

# Boxplot of rental count by month

boxplot(cnt~mnth,data=bike\_df, main="Bike Rental",

xlab="Months", ylab="Rental Count")

# Boxplot of rental count by weather criteria

boxplot(cnt~weathersit,data=bike\_df, main="Bike Rental",

xlab="Weather", ylab="Rental Count")

# Boxplot of rental count by weekday

boxplot(cnt~weekday,data=bike\_df, main="Bike Rental",

xlab="Weekday", ylab="Rental Count")

#Plot b/w temperature and count

ggplot(bike\_df,aes\_string(x=bike\_df$temp,y=bike\_df$cnt))+

geom\_point(inherit.aes = TRUE,size=3)+

theme\_bw()+ylab("Rental Count")+xlab("Temperature")+ggtitle("Scatter Plot b/w Temperature and Rental Count")+

theme(text=element\_text(size = 15))+

scale\_x\_continuous(breaks = pretty\_breaks(10))+

scale\_y\_continuous(breaks = pretty\_breaks(10))

#Plot b/w humidity and count

ggplot(bike\_df,aes\_string(x=bike\_df$hum,y=bike\_df$cnt))+

geom\_point(inherit.aes = TRUE,size=3)+

theme\_bw()+ylab("Rental Count")+xlab("Humidity")+ggtitle("Scatter Plot b/w Humidity and Rental Count")+

theme(text=element\_text(size = 15))+

scale\_x\_continuous(breaks = pretty\_breaks(10))+

scale\_y\_continuous(breaks = pretty\_breaks(10))

#Plot b/w windspeed and count

ggplot(bike\_df,aes\_string(x=bike\_df$windspeed,y=bike\_df$cnt))+

geom\_point(inherit.aes = TRUE,size=3)+

theme\_bw()+ylab("Rental Count")+xlab("Windspeed")+ggtitle("Scatter Plot b/w Windspeed and Rental Count")+

theme(text=element\_text(size = 15))+

scale\_x\_continuous(breaks = pretty\_breaks(10))+

scale\_y\_continuous(breaks = pretty\_breaks(10))

#As distribution with count variable is randomly scattered, so doesn't capture

#any pattern

bike\_df$hum = NULL

bike\_df$windspeed = NULL

#Model Building

#Normality check

vif(bike\_df[,])

vifcor(bike\_df[,7:8],th = 0.9)

#Random Sampling

set.seed(1234)

train\_index = sample(1:nrow(bike\_df),0.8\*nrow(bike\_df))

train = bike\_df[train\_index,]

test = bike\_df[-train\_index,]

#Multiple Linear Regression

lm\_model = lm(cnt~.,data = train)

summary(lm\_model)

predictions\_lr = predict(lm\_model,test[,1:7])

mape = function(actual,predicted){

mean(abs((actual - predicted)/actual))

}

mape(test[,8],predictions\_lr) \* 100

#R sq : 84.39; Accuracy : 100 - 19.44 = 80.56%

#error rate mape: 19.44%

#Decision Tree

dt\_model = rpart(cnt~.,data = train,method = "anova")

predictions\_dt = predict(dt\_model,test[,1:7])

mape(test[,8],predictions\_dt)\*100

#Error rate mape: 23.82%, Accuracy : 100-23.82 = 76.18%

#Random Forest

model\_rf = randomForest(cnt~.,

train,importance = TRUE, ntree = 300)

RF\_predictions = predict(model\_rf,test[,1:7])

mape(test[,8],RF\_predictions)

#Error rate mape: 18.97% , Accuracy : 100-18.97 = 81.03%

#Rsquare : 83.9%

**References**

James, Gareth, Daniela Witten, Trevor Hastie, and Robert Tibshirani. 2013. An Introduction to Statistical Learning. Vol. 6. Springer. Wickham, Hadley. 2009. Ggplot2: Elegant Graphics for Data Analysis. Springer Science & Business Media.