**Predication of bike rental count**

***Piyusha Varshini T***

Table of Contents

[Introduction 3](#_Toc527659929)

[1.1 Problem Statement 3](#_Toc527659930)

[1.2 Data 3](#_Toc527659931)

[Methodology 5](#_Toc527659932)

[2.1 Pre Processing 5](#_Toc527659933)

[2.1.1 Outlier Analysis 6](#_Toc527659934)

[2.1.2 Feature Selection 14](#_Toc527659935)

[2.2 Modeling 18](#_Toc527659936)

[2.2.1 Model Selection 18](#_Toc527659937)

[2.2.2 Linear Regression 18](#_Toc527659938)

[2.2.3 Desicison Tree Regression 19](#_Toc527659939)

[Conclusion 20](#_Toc527659940)

[3.1 Model Evaluation 20](#_Toc527659941)

[3.1.1 Mean Absolute Percentage Error (MAPE) 20](#_Toc527659942)

[3.2 Model Selection 21](#_Toc527659943)

[3.3 R code of the Entire model: 21](#_Toc527659944)

[3.4 Python code 25](#_Toc527659945)

**Chapter 1**

# Introduction

## 1.1 Problem Statement

The aim of this project is predict the bike rental count on a particular time along with season, weather setting, and temperature. The advantage of predicting bike rental count will be scope to the management to maintain exact number of bikes according to the seasons weather conditions without losing the customers lack of bikes.

## 1.2 Data

Our task is to build the regression model upon the training data and verify using the test data. Given below is the sample of data.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| | Bike Rental Count (Columns: 1-10) | | | | | | | | | | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | **instant** | **dteday** | **season** | **yr** | **mnth** | **holiday** | **weekday** | **workingday** | **weathersit** | **temp** | | 1 | 2011-01-01 | 1 | 0 | 1 | 0 | 6 | 0 | 2 | 0.344167 | | 2 | 2011-01-02 | 1 | 0 | 1 | 0 | 0 | 0 | 2 | 0.363478 | | 3 | 2011-01-03 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 0.196364 | | 4 | 2011-01-04 | 1 | 0 | 1 | 0 | 2 | 1 | 1 | 0.200000 | | 5 | 2011-01-05 | 1 | 0 | 1 | 0 | 3 | 1 | 1 | 0.226957 | | 6 | 2011-01-06 | 1 | 0 | 1 | 0 | 4 | 1 | 1 | 0.204348 | |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| | Bike Rental Count (Columns: 11-16) | | | | | | | --- | --- | --- | --- | --- | --- | | **atemp** | **hum** | **windspeed** | **casual** | **registered** | **cnt** | | 0.363625 | 0.805833 | 0.1604460 | 331 | 654 | 985 | | 0.353739 | 0.696087 | 0.2485390 | 131 | 670 | 801 | | 0.189405 | 0.437273 | 0.2483090 | 120 | 1229 | 1349 | | 0.212122 | 0.590435 | 0.1602960 | 108 | 1454 | 1562 | | 0.229270 | 0.436957 | 0.1869000 | 82 | 1518 | 1600 | | 0.233209 | 0.518261 | 0.0895652 | 88 | 1518 | 1606 | |

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

| Predictor Variables | |
| --- | --- |
| **S.No.** | **Predictor** |
| 1 | instant |
| 2 | dteday |
| 3 | season |
| 4 | yr |
| 5 | mnth |
| 6 | holiday |
| 7 | weekday |
| 8 | workingday |
| 9 | weathersit |
| 10 | temp |
| 11 | atemp |
| 12 | hum |
| 13 | windspeed |
| 14 | casual |
| 15 | registered |

**Chapter 2**

# Methodology

## 2.1 Pre Processing

Any predictive modeling requires that we look at the data before we start modeling. However, in data mining terms *looking at data* refers to so much more than just looking. 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**. To start this process we will first try and 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.

Converting data into required format

*#converting the data into required format*

*day$season=as.factor(as.character(day$season))*

*day$yr=as.factor(as.character(day$yr))*

*day$holiday=as.factor(as.character(day$holiday))*

*day$workingday=as.factor(as.character(day$workingday))*

*day$mnth=as.factor(as.character(day$mnth))*

*day$weekday=as.factor(as.character(day$weekday))*

*day$weathersit=as.factor(as.character(day$weathersit))*

**Missing value Anaysis**

Checking data whether there are any missing values in the data.

*#missing value analysis*

*missing\_val=data.frame(apply(day,2,function(x){sum(is.na(x))}))*

*missing\_val*

*###no missing values found in the data*

Output:

apply.day..2..function.x...

instant 0

dteday 0

season 0

yr 0

mnth 0

holiday 0

weekday 0

workingday 0

weathersit 0

temp 0

atemp 0

hum 0

windspeed 0

casual 0

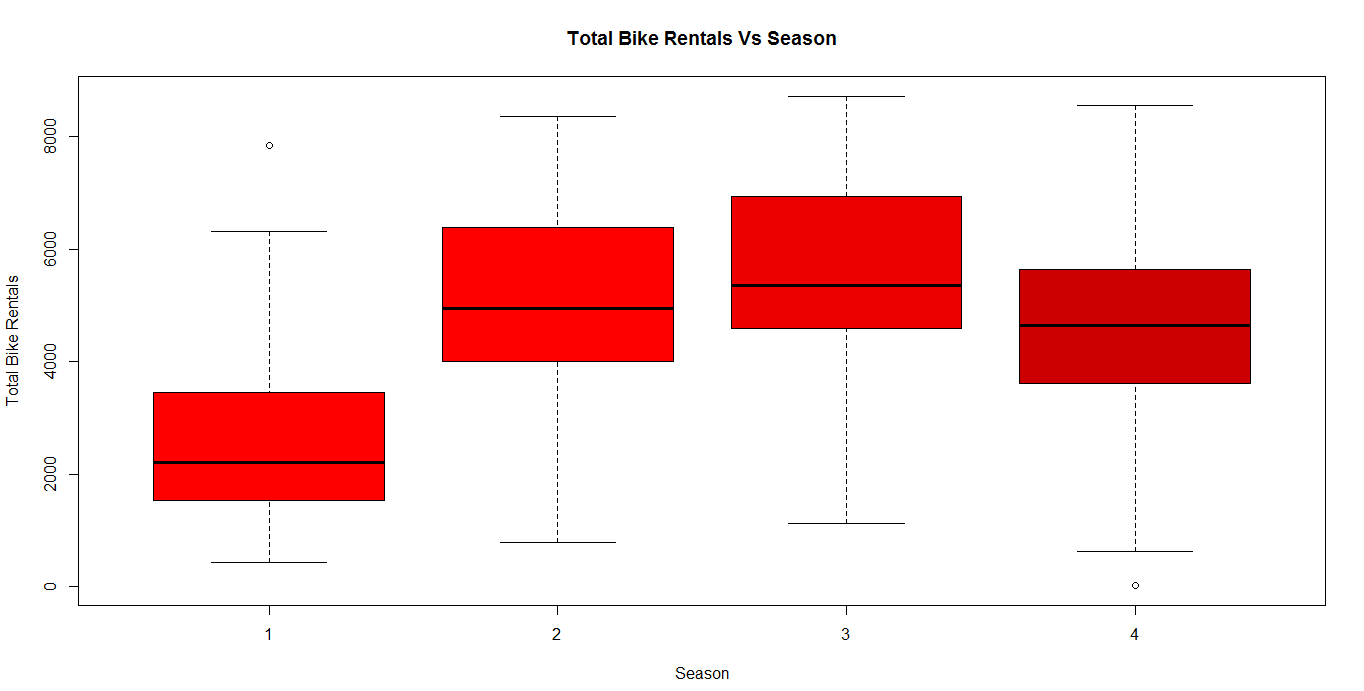
registered 0

cnt 0

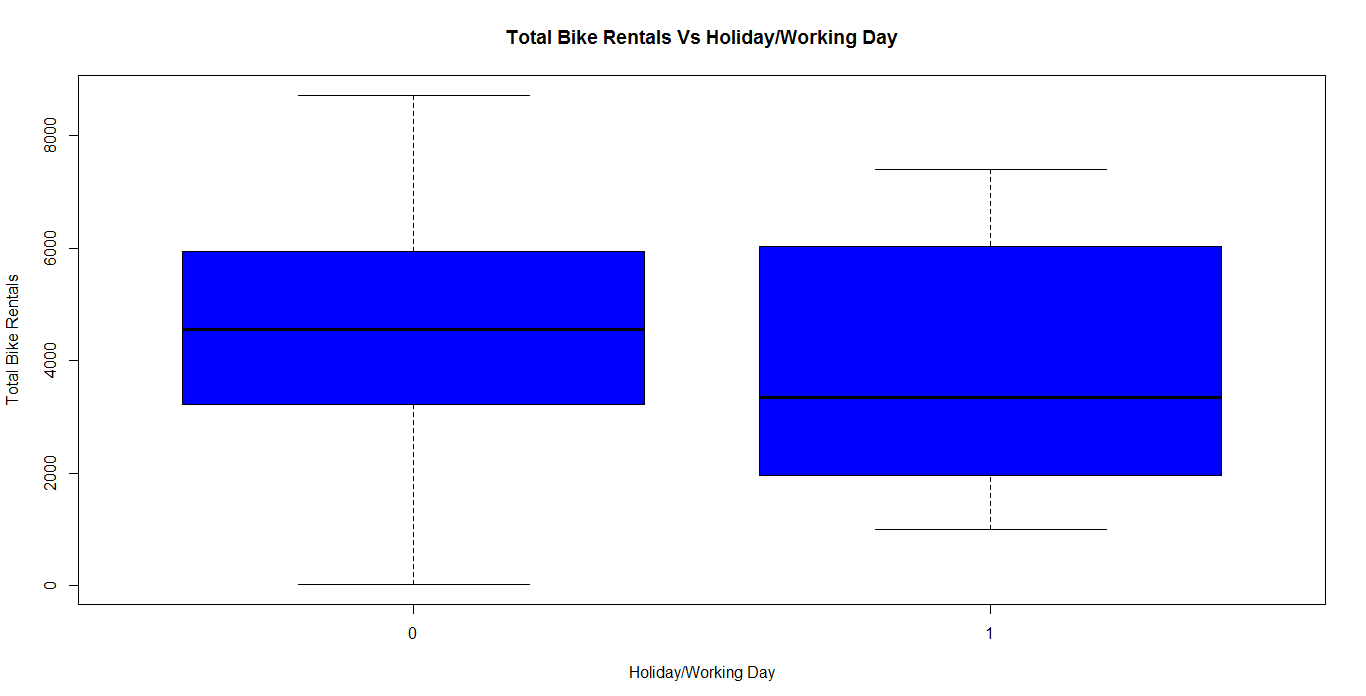
### 2.1.1 Outlier Analysis

Outliers are those data points which are away from the normal data. Outliers cause a skewness in the data and make model inconsistent. There is a need to remove outliers in the Data. Outliers are identified using Boxplots and are removed from the data.

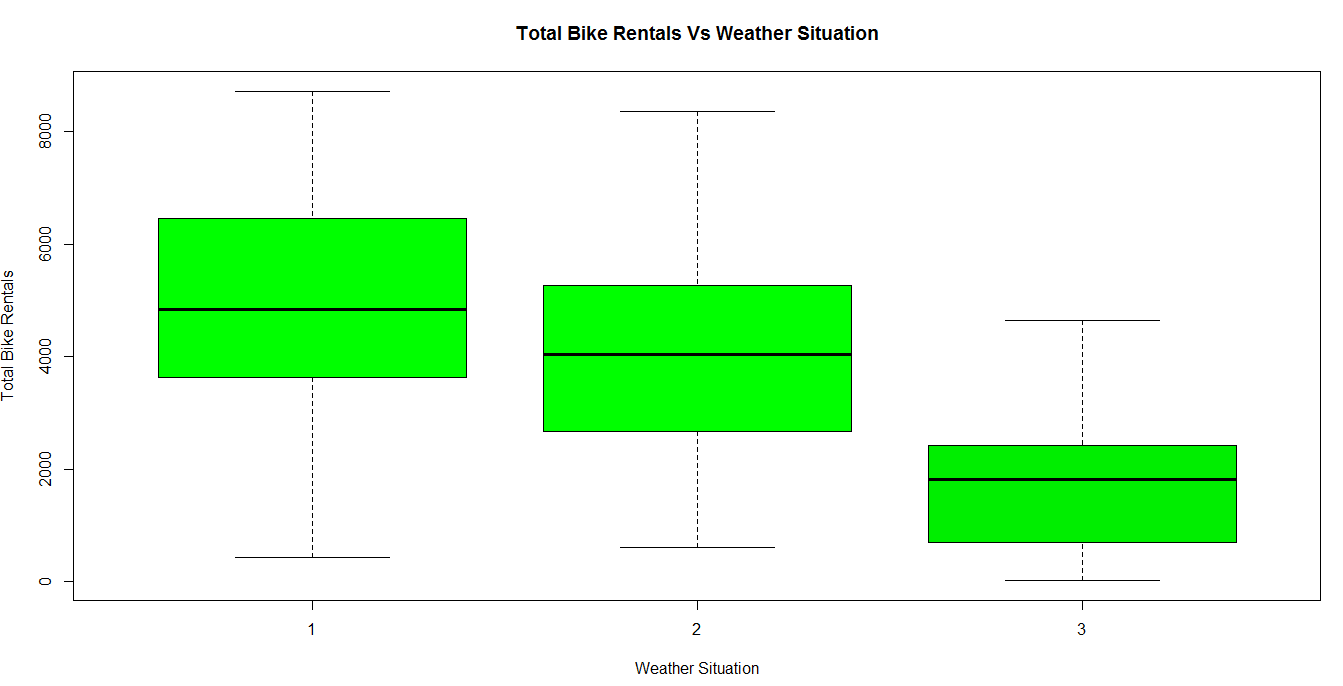
Boxplot of season and bike rentals:



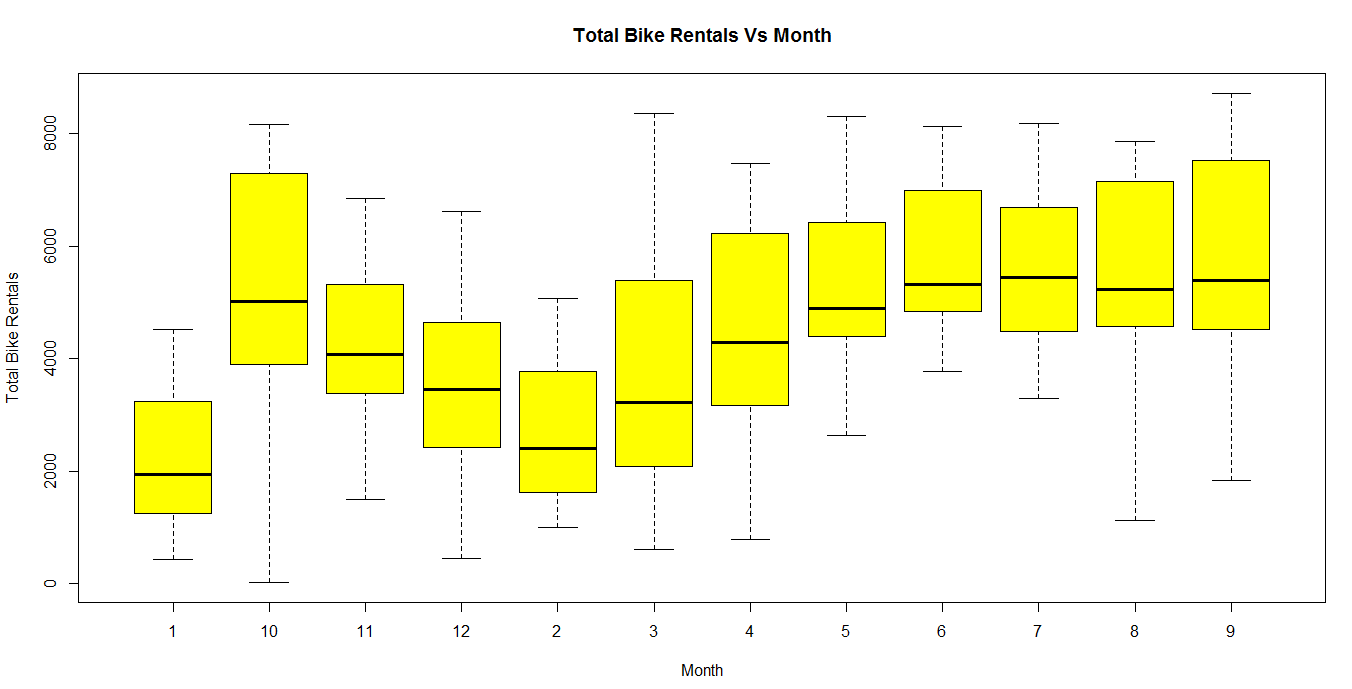
BoxPlot Bikerentals vs Workingday:



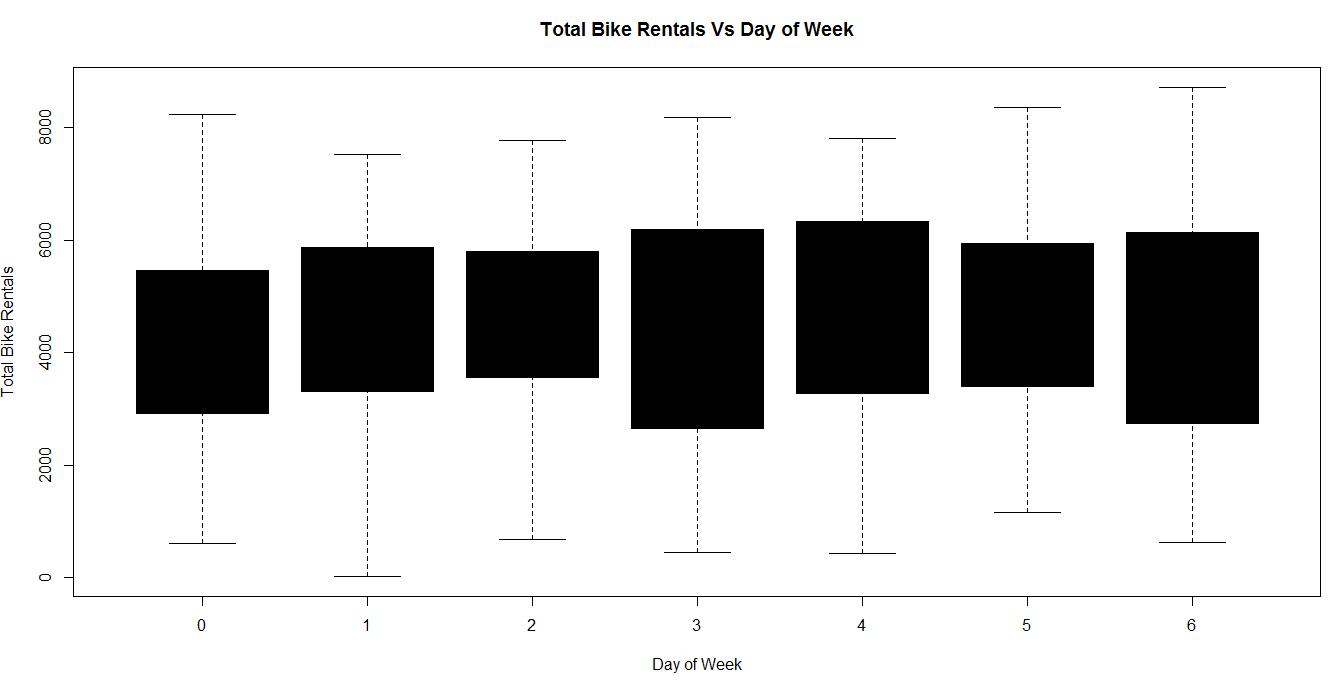
Boxplot for Total bike rental vs weathersit :



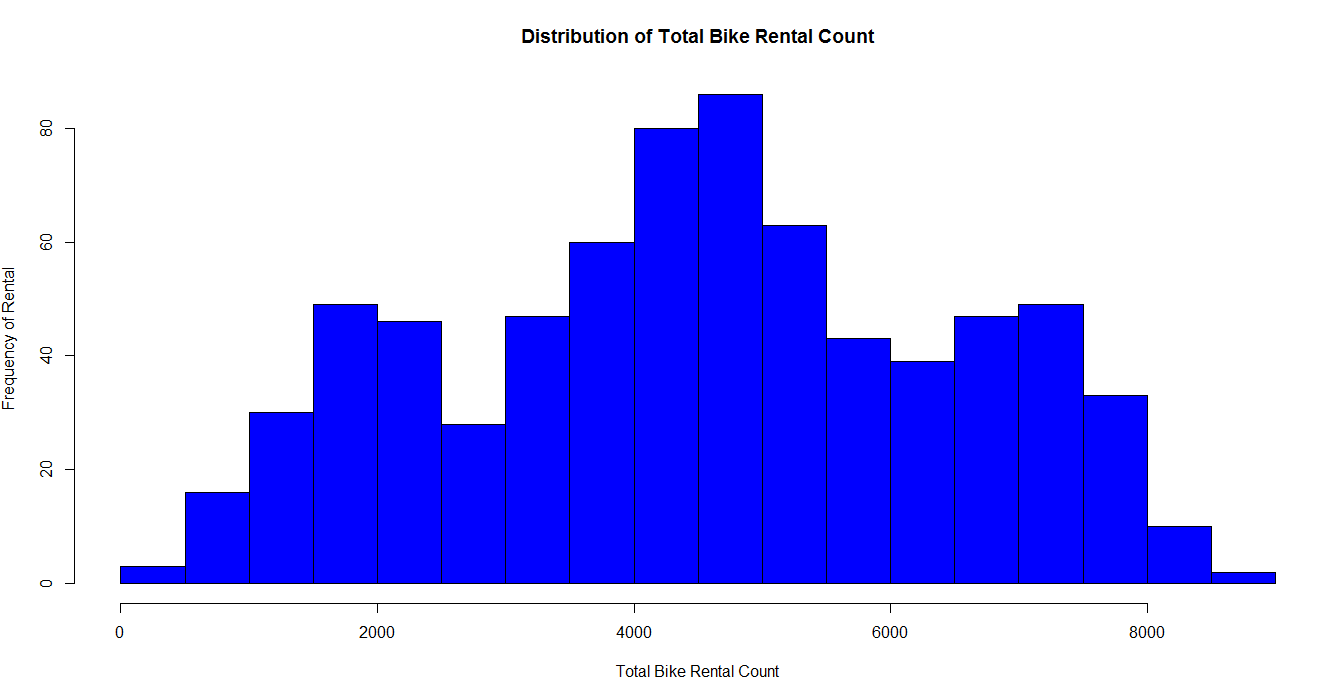
Boxplot for Bike Rental vs Month:

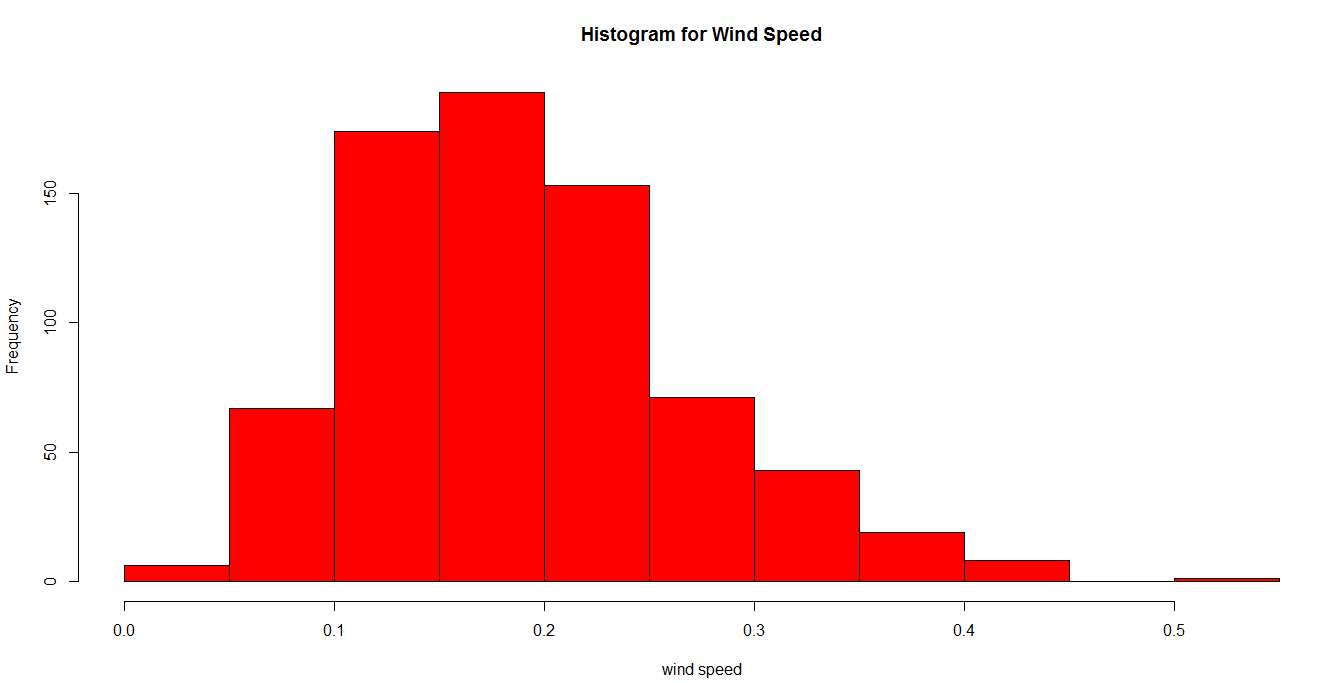


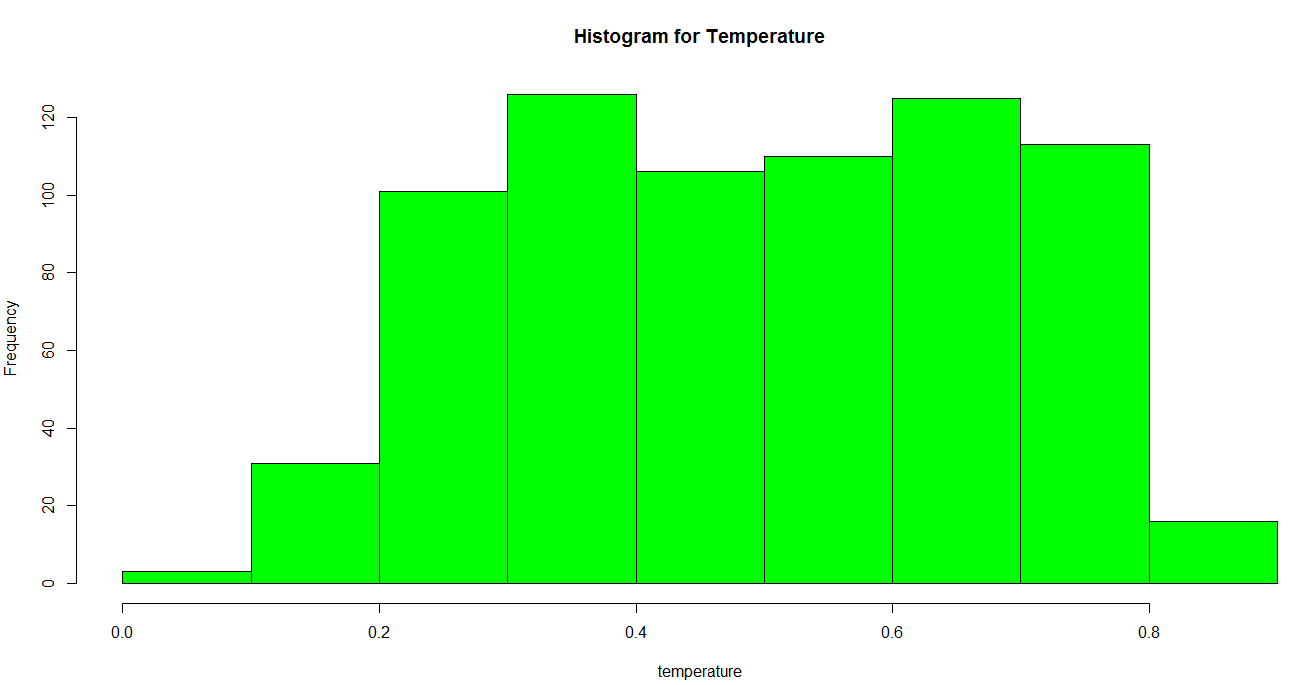
Boxplot for BikeRental vs weekday:

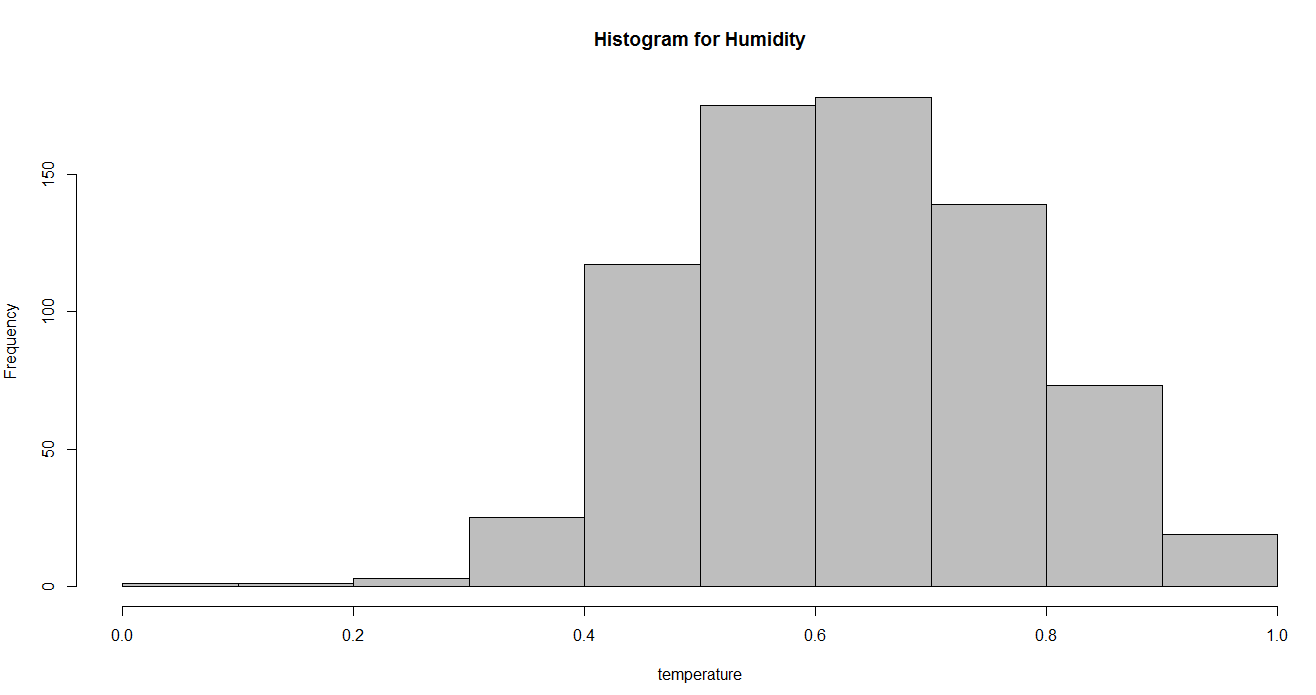


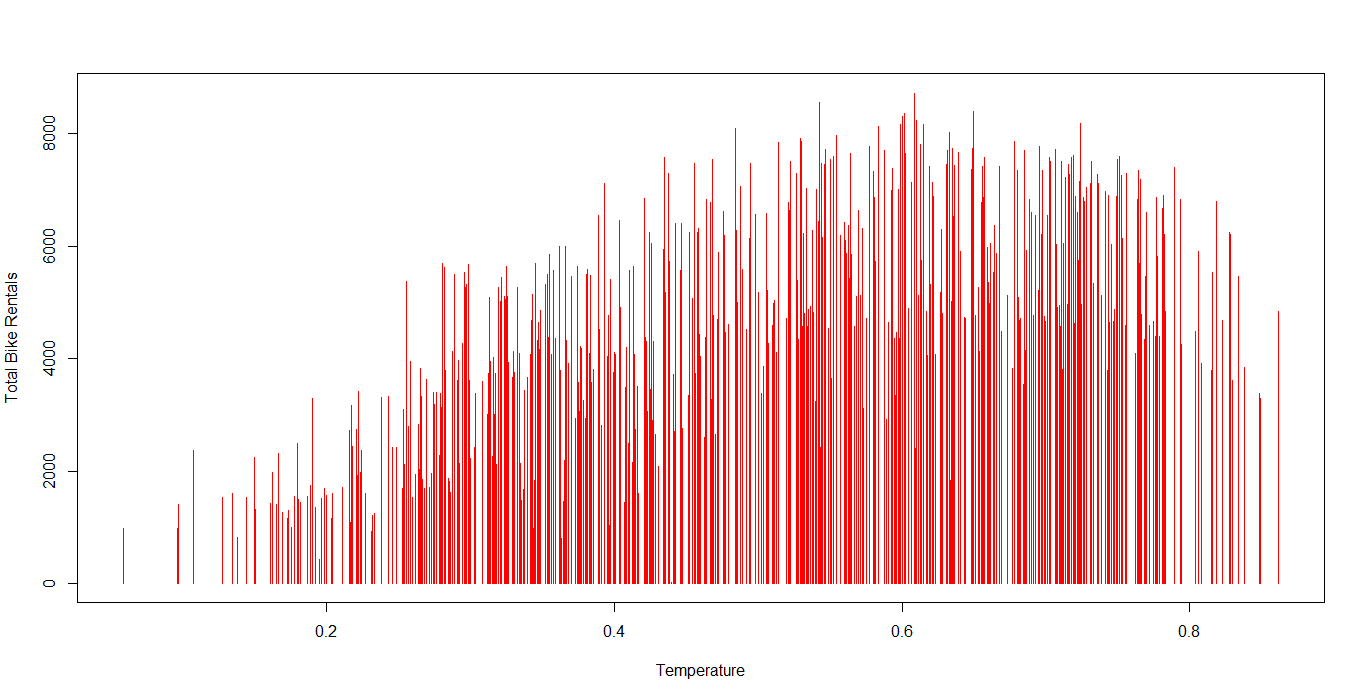
The following figures explain Data distribution of contiuous variables using histograms

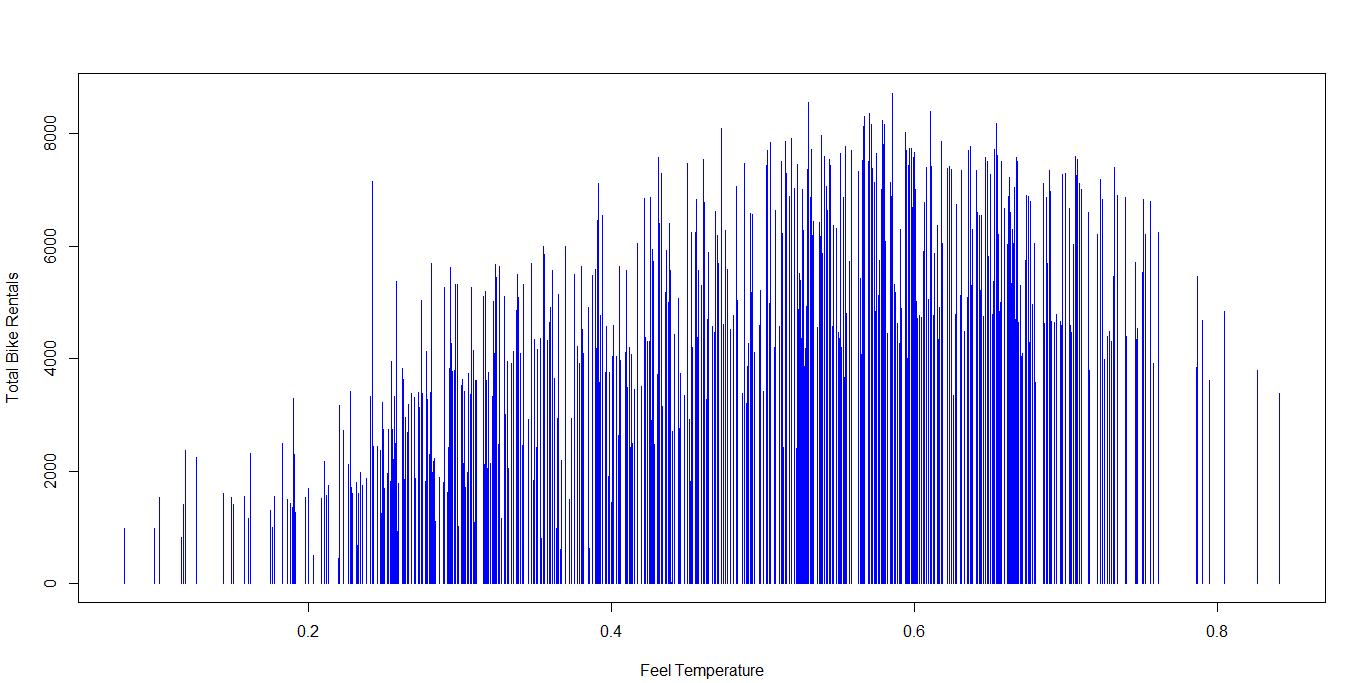


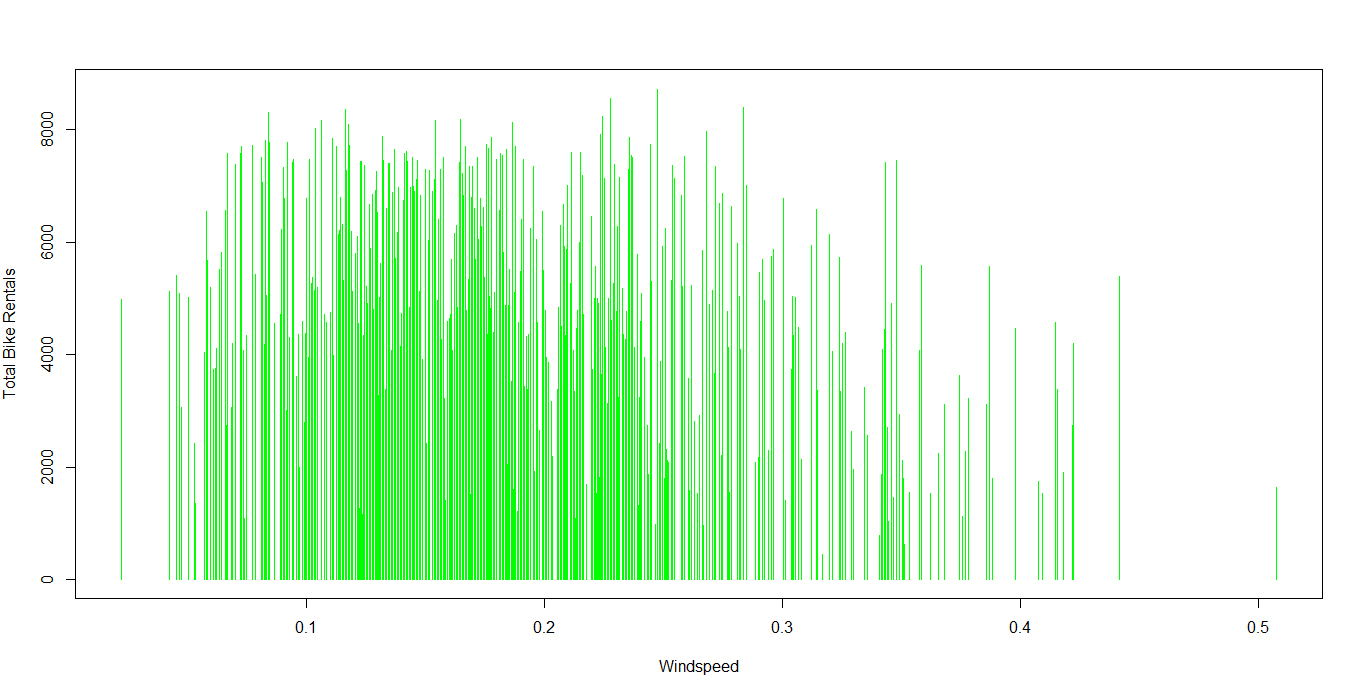


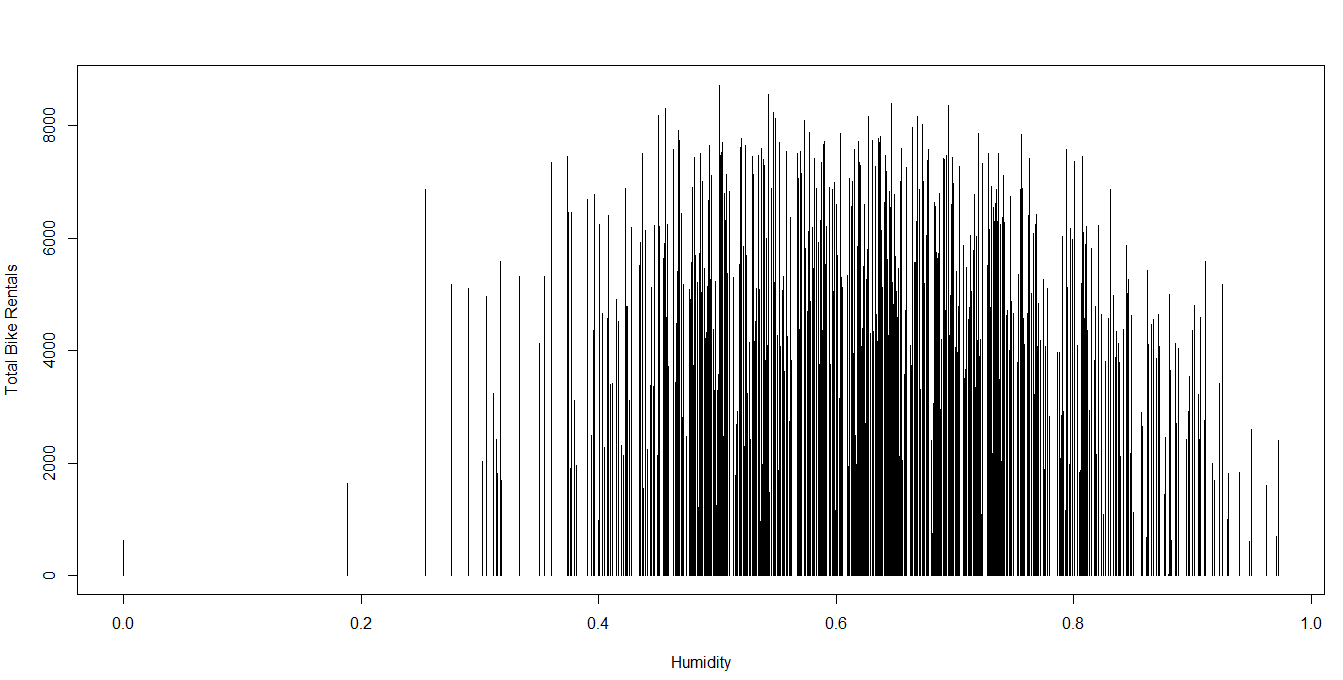


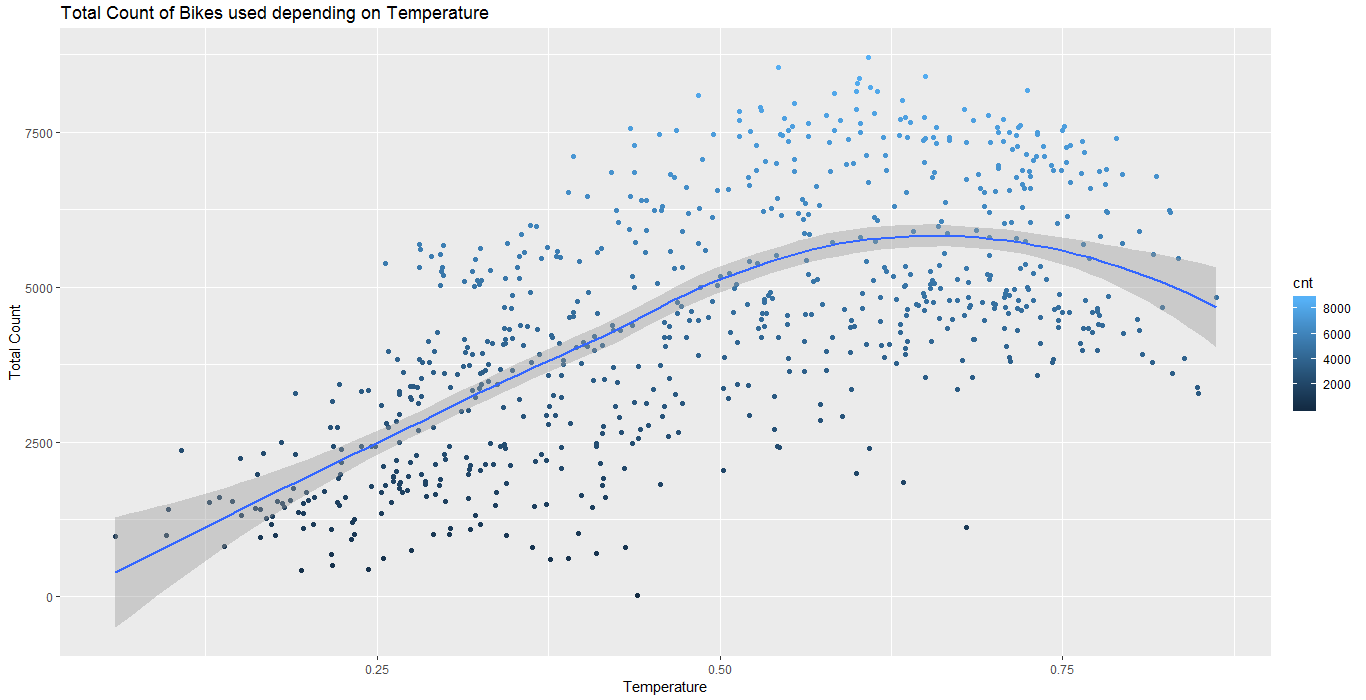












The following inferences can be drawn from the data. it was seen that the number of registered users was overall higher compared to that of casual users. Specifically when classified by working days it was found that more number of registered bikes were rented on occasions when it was neither a weekend nor a holiday as compared to casual bike rentals which were more during situations (1) when it could have been a weekend or a holiday. This possibly helps us to have an understanding of purpose and type of bike rentals. Registered users might use bikes on a daily basis, example for work or other day to day activities where as casual bike rentals are associated with holiday and leisure.

The highest number of bike rentals were between the months of June and August, whereas the lowest number of bike rentals were between the months November and February. This gives us an indication about the role of corresponding seasons associated with these months. Furthermore, I also checked for total count of bike rentals across seasons which confirms that the highest rentals were during summer followed by spring, fall and lastly in winter.

Another important factor to understand the bike rental trends is temperature. There was not a significant difference between real temperature and feeling of the temperature. Total Temperature was significantly correlated with total bike rentals. Irrespective of the type of bike rental the temperature (real) was equally correlated with both casual and registered rentals. However, one important thing to mention again is that the registered users were higher overall compared to casual users, hence the results could be biased or misleading, but overall it seems like temperature plays an important role for total count. Furthermore, weather situation was found to be significant predictor of bike rentals. However,the impact of holidays is not significant. One reason for holidays not being significant could be that there were probably a very few holidays during the years for it to have a significant impact.

### 2.1.2 Feature Selection

Before performing any type of modeling we need to assess the importance of each predictor variable in our analysis. There is a possibility that many variables in our analysis are not important at all to the problem of regression. There are several methods of doing that. Below we have used correlation and anova tests for feature selection.

Another step of Exploratory Data Analysis is to look for highly correlated variables in the data. A very simple way of looking at correlations in the data is shown below.

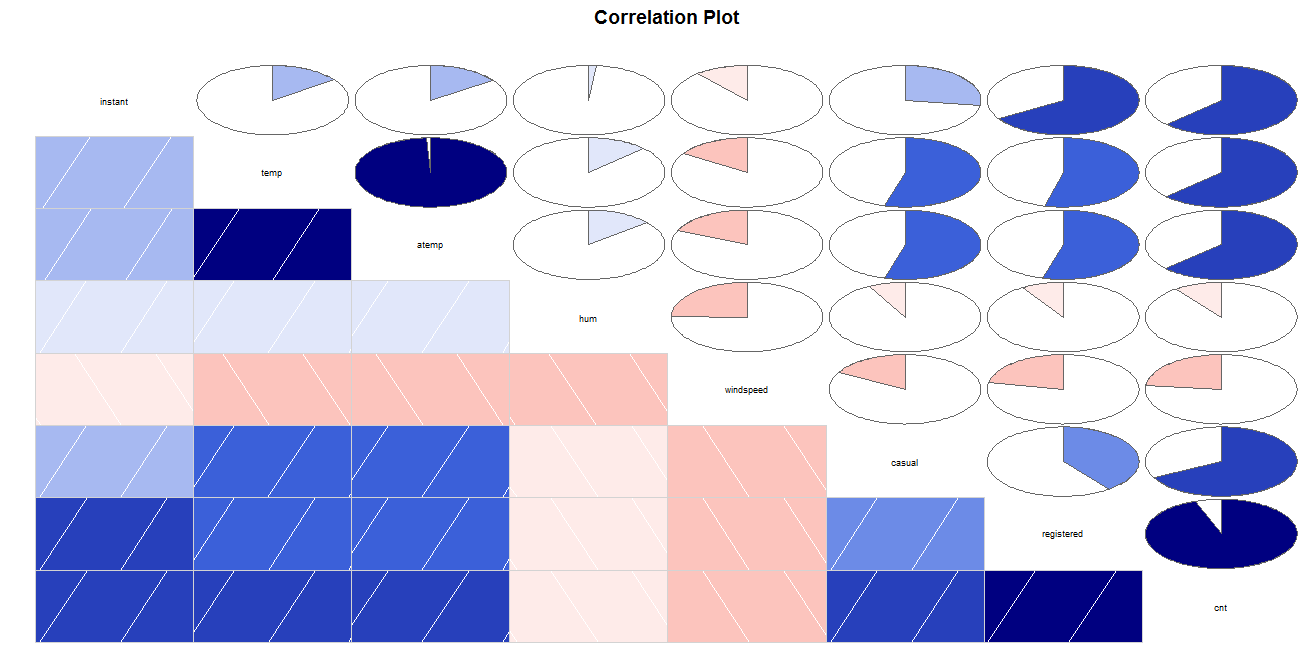
*#plotting corrgram*

*res2 = rcorr(as.matrix(num\_data))*

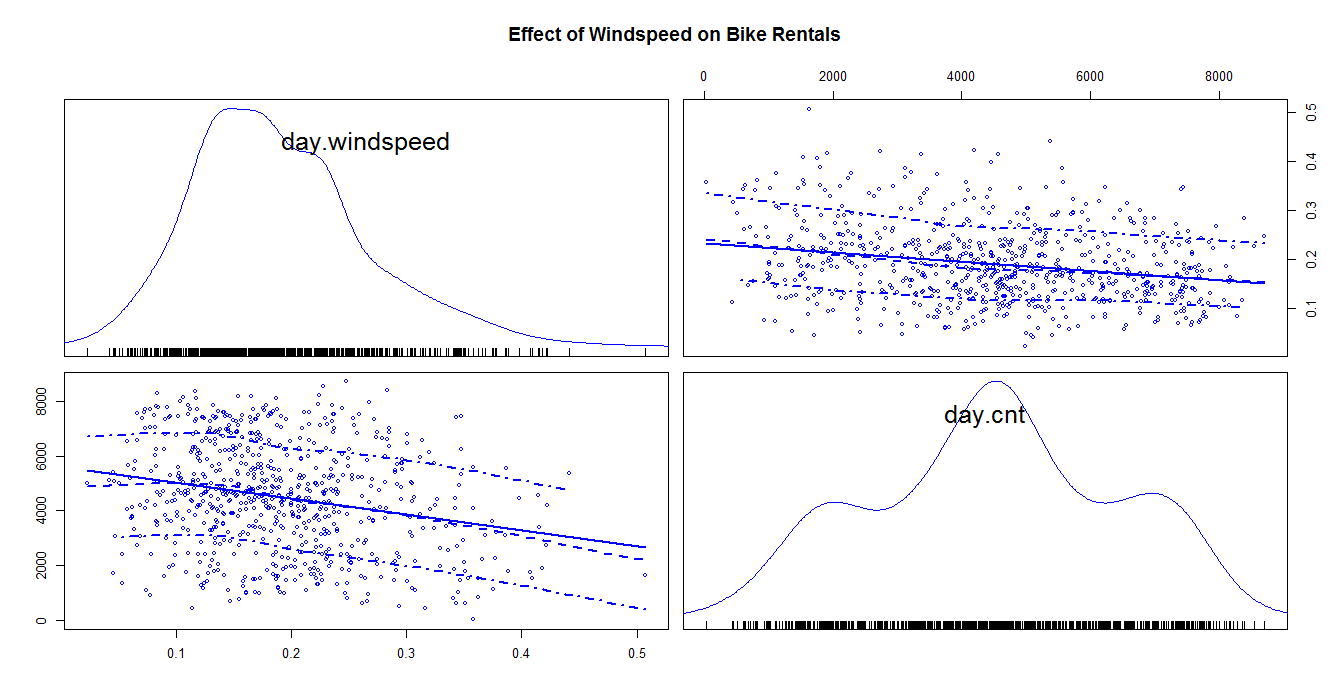
*res2*

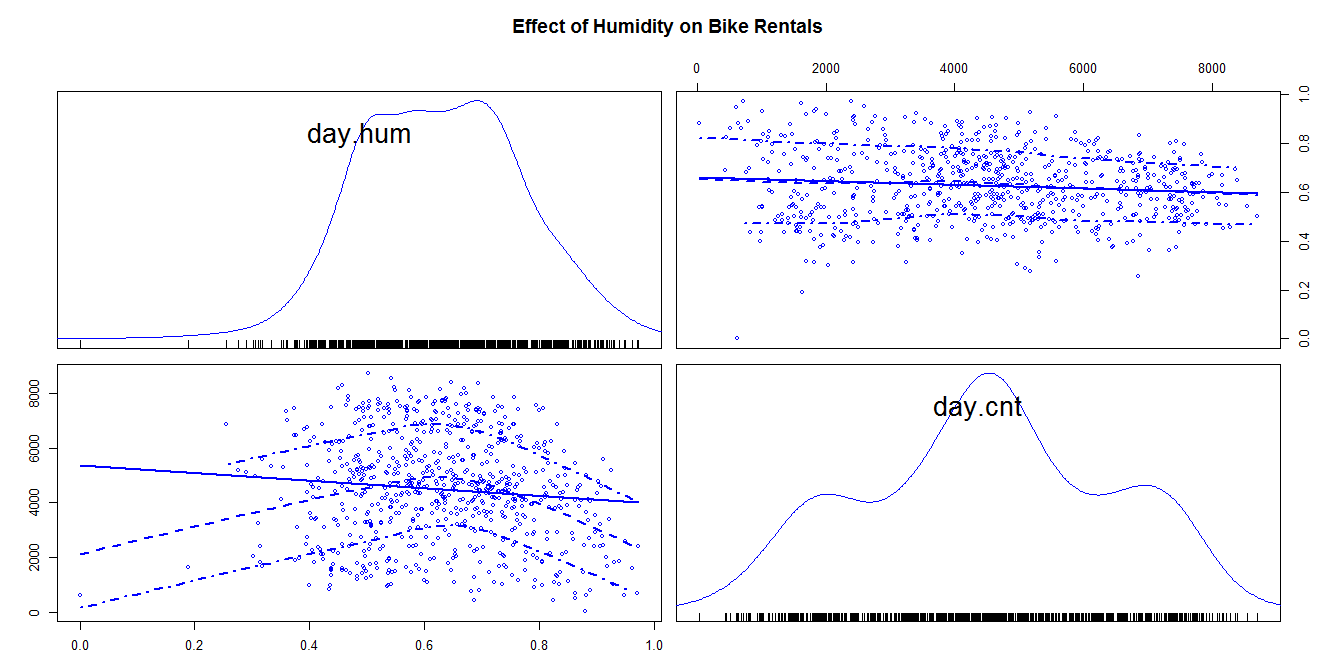
|  |
| --- |
| instant temp atemp hum windspeed casual registered cnt  instant 1.00 0.15 0.15 0.02 -0.11 0.28 0.66 0.63  temp 0.15 1.00 0.99 0.13 -0.16 0.54 0.54 0.63  atemp 0.15 0.99 1.00 0.14 -0.18 0.54 0.54 0.63  hum 0.02 0.13 0.14 1.00 -0.25 -0.08 -0.09 -0.10  windspeed -0.11 -0.16 -0.18 -0.25 1.00 -0.17 -0.22 -0.23  casual 0.28 0.54 0.54 -0.08 -0.17 1.00 0.40 0.67  registered 0.66 0.54 0.54 -0.09 -0.22 0.40 1.00 0.95  cnt 0.63 0.63 0.63 -0.10 -0.23 0.67 0.95 1.00  n= 731  P  instant temp atemp hum windspeed casual registered cnt  instant 0.0000 0.0000 0.6585 0.0023 0.0000 0.0000 0.0000  temp 0.0000 0.0000 0.0006 0.0000 0.0000 0.0000 0.0000  atemp 0.0000 0.0000 0.0001 0.0000 0.0000 0.0000 0.0000  hum 0.6585 0.0006 0.0001 0.0000 0.0374 0.0138 0.0065  windspeed 0.0023 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000  casual 0.0000 0.0000 0.0000 0.0374 0.0000 0.0000 0.0000  registered 0.0000 0.0000 0.0000 0.0138 0.0000 0.0000 0.0000  cnt 0.0000 0.0000 0.0000 0.0065 0.0000 0.0000 0.0000 |
|  |
| |  | | --- | |  | |

*corrgram(day[,num\_index], order = F,upper.panel=panel.pie, text.panel=panel.txt, main = "Correlation Plot")*



Scatterplot for windspeed and humidity





From the above correlation plot and scatterplots it is clear that the variables windspeed and humidity have negative correlation.The slopes of the variables are also negative.This indicates these variables are negatively correlated and will hamper model accuracy and are dropped while creating a model.

**Anova test for categorical variables**

Anova test is used to analyse if there is any (statistically) significant difference in groups of categorical variables.

ANOVA is going to compare means of dependent continuous variable among the groups of categorical variable and check ifdifferences are statistically significant. Here are null and alternative hypothesis:  
  
**Null Hypothesis**: all group means are equal —> there is no relationship between categorical variable and dependent variable, which we can write as follows:

        H0: all means are equal

**Alternative Hypothesis**: not all group means are equal —> there is a relationship between categorical variable and dependent variable.:

        H1: not all means are equal

F statistics = Variation among sample means / Variation within groups

Through the F statistics we can see if the variation among sample means dominates over the variation within groups, or not. In the first case we will have strong evidence against the null hypothesis (means are all equals), while in the second case we would have little evidence against the null hypothesis.

*season\_anv=aov(cnt~season,data=day)*

*summary(season\_anv)*

Results:

Df Sum Sq Mean Sq F value Pr(>F)

season 3 9.506e+08 316865289 128.8 <2e-16 \*\*\*

Residuals 727 1.789e+09 2460715

---

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

 our p-value is less than 0.05 (as suggested by normal scientific standard). Hence we can conclude that for our confidence interval we accept the alternative hypothesis H1 that there is a significant relationship between variables.

*holiday\_anv=aov(cnt~holiday,data=day)*

*summary(holiday\_anv)*

Results:

Df Sum Sq Mean Sq F value Pr(>F)

holiday 1 1.280e+07 12797494 3.421 0.0648 .

Residuals 729 2.727e+09 3740381

---

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

our p-value is more than 0.05 (as suggested by normal scientific standard). Hence we can conclude that for our confidence interval we accept the Null hypothesis H0 that there is a no significant relationship between variables.

*weathersit\_anv=aov(cnt~weathersit,data=day)*

*summary(weathersit\_anv)*

Results:

Df Sum Sq Mean Sq F value Pr(>F)

weathersit 2 2.716e+08 135822286 40.07 <2e-16 \*\*\*

Residuals 728 2.468e+09 3389960

---

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

our p-value is less than 0.05 (as suggested by normal scientific standard). Hence we can conclude that for our confidence interval we accept the alternative hypothesis H1 that there is a significant relationship between variables.

## 2.2 Modeling

### 2.2.1 Model Selection

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

1. Nominal
2. Ordinal
3. Interval
4. Ratio

Based on the dependent of variable of your dataset we use a model accordingly.There are two types of supervised learning models.They are classification and regression.we choose a type from them depending on the predicting variable. In this case our depending variable is continuous variable we use regression models.If the variable is categorical we go classification models.we try different regression models and analyse using error metrics choose the model which is optimal.

## 2.2.2 Linear Regression

*lin.mod =lm(cnt ~ season + workingday+ weathersit +hum+ temp, data =train\_data)*

*summary(lin.mod)*

Results:

Call:

lm(formula = cnt ~ season + workingday + weathersit + hum + temp,

data = train\_data)

Residuals:

Min 1Q Median 3Q Max

-3684.7 -997.7 -187.0 1068.0 3302.4

Coefficients:

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

(Intercept) 1890.4 310.0 6.099 1.96e-09 \*\*\*

season2 1002.5 206.1 4.864 1.49e-06 \*\*\*

season3 771.6 270.2 2.856 0.004446 \*\*

season4 1666.2 175.4 9.500 < 2e-16 \*\*\*

workingday1 146.9 117.1 1.254 0.210284

weathersit2 -448.6 142.6 -3.146 0.001739 \*\*

weathersit3 -2653.6 365.1 -7.268 1.19e-12 \*\*\*

hum -1684.7 488.6 -3.448 0.000606 \*\*\*

temp 5909.3 555.1 10.646 < 2e-16 \*\*\*

---

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

Residual standard error: 1319 on 575 degrees of freedom

Multiple R-squared: 0.5387, Adjusted R-squared: 0.5323

F-statistic: 83.93 on 8 and 575 DF, p-value: < 2.2e-16

As you can see the *Adjusted R-squared* value, we can explain about 53.87% of the data using our linear regression model.

The mean absolute percentage error in this model 30.28.Adding or removing any other variables didn’t make much difference in the error.

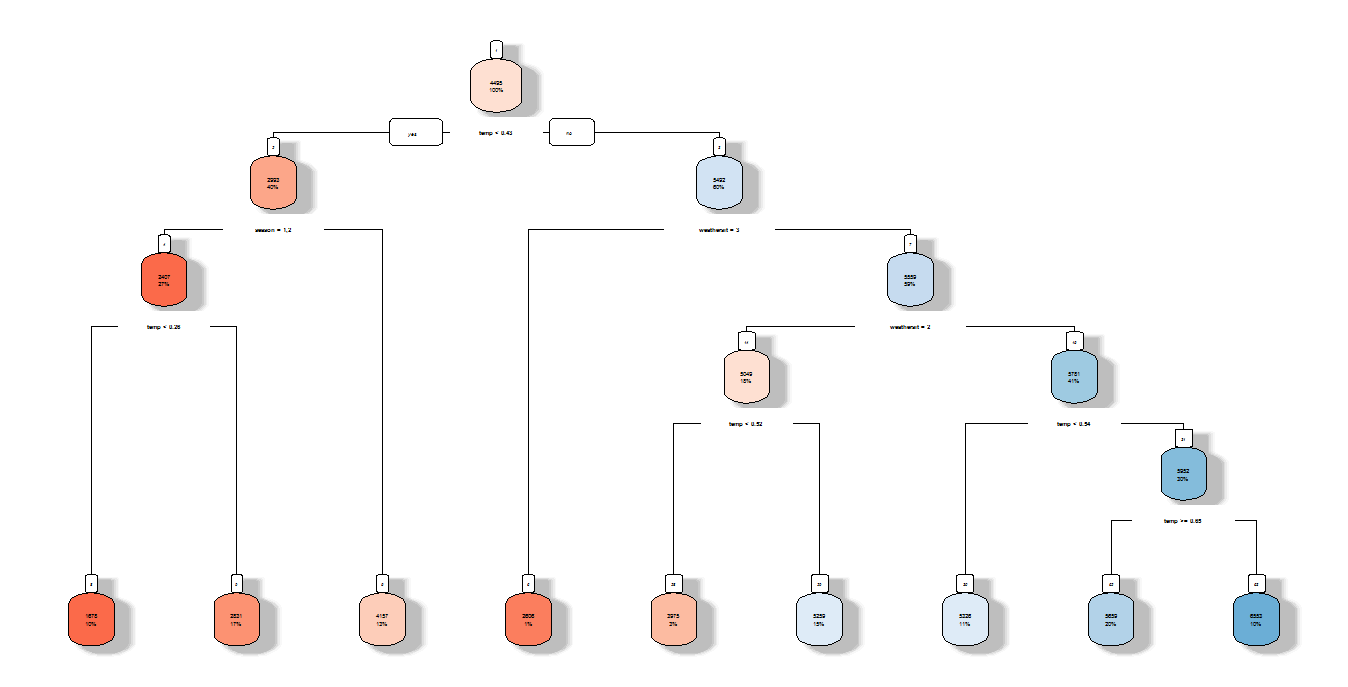
### 2.2.3 Desicison Tree Regression

Now we will try and use a diferent regression model to predict our cnt variable. We will use a regression tree to predict the values of our target variable.

# decission tree regression

fit = rpart(cnt ~ season + workingday+ weathersit + temp, data = train\_data, method = "anova")

pr=predict(fit,test\_data[,-16])



**Chapter 3**

# Conclusion

## 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. We can compare the models using any of the following criteria:

1. Predictive Performance
2. Interpretability
3. Computational Efficiency

In our case of Wine Data, the latter two, Interpretability and Computation Efficiency, do not hold much significance. Therefore we will use Predictive performance as the criteria to compare and evaluate models.

Predictive performance can be measured by comparing Predictions of the models with real values of the taret variables, and calculating some average error measure.

### 3.1.1 Mean Absolute Percentage Error (MAPE)

MAPE is one of the error measures used to calculate the predictive performance of the model. We will apply this measure to our models that we have generated in the previous section.

*#linear regression*

*lin.mod =lm(cnt ~ season + workingday+ weathersit + temp, data =train\_data)*

*predictions=predict(lin.mod,test\_data[,-16])*

*mapee = function(y, yhat){*

*mean(abs((y - yhat)/y))\*100*

*}*

*mapee(test\_data[,16],predictions)*

Results:

#30.4963

*#Decision tree*

*fit = rpart(cnt ~ season + workingday+ weathersit + temp, data = train\_data, method = "anova")*

*pr=predict(fit,test\_data[,-16])*

*rpart.plot(fit,box.palette ="RdBu",shadow.col="gray",nn=TRUE)*

*mapee(test\_data[,16],pr)*

Results:

# 28.44227

## 3.2 Model Selection

We can see that both models perform comparatively on average and therefore we can select either of the two models without any loss of information.

## 3.3 R code of the Entire model:

*rm(list=ls(all=T))*

*setwd("C:/Users/BITTU/Desktop/Project 1")*

*getwd()*

*#Load Libraries*

*x = c("ggplot2", "corrgram", "DMwR", "caret", "unbalanced", "dummies", "e1071", "Information",*

*"MASS", "rpart", "gbm", "ROSE", 'sampling', 'DataCombine', 'inTrees')*

*#install.packages("rpart.plot")*

*library(Hmisc)*

*library(car)*

*library(DMwR)*

*library(Metrics)*

*library(rpart.plot)*

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

*rm(x)*

*#install.packages("sqldf")*

*library(sqldf)*

*library('kableExtra')*

*library(MASS)*

*library(psych)*

*#read the data*

*day=read.csv("day.csv")*

*head(day[, 1:10]) %>% kable(caption = "Bike Rental Count (Columns: 1-10)",*

*booktabs = TRUE, longtable = TRUE)*

*head(day[, 11:16]) %>% kable(caption = "Bike Rental Count (Columns: 11-16)",*

*booktabs = TRUE, longtable = TRUE)*

*var <- colnames(day)[-ncol(day)]*

*num <- 1:length(var)*

*df = data.frame(S.No. = num, Predictor = var)*

*kable(df, caption = "Predictor Variables", booktabs = TRUE,*

*longtable = TRUE)*

*#converting the data into required format*

*day$season=as.factor(as.character(day$season))*

*day$yr=as.factor(as.character(day$yr))*

*day$holiday=as.factor(as.character(day$holiday))*

*day$workingday=as.factor(as.character(day$workingday))*

*day$mnth=as.factor(as.character(day$mnth))*

*day$weekday=as.factor(as.character(day$weekday))*

*day$weathersit=as.factor(as.character(day$weathersit))*

*str(day)*

*#missing value analysis*

*missing\_val=data.frame(apply(day,2,function(x){sum(is.na(x))}))*

*missing\_val*

*###no missing values found in the data*

*#histograms*

*hist(day$cnt, breaks = 25,*

*ylab = 'Frequency of Rental', xlab = 'Total Bike Rental Count',*

*main = 'Distribution of Total Bike Rental Count', col = 'blue' )*

*hist(day$windspeed, main="Histogram for Wind Speed",*

*xlab="wind speed", col = "red")*

*hist(day$temp, main="Histogram for Temperature",*

*xlab="temperature", col = "green")*

*hist(day$hum, main="Histogram for Humidity",*

*xlab="temperature", col = "grey")*

*#plots*

*plot(day$temp, day$cnt ,*

*type = 'h', col= 'red', xlab = 'Temperature', ylab = 'Total Bike Rentals')*

*plot(day$atemp, day$cnt ,*

*type = 'h', col= 'blue', xlab = 'Feel Temperature', ylab = 'Total Bike Rentals')*

*plot(day$windspeed, day$cnt ,*

*type = 'h', col= 'green', xlab = 'Windspeed', ylab = 'Total Bike Rentals')*

*plot(day$hum, day$cnt ,*

*type = 'h', col= 'black', xlab = 'Humidity', ylab = 'Total Bike Rentals')*

*ggplot (day, aes( x= temp, y = cnt, colour = cnt))+*

*geom\_point()+geom\_smooth()+xlab("Temperature") +*

*ylab ("Total Count")+*

*ggtitle("Total Count of Bikes used depending on Temperature")*

*#Boxplot analysis of the variables*

*boxplot(day$cnt ~ day$season,*

*data = day,*

*main = "Total Bike Rentals Vs Season",*

*xlab = "Season",*

*ylab = "Total Bike Rentals",*

*col = c("red", "red1", "red2", "red3"))*

*boxplot(day$cnt ~ day$holiday,*

*data = day,*

*main = "Total Bike Rentals Vs Holiday/Working Day",*

*xlab = "Holiday/Working Day",*

*ylab = "Total Bike Rentals",*

*col = c("blue", "blue1", "blue2", "blue3"))*

*boxplot(day$cnt ~ day$weathersit,*

*data = day,*

*main = "Total Bike Rentals Vs Weather Situation",*

*xlab = "Weather Situation",*

*ylab = "Total Bike Rentals",*

*col = c("green", "green1", "green2", "green3"))*

*boxplot(day$cnt ~ day$mnth,*

*data = day,*

*main = "Total Bike Rentals Vs Month",*

*xlab = "Month",*

*ylab = "Total Bike Rentals",*

*col = c("yellow"))*

*boxplot(day$cnt ~ day$weekday,*

*data = day,*

*main = "Total Bike Rentals Vs Day of Week",*

*xlab = "Day of Week",*

*ylab = "Total Bike Rentals",*

*col = c("black"))*

*num\_index=sapply(day,is.numeric)*

*num\_data=day[,num\_index]*

*cnames=colnames(num\_data)*

*for (i in 1:length(cnames))*

*{*

*assign(paste0("gn",i), ggplot(aes\_string(y = (cnames[i]), x = 'cnt'), data = subset(day))+*

*stat\_boxplot(geom = "errorbar", width = 0.5) +*

*geom\_boxplot(outlier.colour="red", fill = "grey" ,outlier.shape=18,*

*outlier.size=1, notch=FALSE) +*

*theme(legend.position="bottom")+*

*labs(y=cnames[i],x="cnt")+*

*ggtitle(paste("Box plot of count for",cnames[i])))*

*}*

*### Plotting plots together*

*gridExtra::grid.arrange(gn1,gn5,gn2,gn3,gn4,ncol=5)*

*##scaterplots to check variables which are negatively correlated*

*scatterplotMatrix(formula = ~ day$windspeed + day$cnt, cex=0.6,*

*data=day, main = "Effect of Windspeed on Bike Rentals" )*

*scatterplotMatrix(formula = ~ day$hum + day$cnt, cex=0.6,*

*data=day, main = "Effect of Humidity on Bike Rentals" )*

*#plotting corrgram*

*res2 = rcorr(as.matrix(num\_data))*

*res2*

*num\_index=sapply(day,is.numeric)*

*num\_data=day[,num\_index]*

*cnames=colnames(num\_data)*

*corrgram(day[,num\_index], order = F,upper.panel=panel.pie, text.panel=panel.txt, main = "Correlation Plot")*

*str(num\_data)*

*##anova test for independence*

*factor\_data=subset(day,select=c(season,holiday,weekday,workingday,weathersit))*

*names(factor\_data)*

*season\_anv=aov(cnt~season,data=day)*

*summary(season\_anv)*

*holiday\_anv=aov(cnt~holiday,data=day)*

*summary(holiday\_anv)*

*workingday\_anv=aov(cnt~workingday,data=day)*

*summary(workingday\_anv)*

*weathersit\_anv=aov(cnt~weathersit,data=day)*

*summary(weathersit\_anv)*

*#divide data into train and test data*

*train\_ind=sample(1:nrow(day),0.8\*nrow(day))*

*train\_data=day[train\_ind,]*

*test\_data=day[-train\_ind,]*

*# decission tree regression*

*fit = rpart(cnt ~ season + workingday+ weathersit + temp, data = train\_data, method = "anova")*

*pr=predict(fit,test\_data[,-16])*

*rpart.plot(fit,box.palette ="RdBu",shadow.col="gray",nn=TRUE)*

*mapee(test\_data[,16],pr)*

*#error metrics*

*#calculate mape*

*actuals\_preds = data.frame(cbind(actuals=test\_data$cnt, predicteds=predictions))*

*mapee = function(y, yhat){*

*mean(abs((y - yhat)/y))\*100*

*}*

*#linear regression model*

*lin.mod =lm(cnt ~ season + workingday+ weathersit + temp, data =train\_data)*

*predictions=predict(lin.mod,test\_data[,-16])*

*mapee(test\_data[,16],predictions)*

*#mape value for linear regression model is 30.49*

*#mape value for Decision tree regression model is 28.47*

## 3.4 Python code

**import** **os**

**import** **pandas** **as** **pd**

**import** **seaborn** **as** **sns**

**from** **random** **import** randrange, uniform

**import** **numpy** **as** **np**

**import** **matplotlib.pyplot** **as** **plt**

**from** **sklearn.model\_selection** **import** train\_test\_split

**from** **sklearn.tree** **import** DecisionTreeRegressor

**from** **sklearn** **import** metrics

In [2]:

os.chdir("C:/Users/BITTU/Desktop/Project 1")

os.getcwd()

Out[2]:

'C:\\Users\\BITTU\\Desktop\\Project 1'

In [3]:

df\_csv=pd.read\_csv("day.csv",sep=",")

In [4]:

df\_csv.columns

Out[4]:

Index(['instant', 'dteday', 'season', 'yr', 'mnth', 'holiday', 'weekday',

'workingday', 'weathersit', 'temp', 'atemp', 'hum', 'windspeed',

'casual', 'registered', 'cnt'],

dtype='object')

In [6]:

*#these are columns for correlation analysis*

cnames\_corr=['dteday', 'season', 'yr', 'mnth', 'holiday', 'weekday',

'workingday', 'weathersit', 'temp', 'atemp', 'hum', 'windspeed',

'casual', 'registered','cnt']

*#below features contains outliars*

cnames\_ol=['temp','atemp','hum','windspeed']

In [7]:

*#dropping outlier rows*

**for** i **in** cnames\_ol:

q75,q25=np.percentile(df\_csv.loc[:,i],[75,25])

iqr=q75-q25

min=q25-(iqr\*1.5)

max=q75+(iqr\*1.5)

df\_csv=df\_csv.drop(df\_csv[df\_csv.loc[:,i]<min].index)

df\_csv=df\_csv.drop(df\_csv[df\_csv.loc[:,i]>max].index)

In [8]:

*#saving into new dataset called df\_csv\_1*

df\_csv\_1=df\_csv.loc[:,cnames\_corr]

df\_csv\_1.shape

Out[8]:

(717, 15)

In [10]:

*#new dataset called df\_csv\_2 for correlation analysis and also applied on onlynecessary features*

df\_csv\_2=df\_csv\_1.iloc[:,0:12]

In [ ]:

*#below is correlation matrix*

f, ax = plt.subplots(figsize=(7, 5))

*#Generate correlation matrix*

corr = df\_csv\_2.corr()

*#Plot using seaborn library*

sns.heatmap(corr, mask=np.zeros\_like(corr, dtype=np.bool), cmap=sns.diverging\_palette(220, 10, as\_cmap=**True**),

square=**True**, ax=ax)

In [11]:

*#below is the step to remove columns that contain hign +ve and -ve correlations*

df\_csv\_1=df\_csv\_1.drop(['windspeed','hum','dteday','holiday','windspeed','atemp','casual','registered'],axis=1)

x=df\_csv\_1.drop('cnt',axis=1)

y=df\_csv\_1['cnt']

In [12]:

*#is to check column names of removing target feature*

df\_csv\_1.columns

Out[12]:

Index(['season', 'yr', 'mnth', 'weekday', 'workingday', 'weathersit', 'temp',

'cnt'],

dtype='object')

In [13]:

*#below is to divide dataset into train and test data sets*

x\_train,x\_test,y\_train,y\_test=train\_test\_split(x,y,test\_size=0.2)

In [14]:

*#Applying DT regression*

fit\_Dt=DecisionTreeRegressor(max\_depth=50).fit(x\_train,y\_train)

In [15]:

*#storing predicted values into y\_pred using above model called fit\_Dt*

y\_pred=fit\_Dt.predict(x\_test)

In [16]:

*#to see Actual values and predicted values*

df=pd.DataFrame({'Actual':y\_test,'Prediction':y\_pred})

df

Out[16]:

|  | **Actual** | **Prediction** |
| --- | --- | --- |
| **711** | 5319 | 5008.0 |
| **707** | 5582 | 4220.0 |
| **74** | 2192 | 2121.0 |
| **0** | 985 | 801.0 |
| **437** | 5847 | 6153.0 |
| **113** | 4191 | 3348.0 |
| **480** | 6196 | 5585.0 |
| **238** | 1115 | 4150.0 |
| **417** | 4773 | 4363.0 |
| **124** | 4433 | 4189.0 |
| **531** | 7665 | 7736.0 |
| **312** | 4109 | 4068.0 |
| **149** | 4098 | 4507.0 |
| **566** | 5870 | 7415.0 |
| **612** | 6864 | 6784.0 |
| **129** | 4803 | 4182.0 |
| **342** | 3620 | 3614.0 |
| **414** | 2689 | 2947.0 |
| **38** | 1530 | 1712.0 |
| **333** | 3613 | 3331.0 |
| **388** | 4339 | 4509.0 |
| **457** | 5936 | 5918.0 |
| **421** | 3389 | 3129.0 |
| **428** | 3423 | 3129.0 |
| **571** | 8173 | 7148.0 |
| **656** | 7509 | 7852.0 |
| **135** | 3958 | 5020.0 |
| **208** | 4390 | 4541.0 |
| **453** | 6133 | 6871.0 |
| **183** | 4649 | 6043.0 |
| **...** | ... | ... |
| **14** | 1248 | 1096.0 |
| **635** | 7393 | 7328.0 |
| **602** | 6053 | 5976.0 |
| **186** | 4629 | 5302.0 |
| **64** | 605 | 2077.0 |
| **275** | 3570 | 3811.0 |
| **212** | 4266 | 4326.0 |
| **347** | 3740 | 3053.0 |
| **696** | 3959 | 1096.0 |
| **633** | 7538 | 7767.0 |
| **545** | 5463 | 6207.0 |
| **32** | 1526 | 1807.0 |
| **179** | 5225 | 4507.0 |
| **123** | 2633 | 2162.0 |
| **472** | 6691 | 7424.0 |
| **257** | 3659 | 4795.0 |
| **143** | 4492 | 4835.0 |
| **118** | 4595 | 3429.0 |
| **689** | 5634 | 5478.0 |
| **418** | 5062 | 4579.0 |
| **530** | 7363 | 7494.0 |
| **648** | 7691 | 7058.0 |
| **475** | 7290 | 7460.0 |
| **389** | 4270 | 3598.0 |
| **582** | 5464 | 6227.0 |
| **242** | 5058 | 5895.0 |
| **441** | 7836 | 5026.0 |
| **281** | 5511 | 5041.0 |
| **122** | 4451 | 4274.0 |
| **314** | 3368 | 3926.0 |

144 rows × 2 columns

In [17]:

*#caluculation of MAE MQE,RMSE*

print('Mean Absolute Error:', metrics.mean\_absolute\_error(y\_test, y\_pred))

print('Mean Squared Error:', metrics.mean\_squared\_error(y\_test, y\_pred))

print('Root Mean Squared Error:', np.sqrt(metrics.mean\_squared\_error(y\_test, y\_pred)))

Mean Absolute Error: 618.3194444444445

Mean Squared Error: 763241.0

Root Mean Squared Error: 873.6366521615265

In [18]:

*#calculation of MAPE*

**def** MAPE(x,y):

mape=np.mean(np.abs((x-y)/x))\*100

**return** mape

*#Percentage of error in out DT Regression Model*

count=MAPE(y\_test,y\_pred)

count

Out[18]:

21.17055166295553

In [19]:

*#Linear Regression from scratch*

**import** **os**

**import** **pandas** **as** **pd**

**import** **seaborn** **as** **sns**

**from** **random** **import** randrange, uniform

**import** **numpy** **as** **np**

**import** **matplotlib.pyplot** **as** **plt**

**from** **sklearn.model\_selection** **import** train\_test\_split

**import** **statsmodels.api** **as** **sm**

In [604]:

os.chdir("C:/Users/Rajashekar/Videos/python\_project")

os.getcwd()

Out[604]:

'C:\\Users\\Rajashekar\\Videos\\python\_project'

In [605]:

df\_csv=pd.read\_csv("day.csv",sep=",")

In [606]:

*#these are columns for correlation analysis*

cnames\_corr=['dteday', 'season', 'yr', 'mnth', 'holiday', 'weekday',

'workingday', 'weathersit', 'temp', 'atemp', 'hum', 'windspeed',

'casual', 'registered','cnt']

*#below features contains outliars*

cnames\_ol=['temp','atemp','hum','windspeed']

In [607]:

*#dropping outlier rows*

**for** i **in** cnames\_ol:

q75,q25=np.percentile(df\_csv.loc[:,i],[75,25])

iqr=q75-q25

min=q25-(iqr\*1.5)

max=q75+(iqr\*1.5)

df\_csv=df\_csv.drop(df\_csv[df\_csv.loc[:,i]<min].index)

df\_csv=df\_csv.drop(df\_csv[df\_csv.loc[:,i]>max].index)

In [608]:

*#saving into new dataset called df\_csv\_1*

df\_csv\_1=df\_csv.loc[:,cnames\_corr]

df\_csv\_1.shape

Out[608]:

(717, 15)

In [609]:

*#new dataset called df\_csv\_2 for correlation analysis and also applied on onlynecessary features*

df\_csv\_2=df\_csv\_1.iloc[:,0:12]

In [610]:

*#below is correlation matrix*

f, ax = plt.subplots(figsize=(7, 5))

*#Generate correlation matrix*

corr = df\_csv\_2.corr()

*#Plot using seaborn library*

sns.heatmap(corr, mask=np.zeros\_like(corr, dtype=np.bool), cmap=sns.diverging\_palette(220, 10, as\_cmap=**True**),

square=**True**, ax=ax)

Out[610]:

<matplotlib.axes.\_subplots.AxesSubplot at 0x1a8c5e99780>

In [611]:

*#below is the step to remove columns that contain hign +ve and -ve correlations*

df\_csv\_1=df\_csv\_1.drop(['season','windspeed','dteday','holiday','hum','windspeed','atemp','casual','registered'],axis=1)

x=df\_csv\_1.drop('cnt',axis=1)

y=df\_csv\_1['cnt']

In [612]:

*#is to check column names of removing target feature*

df\_csv\_1.columns

Out[612]:

Index(['yr', 'mnth', 'weekday', 'workingday', 'weathersit', 'temp', 'cnt'], dtype='object')

In [613]:

*#below is to divide dataset into train and test data sets*

x\_train,x\_test,y\_train,y\_test=train\_test\_split(x,y,test\_size=0.2)

In [20]:

*# Train the model using the training sets*

model = sm.OLS(y\_train, x\_train).fit()

In [21]:

*#summary of above model*

model.summary()

Out[21]:

Warnings:  
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

In [22]:

*#prediction of values of above mode*

y\_pred= model.predict(x\_test)

In [23]:

*#MAPE1 function*

**def** MAPE1(x,y):

mape=np.mean(np.abs((x-y)/x))\*100

**return** mape

*#Error percentage*

count=MAPE1(y\_test,y\_pred)

count

Out[23]:

20.305142063074385