PROJECT BIKE RENTAL COUNT

By Pavithra Mamallan

06th August 2019

SYNOPSIS

Tables and Figures

Chapter 1

1. Introduction

1.1.Problem Statement

The objective of this Case is to Predication of bike rental count on daily based on the environmental and seasonal settings.

The details of data attributes in the dataset are as follows –

instant: Record index

dteday: Date

season: Season (1:springer, 2:summer, 3:fall, 4:winter)

yr: Year (0: 2011, 1:2012)

mnth: Month (1 to 12)

hr: Hour (0 to 23)

holiday: weather day is holiday or not (extracted fromHoliday Schedule)

weekday: Day of the week

workingday: If day is neither weekend nor holiday is 1, otherwise is 0.

weathersit: (extracted fromFreemeteo)

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

registered: count of registered users

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

Data

The dataset "day.csv" contains 731 observations and 17 variables. The dependent variable is 'cnt' The independent variables are 'instant', 'dteday', 'season', 'yr', 'mnth', 'hr', 'holiday', 'weekday', 'workingday', 'weathersit', 'temp', 'atemp', 'hum', 'windspeed', 'casual', 'registered'.

instant	dteday	season	yr	mnth	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

weathersit	temp	atemp	hum	windspeed	casual	registered	cnt
2	0.344167	0.363625	0.805833	0.160446	331	654	985
2	0.363478	0.353739	0.696087	0.248539	131	670	801
1	0.196364	0.189405	0.437273	0.248309	120	1229	1349
1	0.2	0.212122	0.590435	0.160296	108	1454	1562
1	0.226957	0.22927	0.436957	0.1869	82	1518	1600

Table 1.1 : First five observations in the Dataset

Chapter 2

2. Pre-processing techniques

2.1.Outlier Analysis

Outliers are extreme values that deviate from other observations on data; they may indicate variability in a measurement, experimental errors or a novelty. In other words, an outlier is an observation that diverges from an overall pattern on a sample. Outliers can be detected using one of the four major techniques.

- **Graphical Plot – Box plot**
- **Statistical Technique – Grubb's test**
- **R package – Outlier**
- **Experiment**

Most of the times, we use only Boxplot method which is most suitable method for all kinds of data. If we consider Grubb's test, which is the statistical technique for detecting the outliers, it is limited only to the normally distributed data. We cannot expect the normally distributed data all the time. Often the data will be skewed. Hence, it is not possible to use Grubb's test for all data. R package, "Outlier" works on the mean concept. It calculates the mean of the variable and then detects which values are falling very far from the mean. This may give some wrong answers and also it takes so much time.

A box plot is a highly visually effective way of viewing a clear summary of one or more sets of data. It is particularly useful for quick **summarizing and comparison** of different sets of variables. At a glance, a box plot allows a graphical display of the distribution of results and provides indications of symmetry within the data.

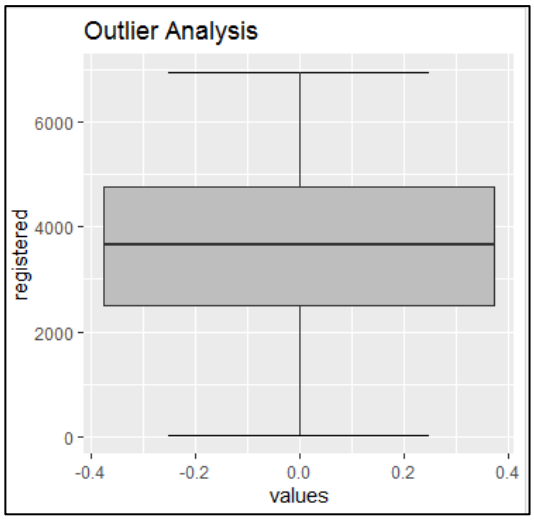


Fig 2.1 : Outlier Analysis of Registered users

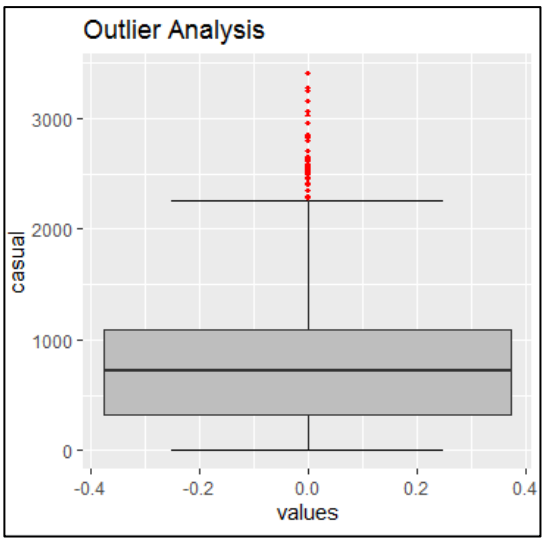
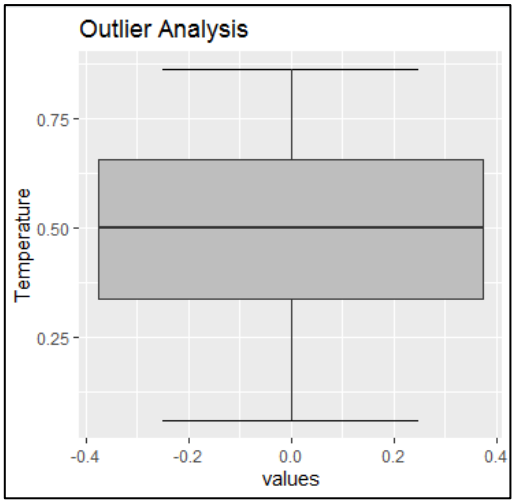


Fig 2.2 : Outlier Analysis of Casual users



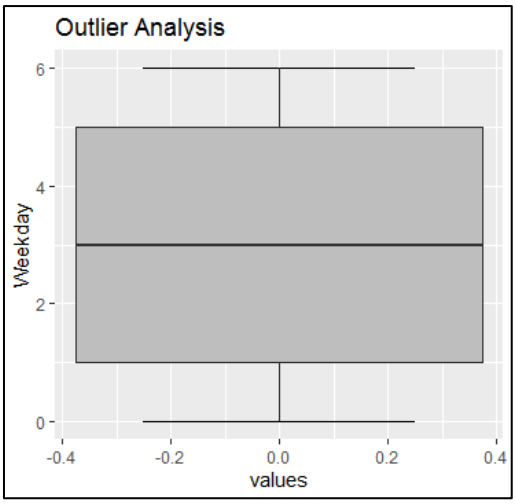


Fig 2.3 : Outlier Analysis of Temperature Fig 2.4 : Outlier Analysis of Weekday

The plots clearly show that the variables contain extreme values or outliers. The numeric variables are selected and plotted using box plot. The outliers are not removed in this case as these observations may contain valuable information, without which the accuracy of the model will be questionable.

2.2.Feature Engineering

Feature engineering efforts mainly have two goals:

- Preparing the proper input dataset, compatible with the machine learning algorithm requirements.
- Improving the performance of machine learning models.

The features you use influence more than everything else the result. No algorithm alone, to my knowledge, can supplement the information gain given by correct feature engineering.

— Luca Massaron

Season, mnth, yr, holiday, weekday, workingday, weathersit are converted into factor variables. Date is extracted from dteday feature and then is converted into factor variable.

2.3.Feature Selection

Feature Selection is one of the core concepts in machine learning which hugely impacts the performance of the model. The data features that are used to train the machine learning models have a huge influence on the performance. Irrelevant or partially relevant features can negatively impact model performance.

VIF

VIF stands for Variance Inflation Factor. During regression analysis, VIF assesses whether factors are correlated to each other (multicollinearity), which could affect p-values and the model isn't going to be as reliable. If a VIF is greater than 10, you have high

multicollinearity and the variation will seem larger and the factor will appear to be more influential than it is. If VIF is closer to 1, then the model is much stronger, as the factors are not impacted by correlation with other factors.

Correlation

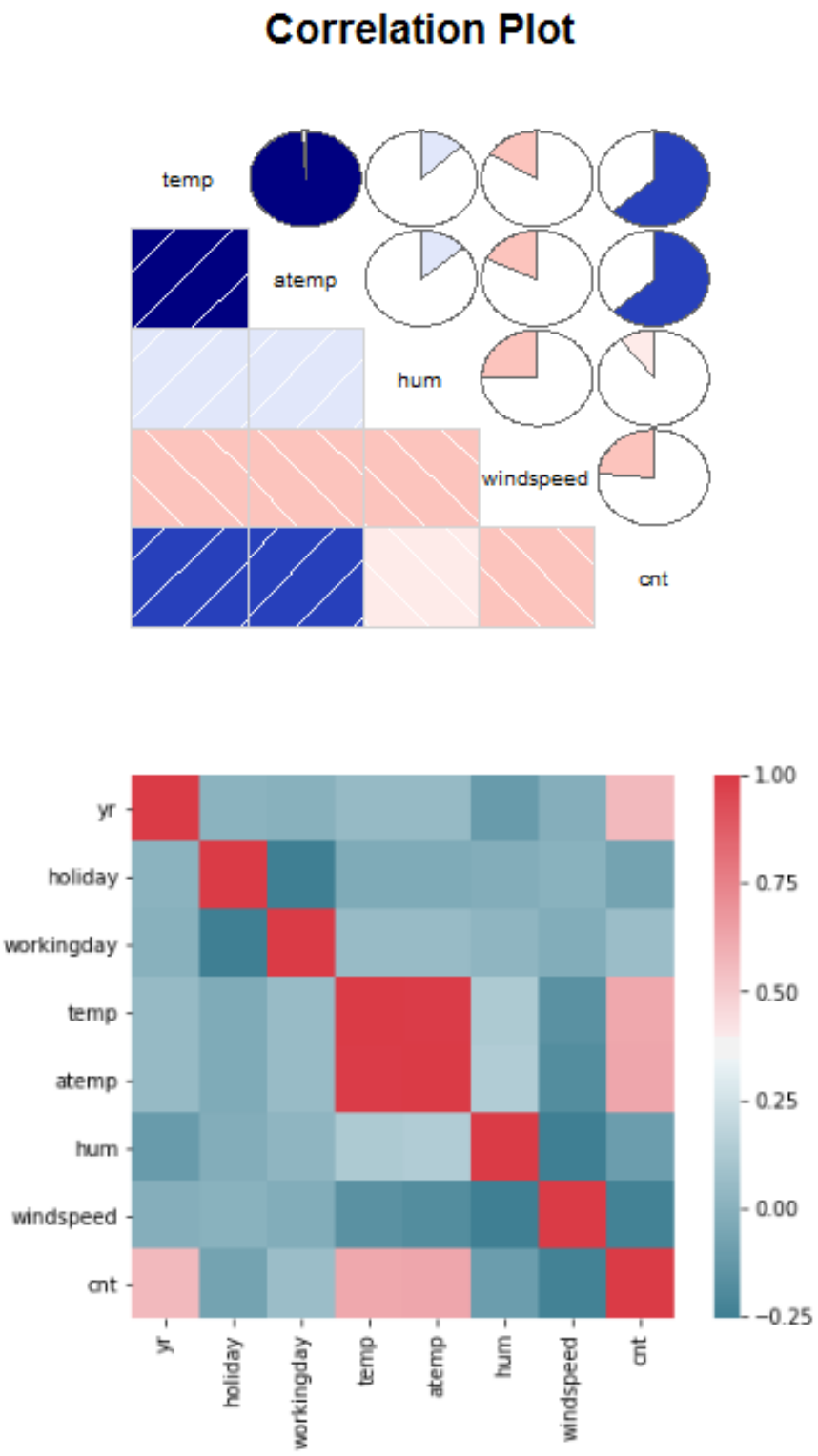


Fig 2.5 : Correlation plot in R and Python

From this Corrgram, Dark blue colour represents highly correlated and light colour represent very less correlated. It is clearly found that ‘temp’ and ‘atemp’ are correlated. Hence it is necessary to deselect one of them for better model performance.

2.4.Sampling

Sampling is the selection of a subset (a statistical sample) of individuals from within a statistical population to estimate characteristics of the whole population.

- **Population parameter**. A population parameter is the true value of a population attribute.
- **Sample statistic**. A sample statistic is an estimate, based on sample data, of a population parameter.

The quality of a sample statistic (i.e., accuracy, precision, representativeness) is strongly affected by the way that sample observations are chosen; that is., by the sampling method. Some of the Sampling methods are

- Simple random sampling
- Stratified sampling
- Systematic sampling

For this dataset, simple random sampling is used since the dependent variable is continuous. The dataset is divided into train and test data. 80% of the data is separated for training the data and the remaining 20% is for testing the data.

The process of training an ML model involves providing an ML algorithm (that is, the learning algorithm) with training data to learn from. The term ML model refers to the model artefact that is created by the training process. The training data must contain the correct answer, which is known as a target or target attribute. The learning algorithm finds **patterns in the training data** that map the input data attributes to the target (the answer that you want to predict), and it outputs an ML model that captures these patterns.

Chapter 3

3. Modeling

In the previous sections we have done all the pre-processing steps in the dataset to develop the model. Now, as our problem statement is to predict the fare amount, which is a **continuous variable**, we build models for **Regression analysis**. Always, we have move from simple to complex. Hence, the first model that we are going to build is Multiple Linear Regression. And then Support Vector Regressor, Decision Tree Regression, Random Forest.

3.1.Multiple Linear Regression

Multiple linear regression attempts to model the relationship between two or more explanatory variables and a response variable by fitting a linear equation to observed data. Every value of the independent variable x is associated with a value of the dependent variable y.

The Formula for Multiple Linear Regression Is

$$\boldsymbol{Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p + \varepsilon}$$

where,

Y=dependent variable

X=explanatory variables

β0=y-intercept (constant term)

βp=slope coefficients for each explanatory variable

ϵ=the model's error term (also known as the residuals)

Summary of Linear Regression model is as follows.

Call:
lm(formula = cnt ~ season + yr + mnth + holiday + weekday + weathersit +
temp + hum + windspeed, data = train)

Residuals:

Min	1Q	Median	3Q	Max
-3817.3	-376.4	51.4	474.8	2900.2

Coefficients:

	Estimate	**Std. Error**	**t value**	**Pr(>|t|)**
(Intercept)	1452.58	263.21	5.519	5.25e-08 ***
season2	708.14	210.26	3.368	0.000810 ***

season3	430.62	242.61	1.775	0.076446 .
season4	1467.11	204.09	7.189	2.12e-12 ***
yr1	1960.15	64.83	30.234	< 2e-16 ***
mnth2	140.95	161.04	0.875	0.381795
mnth3	410.37	187.94	2.184	0.029415 *
mnth4	495.92	281.91	1.759	0.079107 .
mnth5	672.83	305.67	2.201	0.028134 *
mnth6	428.16	322.45	1.328	0.184786
mnth7	78.99	355.50	0.222	0.824250
mnth8	491.55	340.03	1.446	0.148845
mnth9	1260.83	297.48	4.238	2.64e-05 ***
mnth10	569.51	271.55	2.097	0.036420 *
mnth11	-47.62	260.47	-0.183	0.855013
mnth12	53.06	206.63	0.257	0.797452
holiday1	-754.69	189.48	-3.983	7.71e-05 ***
weekday1	221.70	122.13	1.815	0.070013 .
weekday2	333.92	119.23	2.801	0.005278 **
weekday3	429.34	119.54	3.592	0.000358 ***
weekday4	390.68	120.31	3.247	0.001235 **
weekday5	511.21	120.73	4.234	2.68e-05 ***
weekday6	557.06	116.77	4.770	2.35e-06 ***
weathersit2	-488.92	87.35	-5.597	3.43e-08 ***
weathersit3	-2196.29	216.77	-10.132	< 2e-16 ***
temp	5165.13	460.41	11.218	< 2e-16 ***
hum	-1710.32	332.10	-5.150	3.62e-07 ***
windspeed	-2876.55	455.21	-6.319	5.40e-10 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 764.3 on 556 degrees of freedom
Multiple R-squared: 0.8529, Adjusted R-squared: 0.8457
F-statistic: 119.4 on 27 and 556 DF, p-value: < 2.2e-16

StepAIC

The **stepwise regression** (or stepwise selection) consists of iteratively adding and removing predictors, in the predictive model, in order to find the subset of variables in the data set resulting in the best performing model that is a model that **lowers prediction error.** AIC is **Akaike information criterion (AIC)**. Main approaches of stepwise selection are the forward selection, backward elimination and a combination of the two. The procedure has advantages if there are numerous potential explanatory variables, but it is also criticized for being a paradigmatic example of data dredging those significant variables may be obtained from

"noise" variables. The stepAIC() function also allows specification of the range of variables to be included in the model by using the scope argument.

VIF

A variance inflation factor (VIF) **detects multicollinearity** in regression analysis. Multicollinearity is when there's correlation between predictors (i.e. independent variables) in a model; its presence can adversely affect your regression results.

	GVIF	Df	GVIF^(1/(2*Df))
season	195.434341	3	2.408982
yr	1.049393	1	1.024399
mnth	469.866243	11	1.322676
holiday	1.129866	1	1.062952
weekday	1.200792	6	1.015365
weathersit	1.956980	2	1.182760
temp	6.965543	1	2.639232
hum	2.172805	1	1.474044
windspeed	1.206890	1	1.098586

3.2.Support Vector Regression

Support Vector Regression (SVR) works on similar principles as Support Vector Machine (SVM) classification. One can say that SVR is the adapted form of SVM when the dependent variable is numerical rather than categorical. A major benefit of using SVR is that it is a non-parametric technique. Unlike SLR, whose results depend on Gauss-Markov assumptions, the output model from SVR does not depend on distributions of the underlying dependent and independent variables. Instead the SVR technique depends on kernel functions. Another advantage of SVR is that it permits for construction of a non-linear model without changing the explanatory variables, helping in better interpretation of the resultant model. The basic idea behind SVR is not to care about the prediction as long as the error (ϵi) is less than certain value. This is known as the principle of maximal margin. This idea of maximal margin allows viewing SVR as a convex optimization problem. The regression can also be penalized using a cost parameter, which becomes handy to avoid over-fit. SVR is a useful technique provides the user with high flexibility in terms of distribution of underlying variables, relationship between independent and dependent variables and the control on the penalty term. Now let us fit SVR model on our sample data. R package "e1071" is required to call svm function.

3.3.Decision Tree Regression

Decision tree builds regression or classification models in the form of a tree structure. It breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with **decision**

nodes and **leaf nodes**. A decision node has two or more branches, each representing values for the attribute tested. Leaf node represents a decision on the numerical target. The topmost decision node in a tree which corresponds to the best predictor called **root node**. Decision trees can handle both categorical and numerical data.

'rpart' function is used for Decision Tree Regression Analysis. rpart() function helps establish a relationship between a dependant and independent variables so that a business can understand the variance in the dependant variables based on the independent variables.

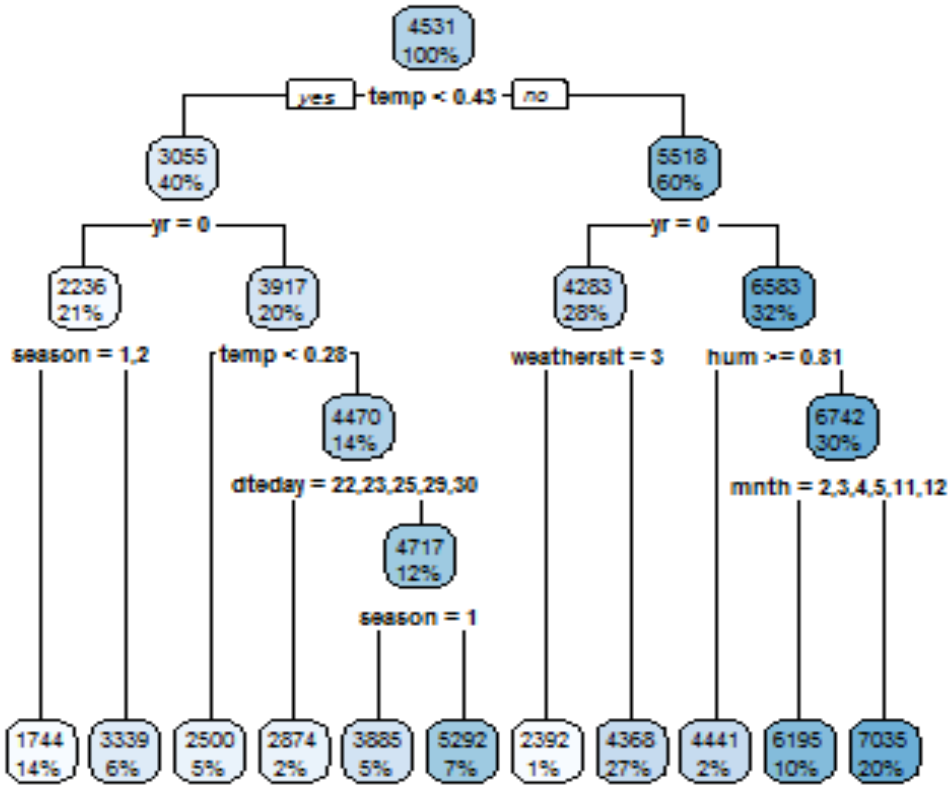


Fig 3.1 : Decision Tree Plot of the Data

3.4.Random Forest

Random forest, like its name implies, consists of a large number of individual decision trees that operate as an ensemble. Each individual tree in the random forest spits out a class prediction and the class with the most votes becomes our model's prediction. A large number of relatively uncorrelated models (trees) operating as a committee will outperform any of the individual constituent models. The low correlation between models is the key. Just like how investments with low correlations (like stocks and bonds) come together to form a portfolio that is greater than the sum of its parts, uncorrelated models can produce ensemble predictions that are more accurate than any of the individual predictions. The reason for this wonderful effect is that the trees protect each other from their individual errors.

‘Randomforest()’ is the function that is used for performing Random Forest Regression. The number of trees is set to 150.

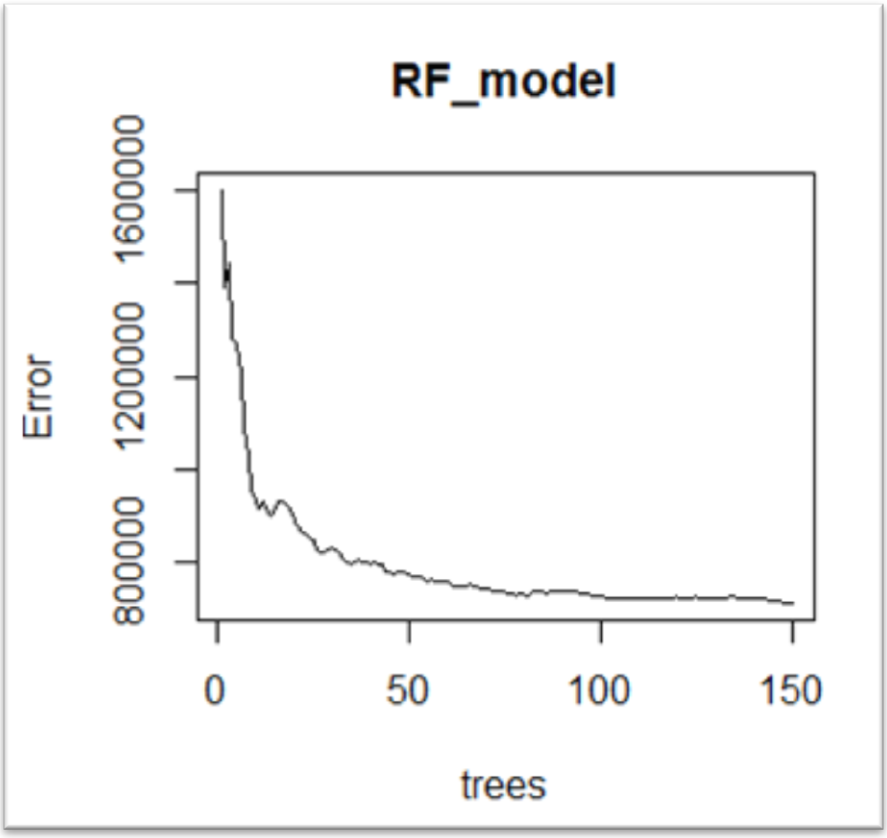


Fig 3.2 : Plot with Error vs Number of Trees

Chapter 4

4. Error Metrics

Predictive Modeling works on constructive feedback principle. You build a model. Get feedback from metrics, make improvements and continue until you achieve a desirable accuracy. Evaluation metrics explain the performance of a model. An important aspect of evaluation metrics is their capability to discriminate among model results. Simply, building a predictive model is not your motive. But, creating and selecting a model which gives high accuracy on out of sample data. Hence, it is crucial to check accuracy of the model prior to computing predicted values.

4.1.R^2 Value

The R^2 (or R Squared) metric provides an indication of the goodness of fit of a set of predictions to the actual values. In statistical literature, this measure is called the coefficient of determination. This is a value between 0 and 1 for no-fit and perfect fit respectively. It is the proportion of variance in the dependent variable that is predictable from the independent variable(s). Another definition is "(total variance explained by model) / total variance." So if it is 100%, the two variables are perfectly correlated, i.e., with no variance at all. A low value would show a low level of correlation, meaning a regression model that is not valid, but not in all cases. The formula for calculating R^2 is

$$R^2 = \frac{\sum (Y_i' - \overline{Y})^2}{\sum (Y_i - \overline{Y})^2}$$

Models	R-squared value
Linear Regression	0.8078
Support Vector Regression	**0.8544**
Decision Tree	0.8166
Random Forest	0.8455

Table 4.1 : R-squared values of Models

4.2.MAE, MSE, RMSE, MAPE

MAE

The Mean Absolute Error (MAE) is the average of the absolute differences between predictions and actual values. It gives an idea of how wrong the predictions were. The measure gives an idea of the magnitude of the error, but no idea of the direction (over or

under predicting). A value of 0 indicates no error or perfect predictions. The formula for MAE is

$$\mathrm{MAE} = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i|$$

Models	MAE
Linear Regression	0.058
Support Vector Regression	**0.054**
Decision Tree	0.058
Random Forest	0.057

Table 4.2 : MAE values of Models

MSE

The Mean Squared Error (or MSE) is much like the mean absolute error in that it provides a gross idea of the magnitude of error. It measures the average of the squares of the errors — that is, the average squared difference between the estimated values and what is estimated. MSE is a risk function, corresponding to the expected value of the squared error loss. The fact that MSE is almost always strictly positive (and not zero) is because of randomness or because the estimator does not account for information that could produce a more accurate estimate. The formula for calculating MSE is

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \tilde{y}_i)^2$$

Models	MSE
Linear Regression	0.000069
Support Vector Regression	**0.000052**
Decision Tree	0.000066
Random Forest	0.000055

Table 4.3 : MSE values of Models

RMSE

Taking the square root of the mean squared error converts the units back to the original units of the output variable and can be meaningful for description and presentation. This is called the Root Mean Squared Error (or RMSE). The 'squared' nature of this metric helps to deliver more robust results, which prevents cancelling the positive and negative error values. In other words, this metric aptly displays the plausible magnitude of error term. It avoids the use of absolute error values which is highly undesirable in mathematical calculations. As compared to mean absolute error, RMSE gives higher weightage and punishes large errors. RMSE metric is given by:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} (Predicted_i - Actual_i)^2}{N}}$$

Models	RMSE
Linear Regression	0.083
Support Vector Regression	**0.072**
Decision Tree	0.081
Random Forest	0.073

Table 4.4 : RMSE values of Models

MAPE

The MAPE (Mean Absolute Percent Error) measures the size of the error in percentage terms. 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 A_t is the actual value and F_t is the forecast value, this is given by:

$$\mathrm{M} = \frac{1}{n} \sum_{t=1}^{n} \left| \frac{A_t - F_t}{A_t} \right|$$

Models	MAPE (in %)
Linear Regression	18.35
Support Vector Regression	**16.50**
Decision Tree	18.84
Random Forest	20.53

Table 4.5 : MAPE values of Models

Chapter 5

5. Conclusion

From all the Error metrics, it is arrived at a conclusion that Multiple Linear Regression is more suitable for this dataset than Decision Tree Regression. R^2, which is giving the accuracy of the model, is high in Linear Regression. MAE, RMSE, MSE, MAPE are very low in Linear Regression.

Hence, Multiple Linear Regression model is used for predicting the test dataset. Before giving the dataset into the model, it is subjected to all the pre-processing techniques.

The Accuracy of Support Vector Regression Model is 85.44%.

MAPE value of Support Vector Regression Model is 16.50%.

This model is selected as best fit as the Accuracy of the model is high and error is very low.

Chapter 6

6.1.Appendix : Graphs

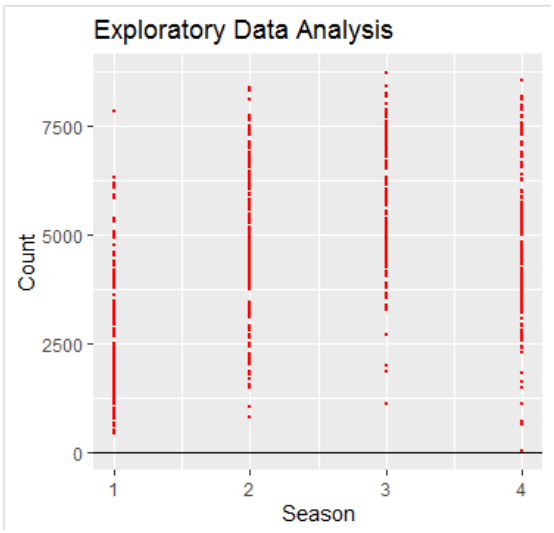


Fig 6.1 : Plot between Season and Count

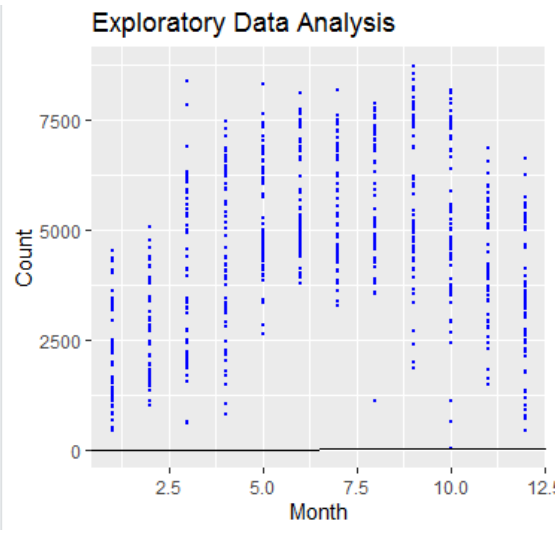


Fig 6.2 : Plot between Month and Count

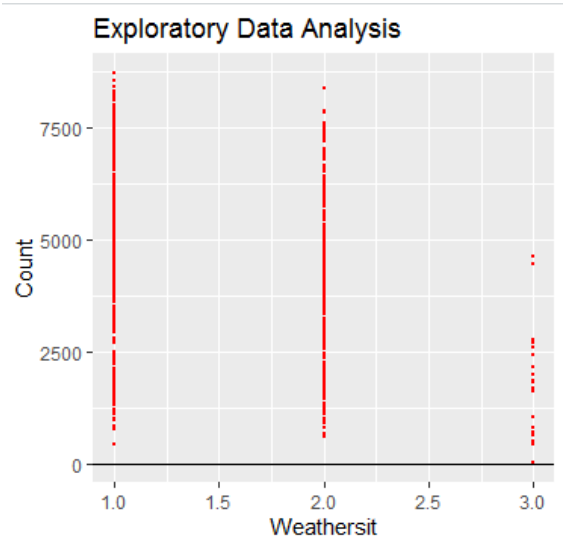


Fig 6.3 : Plot between Weathersit and Count

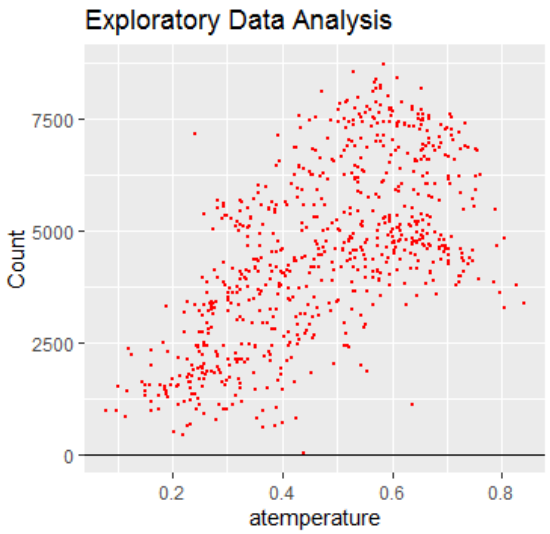


Fig 6.4 : Plot between Atemp and Count

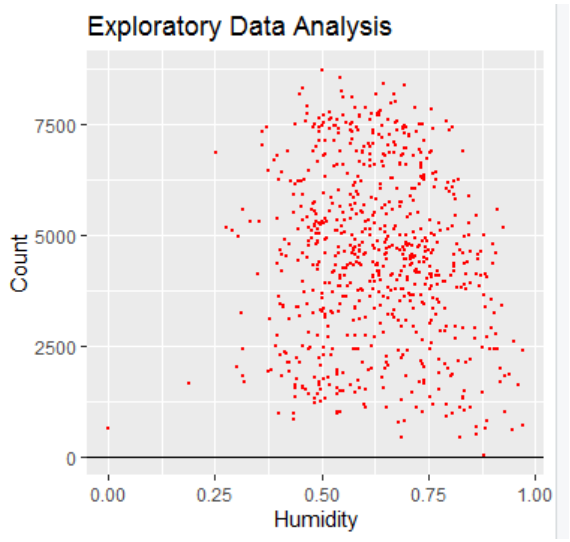


Fig 6.5 : Plot between Humidity and Count

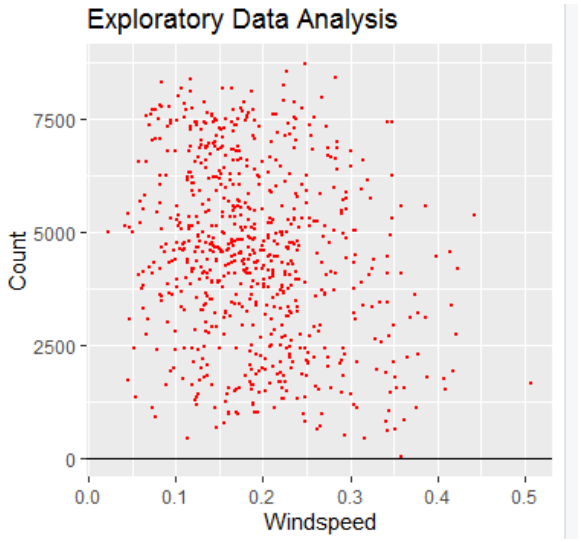


Fig 6.6 : Plot between Windspeed and Count

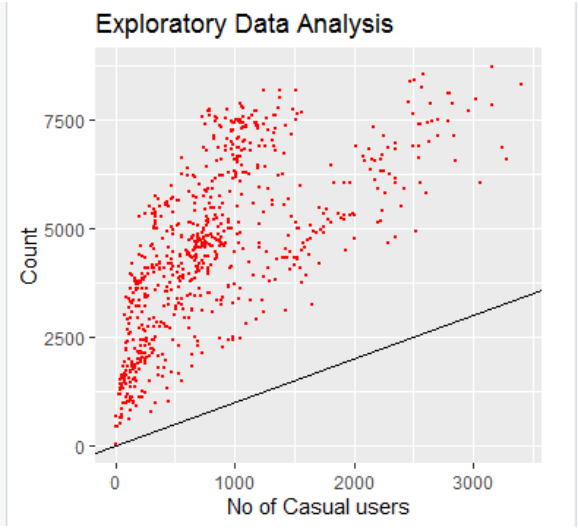


Fig 6.7 : Plot between Casual and Count

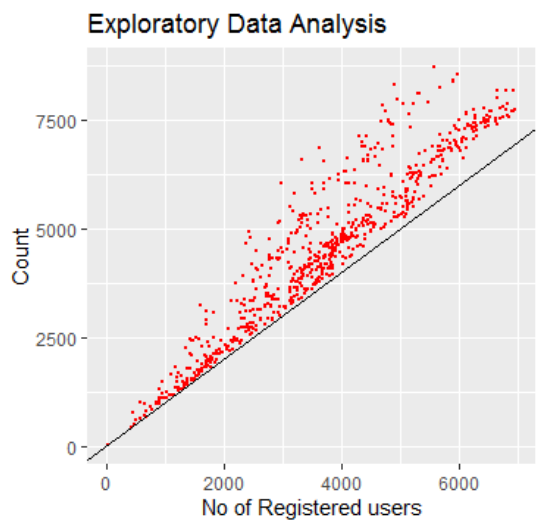


Fig 6.8 : Plot between Registered and Count

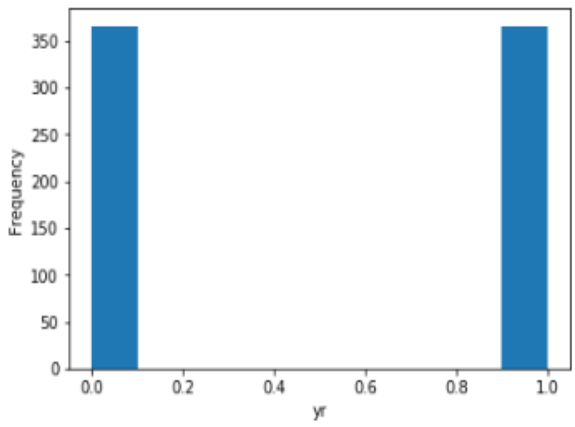


Fig 6.9 : Histogram of Year

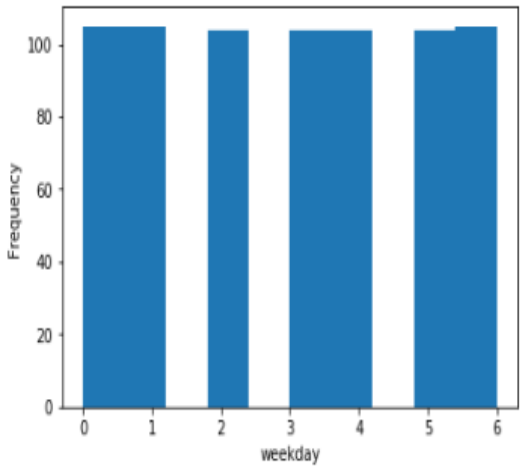


Fig 6.10 : Histogram of Weekday

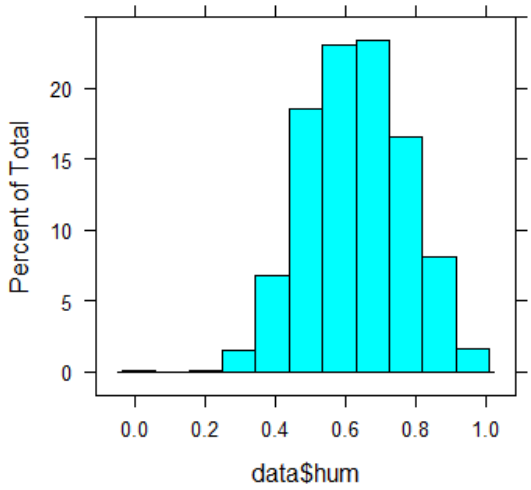


Fig 6.11 : Histogram of Humidity

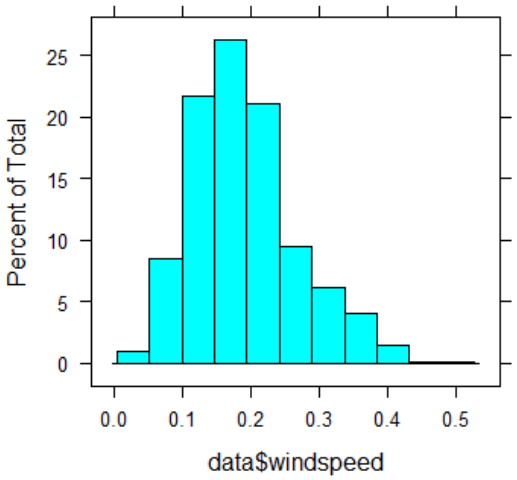


Fig 6.12 : Histogram of Windspeed

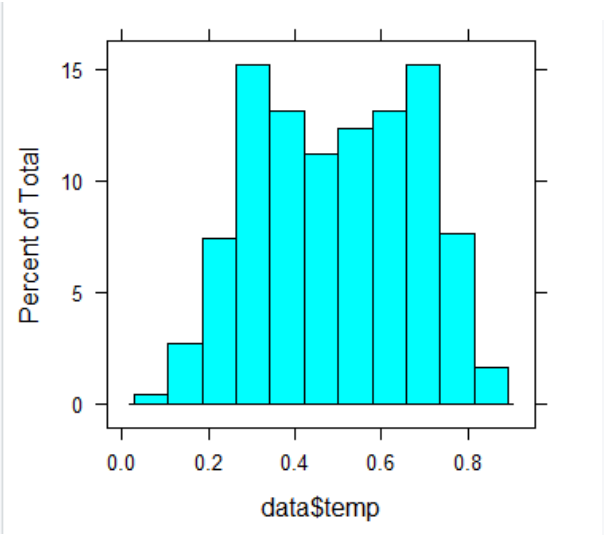


Fig 6.13 : Histogram of Temperature

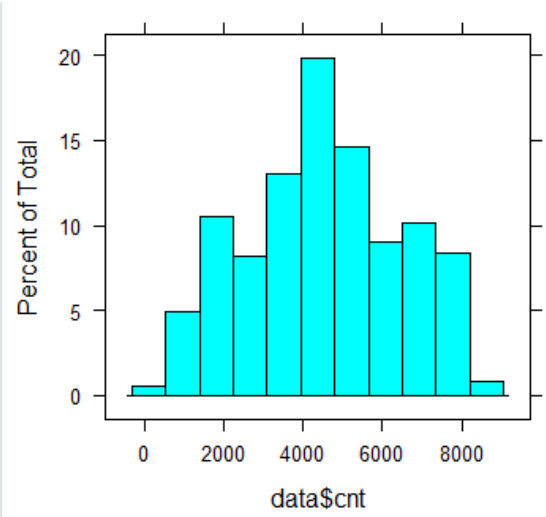


Fig 6.14 : Histogram of Count

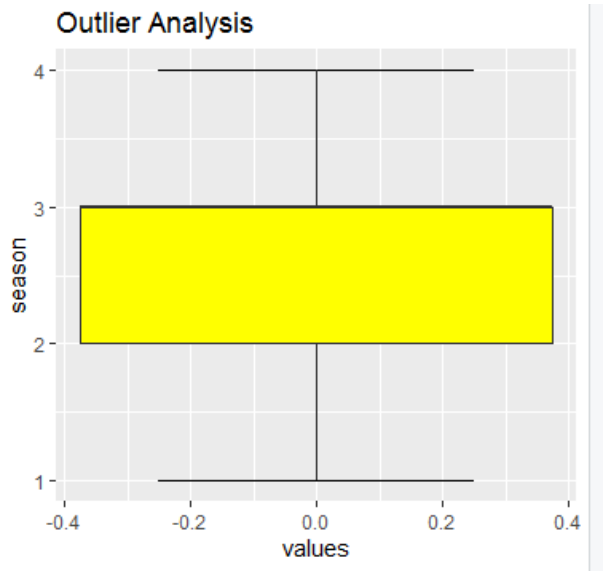


Fig 6.15 : Outlier Analysis of Season

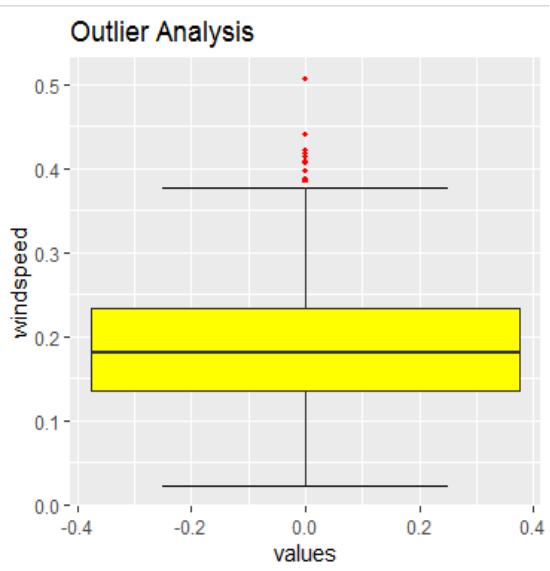


Fig 6.16 : Outlier Analysis of Windspeed

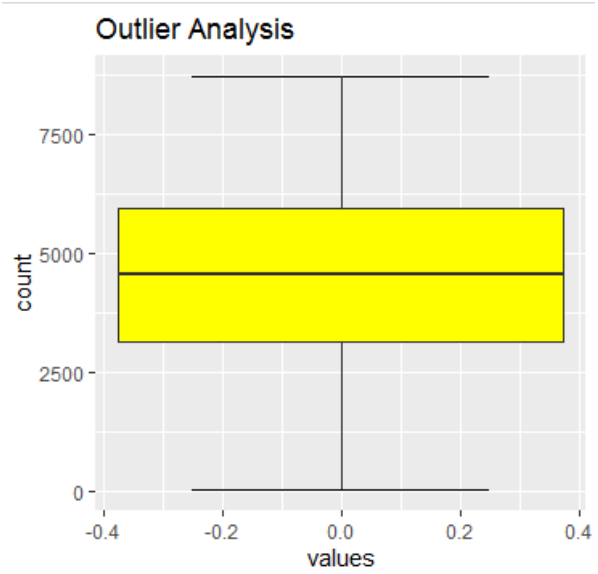


Fig 6.17 : Outlier Analysis of Count

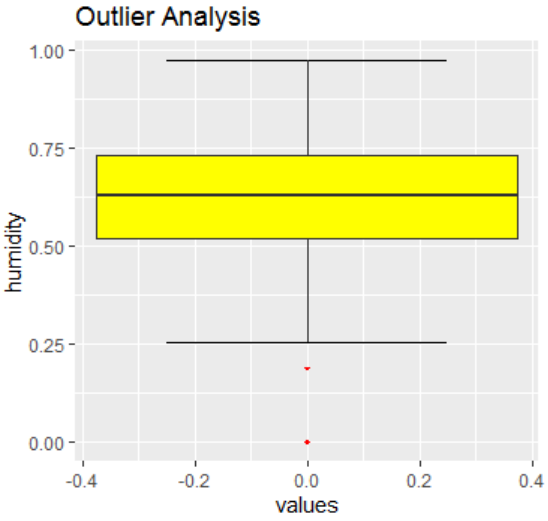


Fig 6.18 : Outlier Analysis of Humidity

6.2.Appendix : R Code

```
#------------------------------Loading the Libraries----------------------
rm(list=ls())
library(ggplot2)
library(caret)
library(DMwR)
library(MASS)
library(car)
library(rpart)
library(dummies)
library(corrgram)
library(randomForest)
library(e1071)
library(rpart.plot)
setwd("F:\\MBA\\Edwisor")
getwd()

#---------------------------------Reading the data-----------------------------

data <- read.csv("day.csv",header = T, na.strings = c("","", NA))

str(data)
sum(is.na(data))

#---------------------------------Understanding the Data-----------------------
ggplot(data=data,aes(y=cnt, x = season))+
  geom_point(mapping=NULL, data= NULL, size=0.5, color = "RED")+
```

```
  labs(x="Season", y="Count")+
  ggtitle("Exploratory Data Analysis")+
  geom_abline()

ggplot(data=data,aes(y=cnt, x = mnth))+
  geom_point(mapping=NULL, data= NULL, size=0.5, color = "BLUE")+
  labs(x="Month", y="Count")+
  ggtitle("Exploratory Data Analysis")+
  geom_abline()

ggplot(data=data,aes(y=cnt, x = weathersit))+
  geom_point(mapping=NULL, data= NULL, size=0.5, color = "RED")+
  labs(x="Weathersit", y="Count")+
  ggtitle("Exploratory Data Analysis")+
  geom_abline()

ggplot(data=data,aes(y=cnt, x = atemp))+
  geom_point(mapping=NULL, data= NULL, size=0.5, color = "RED")+
  labs(x="atemperature", y="Count")+
  ggtitle("Exploratory Data Analysis")+
  geom_abline()

ggplot(data=data,aes(y=cnt, x = hum))+
  geom_point(mapping=NULL, data= NULL, size=0.5, color = "RED")+
  labs(x="Humidity", y="Count")+
  ggtitle("Exploratory Data Analysis")+
  geom_abline()
```

```
ggplot(data=data,aes(y=cnt, x = windspeed))+
  geom_point(mapping=NULL, data= NULL, size=0.5, color = "RED")+
  labs(x="Windspeed", y="Count")+
  ggtitle("Exploratory Data Analysis")+
  geom_abline()

ggplot(data=data,aes(y=cnt, x = casual))+
  geom_point(mapping=NULL, data= NULL, size=0.5, color = "RED")+
  labs(x="No of Casual users", y="Count")+
  ggtitle("Exploratory Data Analysis")+
  geom_abline()

ggplot(data=data,aes(y=cnt, x = registered))+
  geom_point(mapping=NULL, data= NULL, size=0.5, color = "RED")+
  labs(x="No of Registered users", y="Count")+
  ggtitle("Exploratory Data Analysis")+
  geom_abline()

histogram(data$hum,data)
histogram(data$windspeed,data)
histogram(data$temp,data)
histogram(data$cnt,data)
#-----------------------------------Outliers Analysis-------------------------

ggplot(data=data,aes(y=data$hum))+
  stat_boxplot(geom = "errorbar", width = 0.5) +
```

```
  geom_boxplot(outlier.colour="red", fill = "yellow" ,outlier.shape=18, outlier.size=1, notch=FALSE)+
  labs(y="humidity", x="values")+
  ggtitle("Outlier Analysis")
ggplot(data=data,aes(y=data$weekday))+
  stat_boxplot(geom = "errorbar", width = 0.5) +
  geom_boxplot(outlier.colour="red", fill = "grey" ,outlier.shape=18, outlier.size=1, notch=FALSE)+
  labs(y="Weekday", x="values")+
  ggtitle("Outlier Analysis")

#----------------------------------Feature Engineering-------------------------

data$season=as.factor(data$season)
data$mnth=as.factor(data$mnth)
data$yr=as.factor(data$yr)
data$holiday=as.factor(data$holiday)
data$weekday=as.factor(data$weekday)
data$workingday=as.factor(data$workingday)
data$weathersit=as.factor(data$weathersit)
d1=unique(data$dteday)
df=data.frame(d1)
data$dteday=as.Date(df$d1,format="%Y-%m-%d")
df$d1=as.Date(df$d1,format="%Y-%m-%d")
data$dteday=format(as.Date(df$d1,format="%Y-%m-%d"), "%d")
data$dteday=as.factor(data$dteday)
```

```
#----------------------------------Feature Selection----------------------------
library(usdm)
vif(data[,3:15])
data=subset(data,select = -c(instant,casual,registered))

numeric_index = sapply(data,is.numeric)
## Correlation Plot
corrgram(data[,numeric_index], order = F,
        upper.panel=panel.pie, text.panel=panel.txt, main = "Correlation Plot")
cor(data[,numeric_index])

## Dimension Reduction
data = subset(data,select = -c(atemp))



#-------------------------------------Sampling-----------------------------------
set.seed(11994)
train.index = sample(1:nrow(data), 0.8 * nrow(data))

train = data[ train.index,]
test  = data[-train.index,]

#-----------------------------------Model Development--------------------------

detach("package:usdm", unload=TRUE)
lm.fit=lm(cnt~.,data=train)
summary(lm.fit)
```

```
vif(lm.fit)

step=stepAIC(lm.fit)


lm.fit1=lm(cnt~.-dteday,data=train)

summary(lm.fit1)

vif(lm.fit1)


lm.fit2=lm(cnt~season + yr + mnth + holiday + weekday + weathersit + temp + hum +
windspeed, data=train)

summary(lm.fit2)

vif(lm.fit2)


predictions_LM = predict(lm.fit2, test[,-12])

regr.eval(test[,12], predictions_LM)

rss <- sum((predictions_LM - test$cnt) ^ 2)

tss <- sum((test$cnt - mean(test$cnt)) ^ 2)

rsq <- 1 - rss/tss

rsq


#MAE = 587.0

#MSE = 692444

#RMSE = 831

#MAPE = 0.183539

#Rsq = 0.8078589
```

```
#----------------------------------Support Vector Machine-----------------------

svm_fit = svm(cnt ~ ., data = train)
summary(svm_fit)
prediction_SVM = predict(svm_fit, test[-12])
regr.eval(test[,12], prediction_SVM)
rss <- sum((prediction_SVM - test$cnt) ^ 2)
tss <- sum((test$cnt - mean(test$cnt)) ^ 2)
rsq <- 1 - rss/tss
rsq

#MAE = 543.4
#MSE = 524626
#RMSE = 724.31
#MAPE = 0.165052
#Rsq = 0.854425
#----------------------------------Decision Tree---------------------------------

rpart_fit = rpart(cnt ~ ., data = train, method = "anova")
rpart.plot(rpart_fit)
predictions_DT = predict(rpart_fit, test[,-12])
regr.eval(test[,12], predictions_DT)
rss <- sum((predictions_DT - test$cnt) ^ 2)
tss <- sum((test$cnt - mean(test$cnt)) ^ 2)
rsq <- 1 - rss/tss
rsq
```

```
#MAE = 587.8

#MSE = 660668

#RMSE = 812.8

#MAPE = 0.188450

#Rsq = 0.816676

#----------------------------------Random Forest--------------------------------


RF_model = randomForest(cnt ~ ., train, importance = TRUE, ntree = 150)

predictions_RF = predict(RF_model, test[,-12])

plot(RF_model)

regr.eval(test[,12], predictions_RF)

rss <- sum((predictions_RF - test$cnt) ^ 2)

tss <- sum((test$cnt - mean(test$cnt)) ^ 2)

rsq <- 1 - rss/tss

rsq


#MAE = 573.9

#MSE = 556472

#RMSE = 742.6

#MAPE = 0.2053828

#Rsq = 0.8455886
```

6.3.Appendix : Python Code

```
#Load libraries


import os
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import chi2_contingency
import seaborn as sns
from random import randrange, uniform
import datetime as dt
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeRegressor
import statsmodels.api as sm
from sklearn.ensemble import RandomForestRegressor
from sklearn.preprocessing import StandardScaler
from sklearn.svm import SVR
from sklearn.metrics import mean_squared_error,r2_score


#Setting working directory and reading the data


os.chdir("F://MBA//Edwisor")
data = pd.read_csv("day.csv")


# Understanding the Data


data.head(5)
```

```
data.describe()
data.shape

# Outlier Analysis

plt.boxplot(data['mnth'])
plt.ylabel('Month')
plt.boxplot(data['season'])
plt.ylabel('season')
plt.boxplot(data['windspeed'])
plt.ylabel('windspeed')
plt.boxplot(data['hum'])
plt.ylabel('humidity')

# visualization

plt.hist(data['windspeed'], bins = 10)
plt.ylabel('Frequency')
plt.xlabel('windspeed')
plt.show()
plt.hist(data['cnt'], bins = 10)
plt.ylabel('Frequency')
plt.xlabel('cnt')
plt.show()
```

```
# Feature Engineering

# Converting numeric into factor datatypes
data['season']= data['season'].astype('category')
data['yr']=data['yr'].astype('int')
data['mnth']=data['mnth'].astype('category')
data['holiday']=data['holiday'].astype('int')
data['workingday']=data['workingday'].astype('int')
data['weekday']=data['weekday'].astype('category')
data['weathersit']=data['weathersit'].astype('category')
data['dteday'] = pd.to_datetime(data['dteday'], errors='coerce')
data['dteday'] = data['dteday'].dt.day
data['dteday']=data['dteday'].astype('category')
data = data.drop(['instant','casual', 'registered',], axis=1)
data.head(5)

# Feature Selection

#Correlation plot
df_corr = data
#Set the width and hieght of the plot
f, ax = plt.subplots(figsize=(7, 5))

#Generate correlation matrix
corr = df_corr.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)


#droping corelated variable

data = data.drop(['atemp'], axis=1)



# Sampling


#dividing data into train and test

train, test = train_test_split(data, test_size=0.2)



# Modeling


#linear regression

#creating dummy variable

data_lm=data.copy()

cat_names = ["season", "dteday", "weathersit", "mnth","weekday"]

for i in cat_names:

    temp = pd.get_dummies(data_lm[i], prefix = i)

    data_lm = data_lm.join(temp)

drop = ['dteday', 'season', 'weathersit', 'weekday', 'mnth','cnt']

data_lm = data_lm.drop(drop, axis=1)

data_lm=data_lm.join(data['cnt'])
```

```
trainlm, testlm = train_test_split(data_lm, test_size=0.2)
LM_model = sm.OLS(trainlm.iloc[:,63], trainlm.iloc[:,0:62]).fit()
predictions_LM = LM_model.predict(testlm.iloc[:,0:62])
LM_model.summary()
trainlm.head(2)
fit_svr = SVR().fit(train.iloc[:,0:11], train.iloc[:,11])
predictions_SVR = fit_svr.predict(test.iloc[:,0:11])



#Decision Tree


fit_DT = DecisionTreeRegressor(max_depth=2).fit(train.iloc[:,0:11], train.iloc[:,11])
predictions_DT = fit_DT.predict(test.iloc[:,0:11])



#Random forest


RFmodel = RandomForestRegressor(n_estimators = 200).fit(train.iloc[:,0:11],
train.iloc[:,11])
predictions_RF = RFmodel.predict(test.iloc[:,0:11])



# Evaluation
#Defining Mape function
def MAPE(y_act, y_pred):
    mape = np.mean(np.abs((y_act - y_pred) / y_act)) * 100
    return mape
```

```
def RMSE(y_act, y_pred):
    rmse = np.sqrt(np.mean(np.square(y_act - y_pred)))
    return rmse

MAPE_LM = MAPE(testlm['cnt'], predictions_LM)
MAPE_LM
#19.39625%
RMSE_LM = RMSE(testlm['cnt'], predictions_LM)
RMSE_LM
#867.33

MAPE_DT = MAPE(test['cnt'], predictions_DT)
MAPE_DT
#24.0291%

RMSE_DT = RMSE(test['cnt'], predictions_DT)
RMSE_DT
#1146.90

MAPE_RF = MAPE(test.iloc[:,11], predictions_RF)
MAPE_RF
#17.6433%

RMSE_RF = RMSE(test.iloc[:,11], predictions_RF)
RMSE_RF
#710.85
```

```
MAPE_SVR = MAPE(test.iloc[:,11], predictions_SVR)
MAPE_SVR
#18.41555%

RMSE_SVR = RMSE(test.iloc[:,11], predictions_SVR)
RMSE_SVR
#2012.82

r2_LM =r2_score(predictions_LM, testlm['cnt'])
r2_LM
#0.79989

r2_DT =r2_score(predictions_DT, test['cnt'])
r2_DT
#0.47284

r2_RF =r2_score(predictions_RF, test['cnt'])
r2_RF
#0.85239

r2_SVR =r2_score(predictions_SVR, test['cnt'])
r2_SVR
#0.6249
```