SAI SRINIVAS BANALA

(Bike rental project report)

Contents

Chapter 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.

1.2. Data:

The given data contains 731 rows/observations and 16 variables/columns.

instant	dteday	season	yr	mnth	holiday	weekday	workingday	weathersit	temp	atemp	hum	windspeed
1	1/1/2011	1	0	1	0	6	0	2	0.3442	0.363625	0.805833	0.160446
2	1/2/2011	1	0	1	0	0	0	2	0.3635	0.353739	0.696087	0.248539
3	1/3/2011	1	0	1	0	1	1	1	0.1964	0.189405	0.437273	0.248309
4	1/4/2011	1	0	1	0	2	1	1	0.2	0.212122	0.590435	0.160296
5	1/5/2011	1	0	1	0	3	1	1	0.227	0.22927	0.436957	0.1869
6	1/6/2011	1	0	1	0	4	1	1	0.2043	0.233209	0.518261	0.089565
7	1/7/2011	1	0	1	0	5	1	2	0.1965	0.208839	0.498696	0.168726
8	1/8/2011	1	0	1	0	6	0	2	0.165	0.162254	0.535833	0.266804
9	1/9/2011	1	0	1	0	0	0	1	0.1383	0.116175	0.434167	0.36195

Sample of data

In which we have to predict the bike rental count using the given data in which 15 variables are independent variables and one dependent variable which we are going to predict.

1.3. Understanding of data

By looking into the data, we came to know that the dependent variable that which are going to predict is continuous, so it is not the classification model. We need to adopt Regression model for predicting the continuous variable which is count based on environmental and seasonal settings.

Out of 16 variables **cnt** is the target variable which is equal to the sum of causal and registered.

The casual target variable contains total bikes acquired by the customers who are not already registered means at random they hired the bike.

While registered variable represents the hired number of bikes only by the persons who are already registered and their historical customers.

Chapter 2

Methodology

2.1. Data Pre-processing:

This data pre-processing has many stages missing value analysis, outlier analysis, feature selection, feature scaling which the dimension reduction comes into existence.

For every data science problem this is the important stage that the concerned person should take care of. If the processing is perfect, then the model will be successful else we cannot predict the correct output.

Pre-processing of data is nothing but cleaning of the data removing the unwanted data and keeping the significant data which is useful for our model.

2.1.1. Data type Conversion

It means the data which we get from the client is raw data which may or may not contain proper shape so that we need to study the data and convert the data into the required data type like if the variables are continuous we need convert them to numerical variables and if there are classification variable then we need to convert into categories.

To do this firstly we need to import the data into the Working environment like R or python accordingly. Let's look at the snippet of code in both the languages. Firstly, set the working directory and import the data.

data_frame = read.csv("day.csv") - R language

there are different snippets for importing different kind of data like csv file, excel sheet etc. Snippet differs according to the data which we are going to import.

Now according to our dataset by studying on the data we came to know that

Numerical variables:	**Categorical variables:**
Instant	Season
Temp	Yr
atemp	mnth
hum	holiday
windspeed	weekday
casual	workingday
registered	weathersit
cnt	

There is also one more variable in our data set ‘dteday’ some information about this variable

Importing Dates

Dates can be imported from character, numeric, POSIXlt, and POSIXct formats using the *as.Date* function from the **base** package.

If your data were exported from Excel, they will possibly be in numeric format. Otherwise, they will most likely be stored in character format.

Example:

```
dates = c ("05/27/84", "07/07/05")
better Dates = as. Date (dates, format = "%m/%d/%y")
```

2.1.2. Missing value Analysis

There are no any Missing values found in the data so I did not go for it. Missing values is nothing but filling the missing values using mean or mode imputation.

2.1.3. Outlier Analysis:

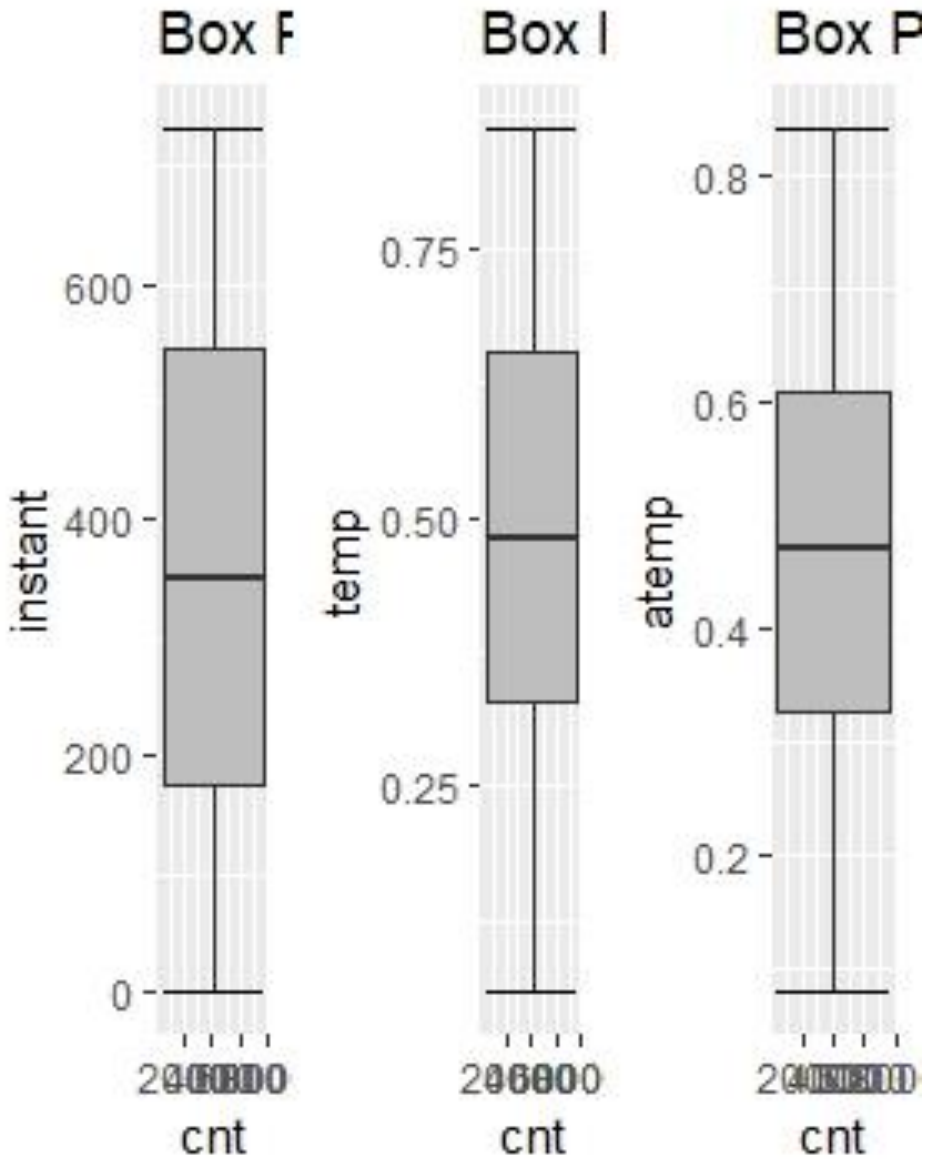
In statistics, an **outlier** is an observation point that is distant from other observations. An **outlier** may be due to variability in the measurement or it may indicate experimental error; the latter are sometimes excluded from the data set. An **outlier** can cause serious problems in statistical **analysis**.

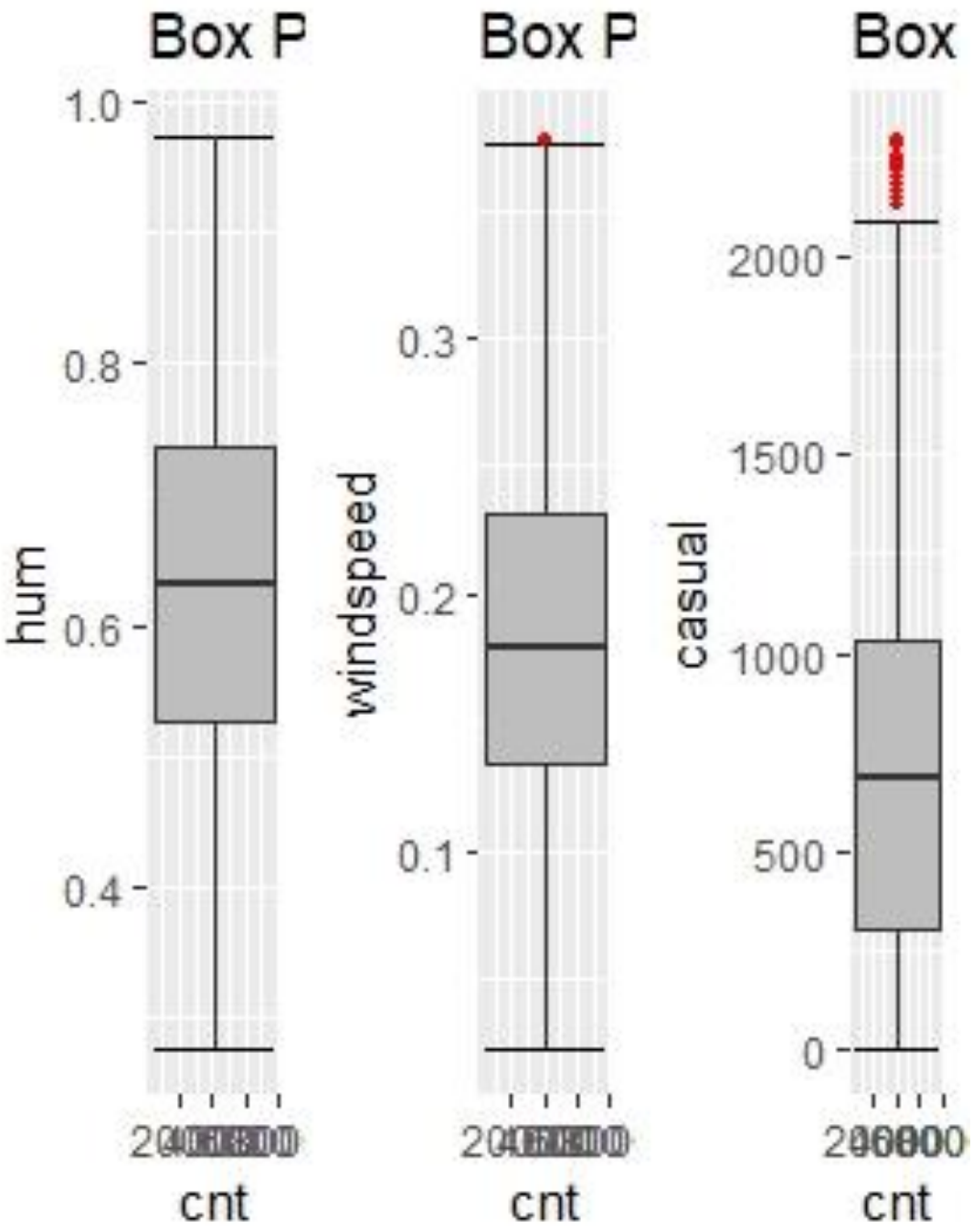
In simpler words outliers means taking all the observations of each column and finding the value which is falling away from the regular values.

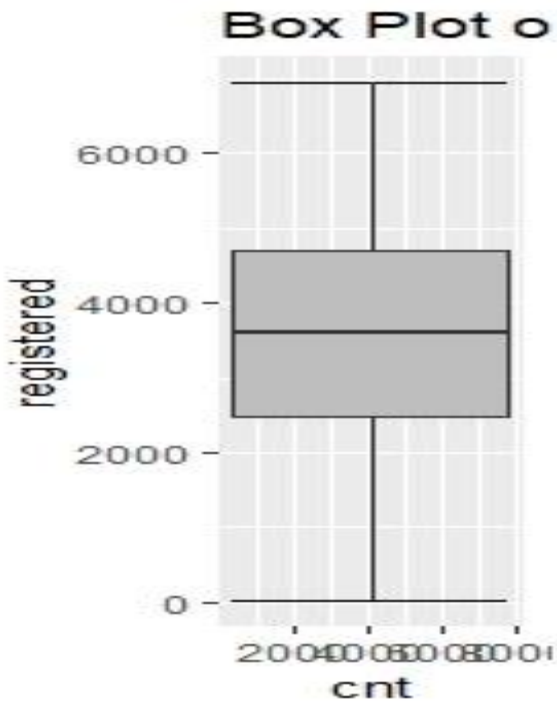
Generally, outliers are detected using boxplot method so that we can visualize clearly using this method, by passing some color coding in snippet we can easily identify the outliers.

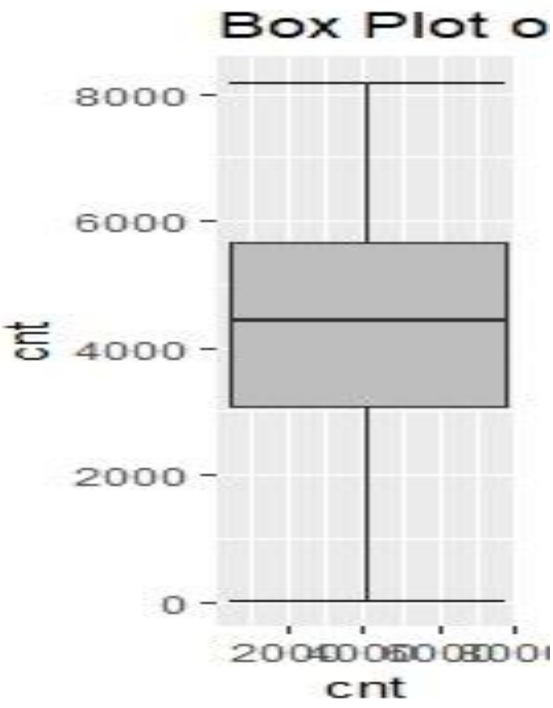
In the given bike rental data also contains outliers we have detected these outliers using boxplot method. We have found around 59 outliers in different variables like hum, windspeed and casual.

Actually there are two ways to deal with the outliers one is imputing the outlier using mean, mode, median (central tendency) and another one is KNN imputation. since, here the outliers are seen less and these might not impact the data so I decided to remove the rows which the outliers exists. After removing the outliers the effective data has been reduced to 676 observations which is not a huge difference So I am not imputing rather deleting the rows.


Box F
Box I
Box P
instant
600
400
200
0
cnt
temp
0.75
0.50
0.25
cnt
atemp
0.8
0.6
0.4
0.2
cnt


Box P
1.0
0.8
0.6
0.4
hum
cnt
Box P
0.3
0.2
0.1
windspeed
cnt
Box
2000
1500
1000
500
0
casual
cnt


Box Plot o
6000
4000
2000
0
registered
cnt


Box Plot o
8000
6000
4000
2000
0
cnt
cnt

2.1.4. Feature Selection

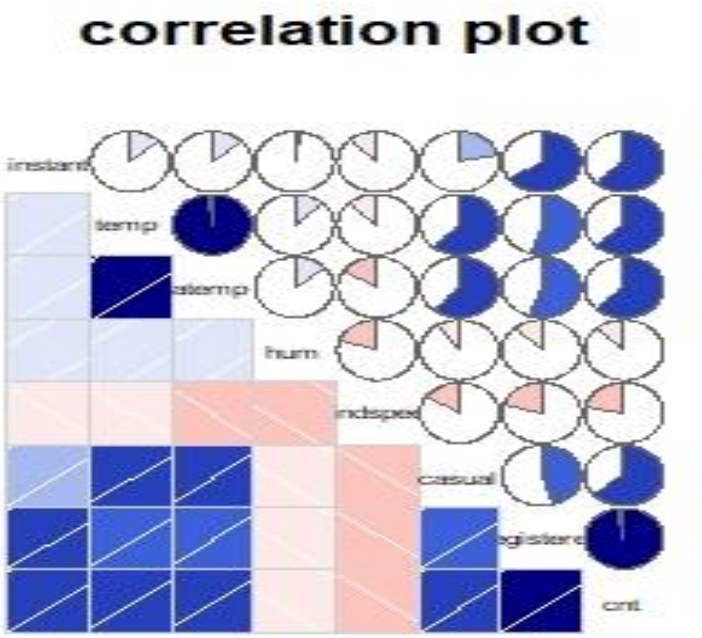
Feature selection is nothing but selection the independent variables which are more significant for implementing and developing the model. Technically, finding the significant variables and finding the highly correlated variables either negatively or positively.

There should be highly correlation between dependent and independent variable and no correlation between independent variables. For this there are different methods like correlation plot, chi-squared test of independence, ANOVA test.

Correlation plot: It is used for finding the highly correlated numerical variables. It is also used to find whether there is any multicollinearity problem.

Chi-Squared Test: It is used to find the independence between one categorical variable and numerical variable.

ANOVA: It is used to find the significance of variables against target variable.



The above pasted correlation plot clearly says that **temp** and **atemp** are highly positively correlated with each other. Registered and cnt are also highly positively correlated with each.

So, I am removing temp because it is only the measured value and it is not value that which the body has felt. I cannot remove **cnt** because it is the target variable. I cannot remove registered lonely because there is dependency between these three variables the sum of **casual** and **registered** gives the **cnt**(from data). So I decided to remove both casual and registered.

Then I went through the ANOVA test to find the significant categorical variable against the target variable. I came to know that every variable is significant.

2.1.5. Feature Scaling:

Feature scaling is a technique in which the dataset having quite different range of values are subjected to scale.

In order to scale the features there are usually two methods:-

1. Standardization
2. Normalization

The variables of the dataset are already normalized given in the problem statement explanation. Thus, we don't need to perform any kind of operation in order to scale the features.

2.1.6. Dimension Reduction:

Dimension reduction is nothing but removing the unwanted variables which are not required for the model that which we have extracted from correlation plot and ANOVA test.

I removed casual, registered, temp and instant variables which the data has been reduced to 12 variables.

2.2. Modelling

2.2.1. Decision Tree :

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 (e.g., Outlook) has two or more branches (e.g., Sunny, Overcast and Rainy), each representing values for the attribute tested. Leaf node (e.g., Hours Played) 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.

2.2.2. Random Forest:

The Random Forest is one of the most effective machine learning models for predictive analytics, making it an industrial workhorse for machine learning.

The **random forest** model is a type of additive model that makes predictions by combining decisions from a sequence of base models. More formally we can write this class of models as:

$$g(x)=f0(x)+f1(x)+f2(x)+...$$

where the final model g is the sum of simple base models fi. Here, each base classifier is a simple **decision tree**. This broad technique of using multiple models to obtain better predictive performance is called **model ensembling**. In random forests, all the base models are constructed independently using a **different subsample** of the data.

2.2.3. Model Selection:

Modelling is nothing but applying the machine learning algorithms according to the data and extract the patterns from the data. When the new data comes we feed the data into the model we implemented and predict the target values.

By observing the we came to that we need to predict the **cnt** variable which is continuous variable therefore, the problem statement does not come under the classification model it comes under regression model.

After applying various regression models, we need to find the error metrics for each model and for which model the error is less we need to freeze that model. Technically, it is called as performance evaluation of the model.

Firstly, we divide the given data into parts train data and test data.

Train data: 80% of the total data is called as train data from which the hidden patterns are extracted by applying machine learning algorithms.

Test data: Remaining 20% of the data is called as train data. test data is fed into the model to predict the target variables and compared with the actual target variable and the error metrics are calculated, to find the accuracy of the model.

After applying several regression models like Decision tree, Random forest and Random regression models, I concluded that Random Forest with 100 trees is giving better performance with less error compared to remaining regression models.

Therefore, I am freezing Random Forest Regression model.

Chapter 3

Model Evaluation

The method here I followed for evaluating the error of the model or performance of model is **RMSLE**(root mean square logarithmic Error).

Firstly, I have used RMSE and MAPE for finding the error, but I was unable to find the errors accurately as I was getting values greater than 1000. So, after going through the data I realized that we are predicting larger values so, I have considered **Root mean Square Logarithmic method** for error calculation. For this evaluation I have installed **mltools** package.

After performing decision tree regression model, RandomForest regression model and linear regression model the error coefficient is less for Random Forest and accuracy is more. So, Random Forest with 100 trees can be used to predict the values

R CODE

```
#Cleaning the R environment

rm(list = ls())


#installing the required packages used for model development and preprocessing techniques


install.packages(c("ggplot2","lsr","corrgram","rpart","DataCombine","DMwR","rattle","mltools","pROC","randomForest","inTrees"

                   ,"usdm","Metrics"))

x = c("ggplot2","lsr","corrgram","rpart","DataCombine","DMwR","rattle","mltools","pROC","randomForest","inTrees"

     ,"usdm","Metrics")


#x = c("ggplot2", "corrgram", "DMwR", "caret", "randomForest", "unbalanced", "C50", "dummies", "e1071", "Information",

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

#cross checking whether all the packages are installed or not


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

#removing the values

rm(x)
```

```
#setting the working directory in which our data set is present
setwd("F:/project/Bike Rental")
#checking the working directory
getwd()

#loading the data set into R environment
data_frame = read.csv("day.csv")

#checking the data types of data
str(data_frame)

#conversion of required data types into numeric a/c data
data_frame$instant = as.numeric(data_frame$instant)
data_frame$temp = as.numeric(data_frame$temp)
data_frame$atemp = as.numeric(data_frame$atemp)
data_frame$hum = as.numeric(data_frame$hum)
data_frame$windspeed = as.numeric(data_frame$windspeed)
data_frame$casual = as.numeric(data_frame$casual)
data_frame$registered = as.numeric(data_frame$registered)
data_frame$cnt = as.numeric(data_frame$cnt)

#converting into categorical variables
data_frame$season = as.factor(as.character(data_frame$season))
```

```
data_frame$yr = as.factor(as.character(data_frame$yr))

data_frame$mnth = as.factor(as.character(data_frame$mnth))

data_frame$holiday = as.factor(as.character(data_frame$holiday))

data_frame$workingday = as.factor(as.character(data_frame$workingday))

data_frame$weathersit = as.factor(as.character(data_frame$weathersit))

data_frame$weekday = as.factor(as.character(data_frame$weekday))

data_frame$dteday = as.Date(data_frame$dteday)

str(data_frame)


#checking is there any missing values in the data

sum(is.na(data_frame))


#no missing values found in the given data


#Outliers detection on nmerical variables

num_var =
c("instant","temp","atemp","hum","windspeed","casual","registered","cnt")


for (i in 1:length(num_var)) {

  assign(paste0("gn",i), ggplot(aes_string(y = (num_var[i]), x = "cnt"),data =
subset(data_frame))+

        stat_boxplot(geom = "errorbar" , width = 0.5) +

        geom_boxplot(outlier.color="red", fill = "grey" , outlier.shape=18,
outlier.size=1, notch=FALSE) +
```

```
        theme(legend.position="bottom")+
        labs(y=num_var[i],x="cnt")+
        ggtitle(paste("Box Plot of responded",num_var[i])))
  print(i)
  print(num_var[i])
}
options(warn = -1)
#plotting for clear  vision of outliers
gridExtra::grid.arrange(gn1,gn2,gn3,ncol=3)
gridExtra::grid.arrange(gn4,gn5,gn6,ncol=3)
gridExtra::grid.arrange(gn7,gn8,ncol = 2)

#--------Getting the outliers data from each numerical variable----------

for (i in num_var) {
  print(i)
  val = data_frame[,i][data_frame[,i] %in% boxplot.stats(data_frame[,i])$out]
  print(length(val))
  print(val)
}
#Remove all the rows which contains outliers because less outliers were observed and it might not impact the model after deletion of rows
for (i in num_var) {
```

```
  val = data_frame[,i][data_frame[,i] %in% boxplot.stats(data_frame[,i])$out]
  data_frame = data_frame[which(!data_frame[,i] %in% val),]

}

#checkingany missing value found
sum(is.na(data_frame))

#checking any outlier found
for (i in num_var) {
  val = data_frame[,i][data_frame[,i] %in% boxplot.stats(data_frame[,i])$out]
}

length(val)

#Correlation plot for detecting the insignificant numeric variables which are highly correlated
library(corrgram)
corrgram(na.omit(data_frame))
dim(data_frame)
corrgram(data_frame[,num_var],order = F, upper.panel = panel.pie, text.panel = panel.txt, main = "correlation plot" )
```

#now we are going for feature selection means which variable is most significant in predicted the dependent variable

cat_var =
c("season","yr","mnth","holiday","workingday","weathersit","weekday","dteday")

#performing ANOVA test on categorical variable against dependent variable

av_test = aov(cnt ~ season + yr + mnth + holiday + workingday + weekday +weathersit , data = data_frame)

summary(av_test)

#by performing anova we came to know that every categorical variable is significant for us and we need not remove any variable

#Dimension reduction(selecting the data required for our model)

data = subset(data_frame,select = -c(instant,casual,registered,temp))

#column names of processed data

names(data)

#writing the processed data into hard disk

write.csv(data,"processed_data.csv", row.names = F)

#----------MODEL ----------

#removing all the objects from R environment except processed data

```
rmExcept("data")
#DIviding the dataset into train and test data using sampling

train_index = sample(1:nrow(data), 0.8* nrow(data))
train = data[train_index,]
test = data[-train_index,]

##__  Decision tree regression model development

fit = rpart(cnt ~. , data = train, method = "anova")

#########  predict results for the test case dataset

predictions_DT_reg = predict(fit , test[,-12])

#names(test)

library("DMwR")
library("mltools")

#Error coefficient method used here is RMSLE Root Mean Square Log Error

rmsle( predictions_DT_reg,test[,12]) #0.25
```

```
library("rattle")
fancyRpartPlot(fit)

text(fit,pretty = 0)

#install.packages("pROC")
library("pROC")

##_______ RANDOM FOREST MODEL DEVELOPMENT _____________________#

#Random Forest Model
library("randomForest")


RandomForest_model = randomForest(cnt~., train, ntree = 100)
str(data)

as.Date(data$dteday)
```

```
#Extract the rules generated as a result of random Forest model
library("inTrees")
rules_list = RF2List(RandomForest_model)

#Extract rules from rules_list
rules = extractRules(rules_list, train[,-12])
rules[1:2,]

#Convert the rules in readable format
read_rules = presentRules(rules,colnames(train))
read_rules[1:2,]

#Determining the rule metric
rule_metric = getRuleMetric(rules, train[,-12], train$cnt)
rule_metric[1:2,]

#Prediction of the target variable data using the random Forest model
RandomForest_prediction = predict(RandomForest_model,test[,-12])
regr.eval(test[,12], RandomForest_prediction, stats = 'rmse')
rmsle( RandomForest_prediction , test[,12]) #0.17
```

####----------STATISTICAL MODEL----------#########

DEVELOPMENT OF LINEAR REGRESSION MODEL

```
library("usdm")
LR_data_select = subset(data, select = -(dteday))
colnames(LR_data_select)
vif(LR_data_select[,-12])
vifcor(LR_data_select[,-12], th=0.9)

####Execute the linear regression model over the data
linearRegression_model = lm(cnt~. , data = train)
summary(linearRegression_model)

colnames(test)

#Predict the data
linearRegression_model_predict_data = predict(linearRegression_model, test[,1:12])
install.packages("Metrics")
```

```
MAPE = function(y, yhat){
  mean(abs((y - yhat)/y))
}

library("Metrics")
rmsle(linearRegression_model_predict_data,test[,12])
```