```
#clearing Envorinment
rm(list=ls())

#Setting Working Directory
setwd("E:/data science and machine learning/BIKE RENTAL PREDICTIONS/R")

#checking the working directory.
getwd()

#Importing CSV file
bdf=read.csv("day.csv",header=T)

View(bdf)

#Getting Colnames
colnames(bdf)

#checking datatype of each variable
str(bdf)

typeof(bdf)

#Converting time stamp into POSIXct
bdf$dteday=as.POSIXct(strptime(bdf$dteday,"%Y-%m-%d"))

#checking Class
class(bdf$dteday)

#Extracting day from date.
bdf$day=as.numeric(format(bdf$dteday,"%d"))

#loading some of the libraries
x=c("ggplot2","Corrgram","DMwR","Caret","randomForest","unbalanced","C50","dummies","e10","MASS","rpart","gbm","ROSE")

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


#finding out number of missing value
missing_val=data.frame(apply(bdf,2,function(x){sum(is.na(x))}))

#giving names in dataframe
missing_val$Columns=row.names(missing_val)
row.names(missing_val)=NULL
names(missing_val)[1]="Missing_Percentage"

#converting to percentage
missing_val$Missing_Percentage=(missing_val$Missing_Percentage/nrow(bdf))*100

#OUTLIER ANALYSIS
numeric_index=sapply(bdf,is.numeric)
numeric_data=bdf[,numeric_index]
cnames=colnames(numeric_data)

a1=bdf$instant
a2=bdf$season
a3=bdf$yr
a4=bdf$mnth
a5=bdf$holiday
a6=bdf$weekday
a7=bdf$workingday
a8=bdf$weathersit
a9=bdf$temp
a10=bdf$atemp
a11=bdf$hum
a12=bdf$windspeed
a13=bdf$casual
a14=bdf$registered
a15=bdf$cnt
a16=bdf$day

#box plot

boxplot(a2,a3,a4,a6,
        main="Multiple boxplots for comparision",
        at=c(1,2,3,4),
        names=c("season","year","month","weekday"),
        las=2,
        col=c("orange","red"),
        border="blue",
        horizontal=FALSE,
        notch=TRUE
        )


boxplot(a5,a7,a8,
```

```
        main="Multiple boxplots for comparision",
        at=c(1,2,3),
        names=c("holiday","working day","weathersit"),
        las=2,
        col=c("orange","red"),
        border="blue",
        horizontal=FALSE,
        notch=TRUE
)

boxplot(a9,a10,a11,a12,
        main="Multiple boxplots for comparision",
        at=c(1,2,3,4),
        names=c("temp","atemp","humidity","windspeed"),
        las=2,
        col=c("orange","red"),
        border="blue",
        horizontal=FALSE,
        notch=TRUE
)

boxplot(a12,
        main="Multiple boxplots for comparision",
        at=c(1),
        names=c("windspeed"),
        las=2,
        col=c("orange","red"),
        border="blue",
        horizontal=FALSE,
        notch=TRUE
)

boxplot(a15,
        main="Multiple boxplots for comparision",
        at=c(1),
        names=c("count"),
        las=2,
        col=c("orange","red"),
        border="blue",
        horizontal=FALSE,
        notch=TRUE
)

#FEATURE SELECTION
install.packages("corrgram",dependencies = TRUE)
library(knitr)
library(fpca)
library(corrgram)

corrgram(bdf[,numeric_index],order=TRUE,upper.panel=panel.pie,text.panel=panel.txt,main="Correlation Plot")
bdf1=bdf

#Converting normalized temp & atemp to actual value

convert_temp=function(x,max,min)
{
  ct=x*(max-min)+min
  return(ct)
}
bdf1$con_temp=convert_temp(x=bdf1$temp,max=39,min=-8)
bdf1$con_atemp=convert_temp(x=bdf1$atemp,max=50,min=-16)

#converting weather situation
as.character(bdf1$weathersit,stringAsFactors=FALSE)
clr=bdf1$weathersit=="1"
bdf1$weathersit[clr]="clear"

cloudy=bdf1$weathersit=="2"
bdf1$weathersit[cloudy]="cloudy"

LR=bdf1$weathersit=="3"
bdf1$weathersit[LR]="Light Rain"

HR=bdf1$weathersit=="3"
bdf1$weathersit[HR]="Heavy Rain"

win=bdf1$season=="4"
bdf1$season[win]="winter"

#converting working day
as.character(bdf1$workingday,stringAsFactors=FALSE)
hd=bdf1$workingday=="0"
bdf1$workingday[hd]="holiday"

wd=bdf1$workingday=="1"
```

```
bdf1$workingday[wd]="working"


#converting season into characters
as.character(bdf1$season,stringAsFactors=FALSE)
spr=bdf1$season=="1"
bdf1$season[spr]="spring"

smr=bdf1$season=="2"
bdf1$season[smr]="summer"

fal=bdf1$season=="3"
bdf1$season[fal]="fall"

win=bdf1$season=="4"
bdf1$season[win]="winter"

table(bdf1$season)
#calculating mean median & sd of temp with season.

#Calculating Mean,Median & SD of temperature with respect to every season
spring_season=subset(bdf1,season=="spring")$con_temp
mean.spring=mean(spring_season)
#5.99
median.spring=median(spring_season)
#5.43
sd.spring=sd(spring_season)
#4.82

summer_season=subset(bdf1,season=="summer")$con_temp
mean.summer=mean(summer_season)
#17.58
median.summer=median(summer_season)
#18.41
sd.summer=sd(summer_season)
#5.76

fall_season=subset(bdf1,season=="fall")$con_temp
mean.fall=mean(fall_season)
#25.19
median.fall=median(fall_season)
#25.58
sd.fall=sd(fall_season)
#3.32

winter_season=subset(bdf1,season=="winter")$con_temp
mean.winter=mean(winter_season)
#11.87
median.winter=median(winter_season)
#11.23
sd.winter=sd(winter_season)
#5.06

#HISTOGRAM OF TEMPERATURE
min(summer_season)
max(summer_season)
hist(x=summer_season,
     main="Temperature in the summer",
     xlab="Temperature in Celcius",
     ylab="days",
     xlim=c(3,33),
     ylim=c(0,45),
     breaks=15,
     border="black",
     col="yellow"
)

abline(v=mean.summer,lwd=2,lty=1,col="red")
text(x=15,y=20,labels=paste("Mean=",round(mean(summer_season),2),sep=""),col="red")

abline(v=median.summer,lwd=2,lty=1,col="blue")
text(x=20,y=25,labels=paste("Median=",round(median(summer_season),2),sep=""),col="blue")


min(winter_season)
max(winter_season)
hist(x=winter_season,
     main="Temperature in the winter",
     xlab="Temperature in Celcius",
     ylab="days",
     xlim=c(2,25),
     ylim=c(0,25),
     breaks=15,
     border="black",
     col="yellow"
```

```
)

abline(v=mean.winter,lwd=2,lty=1,col="red")
text(x=15,y=20,labels=paste("Mean=",round(mean(winter_season),2),sep=""),col="red")

abline(v=median.winter,lwd=2,lty=1,col="blue")
text(x=10,y=11,labels=paste("Median=",round(median(winter_season),2),sep=""),col="blue")


min(fall_season)
max(fall_season)
hist(x=fall_season,
     main="Temperature in the fall",
     xlab="Temperature in Celcius",
     ylab="days",
     xlim=c(14,33),
     ylim=c(0,45),
     breaks=15,
     border="black",
     col="yellow"
)

abline(v=mean.fall,lwd=2,lty=1,col="red")
text(x=22,y=25,labels=paste("Mean=",round(mean(fall_season),2),sep=""),col="red")

abline(v=median.fall,lwd=2,lty=1,col="blue")
text(x=26,y=30,labels=paste("Median=",round(median(fall_season),2),sep=""),col="blue")

min(spring_season)
max(spring_season)
hist(x=spring_season,
     main="Temperature in the spring",
     xlab="Temperature in Celcius",
     ylab="days",
     xlim=c(-5,20),
     ylim=c(0,40),
     breaks=15,
     border="black",
     col="yellow"
)

abline(v=mean.spring,lwd=2,lty=1,col="red")
text(x=7,y=10,labels=paste("Mean=",round(mean(spring_season),2),sep=""),col="red")

abline(v=median.spring,lwd=2,lty=1,col="blue")
text(x=2,y=5,labels=paste("Median=",round(median(spring_season),2),sep=""),col="blue")


#calculating mean,median & sd of count with season
summer_cnt=subset(bdf1,season=="summer")$cnt
mean_sumcnt=mean(summer_cnt)
#4992.3
median_sumcnt=median(summer_cnt)
#4941.5
sd_sumcnt=sd(summer_cnt)
#1695.977

win_cnt=subset(bdf1,season=="winter")$cnt
mean_wincnt=mean(win_cnt)
#4728.16
median_wincnt=median(win_cnt)
#4634.5
sd_wincnt=sd(win_cnt)
#1699.61

spr_cnt=subset(bdf1,season=="spring")$cnt
mean_sprcnt=mean(spr_cnt)
#2604.13
median_sprcnt=median(spr_cnt)
#2209
sd_sprcnt=sd(spr_cnt)
#1399.94

fall_cnt=subset(bdf1,season=="fall")$cnt
mean_fallcnt=mean(fall_cnt)
#5644.303
median_fallcnt=median(fall_cnt)
#5353.5
sd_fallcnt=sd(fall_cnt)
#1459.80


#scatter plot of temperature between registered & casual count
plot(x=1,y=1,xlab="Temperature in Celcius",ylab="Bike Rental",type="n",main="Scatter plot of registered and casual
with respect to count",xlim=c(0,40),ylim=c(0,7000))
```

```
points(bdf1$con_temp,bdf1$casual,pch=10,col="purple")
points(bdf1$con_temp,bdf1$registered,pch=10,col="orange")
legend("topright",legend = c("casual", "registered"), col = c("purple","orange"), pch = c(10, 10), bg = "grey")
abline(lm(bdf1$registered ~ bdf1$con_temp), lty = 6, col = "blue")
abline(lm(bdf1$casual ~ bdf1$con_temp), lty = 6, col = "red")
abline(lm(bdf1$registered ~ bdf1$con_temp), lty = 6, col = "hotpink")
registered=paste("cor = ", round(cor(bdf1$registered, bdf1$con_temp), 2), sep = "")
casual=paste("cor = ", round(cor(bdf1$casual, bdf1$con_temp), 2), sep = "")
legend("left",legend = c(registered, casual) , col = c('red', 'hotpink'),pch = c(10, 10), bg = "grey")

#Converting humidity & windspeed.
bdf1$con_winsspd=bdf1$windspeed*67
bdf1$con_humid=(bdf1$hum*100)

#BAR GRAPH
p1=tapply(bdf1$cnt,bdf1$season,FUN=sum)
barplot(p1,main="Bar plot of count with respect to season",
xlab="Season",
ylab="Count",
border="red",
col="blue",
density=3
)

pie(p1,col=c("yellow","2","3","4"))


#Bar graph between month & count
p2=tapply(bdf1$cnt,bdf1$mnth,FUN=sum)
barplot(p2,main="Bar plot of count with respect to month",
xlab="Month",
ylab="Count",
border="red",
col="blue",
density=3
)

#scatter plot between temperature & count
temp1=bdf1$con_temp
count=bdf1$cnt
plot(temp1,count,
     xlab="Temperature",
     ylab="Count",
     col="Blue")



#Difference between temp & atemp
temp1=bdf1$con_temp
count=bdf1$cnt
mean.temp1=mean(temp1)
plot(temp1,count,
     xlab="Temperature",
     ylab="Count",
     col="Blue")
abline(v=mean.temp1,lwd=2,lty=1,col="red")
text(x=10,y=15,labels=paste("Mean=",round(mean(temp1),2),sep=""),col="red")

temp2=bdf1$con_atemp
count=bdf1$cnt
mean.temp2=mean(temp2)
plot(temp1,count,
     xlab="Temperature",
     ylab="Count",
     col="Blue")
abline(v=mean.temp1,lwd=2,lty=1,col="red")
text(x=10,y=15,labels=paste("Mean=",round(mean(temp1),2),sep=""),col="red")



#bar Graph
p3=tapply(bdf1$cnt,bdf1$weathersit,FUN=sum)
barplot(p3,main="Bar plot of count with respect to weather",
        xlab="weathersit",
        ylab="Count",
        border="red",
        col="blue",
        density=3
)



#bar Graph
p4=tapply(bdf1$cnt,bdf1$workingday,FUN=sum)
barplot(p4,main="Bar plot of count with respect to workingday",
```

```
        xlab="working day",
        ylab="Count",
        border="red",
        col="blue",
        density=3
)

#scatter plot of humidity & windspeed
humidity=bdf1$con_humid
count=bdf1$cnt
plot(humidity,count,
     xlab="Humidty",
     ylab="Count",
     col="brown")

wind=bdf1$con_winsspd
count=bdf1$cnt
plot(wind,count,
     xlab="Windspeed",
     ylab="Count",
     col="darkblue")


#MODELING

bdf2=bdf
install.packages("dplyr")
library(dplyr)
bdf2=select(bdf2,-c(instant,dteday,registered,casual,holiday,atemp))
bdf2=bdf2[,c(10,1,2,3,4,5,6,7,8,9,11)]

#SAMPLING(DIVIDING data into training & test data)
train_index=sample(1:nrow(bdf2),0.8*nrow(bdf2))
train = bdf2[train_index,]
test = bdf2[-train_index,]



#LINEAR REGRESSION
library(rpart)
library(MASS)
library(DMwR)


#Linear regression model
lm_model = lm(cnt ~., data = train)
summary(lm_model)
#Predict
predictions_LR = predict(lm_model, test[,2:11])
rmse_LR=sqrt(mean((test$cnt-predictions_LR)^2))
#RMSE=978.67
SLE_LR=(log(predictions_LR+1)-log(test$cnt+1))^2
rmsle_LR=sqrt(mean(SLE_LR))
#RMSLE_LR=0.32

#DECISION TREE

#rpart for regression
library(rpart)
fit = rpart(cnt ~ ., data = train, method = "anova")

#Predict for new test cases
predictions_DT = predict(fit, test[,2:11])
rmse_DT=sqrt(mean((test$cnt-predictions_DT)^2))
#RMSE=995.34
SLE_DT=(log(predictions_DT+1)-log(test$cnt+1))^2
rmsle_DT=sqrt(mean(SLE_DT))
#RMSLE_DT=0.34

#RANDOM FOREST
library(randomForest)
library(RRF)
library(inTrees)

RF_Model=randomForest(cnt~.,train,importance=TRUE,ntree=500)
#extract rules
treelist=RF2List(RF_Model)

exec=extractRules(treelist,train[,-1])
#visualize some rules
exec[1:2,]
#Make rules more readable
readableRules=presentRules(exec,colnames(train))
readableRules[1:4,]
```

```
#get rule metrics
ruleMetric=getRuleMetric(exec,train[,-1],train$cnt)

ruleMetric[1:2,]
#prediction of test data using RF Model
RF_Prediction=predict(RF_Model,test[,-1])

rmse_RF=sqrt(mean((test$cnt-RF_Prediction)^2))

#RMSE=747.30

SLE_RF=(log(RF_Prediction+1)-log(test$cnt+1))^2
rmsle_RF=sqrt(mean(SLE_RF))
#RMSLE_RF=0.286

#RF with 300 trees
RF_Model=randomForest(cnt~.,train,importance=TRUE,ntree=300)
#extract rules
treelist=RF2List(RF_Model)

exec=extractRules(treelist,train[,-1])
#visualize some rules
exec[1:2,]
#Make rules more readable
readableRules=presentRules(exec,colnames(train))
readableRules[1:4,]

#get rule metrics
ruleMetric=getRuleMetric(exec,train[,-1],train$cnt)

ruleMetric[1:2,]
#prediction of test data using RF Model
RF_Prediction=predict(RF_Model,test[,-1])

rmse_RF=sqrt(mean((test$cnt-RF_Prediction)^2))

#RMSE=746.91

SLE_RF=(log(RF_Prediction+1)-log(test$cnt+1))^2
rmsle_RF=sqrt(mean(SLE_RF))
#RMSLE_RF=0.285


#RF with 700 trees
RF_Model=randomForest(cnt~.,train,importance=TRUE,ntree=700)
#extract rules
treelist=RF2List(RF_Model)

exec=extractRules(treelist,train[,-1])
#visualize some rules
exec[1:2,]
#Make rules more readable
readableRules=presentRules(exec,colnames(train))
readableRules[1:4,]

#get rule metrics
ruleMetric=getRuleMetric(exec,train[,-1],train$cnt)

ruleMetric[1:2,]
#prediction of test data using RF Model
RF_Prediction=predict(RF_Model,test[,-1])

rmse_RF=sqrt(mean((test$cnt-RF_Prediction)^2))

#RMSE=743.04

SLE_RF=(log(RF_Prediction+1)-log(test$cnt+1))^2
rmsle_RF=sqrt(mean(SLE_RF))
#RMSLE_RF=0.284


#RF with 100 trees
RF_Model=randomForest(cnt~.,train,importance=TRUE,ntree=100)
#extract rules
treelist=RF2List(RF_Model)

exec=extractRules(treelist,train[,-1])
#visualize some rules
exec[1:2,]
#Make rules more readable
readableRules=presentRules(exec,colnames(train))
readableRules[1:4,]

#get rule metrics
ruleMetric=getRuleMetric(exec,train[,-1],train$cnt)
```

```
ruleMetric[1:2,]
#prediction of test data using RF Model
RF_Prediction=predict(RF_Model,test[,-1])

rmse_RF=sqrt(mean((test$cnt-RF_Prediction)^2))

#RMSE=748.11

SLE_RF=(log(RF_Prediction+1)-log(test$cnt+1))^2
rmsle_RF=sqrt(mean(SLE_RF))
#RMSLE_RF=0.287


#KNN Prediction

library(class)
##predict test data
KNN_predictions=knn(train[,-1],test[,-1],train$cnt,k=5)
str(KNN_predictions)
KNN_predictions=as.numeric(as.character(KNN_predictions))

rmse_KNN=sqrt(mean((test$cnt-KNN_predictions)^2))
#rmse_knn=2050.061

SLE_Knn=(log(KNN_predictions+1)-log(test$cnt+1))^2
rmsle_KNN=sqrt(mean(SLE_Knn))
#rmsle_knn=0.53

##predict test data
KNN_predictions=knn(train[,-1],test[,-1],train$cnt,k=3)
str(KNN_predictions)
KNN_predictions=as.numeric(as.character(KNN_predictions))

rmse_KNN=sqrt(mean((test$cnt-KNN_predictions)^2))
#rmse_knn=2113.037

SLE_Knn=(log(KNN_predictions+1)-log(test$cnt+1))^2
rmsle_KNN=sqrt(mean(SLE_Knn))
#rmsle_knn=0.561


##predict test data
KNN_predictions=knn(train[,-1],test[,-1],train$cnt,k=7)
str(KNN_predictions)
KNN_predictions=as.numeric(as.character(KNN_predictions))

rmse_KNN=sqrt(mean((test$cnt-KNN_predictions)^2))
#rmse_knn=2224.079

SLE_Knn=(log(KNN_predictions+1)-log(test$cnt+1))^2
rmsle_KNN=sqrt(mean(SLE_Knn))
#rmsle_knn=0.62



##predict test data
KNN_predictions=knn(train[,-1],test[,-1],train$cnt,k=9)
str(KNN_predictions)
KNN_predictions=as.numeric(as.character(KNN_predictions))

rmse_KNN=sqrt(mean((test$cnt-KNN_predictions)^2))
#rmse_knn=2324.114

SLE_Knn=(log(KNN_predictions+1)-log(test$cnt+1))^2
rmsle_KNN=sqrt(mean(SLE_Knn))
#rmsle_knn=0.69
```