FIT5149 Assessment 1 : Mining Knowledge from Data

Student Name: Md. Saadman Hossain

Student ID: 31043313

Libraries used:

```
library(psych)
library(ISLR)
library(ggplot2)
library(GGally)
library(gridExtra)
library(cowplot)
library(lattice)
library(dplyr)
library(ggpubr)
library(randomForest)
library(Metrics)
library(yardstick)
library(car)
```

Importing required libraries

In [ ]:

```
library(psych)
library(ISLR)
library(ggplot2)
library(GGally)
library(gridExtra)
library(cowplot)
library(lattice)
library(dplyr)
library(ggpubr)
library(randomForest)
library(Metrics)
library(yardstick)
library(car)
```

Reading the train data

In [ ]:

```
data = read.csv('train.csv', header = TRUE)
```

In [ ]:

```
dim(data) #dimension if the data
```

In [ ]:

```
head(data)
```

Changing Hour, Seasons, Holiday, Functioning.Day to factor as they are categorical variables

Here the hour attribute is the hour of the day, and i think hour of the day affects the number of bikes rented considerably. turning it into a categorical variable is justifiable in this scenario. Season, Holiday and Functioning.Day also need to be converted to factor for further analysis and exploration.

In [ ]:

```
# changing Hour, Seasons, Holiday, Functioning.Day to factor as they are categorical va
riables

data$Seasons <- as.factor(data$Seasons)
data$Holiday <- as.factor(data$Holiday)
data$Functioning.Day <- as.factor(data$Functioning.Day)
data$Hour <- as.factor(data$Hour)
```

In [ ]:

```
head(data)
```

Changing a few attribute names

Rented.Bike.Count = BikeCount

Temperature = Temp

Dew.point.temperature = DPtemp

Solar.Radiation = SR

Functioning.Day = Fday

In [ ]:

```
names(data)[c(2,4,8,9,14)]<-c("BikeCount","Temp","DPtemp", "SR","Fday")
head(data)
```

Adding a day of the week, months variable and converting them to factor

In [ ]:

```r
## Day of the Week
data$Date = substr(data$Date,1,10)
days<-weekdays(as.Date(data$Date))
data$days=days
data$days <- factor(data$days, levels=c("Monday","Tuesday", "Wednesday", "Thursday", "Friday", "Saturday","Sunday"))
```

In [ ]:

```r
## Converting date to a date format in R
data$Date <- as.POSIXct(data$Date)
## Extracting Month Name
data$months <- format(data$Date,"%B")
data$months <- factor(data$months, levels=c("January","February", "March",
                                            "April", "May", "June","July", "August",
                                            "September", "October", "November", "December"))
```

In [ ]:

```r
head(data)
```

Here i am generating histograms for the numeric attributes to analyse their distribution

In [ ]:

```r
par(mfrow=c(4,3))
par(mar = rep(2, 4))
hist(data$BikeCount, col = 'lightblue')
hist(data$Temp, col = 'lightblue')
hist(data$Humidity, col = 'lightblue')
hist(data$Wind.speed, col = 'lightblue')
hist(data$Visibility, col = 'lightblue')
hist(data$DPtemp, col = 'lightblue')
hist(data$SR, col = 'lightblue')
hist(data$Rainfall, col = 'lightblue')
hist(data$Snowfall, col = 'lightblue')
```

Inference:

1.We can observe that BikeCount (response) attribute is skewed to the left. infact most of the numerical attributed other than Temp and Humidity are skewed. skewed data suggests that there will be outliers in the data, thus a log transformation for certain variables might be an option for modelling purposes.

2.Temp and humidity seems to be close to normal in distribution

3.Most of the data for solar radiation, rainfall and snowfall seems to be leaning towards zero, suggesting the lack of these attributes affecting the metropolitan area.

4.The data for visibility is also highly skewed to the right, which usually means suitable condition for bike riders.

In [ ]:

```
# visualising the categorical variables

par(mfrow=c(4,2))
barplot(table(data$Hour), col = 'lightblue')

barplot(table(data$Seasons), col = 'lightblue')
barplot(table(data$Holiday), col = 'lightblue')
barplot(table(data$Fday), col = 'lightblue')
barplot(table(data$days), col = 'lightblue')
barplot(table(data$months), col = 'lightblue')
```

inference

1. we can see here that there are four seasons which are almost equally distributed.
2. functioning day has way more on yes, holiday has way more on no holiday.
3. day of the week and month also seem to be approximately equally distributed.

In [ ]:

```
# distributions of users for each hour in the day


boxplot(data$BikeCount~data$Hour,xlab="Hour of day", ylab="Bike rental counts", col = 'lightblue')

#inference

# fairly low avg bike counts from 0-7 hours, moderate from 8-16, 17-22 high peaking at 18,
```

Inference

```
 fairly low avg bike counts from 0-7 hours (till 7 am), moderate from 8-16 (8 am to 4pm), 17-22 (5pm to 10 pm) high and peaking at 18 (6pm). This makes sense as
 around 8 am people start heading off to work/study and at around 6 pm they are
 heading home. highlights the 9-5 workday for regular people.
```

In [ ]:

```

```

In [ ]:

```
# using ggplot2 for boxplots of bike rentals over the 12 months.

ggplot(data, aes(BikeCount, months)) + geom_boxplot()

# considerable variation for each month indicating it will have effect on response variable
```

Inference

```
1.very low average bikecounts for the months of december,january and february. This probably happens due to holiday season.
2.Highest average is the month of june
3.we can observe as months passes, average bike counts increases. it peaks at june, then slowly starts decreasing
```

In [ ]:

```
#boxplot(data$BikeCount~data$days,xlab="Days", ylab="Bike rental counts", col = 'lightblue')

ggplot(data, aes(BikeCount, days)) + geom_boxplot()
```

not much variation in the data, probably means that it wont affect the demand for bike rentals as much. only small variation is noticed for friday with slightly less numbers on average.

In [ ]:

```
head(data)
```

creating a correlation matrix

In [ ]:

```
# creating a correlation matrix
Panel <- function(x, y, z, ...) {
    panel.levelplot(x,y,z,...)
    panel.text(x, y, round(z, 2))
}
#Define the color scheme
cols = colorRampPalette(c("Yellow","Red"))
#Plot the correlation matrix.
levelplot(round(cor(data[c(2, 4:11)]),3), col.regions = cols(100), main = "Correlation
 Matrix for Numeric Variables",
          xlab = 'Numeric Variables', ylab = 'Numeric Variables',
          scales = list(x = list(rot = 90)), panel = Panel)
```

inference

```
 1.High correlation between temp and dptemp attributes.
 2.Temp, dptemp, sr positive correlation with response variable BikeCount.
 3.Humidity, rainfall, snowfall, negative correlation with response variable Bik
eCount.
 4.Windspeed not very correlated with response variable BikeCount.
 5.Humidity has moderate negative correlation with windspeed, visibility and sol
ar radiation.
```

In [ ]:

```

```

Getting the structure of the data

In [ ]:

```
str(data)
```

In [ ]:

```
# descriptive stats

summary(data)
```

Generate observations from summary stats

```
Huge range for bike count, might indicate count varies through different hours
Data spread equally for all seasons
Huge range for snowfall, rain fall, solar radiation
```

In [ ]:

```
#using psych describe function
round(describe(data), 4)
```

atrributes with high skewness has high standard deviation. only temp and humidity seems to be close to normal distributions, other numeric variables are skewed to either left or right. this indicates volatality in the data.

Firstly i am attempting to fit the data with a multiple linear regression model with all the variables except Date

In [ ]:

```
model1 = lm(BikeCount~.-Date, data = data)
summary(model1)
```

We can see that the adjusted R squared is 0.6956, which means that the model is only able to expain only 66 percent of the data. The model needs improvment

In [ ]:

```

```

In [ ]:

```

```

Fit a multiple regression model to predict bikecount

here im taking log(BikeCount+1) to log transform the count as the response variable BikeCount has natural outliers. we also saw previously that the distribution for bikecount is skewed.

In [ ]:

```
# Fit a multiple regression model to predict bikecount

# here im taking log(BikeCount+1) to log transform the count as the response variable BikeCount has natural outliers

model1 = lm(log(BikeCount+1)~.-Date, data = data) #removing date as its useless
summary(model1)
```

Already improvement in adjusted R squared can be seen, as it increased to 0.8506 from 0.6956

In [ ]:

In [ ]:

which of the predictors can i reject the null hypothesis H0: j = 0?

In order to answer this question, we need to look at the p-value in the summary, which is indicated by Pr(>|t|). For the predictors which have the low p-value (less than 0.01), we can reject the null hypothesis. What are the predictors that have a strong association with the response variable?

We can see that snowfall and days doesnt have significant association with respone variable, so we remove those form the model

```
Hour,
Temp,
humidity,
visibility,
Dptemp,
SR,
rainfall,
wind speed,
seasons,
holiday,
fday


have strongest association with the response variable
```

I am removing snowfall and days from the model as they have weak association with the BikeCount

In [ ]:

```
model2 = lm(log(BikeCount+1)~.-Date, data = subset(data, select=c( -Snowfall, -days)))
summary(model2)
```

We can see that now model is simpler and R squared hasnt changed much, which is good indication.

In [ ]:

```
par(mfcol=c(2,2))
plot(model1, which = 1)
plot(model1, which = 2)
plot(model2, which = 1)
plot(model2, which = 2)
```

Residuals seem to be normal, although for a big dataset it isnt a huge concern.

In [ ]:

```
anova(model1,model2) # not much difference in the models
```

low p value form anova test indicates these models are statistically different.

categories for the categorical variables

In [ ]:

```
sapply(data[c(3,12,13,14,15,16)], unique)
```

Earlier we observed that Temp and Dtemp are highly correlated

In [ ]:

```
model3 = update(model2, . ~ . + Temp: DPtemp ) # using interaction term
summary(model3)
```

In [ ]:

```
anova(model2, model3)
```

the anova table small p value indicated that addinf the interaction term Temp: DPtemp was a success as both models are statistically different.

wind speed is no longer well associated (p value 0.127942) so removing it.

In [ ]:

```
model3 = update(model3, . ~ . - Wind.speed)
summary(model3)
```

Doing some more testing

In [ ]:

```
summary(update(model3, . ~ . - Rainfall + log(Rainfall+1)- SR + log(SR+1)))
```

we can see that log transformation of rainfall and solar radiation was a success as Adjusted R-squared improved quite a lot

In [ ]:

```
model4 = update(model3, . ~ . - Rainfall + log(Rainfall+1)- SR + log(SR+1))
summary(model4)
```

In [ ]:

```
# removing temp
model4 = update(model4, . ~ . - Temp)
summary(model4)
```

obtain 95% confidence intervals for the coefficients.

In [ ]:

```
influencePlot(model4, scale=5, id.method="noteworthy", main="Influence Plot", sub="Circ
le size is proportial to Cook's Distance" )
```

In [ ]:

```
par(mfcol=c(2,2))
plot(model4)
```

In [ ]:

```
test = read.csv("test.csv") # reading testing data

# changing Hour, Seasons, Holiday, Functioning.Day to factor as they are categorical va
riables

test$Seasons <- as.factor(test$Seasons)
test$Holiday <- as.factor(test$Holiday)
test$Functioning.Day <- as.factor(test$Functioning.Day)
test$Hour <- as.factor(test$Hour)

# changing a few names

# Rented.Bike.Count = BikeCount
# Temperature = Temp
# Dew.point.temperature = DPtemp
# Solar.Radiation = SR
# Functioning.Day = Fday



names(test)[c(2,4,8,9,14)]<-c("BikeCount","Temp","DPtemp", "SR","Fday")

## Day of the Week
test$Date = substr(test$Date,1,10)
days<-weekdays(as.Date(test$Date))
test$days=days
test$days <- factor(test$days, levels=c("Monday","Tuesday", "Wednesday", "Thursday", "F
riday", "Saturday","Sunday"))

## Converting date to a date format in R
test$Date <- as.POSIXct(test$Date)
## Extracting Month Name
test$months <- format(test$Date,"%B")
test$months <- factor(test$months, levels=c("January","February", "March",
                                            "April", "May", "June","July", "August",
                                            "September", "October", "November", "Decemb
er"))
```

MLR prediction and outputting prediction values into new column in the test data

In [ ]:

```
# predicting the test dataset BikeCount using the Multiple linear regression model
prediction_1=predict(model4,test)
test$logBikeCount=prediction_1

# transforming the log predictions back to real numbers for model efficiency testing pu
rpose
test$predicted_BikeCount_MLR=round(exp(test$logBikeCount),digits=0)-1

head(test)
```

In [ ]:

```

```

MAE

is measured by taking the average of the absolute difference between actual values and the predictions.

The Root Mean Square Error

is measured by taking the square root of the average of the squared difference between the prediction and the actual value. It represents the sample standard deviation of the differences between predicted values and observed values.

RSQ (R^2)

helps you to understand how well the independent variable adjusted with the variance in your model. That means how good is the model for a dataset.

In [ ]:

```
#testing using yardstick package, function metrics

metrics(test,BikeCount,predicted_BikeCount_MLR)
```

FOR MLR MODEL

According to this analysis, root mean squared error is 322.9604850, mean absolute error is 204.3744292 and r squared is 0.7546 which are decent results for a MLR implementation.

In [ ]:

```

```

Random Forest Model

Random forest (or random forests) is an ensemble classifier that consists of many decision trees and outputs the class that is the mode of the class's output by individual trees.

In [ ]:

```
set.seed(999)






# here im taking log(BikeCount+1) to log transform the count as the response variable BikeCount has natural outliers.
# number of trees selected is 100, should be suitable for this dataset.
forest <- randomForest(log(BikeCount+1) ~.,data,importance=TRUE, ntree=100)

forest
```

In [ ]:

```
```

%explained variance is a measure of how well out-of-bag predictions explain the target variance of the data. the RF model is able to explain 93.17 percent of the data.

Prediction for the RF model

In [ ]:

```
# predicting the test dataset BikeCount using the random forest algorithm model
prediction_2=predict(forest,test)
test$logBikeCount_1=prediction_2

# transforming the log predictions back to real numbers for model efficiency testing purpose
test$predicted_BikeCount_RF=round(exp(test$logBikeCount_1),digits=0)-1

head(test)
```

Highlighting the important attributes RF

In [ ]:

```
#highlighting the important attributes

importance(forest)
varImpPlot(forest)
```

Inference

```
1.Fday is most important as it direcly influences if bikes can be rented or not.
if its non functioning day, then bike rented would be zero.
2. Hour of the day is also hugely influencial. this is easy to explain as demand
s are higher at 8-18 hours as opposed to 0-6 for the day.
3. Rainfall affects whether people are able to ride bikes; if its raining, peopl
e are unlikely to rent bikes.
4. The heirarchy of important variables just makes sense.
```

In [ ]:

```
#testing using yardstick package, function metrics

metrics(test,BikeCount,predicted_BikeCount_RF)
```

In [ ]:

```

```

FOR RANDOM FOREST MODEL

According to this analysis, root mean squared error is 230.6900929, mean absolute error is 132.9063927 and r squared is 0.8830867

From the above testing it is pretty safe to say that the random forest model is able to predict the rental BikeCounts more accurately than the MLR model.

Some reasons for Random forest model outperforming MLR model for prediction task

```
1.Decision trees such as random forest (second model used in this assignment) ar
e able to handle messy data and relationships way better than regression models
 in general. In our case, there are many variables with complex relationships wh
ich allows the random forest model to shine.
2. Linear regression models require regularization to overcome overfitting where
as random forests have regularization inbuilt.
3. Random forest is a sum of piecewise function ( is a function defined by multi
ple sub-functions, where each sub-function applies to a different sub-domain).
4. Random forest runs efficiently on large datasets.
5. RF is able to handle thousands of variables if required, and is not required
 to delete attributes unlike MLR.
6. Missing data estimation in RF is a strong feature, wherese in MLR it needs to
be done manually.
7. RF also provides robust variable interaction detection methodology which need
s to be dsone manually in MLR (by traial and error and human intuition).
8. Overall RFs are just more suitable for the prediction task for this assignmen
t for the reasons stated above. If the data was less complex and variabled had s
traighforward linear relationships, MLR would be a decent option.
```

In [ ]:

In [ ]: