Predication of Bike Rental Count

Prajakta Deshmukh

(11/8/2020)

Contents

<u>Chapter 1 : Introduction</u>

1.1 Problem statement

The objective of this problem statement is **to Predication of bike rental count on daily** based on some parameters like the environmental and seasonal conditions.

Aim of this project is to understand the rental count and how to increase the system capacity. At the end we will able to predict the bike rent count on particular day so according to that we can improve business model.

With help of historical data we are able to predict bike rent count in future business.

1.2 Data

In data there are total 16 variables are given below

- instant: Record index number
- dteday: Date
- season: Season (1:springer, 2:summer, 3:fall, 4:winter)
- yr: Year (0: 2011, 1:2012)
- mnth: Month (1 to 12)
- holiday: weather day is holiday or not (extracted from Holiday 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

Predictor variables or independent variables are as follow:

Predictor variables
Season
Year
Month
Holiday
Weekdays
Workingdays
Temp
Weatherlist
Atemp
Windspeed
Cnt

Given below is the dataset which we will be using to predict the Bike rental count -

1	instant	dteday	season	yr	mnth	holiday	weekday	workingday	weathersit	temp	atemp	hum	windspeed	casual	registered	cnt
2	1	01-01-2011	1	0	1	0	6	0	2	0.344167	0.363625	0.805833	0.160446	331	654	985
3	2	02-01-2011	1	0	1	0	0	0	2	0.363478	0.353739	0.696087	0.248539	131	670	801
4	3	03-01-2011	1	0	1	0	1	1	1	0.196364	0.189405	0.437273	0.248309	120	1229	1349
5	4	04-01-2011	1	0	1	0	2	1	1	0.2	0.212122	0.590435	0.160296	108	1454	1562
6	5	05-01-2011	1	0	1	0	3	1	1	0.226957	0.22927	0.436957	0.1869	82	1518	1600
7	6	06-01-2011	1	0	1	0	4	1	1	0.204348	0.233209	0.518261	0.0895652	88	1518	1606
8	7	07-01-2011	1	0	1	0	5	1	2	0.196522	0.208839	0.498696	0.168726	148	1362	1510
9	8	08-01-2011	1	0	1	0	6	0	2	0.165	0.162254	0.535833	0.266804	68	891	959
10	9	09-01-2011	1	0	1	0	0	0	1	0.138333	0.116175	0.434167	0.36195	54	768	822

Chapter 2 : Methodology

Data Pre-processing

Data pre-processing first step is to Analysis of Data means transforming raw data in proper format required for modeling. Explore data and transform it, means the analysis of data.

EDA (Exploratory Data Analysis) plays important role in data analysis.

In Bike data set we analyzed data type of each column in dataset. We analyzed data distribution of dataset.

After exploring of dataset it is clear that season, year, month, holiday, weekdays, workingday, weathersit are categorical variables.

Data variables such as instant, dteday have no relation with cnt. Data variable cnt is combination of casual and registered.

From dteday we extracted day and visualization between day and cnt clearly show that there is no relationship between day and cnt.

After analysis it is clear that some data variables have no relationship with cnt:

- Data variable 'dteday' is combination of day, month, and year. In dataset we have 'year' and 'month' separate columns and 'day' extracted column has no relationship with cnt so we can drop 'dteday' variable.
- Data variable 'instant' which gives information about index number which also have no relationship with 'cnt' so we can drop 'instant' variable.
- Data variable 'cnt' is combination of 'casual' and 'registered' so we can drop 'casual' and 'registered' data variables.

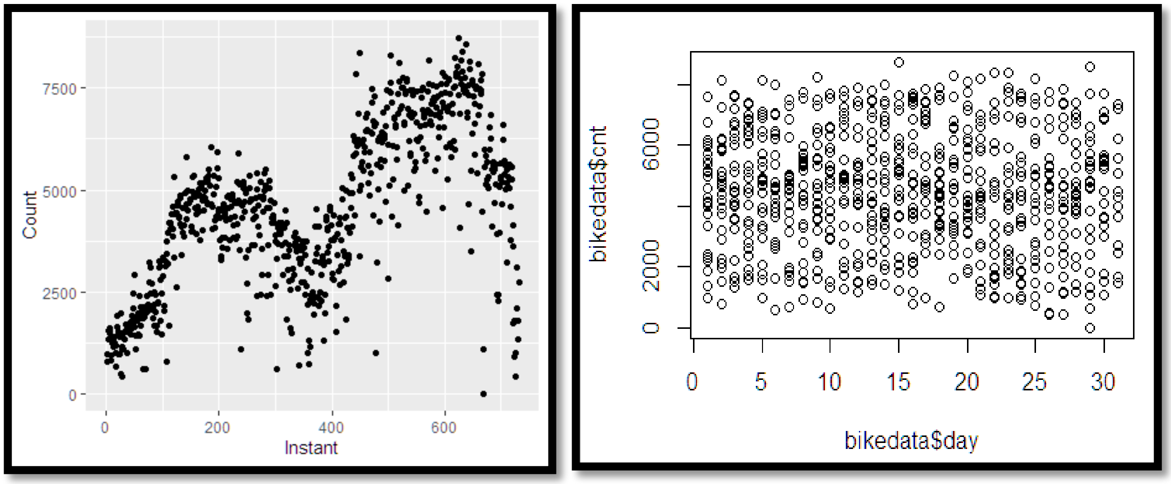


Fig: Visualization of instant and day with cnt variable

2.2.1 Missing value analysis

After exploring dataset next set is to check missing value in data set.

Presence of missing value causes error in prediction of variable. So, we need to detect and treat the missing values.

There is no missing value in data set.

2.2.2 Outlier analysis

Observations which are inconsistent with dataset are outlier in dataset. Outliers cause huge difference in mean of variable. And also cause error in prediction so, need to detect and treat outliers in dataset.

<u>Detection of outliers:</u>

To detect outlier in dataset, need to plot boxplot.

Outliers are present in continuous variable. Now in given data set the continuous variables are ‘hum’, ‘temp, ‘atemp’, ‘windspeed’.

Observations which are away from boxframe are outliers. **Data variable 'windspeed' and 'hum' has outliers**. Values which are above upper quartile and below lower quartile are outliers.

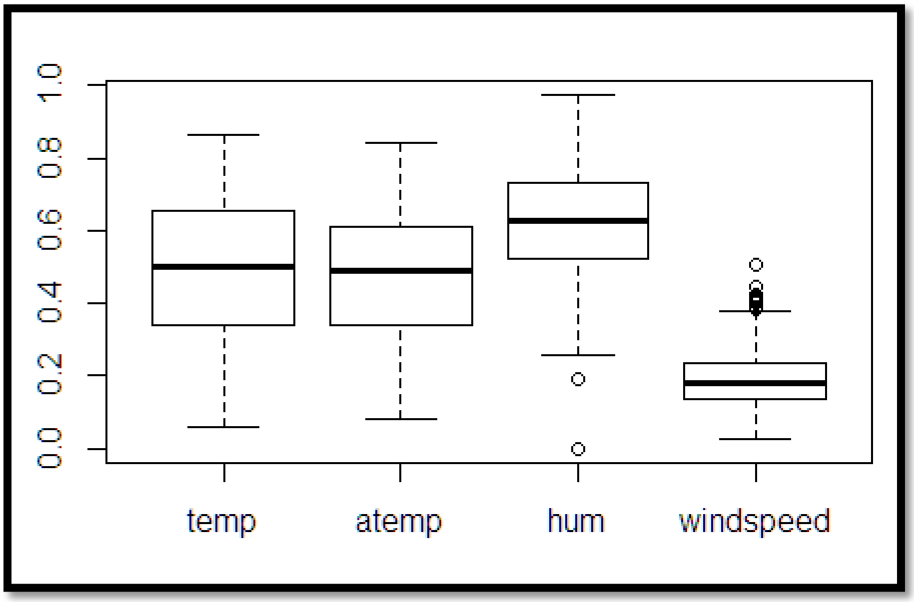


Fig: Visualization to detect outliers

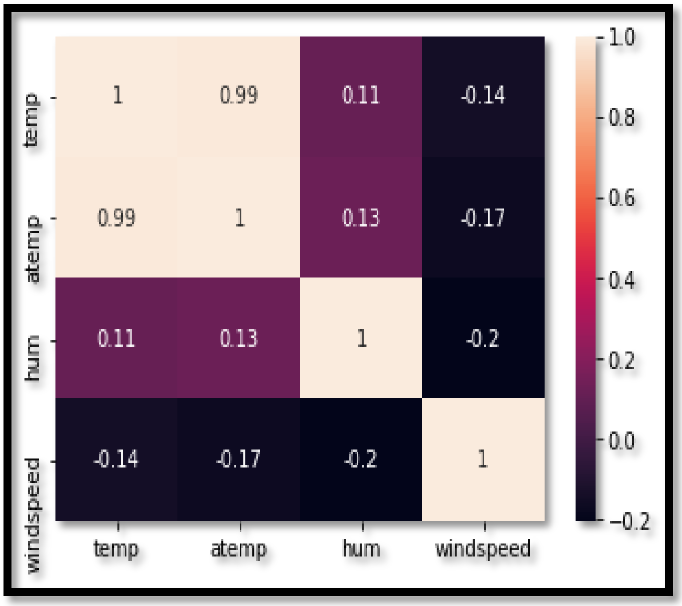
<u>Removal of outliers:</u>

Now we drop values which cause the outliers in dataset.

2.2.3 Feature selection

Feature selection is extracting meaningful features from dataset. It is also called as dimension reduction process.

For our model we have selected correlation analysis method. It gives association between data variables in range of -1 to 1.



From above visualization it is clear that ‘temp’ and ‘atemp’ are highly correlated so we can drop any of them to reduce dimension of data.

So, here we drop data variable ‘atemp’.

Now feature selection for categorical variable

For this we have performed **ANOVA test.**

With the help of this test we understood that some data variables have p-value less than 0.5 such data variables are ‘weekday’, ‘workingday’.

```
             df        sum_sq       mean_sq           F         PR(>F)
season      3.0  9.218466e+08  3.072822e+08  124.840203  5.433284e-65
Residual  713.0  1.754981e+09  2.461404e+06         NaN           NaN
             df        sum_sq       mean_sq           F         PR(>F)
yr          1.0  8.813271e+08  8.813271e+08  350.959951  5.148657e-64
Residual  715.0  1.795501e+09  2.511190e+06         NaN           NaN
             df        sum_sq       mean_sq          F         PR(>F)
mnth       11.0  1.042307e+09  9.475520e+07  40.869727  2.557743e-68
Residual  705.0  1.634521e+09  2.318469e+06        NaN           NaN
             df        sum_sq       mean_sq        F    PR(>F)
holiday     1.0  1.377098e+07  1.377098e+07  3.69735  0.054896
Residual  715.0  2.663057e+09  3.724555e+06      NaN       NaN
             df        sum_sq       mean_sq         F    PR(>F)
weekday     6.0  1.757122e+07  2.928537e+06  0.781896  0.584261
Residual  710.0  2.659257e+09  3.745432e+06       NaN       NaN
               df        sum_sq       mean_sq         F    PR(>F)
workingday    1.0  8.494340e+06  8.494340e+06  2.276122  0.131822
Residual    715.0  2.668333e+09  3.731935e+06       NaN       NaN
               df        sum_sq       mean_sq          F        PR(>F)
weathersit    2.0  2.679982e+08  1.339991e+08  39.718604  4.408358e-17
Residual    714.0  2.408830e+09  3.373711e+06        NaN           NaN
```

From the ANOVA Test analysis, it is clear that the variables 'workingday' and ‘weekday’ have p-values > 0.05. Thus, we accept the Null Hypothesis

Here we drop the ‘workingday’ and ‘weekday’ values.

2.2.4 Feature scaling

To check feature scaling in give data set we need to perform some test as follow:

1. Skewness test

```
#Skewness Test
from scipy.stats import skew
for x in num_col:
    print(x)
    skew_test = skew(bikedata.loc[:,x])
    print(skew_test)
```

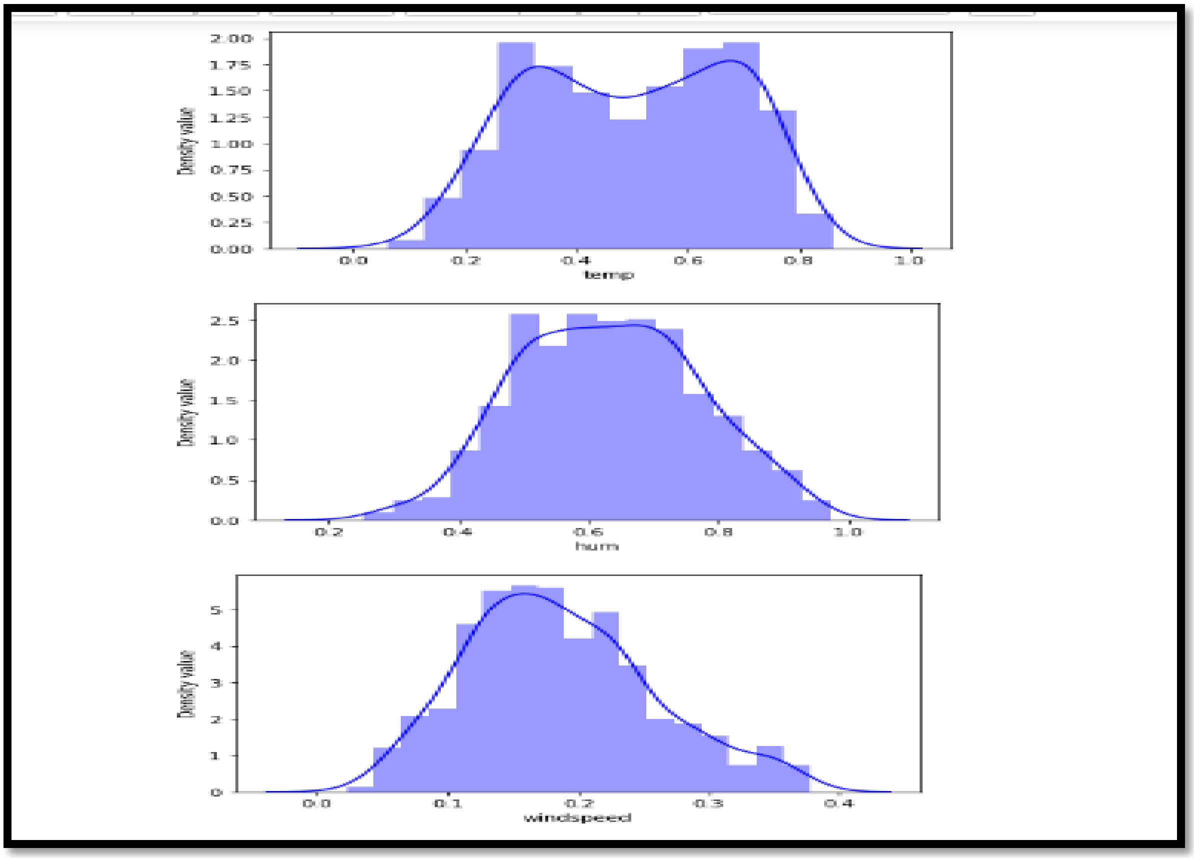
```
temp
-0.0690698243507108
hum
0.05235661609568474
windspeed
0.4400548001440976
```

From the Skewness test, it is clear that the data variables ‘temp’, ‘hum’, ‘windspeed’ are normally distributed as the skewness lies between -0.5 to +0.5

2. Normality check

If data is normally distributed then there is no need for feature scaling

To check this we need to plot histogram, Q-Q plot.



After skewness test and normality distribution check it is clear that data is **normally distributed there is no need for scaling.**

2.2.5 Sampling

After above all process the next step is modeling but before modeling we need to perform data sampling.

Data sampling is diving data in training and testing

Chapter 3 : Modeling

From the above Data Pre-processing, we draw the following conclusions:

1. The final variables of the data set are:
    - season
    - yr
    - mnth
    - holiday
    - temp
    - hum
    - windspeed
    - weathersit

2. The dependent variable of the dataset is cnt, which is a continuous data variable.

In Data Modeling at first, we need to identify the type of Problem statement.

In general, there are 4 kinds of Problem statement—

1. Predictive/Forecasting: The dependent variable has to be of type continuous.
2. Classification: The dependent variable has to be of type categorical.
3. Optimization
4. Unsupervised Learning

Thus, we conclude that **Bike Rental Count Prediction is a Predictive i.e. Regression Problem**.

We will be using the following Machine Learning Algorithms to analyze and build our model:

- Linear regression
- Decision tree
- Random forest tree
- K nearest neighbor approach (KNN)

3.1 Linear Regression:

The linear regression model explains the relationship between the continuous dependent variable and the independent variables (continuous or categorical).

We applied linear regressing to build model and the following was observed.

```
Call:
lm(formula = cnt ~ ., data = train_data)

Residuals:
    Min      1Q  Median      3Q     Max 
-3895.5  -360.0    61.4   459.5  3051.8 

Coefficients: (5 not defined because of singularities)
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  2514.28     473.91   5.305 1.63e-07 ***
season1     -1506.55     206.17  -7.307 9.55e-13 ***
season2      -535.74     245.01  -2.187 0.029191 *  
season3      -807.04     218.77  -3.689 0.000247 ***
season4           NA         NA      NA       NA    
yr0         -2006.43      67.31 -29.810  < 2e-16 ***
yr1               NA         NA      NA       NA    
mnth1         126.58     205.00   0.617 0.537178    
mnth2         255.22     207.14   1.232 0.218419    
mnth3         593.48     209.06   2.839 0.004694 ** 
mnth4         447.71     279.12   1.604 0.109284    
mnth5         642.41     299.31   2.146 0.032280 *  
mnth6         381.19     305.91   1.246 0.213267    
mnth7          25.50     322.93   0.079 0.937081    
mnth8         445.38     309.15   1.441 0.150253    
mnth9        1162.05     252.66   4.599 5.26e-06 ***
mnth10        629.27     194.88   3.229 0.001316 ** 
mnth11        139.04     183.13   0.759 0.448035    
mnth12            NA         NA      NA       NA    
holiday0      873.17     191.27   4.565 6.15e-06 ***
holiday1          NA         NA      NA       NA    
weathersit1  1853.15     239.42   7.740 4.74e-14 ***
weathersit2  1425.05     218.80   6.513 1.65e-10 ***
weathersit3       NA         NA      NA       NA    
temp         4840.27     476.72  10.153  < 2e-16 ***
hum         -1742.72     353.26  -4.933 1.07e-06 ***
windspeed   -2590.45     490.78  -5.278 1.88e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 785.6 on 554 degrees of freedom
Multiple R-squared:  0.8437,	Adjusted R-squared:  0.8377 
F-statistic: 142.4 on 21 and 554 DF,  p-value: < 2.2e-16
```

3.2 Decision Tree

Decision Tree is supervised learning algorithms. The goal of using a Decision Tree is to predict the value of the target variable by learning simple decision rules inferred from prior data (training data).

Further we applied Decision Tree to build model and the following was observed.

```
DecisionTreeRegressor(ccp_alpha=0.0, criterion='mse', max_depth=5,
                      max_features=None, max_leaf_nodes=None,
                      min_impurity_decrease=0.0, min_impurity_split=None,
                      min_samples_leaf=1, min_samples_split=2,
                      min_weight_fraction_leaf=0.0, presort='deprecated',
                      random_state=None, splitter='best')
```

3. 3 Random Forest

Random forest is method can be used for regression. It constructs a multitude of decision trees at training time and outputting the class that is prediction of the individual trees. It uses bagging technique to select tree.

We applied also this algorithm to build model and the following was observed.

```
RandomForestRegressor(bootstrap=True, ccp_alpha=0.0, criterion='mse',
                      max_depth=None, max_features='auto', max_leaf_nodes=None,
                      max_samples=None, min_impurity_decrease=0.0,
                      min_impurity_split=None, min_samples_leaf=1,
                      min_samples_split=2, min_weight_fraction_leaf=0.0,
                      n_estimators=300, n_jobs=None, oob_score=False,
                      random_state=None, verbose=0, warm_start=False)
```

3.4 KNN

KNN algorithm can be used for regression problems. KNN works on the concept of Euclidean distance among the centroid of data points specified.

In our model, we did try to fit the KNN model and the following was observed.

```
KNeighborsRegressor(algorithm='auto', leaf_size=30, metric='minkowski',
                    metric_params=None, n_jobs=None, n_neighbors=3, p=2,
                    weights='uniform')
```

Chapter 4 : Conclusion

After modeling it is very important to check accuracy and find perfect model which can predict accurate values.

4.1 Model evaluation

For evaluation we are using

1. **MAPE (Mean Absolute Percentage Error)**

It represents the mean of the absolute v/s predicted error percentile. We can say, that less the MAPE, better is the Model.

	Linear Regressing	Decision Tree	Random Forest	KNN
MAPE R	18.41743	25.83608	16.55055	47.91161
MAPE Python	16.71874517	18.076637	15.73214525	17.44366877

2. **R Square – Coefficient of Determination**

R square values are also known as the goodness fit for a model. It depicts the percentile of the dependent variable's variance that is collectively expressed by the independent variables. The more the R-square value, better is the model.

	Linear Regressing	Decision Tree	Random Forest	KNN
R-Square R	0.8220462	0.7299564	0.8827221	
R-Square Python	0.874478	0.8754469	0.92008377	0.8827864825

3. Accuracy

Thus, from the results of MAPE and R-SQUARE, we have found out Random Forest or Decision tree can be fit for our problem statement.

	Linear Regressing	Decision Tree	Random Forest	KNN
Accuracy R	82.38326%	73.6672%	83.43469%	52.08839%
Accuracy Python	83.28%.	81.92%.	84.27%.	82.56%.

After all above model evolution it is clear that **Random Forest** is best model to predict values in Python as well as in R.

Appendix A : R Code

```
#Removed all the existing objects
rm(list = ls())
# set working directory
setwd("C:/Users/HP/Desktop/Bikerent")
getwd()
#Load data
bikedata = read.csv("bike.csv",header=TRUE)

####################### Understand data (Data anlysis) ########################

##to understand values of col
str(bikedata)

##
class(bikedata)


## to understand summary
summary(bikedata)

## to understand dimension
dim(bikedata)
```

from above understanding of data it is clear that some data type not correct

##data type change

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

str(bikedata)
```

now we find dependent and indepndent variables

after understanding variables as 'dteday','instant''casual','registered'does not

removal of variable which are not required for further process

```
#Extracting the day values from the date and storing into a new column - 'day'
bikedata$day=format(bikedata$dteday,"%d")
unique(bikedata$day)
```

#Using plot() function to visualize the relationship between the data column 'day' and dependent variable 'cnt'

```
plot(bikedata$day,bikedata$cnt)


library(ggplot2)

ggplot(bikedata, aes(instant, cnt)) + geom_point() + scale_x_continuous("Instant")+
scale_y_continuous("Count")


bikedata=subset(bikedata,select = -c(instant,dteday,casual,registered))

str(bikedata)

dim(bikedata)

########################## Missing value anlaysis ##############################


sum(is.na(bikedata))

summary(is.na(bikedata))

###From this it is clear that data has no missing value


## there is no missing value in dataset


############################ Outlier Anlaysis ################################

numeric_col = c('temp','atemp','hum','windspeed')

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


###  to detect outliers in continous variables

boxplot(bikedata[,c('temp','atemp','hum','windspeed')])
```

```
### With help of box plot we are able to understand that there are outliers in data
#### values above and below quartile are outliers now replace it with NULL

for (x in c('hum','windspeed'))
{
  value = bikedata[,x][bikedata[,x] %in% boxplot.stats(bikedata[,x])$out]
  bikedata[,x][bikedata[,x] %in% value] = NA
}

####Checking whether the outliers in the above defined columns are replaced by NULL or not
##
sum(is.na(bikedata$hum))
sum(is.na(bikedata$windspeed))
as.data.frame(colSums(is.na(bikedata)))
#Removing the null values
library(tidyr)
bikedata = drop_na(bikedata)
as.data.frame(colSums(is.na(bikedata)))

####################### feature selection  #######################

### Numeric/Continuous data variables of the dataset
print(numeric_col)

library(corrgram)
```

```
corrgram(bikedata[,numeric_col],order=FALSE,upper.panel = panel.pie,
         text.panel = panel.txt,
         main= "Correlation Analysis Plot of the Continuous variables")
##### From above it is clear that temp and atemp are highly corealted
bikedata = subset(bikedata,select = -c(atemp))
str(bikedata)

#### Categorical variables of dataset.

print(categorical_col)
for(x in categorical_col)
{
  print(x)
  anova_test = aov(cnt ~ bikedata[,x],bikedata)
  print(summary(anova_test))
}

######From the ANOVA Test analysis, it is clear that the variables
######['holiday','workingday'and 'weekday'] have p-values > 0.05.
######Thus, we drop these data variables.
bikedata = subset(bikedata, select=-c(weekday,workingday))
str(bikedata)


###################### Feature Scaling ##############################
```

```
#### Before performing data scaling we need to check data distribution

#### If data is normally distributed then no need to apply scaling technique.

#### If data is skewed then there is need to normalization of data by using scaling technique.

### QQCure,Histogram, skewness test on continous variables


qqnorm(bikedata$temp)

qqnorm(bikedata$hum)

qqnorm(bikedata$windspeed)


hist(bikedata$temp)

hist(bikedata$hum)

hist(bikedata$windspeed)


library(e1071)

num_col = c('temp','hum','windspeed')

for(x in num_col)

{

  print(x)

  skewtest = skewness(bikedata[,x])

  print(skewtest)

}


##### From above it si clear that data is normally distributed

##### There are total 8 variables 1 is dependent and other are independent variables

##### So, after pre-processing of data we get to know that problem statemnt is of predictive,
```

```
##### that is Regression type of business problem statement.

############################## Sampling of data ##############################



##### sampling of data into training and testing

categorical_col_updated = c('season','yr','mnth','weathersit','holiday')
library(dummies)
bike = bikedata
bike = dummy.data.frame(bike,categorical_col_updated)
dim(bike)
#Separating the depenedent and independent data variables into two dataframes.
library(caret)
set.seed(101)
split_val = createDataPartition(bike$cnt, p = 0.80, list = FALSE)
train_data = bike[split_val,]
test_data = bike[-split_val,]



############################## Modeling of data ##############################



#1. MEAN ABSOLUTE PERCENTAGE ERROR (MAPE)
MAPE = function(y_actual,y_predict){
```

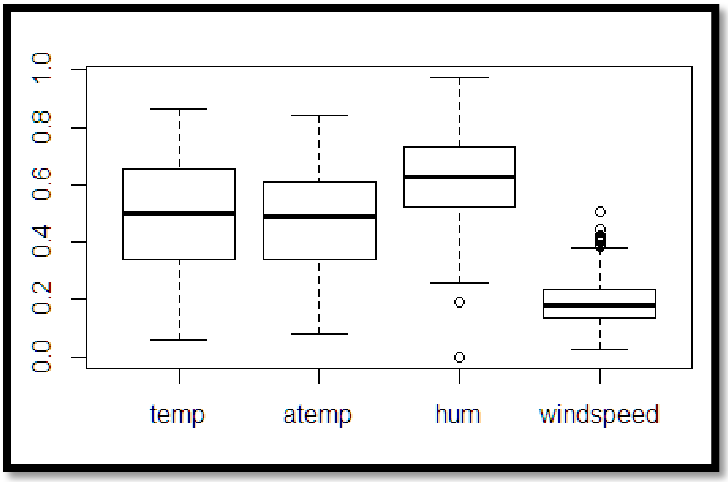
```
  mean(abs((y_actual-y_predict)/y_actual))*100
}

#2. R SQUARE error metric -- Coefficient of Determination
RSQUARE = function(y_actual,y_predict){
  cor(y_actual,y_predict)^2
}


##MODEL 1: DECISION TREES
library(rpart)
DT_model =rpart(cnt~., train_data, method = "anova" , minsplit=5)
DT_predict = predict(DT_model,test_data[-27])
DT_MAPE = MAPE(test_data[,27],DT_predict)
DT_R = RSQUARE(test_data[,27],DT_predict)
Accuracy_DT = 100 - DT_MAPE
print("MAPE: ")
print(DT_MAPE)
print("R-Square: ")
print(DT_R)
print('Accuracy of Decision Tree: ')
print(Accuracy_DT)

##MODEL 3: LINEAR REGRESSION
linear_model = lm(cnt~., train_data) #Building the Linear Regression Model on our dataset
```

```
summary(linear_model)

linear_predict=predict(linear_model,test_data[-27]) #Predictions on Testing data

LR_MAPE = MAPE(test_data[,27],linear_predict) # Using MAPE error metrics to check for the error rate and accuracy level

LR_R = RSQUARE(test_data[,27],linear_predict) # Using R-SQUARE error metrics to check for the error rate and accuracy level

Accuracy_Linear = 100 - LR_MAPE

print("MAPE: ")

print(LR_MAPE)

print("R-Square: ")

print(LR_R)

print('Accuracy of Linear Regression: ')

print(Accuracy_Linear)


##MODEL 2: RANDOM FOREST

library(randomForest)

RF_model = randomForest(cnt~., train_data, ntree = 300, importance = TRUE)

RF_predict=predict(RF_model,test_data[-27])

RF_MAPE = MAPE(test_data[,27],RF_predict)

RF_R = RSQUARE(test_data[,27],RF_predict)

Accuracy_RF = 100 - RF_MAPE

print("MAPE: ")

print(RF_MAPE)

print("R-Square: ")

print(RF_R)

print('Accuracy of Random Forest: ')
```

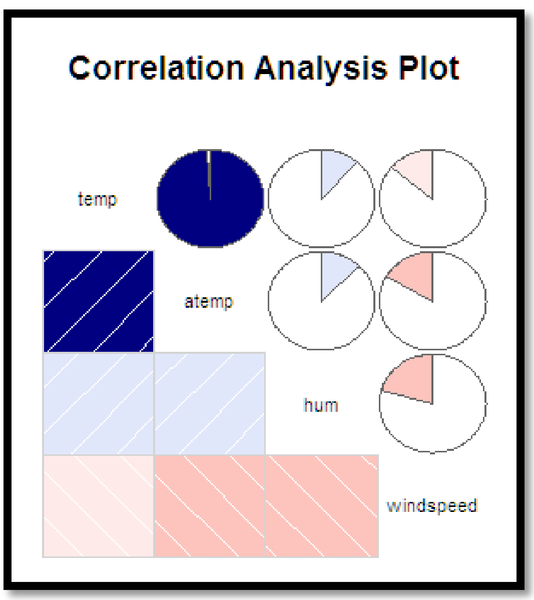
```
print(Accuracy_RF)


##MODEL 4: KNN
library('FNN')
set.seed(123)
KNN_model = FNN::knn.reg(train = train_data, test = test_data, y = train_data[,27], k = 3)
KNN_predict=ceiling(KNN_model$pred[1:27]) #Predicted values
KNN_MAPE = MAPE(test_data[,27],KNN_predict)
Accuracy_KNN = 100 - KNN_MAPE
print("MAPE: ")
print(KNN_MAPE)
print('Accuracy of KNN: ')
print(Accuracy_KNN)



Bike_res = data.frame('Actual_count' = test_data[,27], 'Predicted_count' = RF_predict )
write.csv(Bike_res,"BIKE_RESULT_R.csv",row.names=FALSE)
```

Appendix B : Figures

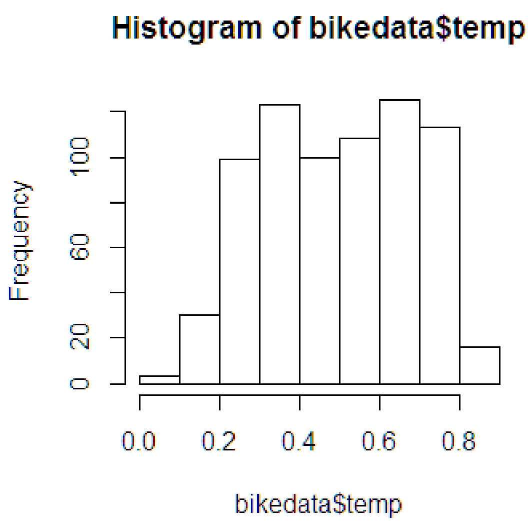
Outlier

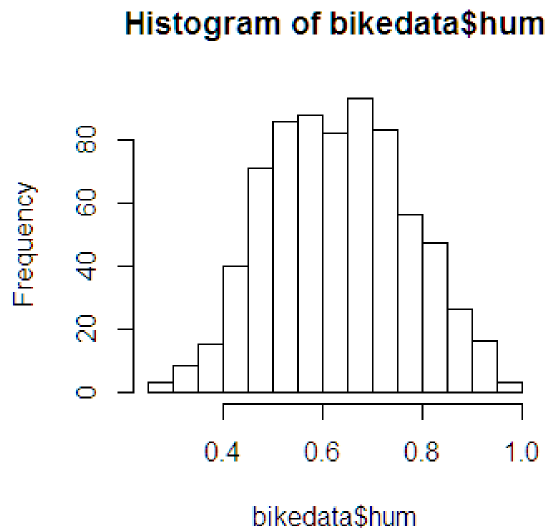


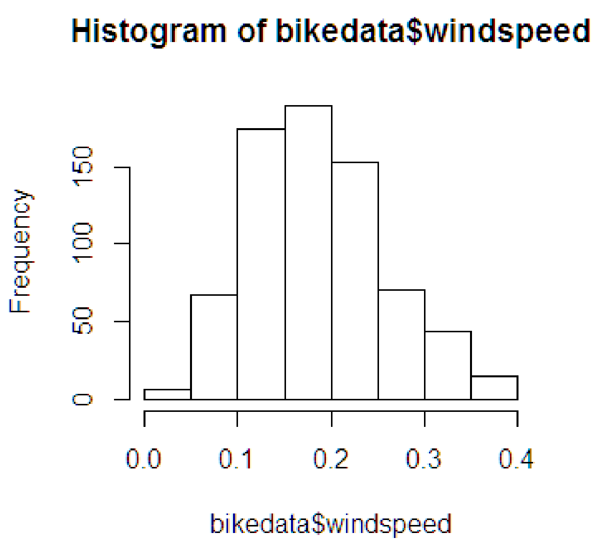
Feature selection



Feature scaling







References

- Medium.com
- towardsdatascience.com
- r tutorial