

**Prediction of Rental Bike Count**

ABHIRUP BHADRA



February 7, 2020

**Table of Contents**

Introduction .................................................................................................................................... 2

1.1 Problem Statement ................................................................................................................... 2

1.2 Dataset ....................................................................................................................................... 2

Methodology .................................................................................................................................... 3

2.1 Pre-Processing and EDA ............................................................................................................. 3

2.2 Target Variable – ‘cnt’ ................................................................................................................ 3

2.3 Missing value Analysis ................................................................................................................. 3

2.4 Distribution of Categorical variables .......................................................................................... 3

2.5 Distribution of continuous variables ……………………………………………………………………………………… 4

2.6 Relationship of Continuous variables against bike count……………………………………………………….. 5

2.7 Detection & Removal of outliers……………………………………………………………………………………………. 6

2.8 Feature Selection…………………………………………………………………………………………………………………… 6

Modelling……………………………………………………………………………………………………………………………………… 7

3.1 Model Selection………………………………………………………………………………………………………………………. 7

3.2 Decision Tree………………………………………………………………………………………………………………………..... 7

3.3 Random Forest……………………………………………………………………………………………………………………….. 7

3.4 Linear Regression……………………………………………………………………………………………………………………. 7

Conclusion……………………………………………………………………………………………………………………………………. 7

4.1 Mean Absolute Error(MAE)……………………………………………………………………………………………………. 8

Appendix A..………………………………………………………………………………………………………………………………… 8

5.1 Python Code……………………………………………………………………………………………………………………… 8-10

5.1 R Code……………………………………………………………………………………………………………………………… 10-15

**Introduction**

Many bicycle rental shops are now featuring inline skate rentals, especially in the western countries. The bike rental service has a great potential as a business opportunity. In our case this project is to predict the bike rental count based on the environmental and seasonal settings. So that required bikes would be arranged and managed by the shops according to environmental and seasonal conditions.

* 1. Problem Statement

The objective of this project is to predict the bike rental count on daily based on the

environmental and seasonal settings. The dataset contains 731 observations, 15 predictors and 1 target variable. The predictors are describing various environment factors and settings like season, weather, windspeed, humidity etc. We need to build a prediction model to predict estimated count or demand of bikes on a particular day based on the environmental factors.

* 1. Dataset

The data set consist of 731 observation recorded over a period of 2 years, between 2011 and 2012. It has 15 predictors or variables and 1 target variable. All the variables are described in below table.

|  |  |
| --- | --- |
| **Variable names** | **Description** |
| 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 from Holiday Schedule) |
| Weekday | Day of the week |
| Workingday | If day is neither weekend nor holiday is 1, otherwise is 0. |
| Weathersit | 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- t\_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 |

Table1. Description of variables

The data set consist of 7 continuous and 8 categorical variables. Sample data is shown below.

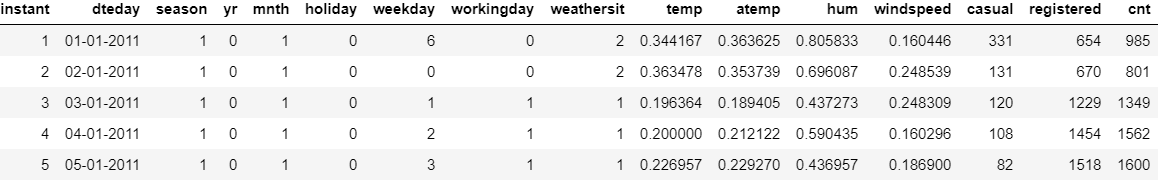


Table2. Sample data

**Methodology**

The solution of this problem is divided into three parts. First was EDA (Exploratory Data analysis) and pre-processing, followed by modelling and performance tuning and comparison. During first part data pre-processing step like missing value analysis, outlier analysis, univariate and bi-variate analysis etc. were performed. After that data was split into train and test. The target variable is a continuous variable, so it a regression problem. Decision tree, Linear regression and Random forest regression were used for modelling and their performance comparison was performed. Both the algorithms were implemented in R and python.

2.1 Pre-Processing and EDA

Pre-processing and EDA was performed. The dataset consists of 731 observations, and 15 predictors. The process of pre-processing and EDA is described below. We have 1 target variable. Cnt is our target variable. In the dataset season, yr, mnth, holiday, weekday, workingday, weathersit predictors should be categorical type, but they are int64. So mapping and categorical transformation is done.

2.2 Target Variable – ‘cnt’

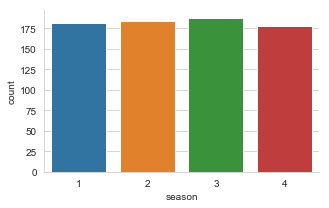
The target variable in the problem statement is the total count of registered and casual users of bikes on a single day. ‘*cnt’* is the combined value of *‘registered’* and *‘casual’* variables. Dropped the variables which are not requried.

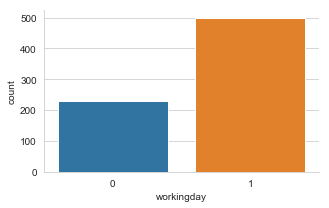
2.3 Missing value Analysis

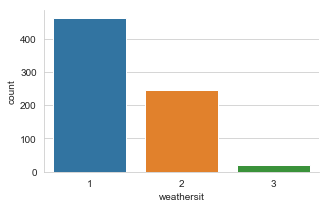
Missing value analysis was performed on the dataset. No missing values were found.

2.4 Distribution of Categorical variables

The distribution of categorical variables is as shown in the below figure:

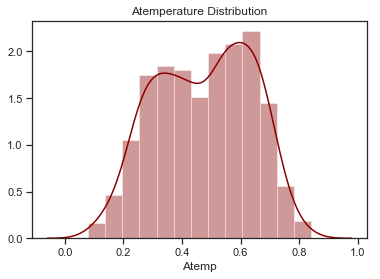
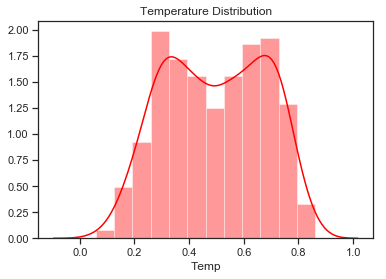


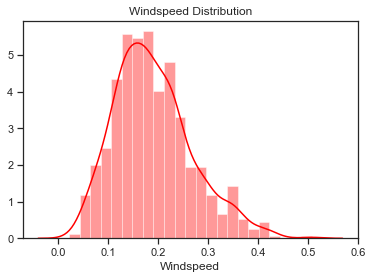
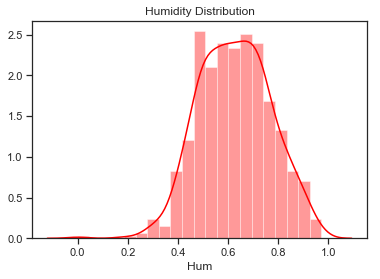




2.5 Distribution of continuous variables

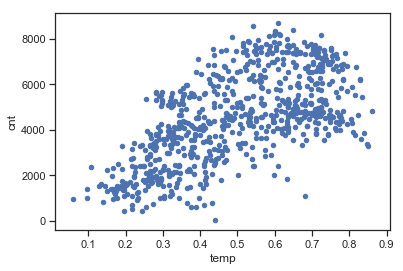
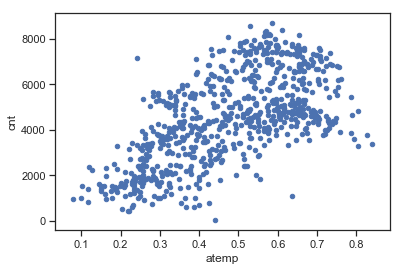
It can be observed from the below histograms is that temperature and feel temperature are normally distributed, whereas the variables windspeed and humidity are slightly skewed. The skewness is likely because of the presence of outliers and extreme data in those variables.

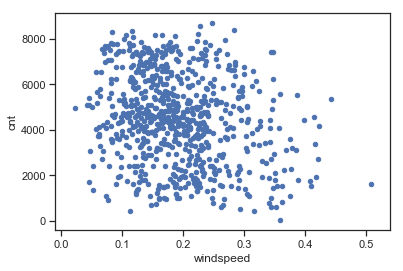
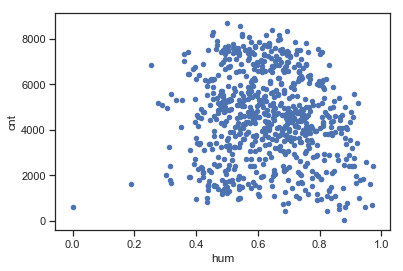




2.6 Relationship of Continuous variables against bike count

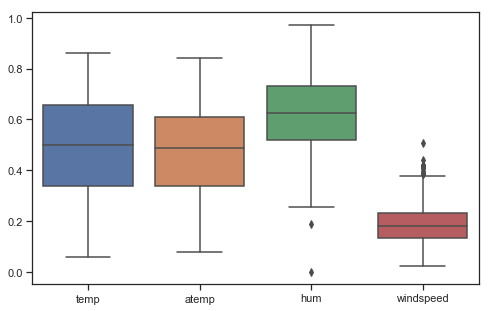
The below figure shows the relationship between continuous variables and the target variable using scatter plot. It can be observed that there exists a linear positive relationship between the variables temperature and feel temperature with the bike rental count. There also exists a negative linear relationship between the variable’s humidity and windspeed with the bike rental count.





2.7 Detection & Removal of outliers

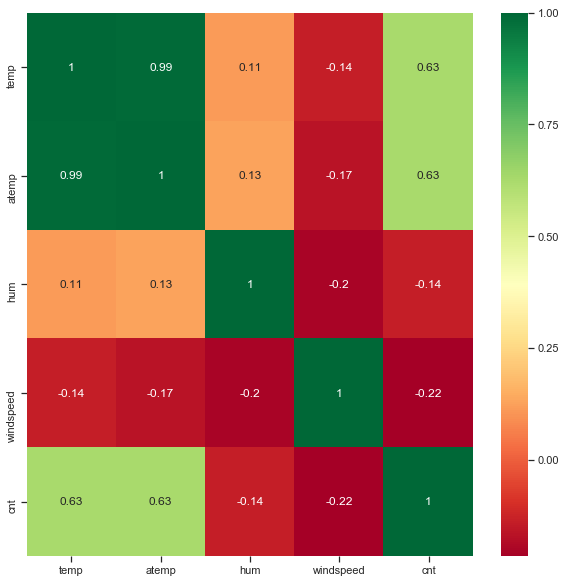
Outliers are detected using boxplots. Below figure illustrates the boxplots for all the continuous variables.



Outliers can be removed using the Boxplot stats method, wherein the Inter Quartile Range (IQR) is calculated and the minimum and maximum value are calculated for the variables. Any value ranging outside the minimum and maximum value are discarded.

2.8 Feature Selection

Feature selection can be done by correlation analysis. This is used for checking a linear relationship between continuous predictor and target. It is also used to check for multicollinearity among predictors. Multicollinearity exists whenever two or more of the predictors in a regression model are moderately or highly correlated. Multicollinearity is the condition when one predictor can be used to predict other. The basic problem is multicollinearity results in unstable estimation of coefficients which makes it difficult to access the effect of independent variable on dependent variable. Below figure is showing the correlation matrix for bike rent dataset.



From the correlation matrix, it is revealed that

1. Temp (temperature)and atemp (ambient temperature) are highly collinear. So atemp is removed from the dataset.

2. ‘cnt’ (demand count) have a strong and positive relationship with temperature and ambient temperature which is logical. People tends to rent bikes more which temperature is higher.

3. ‘cnt’ (demand count) is negative relationship with hum(humidity) and windspeed. People tends to rent bike more when there is less humidity and wind speed.

4. Also the relationship between ‘hum’, ’windspeed’ and ‘cnt’ is very weak. These are not very strong predictors.

**Modelling**

3.1 Model Selection

The dependent variable in our model is a continuous variable i.e., Count of bike rentals. Hence the models that we choose are Linear Regression, Decision Tree and Random Forest. The error metric chosen for the problem statement is Mean Absolute Error (MAE).

3.1 Decision Tree

Using decision tree, we can predict the value of bike count. In R MAE for this model is 618, MAPE is 18.54% hence the accuracy is 81.46%. In python the MAPE for this decision tree is 18.83%. Hence the accuracy for this model is 81.87%.

3.2 Random Forest

The number of ntrees used for prediction in the forest is 500 in R. MAE for this model is 384, MAPE is 12.12% hence accuracy is 87.88%. In python the MAPE for this random forest is 13.21%. Hence the accuracy for this model is 86.79%.

3.3 Multiple Linear Regression

In R MAE for this model is 519, MAPE is 13.36% hence the accuracy is 86.64%. In python the MAPE for this decision tree is 18.96%. Hence the accuracy for this model is 81.04%.

**Conclusion**

Now that we have a few models for predicting the target variable, we need to decide which one to choose. There are several criteria that exist for evaluating and comparing models. We can compare the models using any of the following criteria:

1. Predictive Performance

2. Interpretability

3. Computational Efficiency

In our case of Bike count prediction Data, Interpretability and Computation Efficiency, do not hold much significance. Therefore, we will use Predictive performance as the criteria to compare and evaluate models.

Predictive performance can be measured by comparing Predictions of the models with real values of the target variables, and calculating some average error measure.

4.1 Mean Absolute Error (MAE)

MAE is one of the error measures used to calculate the predictive performance of the model. We will apply this measure to our models that we have generated in the previous section.

**Linear Regression Model: MAE = 519**

**Decision Tree: MAE = 618.**

**Random Forest: MAE = 384**

Based on the above error metrics, Random Forest is the better model for our analysis. Hence Random Forest is chosen as the model for prediction of bike rental count. We can see both in R and python the accuracy is better with the Random Forest model.

**Appendix A**

5.1 Python Code

#Setting working directory

os.chdir("E:\data science\_edwisor\Project 1 bike prediction")

#Reading dataset

data = pd.read\_csv('day.csv')

backup\_data = data

#Converting to categorical variable

categorical\_variable = ["season","yr","mnth","holiday","weekday","workingday","weathersit"]

for var in categorical\_variable:

data[var]=data[var].astype("category")

# Dropping unwanted column

data1 = data.drop(["instant","dteday","casual","registered"],axis=1)

#Missing data

data1.isnull().sum()

# Distribution of the Categorical variable using factorplot

sns.set\_style("whitegrid")

sns.factorplot(data=data1, x='season', kind='count', size=3, aspect=1.5)

sns.factorplot(data=data1, x='workingday', kind='count', size=3, aspect=1.5)

sns.factorplot(data=data1, x='weathersit', kind='count', size=3, aspect=1.5)

# Distribution of the continuous or numerical variable using histogram

sns.set(style="ticks")

sns.distplot(data1['temp'],color='red') # .set\_title("Temperature Distribution")

plt.title("Temperature Distribution")

plt.xlabel("Temp")

sns.distplot(data1['atemp'],color='darkred')

plt.title('Atemperature Distribution')

plt.xlabel('Atemp')

sns.distplot(data1['hum'],color='red')

plt.title('Humidity Distribution')

plt.xlabel('Hum')

sns.distplot(data1['windspeed'],color='red')

plt.title('Windspeed Distribution')

plt.xlabel('Windspeed')

# Relation between continuous or numeric variables against bike count using scatter plot

data1.plot.scatter(x='temp',y='cnt')

data1.plot.scatter(x='atemp',y='cnt')

data1.plot.scatter(x='hum',y='cnt')

data1.plot.scatter(x='windspeed',y='cnt')

#outlier analysis

fig=plt.gcf()

fig.set\_size\_inches(8,5)

sns.boxplot(data=data1[['temp','atemp','hum','windspeed']])

# Removing outlire from Humidity

q75, q25 = np.percentile(data1.loc[:,'hum'],[75,25])

(q75,q25)

iqr=q75-q25

iqr

min= q25-(iqr\*1.5)

max= q75+(iqr\*1.5)

min,max

data1 = data1.drop(data1[data1.loc[:,'hum'] < min].index)

data1 = data1.drop(data1[data1.loc[:,'hum'] > max].index)

#Removing outlier from windspeed

q75, q25 = np.percentile(data1.loc[:,'windspeed'],[75,25])

iqr=q75-q25

min= q25-(iqr\*1.5)

max= q75+(iqr\*1.5)

min,max

data1 = data1.drop(data1[data1.loc[:,'windspeed'] < min].index)

data1 = data1.drop(data1[data1.loc[:,'windspeed'] > max].index)

# Correlation

corrmat= data1.corr()

top\_corr\_features = corrmat.index

plt.figure(figsize=(10,10))

sns.heatmap(data1[top\_corr\_features].corr(),annot=True,cmap='RdYlGn')

#droping corelated variable

data1 = data1.drop(['atemp'], axis=1)

# Importing Libraries

from sklearn.model\_selection import train\_test\_split

from sklearn.tree import DecisionTreeRegressor

train, test = train\_test\_split(data1, test\_size=0.2,random\_state=123)

#train the model

dt\_data = DecisionTreeRegressor(random\_state=123).fit(train.iloc[:,0:10],train.iloc[:,10])

#Prediction of the cnt

dt\_data\_pred = dt\_data.predict(test.iloc[:,0:10])

#Creating dataframe for actual and predicted value

dtpred\_data = pd.DataFrame({'actual': test.iloc[:,10], 'pred': dt\_data\_pred})

# Function for Mean Absolute Percentage Error

def MAPE(y\_actual,y\_pred):

mape = np.mean(np.abs((y\_actual - y\_pred)/y\_actual))\*100

return mape

#Calculate MAPE for decision tree

MAPE(test.iloc[:,10],dt\_data\_pred)

#Import library for RandomForestRegressor

from sklearn.ensemble import RandomForestRegressor

#Train the model

rf\_data = RandomForestRegressor(n\_estimators=200,random\_state=123).fit(train.iloc[:,0:10], train.iloc[:,10])

#Prediction on cnt

rf\_data\_prd = rf\_data.predict(test.iloc[:,0:10])

#Creating dataframe for actual and predicted value

rfpred\_data = pd.DataFrame({'actual': test.iloc[:,10], 'pred': rf\_data\_prd})

rfpred\_data.head()

#Calculate MAPE for decision tree

MAPE(test.iloc[:,10],rf\_data\_prd)

#import libraries for Linear regression

import statsmodels.api as sm

from sklearn.metrics import mean\_squared\_error

#Train the model

lr\_data = sm.OLS(train.iloc[:,10].astype(float), train.iloc[:,0:10].astype(float)).fit()

#Prediction of cnt

lr\_data\_prd = lr\_data.predict(test.iloc[:,0:10])

#Creating dataframe for actual and predicted value

rlpred\_data = pd.DataFrame({'actual': test.iloc[:,10], 'pred': lr\_data\_prd})

rlpred\_data.head()

#Calculate MAPE for decision tree

MAPE(test.iloc[:,10],lr\_data\_prd)

# saving the best model(Random Forest) output data

result=pd.DataFrame(test.iloc[:,0:10])

result['pred\_cnt'] = (rf\_data\_prd)

result.to\_csv("Random forest output python.csv",index=False)

5.2 R Code

# Cleaning the environment

rm(list = ls())

# Setting the working directory

setwd("E:/data science\_edwisor/Project 1 bike prediction")

getwd()

#Reading the file

data1 = read.csv('day.csv', header = TRUE)

#Loading libraries

x = c("ggplot2","usdm","caret","corrgram","randomForest","plyr","dplyr","rpart","DataCombine")

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

#EDA

dim(data1)

str(data1)

names(data1)

summary(data1)

# Checking missing value

sum(is.na(data1))

sapply(data1, function(x){

sum(is.na(x))

})

# Dropping the variable which are not required

data1 = subset(data1,select= -c(instant,dteday,casual,registered))

# Converting into proper data type

cat\_var= c('season','yr','mnth','holiday','weekday','workingday','weathersit')

num\_var= c('temp','atemp','hum','windspeed')

data\_conv = function(data1,var,type){

data1[var] = lapply(data1[var], type)

return(data1)

}

data1= data\_conv(data1,cat\_var,factor)

# Data Visualization

# CNT according to Season

ggplot(data1, aes(fill=cnt, x=season)) +

geom\_bar(position="dodge") + labs(title="cnt ~ season")

# CNT according to holiday

ggplot(data1, aes(fill=cnt, x=holiday)) +

geom\_bar(position="dodge") + labs(title="cnt ~ holiday")

# CNT according to season by yr

ggplot(data1, aes(fill=cnt, x=season)) +

geom\_bar(position="dodge") + facet\_wrap(~yr)+

labs(title="CNT according to season by yr")

# CNT according to season by workingday

ggplot(data1, aes(fill=cnt, x=season)) +

geom\_bar(position="dodge") + facet\_wrap(~workingday)+

labs(title="CNT according to season by workingday")

# CNT according to season by weekday

ggplot(data1, aes(fill=cnt, x=workingday)) +

geom\_bar(position="dodge") + facet\_wrap(~weekday)+

labs(title="CNT according to workingday by weekday")

#Check the distribution of categorical Data using bar graph

bar1 = ggplot(data = data1, aes(x = season)) + geom\_bar() + ggtitle("Count of Season")

bar2 = ggplot(data = data1, aes(x = weathersit)) + geom\_bar() + ggtitle("Count of Weather")

bar3 = ggplot(data = data1, aes(x = holiday)) + geom\_bar() + ggtitle("Count of Holiday")

bar4 = ggplot(data = data1, aes(x = workingday)) + geom\_bar() + ggtitle("Count of Working day")

# ## Plotting plots together

gridExtra::grid.arrange(bar1,bar2,bar3,bar4,ncol=2)

#Check the distribution of numerical data using histogram

hist1 = ggplot(data = data1, aes(x =temp)) + ggtitle("Distribution of Temperature") + geom\_histogram(bins = 25)

hist2 = ggplot(data = data1, aes(x =hum)) + ggtitle("Distribution of Humidity") + geom\_histogram(bins = 25)

hist3 = ggplot(data = data1, aes(x =atemp)) + ggtitle("Distribution of Feel Temperature") + geom\_histogram(bins = 25)

hist4 = ggplot(data = data1, aes(x =windspeed)) + ggtitle("Distribution of Windspeed") + geom\_histogram(bins = 25)

gridExtra::grid.arrange(hist1,hist2,hist3,hist4,ncol=2)

#Check the distribution of numerical data using scatterplot

scat1 = ggplot(data = data1, aes(x =temp, y = cnt)) + ggtitle("Distribution of Temperature") + geom\_point(color="blue") + xlab("Temperature") + ylab("Bike Count")

scat2 = ggplot(data = data1, aes(x =hum, y = cnt)) + ggtitle("Distribution of Humidity") + geom\_point(color="red") + xlab("Humidity") + ylab("Bike Count")

scat3 = ggplot(data = data1, aes(x =atemp, y = cnt)) + ggtitle("Distribution of Feel Temperature") + geom\_point(color="blue") + xlab("Feel Temperature") + ylab("Bike Count")

scat4 = ggplot(data = data1, aes(x =windspeed, y = cnt)) + ggtitle("Distribution of Windspeed") + geom\_point(color="red") + xlab("Windspeed") + ylab("Bike Count")

gridExtra::grid.arrange(scat1,scat2,scat3,scat4,ncol=2)

#Check for outliers in data using boxplot

cnames = colnames(data1[,c("temp","atemp","windspeed","hum")])

for (i in 1:length(cnames))

{

assign(paste0("gn",i), ggplot(aes\_string(y = cnames[i]), data = data1)+

stat\_boxplot(geom = "errorbar", width = 0.5) +

geom\_boxplot(outlier.colour="red", fill = "grey" ,outlier.shape=18,

outlier.size=1, notch=FALSE) +

theme(legend.position="bottom")+

labs(y=cnames[i])+

ggtitle(paste("Box plot for",cnames[i])))

}

gridExtra::grid.arrange(gn1,gn3,gn2,gn4,ncol=2)

#Remove outliers in Windspeed

windoutlr = data1[,11][data1[,11] %in% boxplot.stats(data1[,11])$out]

data1 = data1[which(!data1[,11] %in% windoutlr),]

boxplot(data1$windspeed)

#Remove outliers in humidity

humoutlr = data1[,10][data1[,10] %in% boxplot.stats(data1[,10])$out]

data1 = data1[which(!data1[,10] %in% humoutlr),]

boxplot(data1$hum)

#correlation plot

data2 = subset(data1, select= -c(season,yr,mnth,holiday,weekday,workingday,weathersit))

corr = round(cor(data2),2)

ggcorrplot(corr,hc.order = T,

type = "full",

lab = T,

lab\_size = 3,

method = "square",

colors = c("blue","white","darkgreen"),

title = "Correlation Plot",

ggtheme = theme\_bw)

#Removing atemp from the data1 dataset

data1 = subset(data1, select = -(atemp))

### Modeling

#Dividing into test and train seta

t\_idx = sample(1:nrow(data1), 0.8\*nrow(data1))

train = data1[t\_idx,]

test = data1[-t\_idx,]

# Removing all the custom variables from the memory

rmExcept(c("test","train","data1"))

# MAPE

mape = function(actual, predict){

mean(abs((actual-predict)/actual))\*100

}

##### Decission Tree

# rpart for regression. Train the model

dt\_model = rpart(cnt~ ., data= train, method = "anova")

#Prediction of cnt from test data

dt\_pred = predict(dt\_model, test[,-11])

#Creating dataframe for actual and predicted value

dt\_df = data.frame("actual" = test[,11], "pred"= dt\_pred)

head(dt\_df)

plot(test$cnt, dt\_pred,

xlab='Actual values',

ylab= 'Prediction Values',

main = 'DT model')

# Evaluation

postResample(dt\_pred, test$cnt)

mape(test$cnt, dt\_pred)

#MAPE: 18.54%

#MAE: 618.11

#RMSE: 796.51

#Accuracy: 81.46%

#### Random Forest

# Train the model

rf\_model = randomForest(cnt~., data = train, ntree = 500)

#Prediction of cnt from test data

rf\_pred = predict(rf\_model, test[,-11])

#Creating dataframe for actual and predicted value

rf\_df = data.frame("actual" = test[,11], "pred"= rf\_pred)

head(rf\_df)

plot(test$cnt, rf\_pred,

xlab='Actual values',

ylab= 'Prediction Values',

main = 'RF model')

# Evaluation

postResample(rf\_pred, test$cnt)

mape(test$cnt, rf\_pred)

#MAPE: 12.12%

#MAE: 384.28

#RMSE: 521.73

#Accuracy: 87.88%

#### Linear Regression

#Train the data using linear regression

lr\_model = lm(formula = cnt~., data = train)

#Check the summary of the model

summary(lr\_model)

#Predict the test cases

lr\_pred = predict(lr\_model, test[,-11])

#Creating dataframe for actual and predicted value

lr\_df = data.frame("actual" = test[,11], "pred"= lr\_pred)

head(lr\_df)

plot(test$cnt, lr\_pred,

xlab='Actual values',

ylab= 'Prediction Values',

main = 'LR model')

# Evaluation

postResample(lr\_pred, test$cnt)

mape(test$cnt, lr\_pred)

#MAPE: 13.36%

#MAE: 519.52

#RMSE: 701.62

#Accuracy: 86.64%

# saving the best model(Random Forest) output data

results = data.frame(test, pred\_cnt = rf\_pred)

write.csv(results, file = 'RF output R .csv', row.names = FALSE, quote=FALSE)