**Predication of bike rental count**

***Minumula Rajashekar***

Contents

[Introduction 3](#_Toc532651784)

[1.1 Problem Statement 3](#_Toc532651785)

[1.2 Data 3](#_Toc532651786)

[Methodology 5](#_Toc532651787)

[2.1 Pre Processing 5](#_Toc532651788)

[**2.1.1** **Outlier Analysis** 6](#_Toc532651789)

[**2.1.2** **Feature Selection** 15](#_Toc532651790)

[2.2 Modeling 19](#_Toc532651791)

[**2.2.1** **Model Selection** 19](#_Toc532651792)

[**2.2.2 Linear Regression** 19](#_Toc532651793)

[**2.2.3** **Desicison Tree Regression** 20](#_Toc532651794)

[**2.2.4** **Random Forest Regression** 21](#_Toc532651795)

[**2.2.5** **KNN Regression** 21](#_Toc532651796)

[Conclusion 22](#_Toc532651797)

[3.1 Model Evaluation 22](#_Toc532651798)

[**3.1.1** **Mean Absolute Percentage Error (MAPE)** 22](#_Toc532651799)

[3.2 Model Selection 23](#_Toc532651800)

[3.3 R Code: 24](#_Toc532651801)

[3.4 Python code 28](#_Toc532651802)

**Chapter 1**

# Introduction

## 1.1 Problem Statement

The aim of this project is predict the bike rental count on a particular time along with season, weather setting, and temperature. The advantage of predicting bike rental count will be scope to the management to maintain exact number of bikes according to the seasons weather conditions without losing the customers lack of bikes.

## 1.2 Data

Our task is to build the regression model upon the training data and verify using the test data. Given below is the sample of data.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| | Bike Rental Count (Columns: 1-10) | | | | | | | | | | | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | **instant** | **dteday** | **season** | **yr** | **mnth** | **holiday** | **weekday** | **workingday** | **weathersit** | **temp** | | 1 | 2011-01-01 | 1 | 0 | 1 | 0 | 6 | 0 | 2 | 0.344167 | | 2 | 2011-01-02 | 1 | 0 | 1 | 0 | 0 | 0 | 2 | 0.363478 | | 3 | 2011-01-03 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 0.196364 | | 4 | 2011-01-04 | 1 | 0 | 1 | 0 | 2 | 1 | 1 | 0.200000 | | 5 | 2011-01-05 | 1 | 0 | 1 | 0 | 3 | 1 | 1 | 0.226957 | |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| | Bike Rental Count (Columns: 11-16) | | | | | | | --- | --- | --- | --- | --- | --- | | **atemp** | **hum** | **windspeed** | **casual** | **registered** | **cnt** | | 0.363625 | 0.805833 | 0.1604460 | 331 | 654 | 985 | | 0.353739 | 0.696087 | 0.2485390 | 131 | 670 | 801 | | 0.189405 | 0.437273 | 0.2483090 | 120 | 1229 | 1349 | | 0.212122 | 0.590435 | 0.1602960 | 108 | 1454 | 1562 | | 0.229270 | 0.436957 | 0.1869000 | 82 | 1518 | 1600 | |

As you can see in the table below we have the following 16 variables, using which we have to correctly predict the count of bike rental

| Predictor Variables | |
| --- | --- |
| **S.No.** | **Predictor** |
| 1 | instant |
| 2 | dteday |
| 3 | season |
| 4 | yr |
| 5 | mnth |
| 6 | holiday |
| 7 | weekday |
| 8 | workingday |
| 9 | weathersit |
| 10 | temp |
| 11 | atemp |
| 12 | hum |
| 13 | windspeed |
| 14 | casual |
| 15 | registered |

**Chapter 2**

# Methodology

## 2.1 Pre Processing

Any predictive modeling requires that we look at the data before we start modeling. However, in data mining terms *looking at data* refers to so much more than just looking. Looking at data refers to exploring the data, cleaning the data as well as visualizing the data through graphs and plots. This is often called as **Exploratory Data Analysis**. To start this process we will first try and look at all the probability distributions of the variables. Most analysis like regression, require the data to be normally distributed. We can visualize that in a glance by looking at the probability distributions or probability density functions of the variable.

**Missing value Anaysis**

Checking data whether there are any missing values in the data.

#missing value Analysis ‘R’ Code

missing\_val=data.frame(apply(bike\_data,2,function(x){sum(is.na(x))}))

missing\_val[2]=colnames(bike\_data)

row.names(missing\_val)=NULL

names(missing\_val)[1]="MissingVal"

names(missing\_val)[2]='ColumnName'

missing\_val=missing\_val[,c(2,1)]

missing\_val

ColumnName MissingVal

1 instant 0

2 dteday 0

3 season 0

4 yr 0

5 mnth 0

6 holiday 0

7 weekday 0

8 workingday 0

9 weathersit 0

10 temp 0

11 atemp 0

12 hum 0

13 windspeed 0

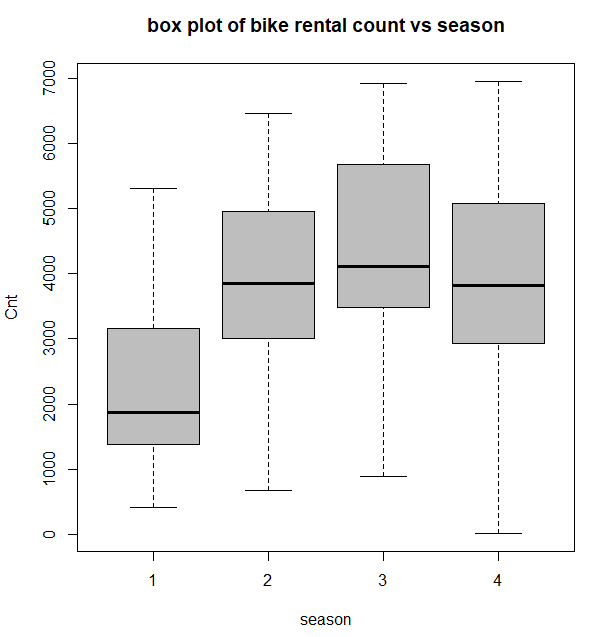
14 casual 0

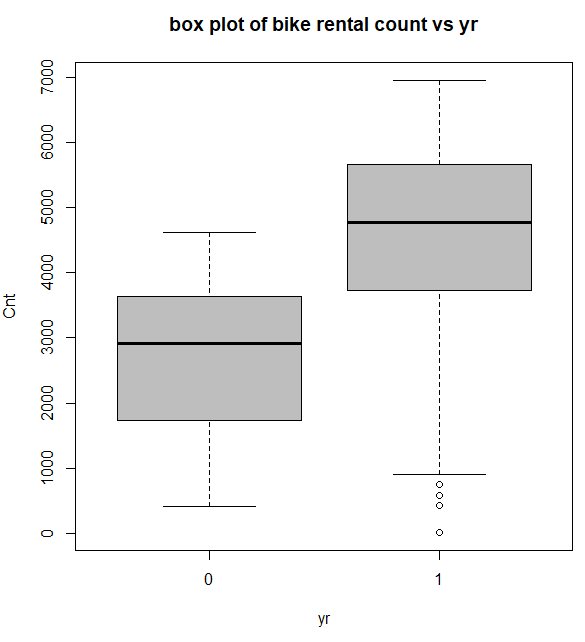
15 registered 0

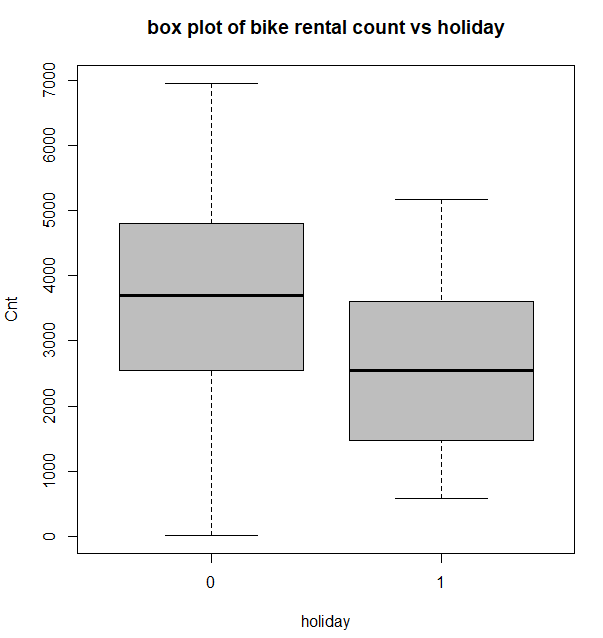
16 cnt 0

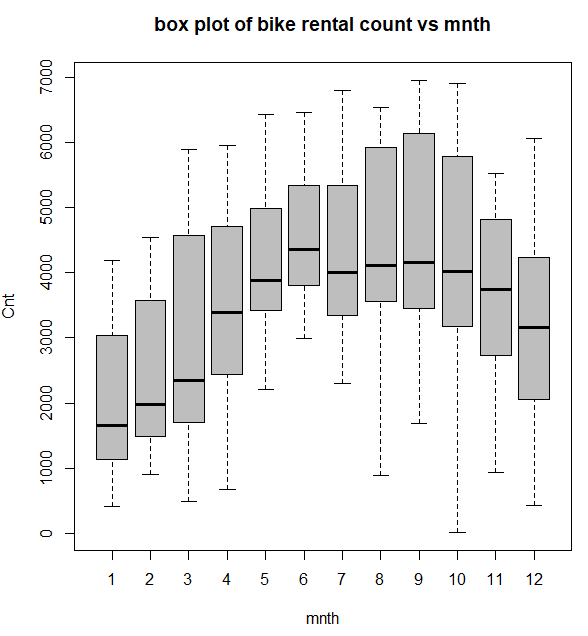
### **2.1.1 Outlier Analysis**

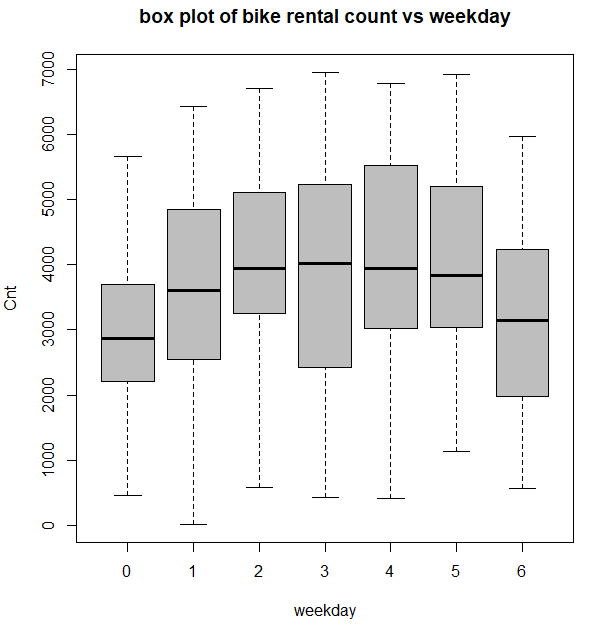
Outliers are those data points which are away from the normal data. Outliers cause a skewness in the data and make model inconsistent. There is a need to remove outliers in the Data. Outliers are identified using Boxplots and are removed from the data.

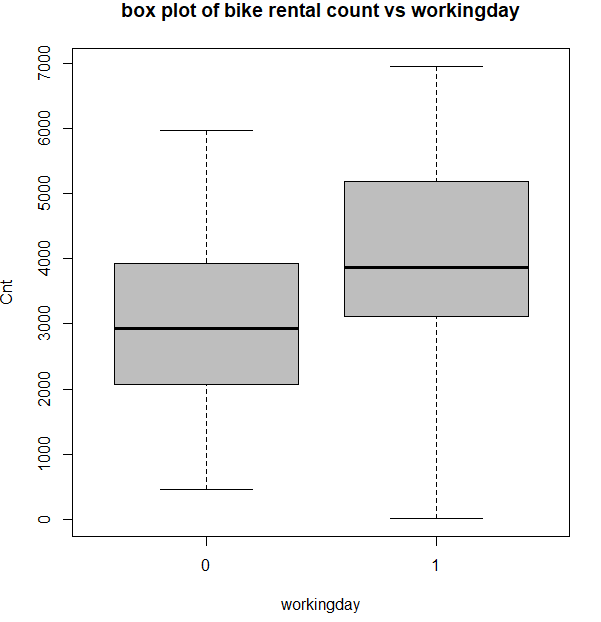


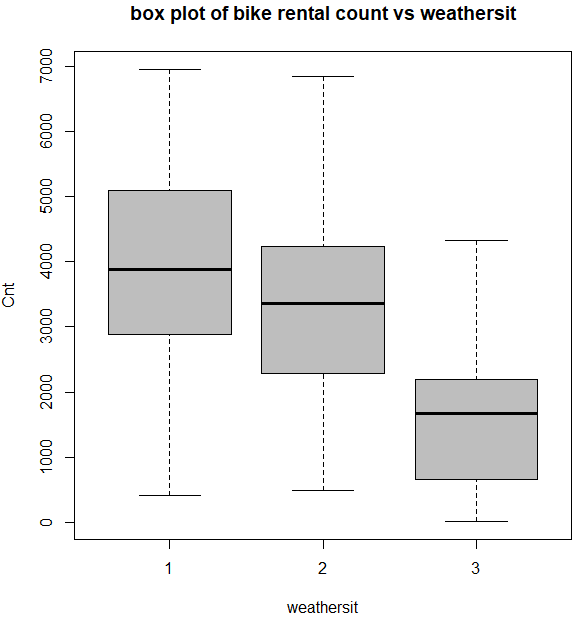






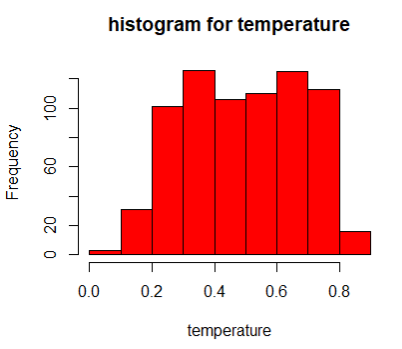


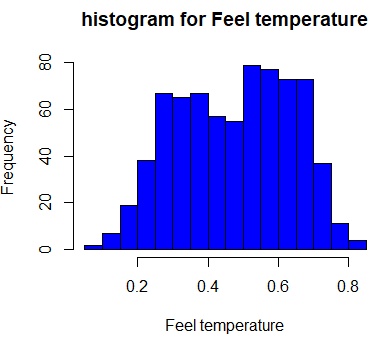


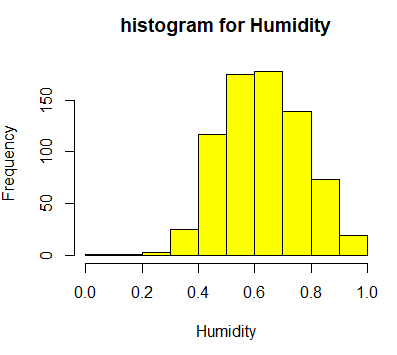


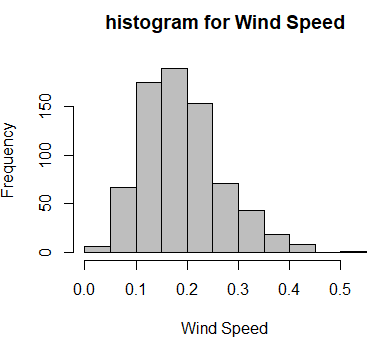
From the above boxplots it’s clear that there are not much outliers present in categorical variables.

**The following figures explain Data distribution of continuous variables using histograms**

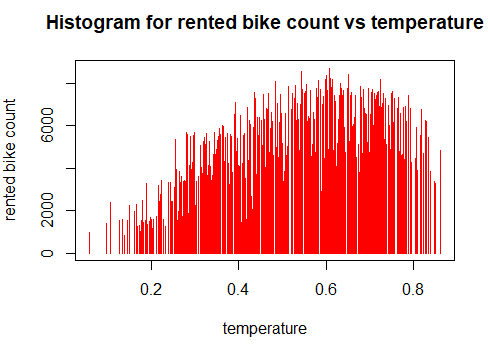


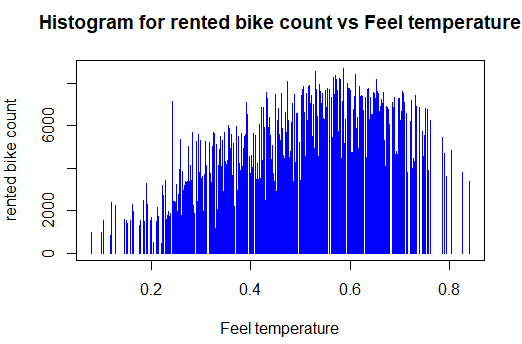


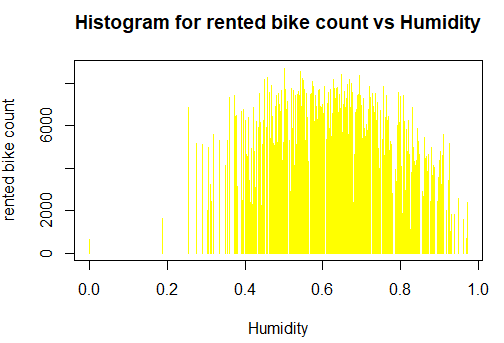


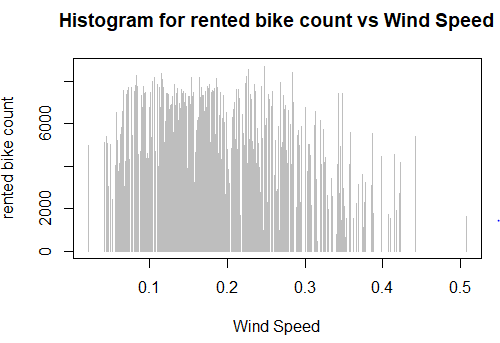


**Effect of categorical variables on Bike rental count**









### **2.1.2 Feature Selection**

Before performing any type of modeling we need to assess the importance of each predictor variable in our analysis. There is a possibility that many variables in our analysis are not important at all to the problem of regression. There are several methods of doing that. Below we have used correlation for continuous variables and anova tests for categorical varaibles under feature selection.

Another step of Exploratory Data Analysis is to look for highly correlated variables in the data. A very simple way of looking at correlations in the data is shown below.

#correlation analysis for indepedent numeric variables

library(corrgram)

str(bike\_data)

numeric\_data =subset(bike\_data,select =c("temp","atemp","hum","windspeed","cnt"))

correlation=round(cor(numeric\_data),2)

correlation

temp atemp hum windspeed cnt

temp 1.00 0.99 0.13 -0.16 0.63

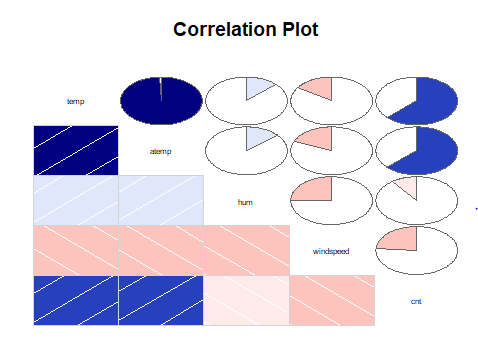
atemp 0.99 1.00 0.14 -0.18 0.63

hum 0.13 0.14 1.00 -0.25 -0.10

windspeed -0.16 -0.18 -0.25 1.00 -0.23

cnt 0.63 0.63 -0.10 -0.23 1.00

corrgram(numeric\_data, order = F,upper.panel=panel.pie, text.panel=panel.txt, main = "Correlation Plot")



From the above correlation plot we can observe temp and atemp varibles are highly correlated so we can drop atemp from our modeling.

**Anova test for categorical variables**

Anova test is used to analyse if there is any (statistically) significant difference in groups of categorical variables.

ANOVA is going to compare means of dependent continuous variable among the groups of categorical variable and check ifdifferences are statistically significant. Here are null and alternative hypothesis:  
  
**Null Hypothesis**: all group means are equal —> there is no relationship between categorical variable and dependent variable, which we can write as follows:

        H0: all means are equal

**Alternative Hypothesis**: not all group means are equal —> there is a relationship between categorical variable and dependent variable.:

        H1: not all means are equal

F statistics = Variation among sample means / Variation within groups

Through the F statistics we can see if the variation among sample means dominates over the variation within groups, or not. In the first case we will have strong evidence against the null hypothesis (means are all equals), while in the second case we would have little evidence against the null hypothesis.

season\_anova=aov(cnt~season,data=bike\_data)

summary(season\_anova)

Df Sum Sq Mean Sq F value Pr(>F)

season 1 4.518e+08 451797359 144 <2e-16 \*\*\*

Residuals 729 2.288e+09 3138187

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

our p-value is less than 0.05 (as suggested by normal scientific standard). Hence we can conclude that for our confidence interval we accept the alternative hypothesis H1 that there is a significant relationship between variables.

yr\_anova=aov(cnt~yr,data=bike\_data)

summary(yr\_anova)

Df Sum Sq Mean Sq F value Pr(>F)

yr 1 8.798e+08 879828893 344.9 <2e-16 \*\*\*

Residuals 729 1.860e+09 2551038

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

our p-value is less than 0.05 (as suggested by normal scientific standard). Hence we can conclude that for our confidence interval we accept the alternative hypothesis H1 that there is a significant relationship between variables.

month\_anova=aov(cnt~mnth,data=bike\_data)

summary(month\_anova)

Df Sum Sq Mean Sq F value Pr(>F)

mnth 1 2.147e+08 214744463 62.01 1.24e-14 \*\*\*

Residuals 729 2.525e+09 3463362

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

our p-value is less than 0.05 (as suggested by normal scientific standard). Hence we can conclude that for our confidence interval we accept the alternative hypothesis H1 that there is a significant relationship between variables.

holiday\_anova=aov(cnt~holiday,data=bike\_data)

summary(holiday\_anova)

Df Sum Sq Mean Sq F value Pr(>F)

holiday 1 1.280e+07 12797494 3.421 0.0648 .

Residuals 729 2.727e+09 3740381

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

our p-value is more than 0.05 (as suggested by normal scientific standard). Hence we can conclude that for our confidence interval we accept the null hypothesis H0 that there is a no significant relationship between variables.

weekday\_anova=aov(cnt~weekday,data=bike\_data)

summary(weekday\_anova)

Df Sum Sq Mean Sq F value Pr(>F)

weekday 1 1.246e+07 12461089 3.331 0.0684 .

Residuals 729 2.727e+09 3740843

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

our p-value is more than 0.05 (as suggested by normal scientific standard). Hence we can conclude that for our confidence interval we accept the null hypothesis H0 that there is a no significant relationship between variables.

workingDay\_anova=aov(cnt~workingday,data=bike\_data)

summary(workingDay\_anova)

Df Sum Sq Mean Sq F value Pr(>F)

workingday 1 1.025e+07 10246038 2.737 0.0985 .

Residuals 729 2.729e+09 3743881

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

our p-value is more than 0.05 (as suggested by normal scientific standard). Hence we can conclude that for our confidence interval we accept the null hypothesis H0 that there is a no significant relationship between variables.

weatherSit\_anova=aov(cnt~weathersit,data=bike\_data)

summary(weatherSit\_anova)

Df Sum Sq Mean Sq F value Pr(>F)

weathersit 1 2.423e+08 242288753 70.73 <2e-16 \*\*\*

Residuals 729 2.497e+09 3425578

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

our p-value is less than 0.05 (as suggested by normal scientific standard). Hence we can conclude that for our confidence interval we accept the alternative hypothesis H1 that there is a significant relationship between variables.

From the above ANOVA test we can conclude that holiday, week day and working day have no significant relationship to that bike rental count so we drop holiday, week day and working day from our modelling and we include season, yr, mnth and weathersit in our modelling.

## 2.2 Modeling

### **2.2.1 Model Selection**

The dependent variable can fall in either of the four categories:

1. Nominal
2. Ordinal
3. Interval
4. Ratio

Based on the dependent of variable of your dataset we use a model accordingly.There are two types of supervised learning models.They are classification and regression.we choose a type from them depending on the predicting variable. In this case our depending variable is continuous variable we use regression models.If the variable is categorical we go classification models.we try different regression models and analyse using error metrics choose the model which is optimal.

### **2.2.2 Linear Regression**

#Linear Regreesion

linear\_model=lm(cnt~season+yr+mnth+weathersit+temp+hum+windspeed,data=train\_data)

summary(linear\_model)

Call:

lm(formula = cnt ~ season + yr + mnth + weathersit + temp + hum +

windspeed, data = train\_data)

Residuals:

Min 1Q Median 3Q Max

-4162.3 -465.7 45.5 533.1 3062.4

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) 1949.50 241.03 8.088 3.60e-15 \*\*\*

season 501.59 61.49 8.158 2.15e-15 \*\*\*

yr 2085.76 73.45 28.397 < 2e-16 \*\*\*

mnth -39.33 19.31 -2.037 0.042125 \*

weathersit -551.93 90.54 -6.096 1.99e-09 \*\*\*

temp 5382.97 216.43 24.872 < 2e-16 \*\*\*

hum -1315.15 366.57 -3.588 0.000362 \*\*\*

windspeed -2846.12 498.32 -5.711 1.80e-08 \*\*\*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 870 on 576 degrees of freedom

Multiple R-squared: 0.8054, Adjusted R-squared: 0.803

F-statistic: 340.5 on 7 and 576 DF, p-value: < 2.2e-16

As you can see the ***Adjusted R-squared*** value, we can explain about 80.3% of the data using our linear regression model.

The mean absolute percentage error in this model 20.817

### **2.2.3 Desicison Tree Regression**

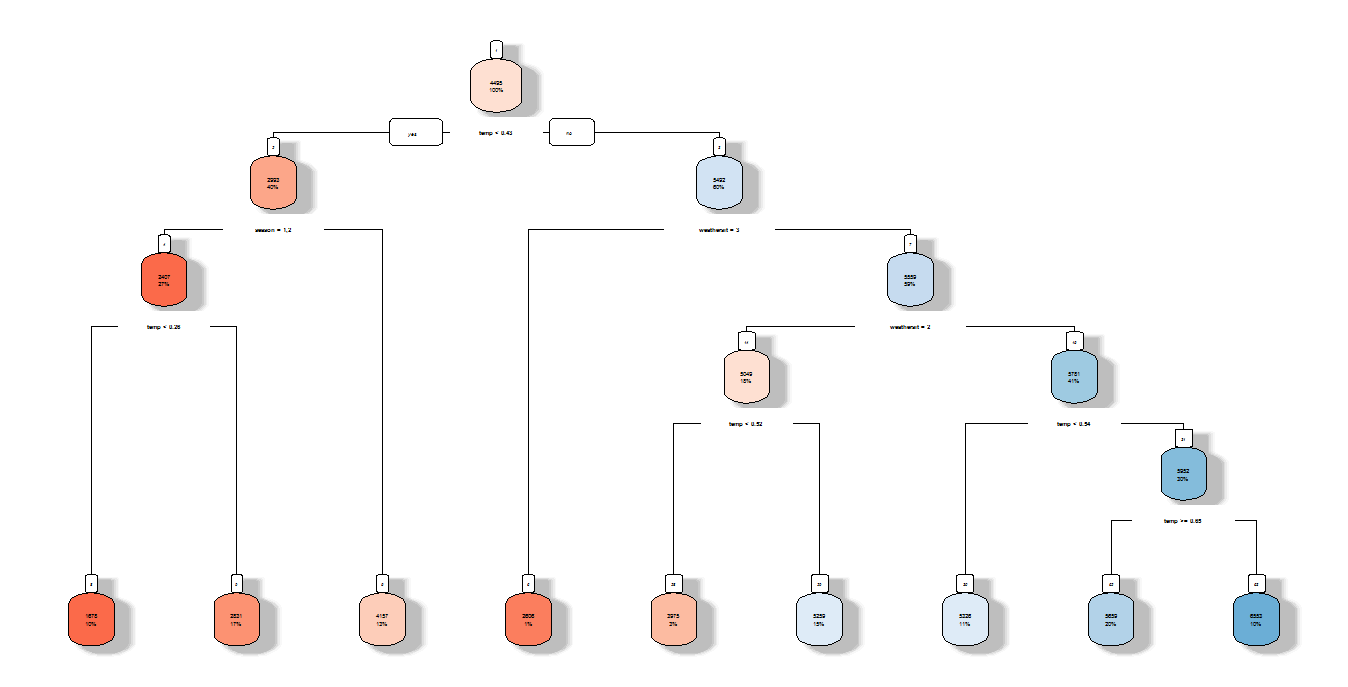
# decission tree regression

library(rpart)

dt\_model=rpart(cnt~season+yr+mnth+weathersit+temp+hum+windspeed,data=train\_data)

summary(dt\_model)

Now we will try and use a different regression model to predict our cnt variable. We will use a regression tree to predict the values of our target variable.



### **2.2.4 Random Forest Regression**

#RandomForest

library(randomForest)

library(inTrees)

RF\_Model=randomForest(cnt~season+yr+mnth+weathersit+temp+hum+windspeed,data = train\_data,importance=TRUE,ntree=500)

Now we will try and use a different regression model to predict our cnt variable. We will use a random forest to predict the values of our target variable. Comparing random forest with decision tree regression it will avoid over fitting.

### **2.2.5 KNN Regression**

#KNN Regression

KNN\_Model=knnreg(train\_data[,1:7],train\_data$cnt,k=25)

k-nearest neighbour regression that can return the average value for the neighbour.

**Chapter 3**

# Conclusion

## 3.1 Model Evaluation

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 Wine Data, the latter two, 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 taret variables, and calculating some average error measure.

### **3.1.1 Mean Absolute Percentage Error (MAPE)**

MAPE 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.

#MAPE Function

Mape = function(act, pre){

mean(abs((act - pre)/act))\*100

}

#linear regression

linear\_model=lm(cnt~season+yr+mnth+weathersit+temp+hum+windspeed,data=train\_data)

summary(linear\_model)

predictions=predict(linear\_model,test\_data[,-8])

Mape(test\_data[,8],predictions)

Results:

20.03628

# decission tree regression

library(rpart)

dt\_model=rpart(cnt~season+yr+mnth+weathersit+temp+hum+windspeed+cnt,data=train\_data)

summary(dt\_model)

predictions=predict(dt\_model,test\_data[,-8])

Mape(test\_data[,8],predictions)

Results:

24.15448

#RandomForest

RF\_Model=randomForest(cnt~season+yr+mnth+weathersit+temp+hum+windspeed,data = train\_data,importance=TRUE,ntree=500)

RF\_Predictions=predict(RF\_Model,test\_data[,-8])

Mape(test\_data[,8],RF\_Predictions)

Results:

17.17348

#KNN Regression

KNN\_Model=knnreg(train\_data[,1:7],train\_data$cnt,k=25)

KNN\_Predictions=predict(KNN\_Model,test\_data[,1:7])

Mape(test\_data[,8],KNN\_Predictions)

Results:

25.99737

## 3.2 Model Selection

We can see that comparing above all models Random Forest performs well and it avoids over fitting problem comparing all other models.

## 3.3 R Code:

rm(list=ls())

setwd("C:/Users/Rajashekar/Videos/Project/Bike Rental")

getwd()

bike\_data=read.csv("day.csv")

summary(bike\_data)

#Data Display

library(kableExtra)

length(colnames(bike\_data))

head(bike\_data[1:5,1:10]) %>% kable(caption="bike renting(columns:1-5) ",booktabs=TRUE,longtable=TRUE)

head(bike\_data[1:5,11:16]) %>% kable(caption="bike renting(columns:1-5) ",booktabs=TRUE,longtable=TRUE)

#Libraries required for Data

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

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

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

#missing value Analysis

missing\_val=data.frame(apply(bike\_data,2,function(x){sum(is.na(x))}))

missing\_val[2]=colnames(bike\_data)

row.names(missing\_val)=NULL

names(missing\_val)[1]="MissingVal"

names(missing\_val)[2]='ColumnName'

missing\_val=missing\_val[,c(2,1)]

missing\_val

#outlier analysis

cat\_columns=c("season","yr","mnth","holiday","weekday","workingday","weathersit")

for(i in 1:length(cat\_columns)){

boxplot(bike\_data[,15]~bike\_data[,cat\_columns[i]],

data=bike\_data,

main=paste("box plot of bike rental count vs",cat\_columns[i]),

xlab=cat\_columns[i],

ylab="Cnt",

col="gray",border="black")

}

#Data Distribution and histograms

bike\_data

hist\_data=subset(bike\_data,select=c("temp","atemp","hum","windspeed","cnt"))

names(hist\_data)[1]="temperature"

names(hist\_data)[2]="Feel temperature"

names(hist\_data)[3]="Humidity"

names(hist\_data)[4]="Wind Speed"

names(hist\_data)[5]="Total Bike Rentals"

hist\_columns=colnames(hist\_data)

colnames(hist\_data)

for(i in 1:length(hist\_columns[-5])){

hist(hist\_data[,i],col = color[i],main=paste0("histogram for ",hist\_columns[i]),xlab = hist\_columns[i])

}

#plotting categorical vs bike rental count

library(RColorBrewer)

color=c("red","blue","yellow","grey")

for(i in 1:length(hist\_columns[-5])){

plot(hist\_data[,i],hist\_data$`Total Bike Rentals`,

main=paste0("Histogram for rented bike count vs ",hist\_columns[i]),

xlab = hist\_columns[i],

ylab="rented bike count",

type="h",

col=color[i])

}

#correlation analysis for indepedent numeric variables

library(corrgram)

str(bike\_data)

numeric\_data =subset(bike\_data,select =c("temp","atemp","hum","windspeed","cnt"))

correlation=round(cor(numeric\_data),2)

correlation

corrgram(numeric\_data, order = F,upper.panel=panel.pie, text.panel=panel.txt, main = "Correlation Plot")

#ANOVA for categorical variables

season\_anova=aov(cnt~season,data=bike\_data)

summary(season\_anova)#season

yr\_anova=aov(cnt~yr,data=bike\_data)

summary(yr\_anova)#yr

month\_anova=aov(cnt~mnth,data=bike\_data)

summary(month\_anova)#mnth

holiday\_anova=aov(cnt~holiday,data=bike\_data)

summary(holiday\_anova)#-holiday

weekday\_anova=aov(cnt~weekday,data=bike\_data)

summary(weekday\_anova)#-weekday

workingDay\_anova=aov(cnt~workingday,data=bike\_data)

summary(workingDay\_anova)#-working day

weatherSit\_anova=aov(cnt~weathersit,data=bike\_data)

summary(weatherSit\_anova)

#MODELLING

bike\_data\_final=subset(bike\_data,select=c("season","yr","mnth","weathersit","temp","hum","windspeed","cnt"))

train\_index=sample(1:nrow(bike\_data\_final),0.8\*nrow(bike\_data\_final))

train\_data=bike\_data\_final[train\_index,]

test\_data=bike\_data\_final[-train\_index,]

#MAPE function

Mape = function(act, pre){

mean(abs((act - pre)/act))\*100

}

#linear regression

linear\_model=lm(cnt~season+yr+mnth+weathersit+temp+hum+windspeed,data=train\_data)

summary(linear\_model)

predictions=predict(linear\_model,test\_data[,-8])

Mape(test\_data[,8],predictions)

# decission tree regression

library(rpart)

dt\_model=rpart(cnt~season+yr+mnth+weathersit+temp+hum+windspeed+cnt,data=train\_data)

summary(dt\_model)

predictions=predict(dt\_model,test\_data[,-8])

Mape(test\_data[,8],predictions)

#RandomForest

library(randomForest)

library(inTrees)

RF\_Model=randomForest(cnt~season+yr+mnth+weathersit+temp+hum+windspeed,data = train\_data,importance=TRUE,ntree=500)

treeList=RF2List(RF\_Model)

exec=extractRules(treeList = treeList,train\_data[,-8]) #extract rules

readableRules=presentRules(exec,colnames(train\_data))

readableRules[1:2,]

RF\_Predictions=predict(RF\_Model,test\_data[,-8])

test\_data$RF\_Predictions=predict(RF\_Model,test\_data[,-8])

test\_data

Mape(test\_data[,8],RF\_Predictions)

#KNN Predictions

library(class)

library(BOSTON)

library(FNN)

z=c()

for(i in 1:range(100))

{

z[i]=999

if(i%%2==1){

bike\_data\_final=subset(bike\_data,select=c("season","yr","mnth","weathersit","temp","hum","windspeed","cnt"))

train\_index=sample(1:nrow(bike\_data\_final),0.8\*nrow(bike\_data\_final))

train\_data=bike\_data\_final[train\_index,]

test\_data=bike\_data\_final[-train\_index,]

print(i)

KNN\_Model=knnreg(train\_data[,1:7],train\_data$cnt,k=i)

KNN\_Predictions=predict(KNN\_Model,test\_data[,1:7])

z[i]=Mape(test\_data[,8],KNN\_Predictions)

print(z[i])

}

}

print(min(z)) # It gives minimum MAPE value

match(min(z),z) #it gives k value for the minimum MAPE value

## 3.4 Python code

#!/usr/bin/env python

# coding: utf-8

#libraries for python project

import os

import pandas as pd

import seaborn as sns

from random import randrange, uniform

import numpy as np

import matplotlib.pyplot as plt

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

from sklearn.tree import DecisionTreeRegressor

from sklearn import metrics

os.chdir("C:/Users/Rajashekar/Videos/python\_project")

os.getcwd()

# In[12]:

bike\_data=pd.read\_csv("day.csv",sep=",")

# In[13]:

bike\_data.columns

# In[14]:

#these are features for anova test

cnames\_anova=['season', 'yr', 'mnth', 'holiday', 'weekday',

'workingday', 'weathersit','cnt']

#below features for correlation analysis

cnames\_corr=['temp','atemp','hum','windspeed','cnt']

# In[15]:

#saving into new dataset called bike\_data\_1

bike\_data\_1=bike\_data.loc[:,cnames\_corr]

bike\_data\_1.shape

# In[16]:

#below is correlation matrix

f, ax = plt.subplots(figsize=(7, 5))

#Generate correlation matrix

corr = bike\_data\_1.corr()

#Plot using seaborn library

sns.heatmap(corr, mask=np.zeros\_like(corr, dtype=np.bool), cmap=sns.diverging\_palette(220, 10, as\_cmap=True),

square=True, ax=ax)

# In[17]:

#Anova for categorical variables

import statsmodels.api as sm

from statsmodels.formula.api import ols

bike\_data\_2=bike\_data.loc[:,cnames\_anova]

#bike\_data\_2.boxplot('cnt',by='season')

season\_anova=ols('cnt~season',data=bike\_data\_2).fit()

season\_anova\_table=sm.stats.anova\_lm(season\_anova,type=2)

print(season\_anova\_table)

yr\_anova=ols('cnt~yr',data=bike\_data\_2).fit()

yr\_anova\_table=sm.stats.anova\_lm(yr\_anova,type=2)

print(yr\_anova\_table)

month\_anova=ols('cnt~mnth',data=bike\_data\_2).fit()

month\_anova\_table=sm.stats.anova\_lm(month\_anova,type=2)

print(month\_anova\_table)

weekday\_anova=ols('cnt~weekday',data=bike\_data\_2).fit()

weekday\_anova\_table=sm.stats.anova\_lm(weekday\_anova,type=2)

print(weekday\_anova\_table)

workingday\_anova=ols('cnt~workingday',data=bike\_data\_2).fit()

workingday\_anova\_table=sm.stats.anova\_lm(workingday\_anova,type=2)

print(workingday\_anova\_table)

weathersit\_anova=ols('cnt~weathersit',data=bike\_data\_2).fit()

weathersit\_anova\_table=sm.stats.anova\_lm(weathersit\_anova,type=2)

print(weathersit\_anova\_table)

holiday\_anova=ols('cnt~holiday',data=bike\_data\_2).fit()

holiday\_anova\_table=sm.stats.anova\_lm(holiday\_anova,type=2)

print(holiday\_anova\_table)

# In[18]:

#below is the step to remove columns that contain hign +ve and -ve correlations and also categorical features that gives no

#information to the target variable

bike\_data\_final=bike\_data.drop(['instant','dteday','weekday','workingday','holiday','atemp','casual','registered'],axis=1)

x=bike\_data\_final.drop('cnt',axis=1)

y=bike\_data\_final['cnt']

#is to check column names of removing target feature

bike\_data\_final.columns

#below is the steps to divide dataset into train and test data sets

x\_train,x\_test,y\_train,y\_test=train\_test\_split(x,y,test\_size=0.2)

#Applying DT regression

fit\_Dt=DecisionTreeRegressor(max\_depth=50).fit(x\_train,y\_train)

#storing predicted values into y\_pred using above model called fit\_Dt

y\_pred=fit\_Dt.predict(x\_test)

#to see Actual values and predicted values

df=pd.DataFrame({'Actual':y\_test,'Prediction':y\_pred})

df

#calculation of MAPE

def MAPE(x,y):

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

return mape

#Percentage of error in out DT Regression Model

count=MAPE(y\_test,y\_pred)

count

#below is to divide dataset into train and test data sets for linear regression

bike\_data\_final=bike\_data.drop(['instant','dteday','weekday','workingday','holiday','atemp','casual','registered'],axis=1)

x=bike\_data\_final.drop('cnt',axis=1)

y=bike\_data\_final['cnt']

x\_train,x\_test,y\_train,y\_test=train\_test\_split(x,y,test\_size=0.2)

# Train the model using the linear regression

model = sm.OLS(y\_train, x\_train).fit()

#summary of above model

model.summary()

#prediction of values of above model

y\_pred= model.predict(x\_test)

#to see Actual values and predicted values

df=pd.DataFrame({'Actual':y\_test,'Prediction':y\_pred})

#MAPE1 function

def MAPE1(x,y):

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

return mape

#Error percentage

count=MAPE1(y\_test,y\_pred)

count

#below is to divide dataset into train and test data sets for RandomForestRegressor

from sklearn.ensemble import RandomForestRegressor

bike\_data=pd.read\_csv("day.csv",sep=",")

bike\_data\_final=bike\_data.drop(['instant','dteday','weekday','workingday','holiday','atemp','casual','registered'],axis=1)

x=bike\_data\_final.drop('cnt',axis=1)

y=bike\_data\_final['cnt']

x\_train,x\_test,y\_train,y\_test=train\_test\_split(x,y,test\_size=0.2)

#RandomForest Regressor

model\_RF=RandomForestRegressor(n\_estimators= 1000, random\_state=0).fit(x\_train,y\_train)

y\_pred= model.predict(x\_test)

count=MAPE1(y\_test,y\_pred)

count

#prediction of values of above model

y\_pred= model.predict(x\_test)

#to see Actual values and predicted values

df\_RF=pd.DataFrame({'Actual':y\_test,'Prediction':y\_pred})

#Error percentage

count=MAPE1(y\_test,y\_pred)

count