Bike Renting

Vinayak Chaturvedi

27-April-2019

Contents

Chapter 1

Introduction

1.1 Problem Statement

The objective of this Case is to Predication of bike rental count on daily based on the environmental and seasonal settings.

1.2 Data

Our task is to build regression mode which will predict the daily bike rental count depending on multiple environmental and seasonal settings. Given below is a sample of the data set that we are using to predict the bike count:

Table 1.1: Sample Data Set (Columns: 1-8)

instant	dteday	season	yr	mnth	holiday	weekday	workingday
1	01-01-2011	1	0	1	0	6	0
2	02-01-2011	1	0	1	0	0	0
3	03-01-2011	1	0	1	0	1	1
4	04-01-2011	1	0	1	0	2	1
5	05-01-2011	1	0	1	0	3	1

Table 1.2: Sample Data Set (Columns: 9-16)

weathersit	temp	atemp	hum	windspeed	casual	registered	cnt
2	0.344167	0.363625	0.805833	0.160446	331	654	985
2	0.363478	0.353739	0.696087	0.248539	131	670	801
1	0.196364	0.189405	0.437273	0.248309	120	1229	1349
1	0.2	0.212122	0.590435	0.160296	108	1454	1562
1	0.226957	0.22927	0.436957	0.1869	82	1518	1600

Table 1.3: Sample Data Set (Columns: 1-8)

instant	dteday	season	yr	mnth	holiday	weekday	workingday
6	06-01-2011	1	0	1	0	4	1
7	07-01-2011	1	0	1	0	5	1
8	08-01-2011	1	0	1	0	6	0
9	09-01-2011	1	0	1	0	0	0
10	10-01-2011	1	0	1	0	1	1

Table 1.4: Sample Data Set (Columns: 9-16)

weathersit	temp	atemp	hum	windspeed	casual	registered	cnt
1	0.204348	0.233209	0.518261	0.089565	88	1518	1606
2	0.196522	0.208839	0.498696	0.168726	148	1362	1510
2	0.165	0.162254	0.535833	0.266804	68	891	959
1	0.138333	0.116175	0.434167	0.36195	54	768	822
1	0.150833	0.150888	0.482917	0.223267	41	1280	1321

As you can see in the table below we have the following 15 variables, using which we have to correctly predict the bike rental count:

Table 1.5: 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

Predictor Details:

1. instant: Record index
2. dteday: Date
3. season: Season (1: springer, 2: summer, 3:fall, 4:winter)
4. yr: Year (0: 2011, 1:2012)
5. mnth: Month (1 to 12) hr: Hour (0 to 23)
6. holiday: weather day is holiday or not (extracted from Holiday Schedule)
7. weekday: Day of the week
8. workingday: If day is neither weekend nor holiday is 1, otherwise is 0.
9. weathersit: (extracted from Free meteo)
    a. Clear, Few clouds, Partly cloudy, Partly cloudy
    b. Mist + Cloudy, Mist + Broken clouds, Mist + Few clouds, Mist
    c. Light Snow, Light Rain + Thunderstorm + Scattered clouds, Light Rain + Scattered clouds
    d. Heavy Rain + Ice Pallets + Thunderstorm + Mist, Snow + Fog
10. 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)
11. atemp: Normalized feeling temperature in Celsius. The values are derived via (t-t_min)/(t_max- t_min), t_min=-16, t_max=+50 (only in hourly scale)
12. hum: Normalized humidity. The values are divided to 100 (max)
13. windspeed: Normalized wind speed. The values are divided to 67 (max)
14. casual: count of casual users
15. registered: count of registered users
16. cnt: count of total rental bikes including both casual and 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 our data and check its impact on the dependent variable.

In the upcoming figures we can see various insights about the dataset

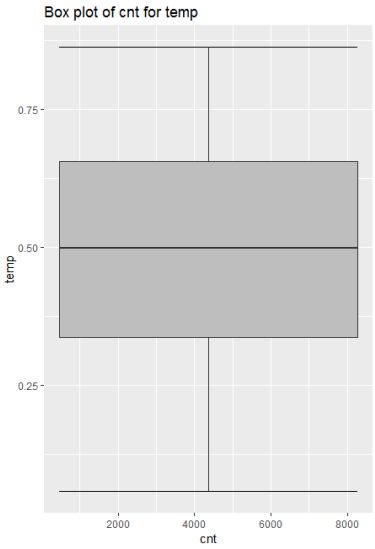
2.1.1 Outlier Analysis

One of the important steps in data pre-processing is outlier analysis.
In statistics, an **Outlier** is an observation point that is distant from other observations. An **outlier** may be due to variability in the measurement or it may indicate experimental error; the latter are sometimes excluded from the data set. Outliers are only present in continuous variable not in categorical variable.

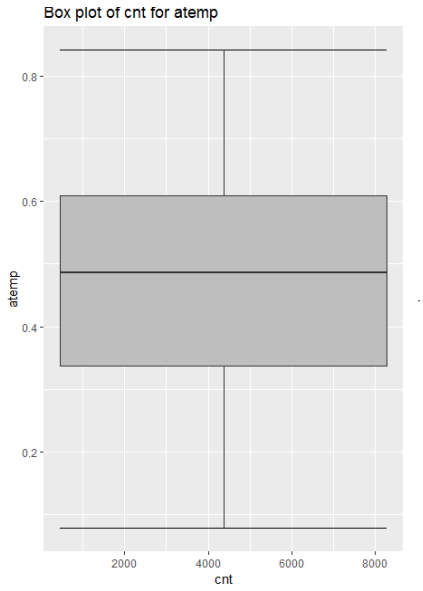
An **outlier** can cause serious problems in statistical analyses.
So to prevent these problems we need to take care of outliers that means we need to remove them.

Before removing them first we need to identify outliers. We can identify using **BoxPlot**
In the BoxPlot the points that are present above the header line or below the tailor line are the outliers
We can draw a charts to identify them.

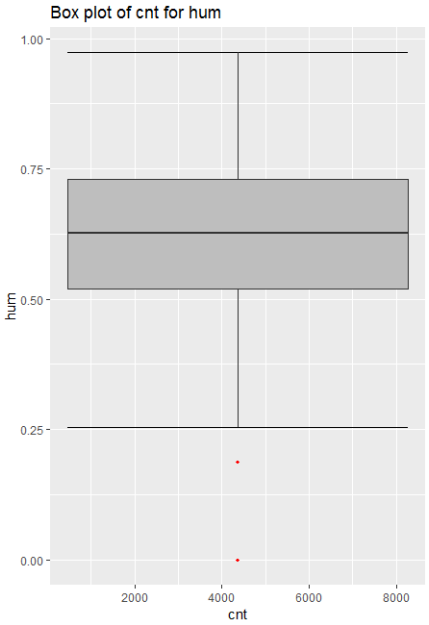
1. BoxPlot for Variable “Temp” : There is not outlier present in Temp.



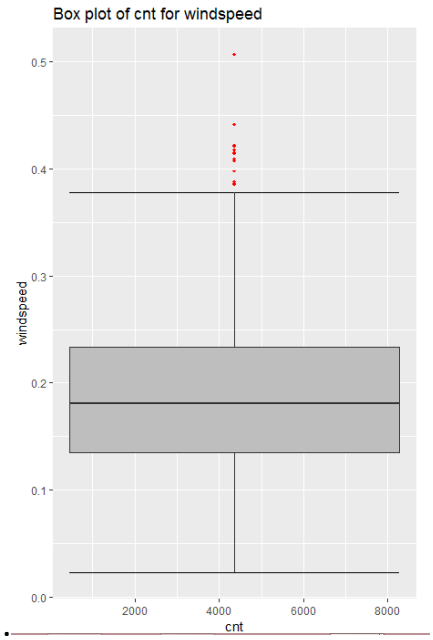
2. BoxPlot for Variable “ATemp” : There is not outlier present in ATemp.



3. BoxPlot for Variable “Hum” : There are 2 outliers present in Hum at the bottom.



4. BoxPlot for Variable “Windspeed” : There are various outliers present in Windspeed at the top.



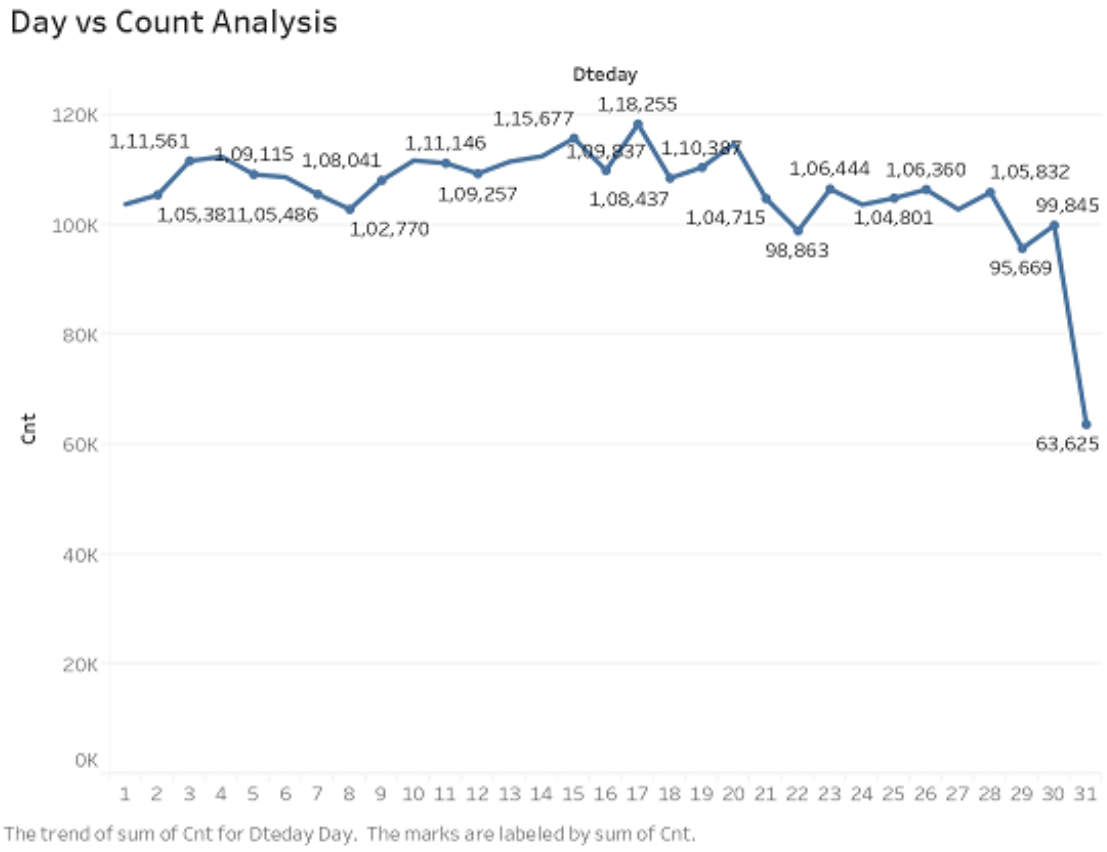
So after identifying them there are various ways to remove them:

1. Remove entire observation.
2. Replace that variable’s value with NA and then use missing value analysis to fill those values.

2.1.2 Feature Selection

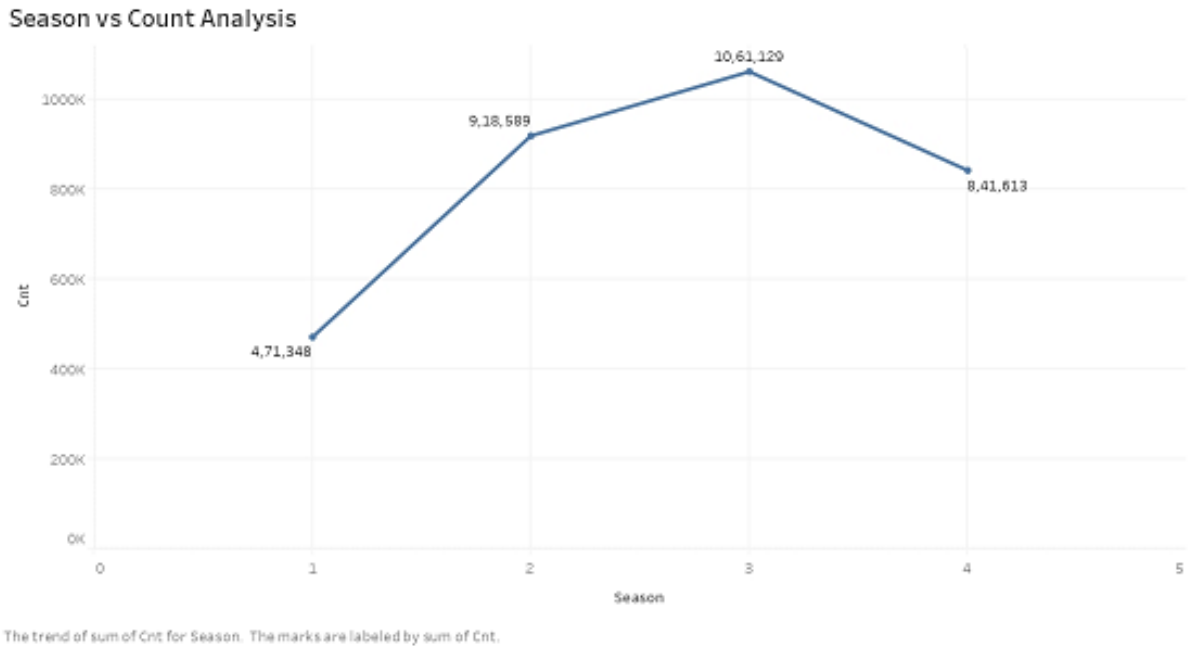
Before performing any type of modeling we need to assess the importance of each predictor variable in our analysis. There is a possibility that many variables in our analysis are not important at all to the problem of class prediction. There are several methods of doing that. Below we used graph (made using Tableau) to identify the impact of independent variable on the dependent variable and if the impact is not that much then we can remove that variable.

1. DteDay: format(dd-mm-yyyy) We do not actually need to add all the details from a date month and year and already present in the dataset in separate column, so we can consider the date only for our analysis.



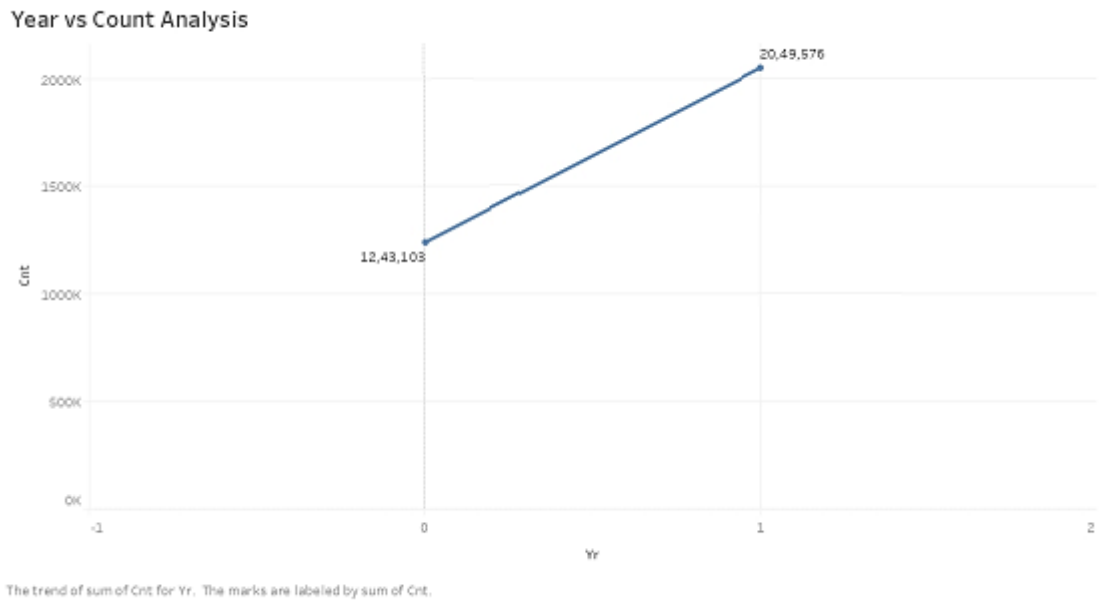
As we can see in the above graph that date not varying much, there is only 1 higher difference that on date: 31 the bike rental count is very less that's because only 7 months in a year have 31 days. And if we do not add this variable then it does not affect the model performance.

2. Season:



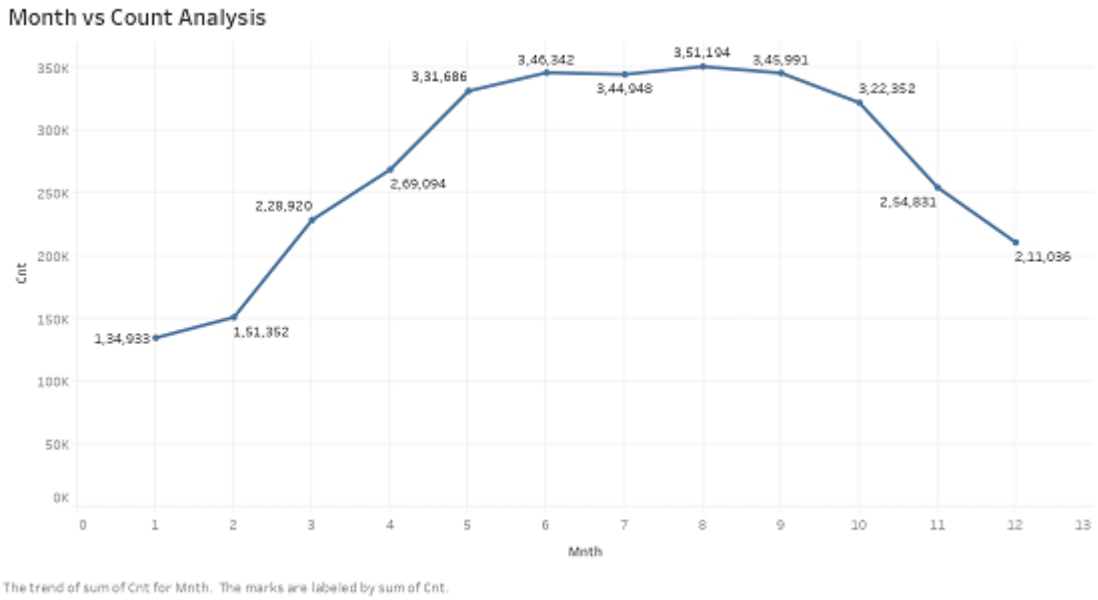
As we can see in the above graph highest number of bikes were rented in season 3: "Fall" and least in season 1: spring

3. Year



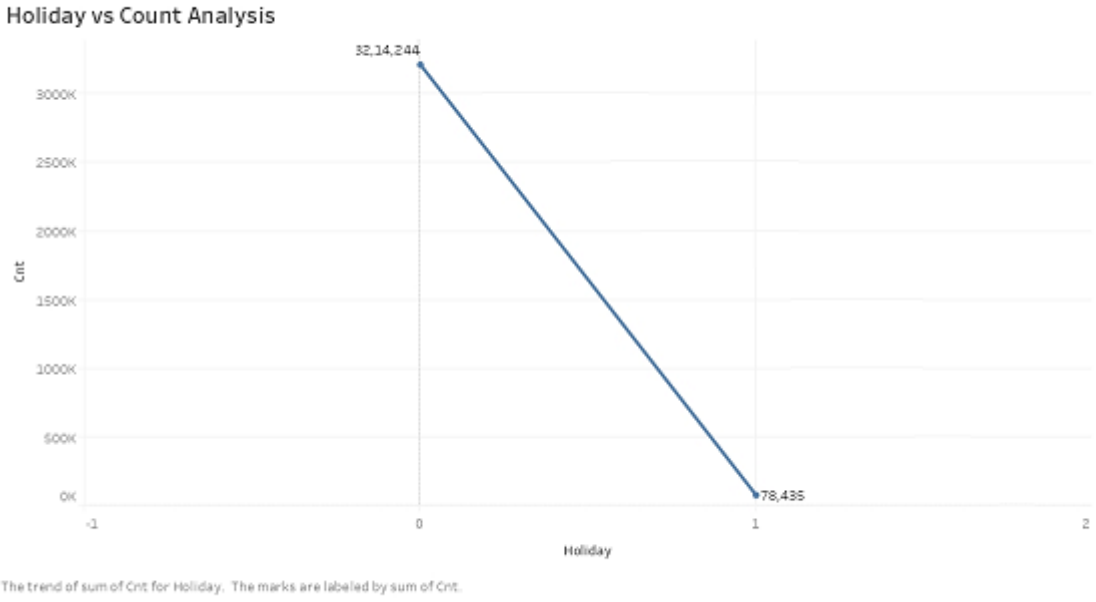
In year 2012 bike rental count is more than 2011

4. Month



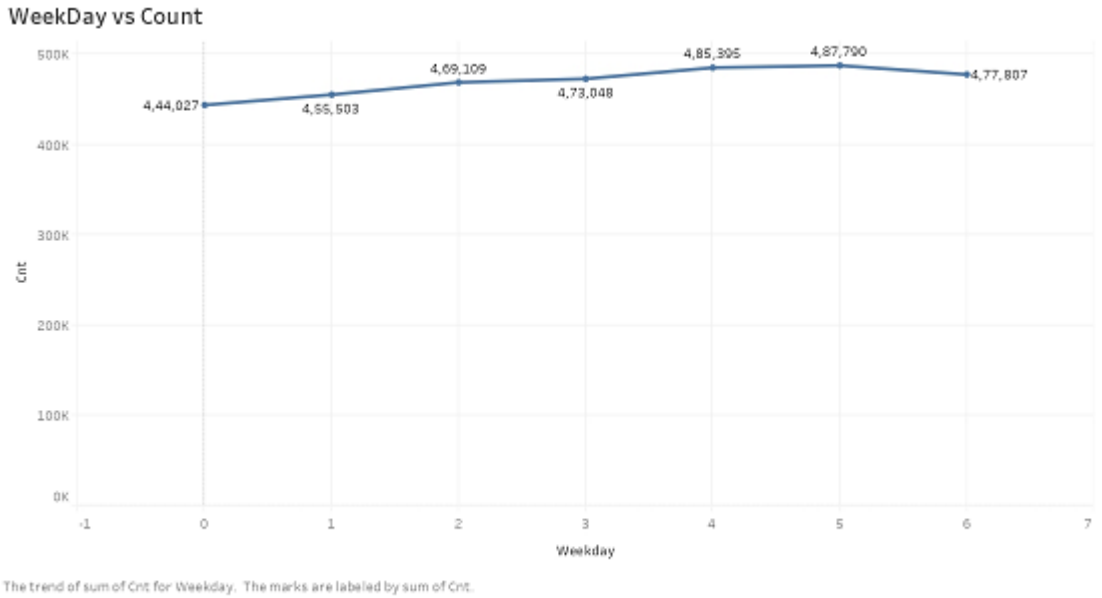
In beginning of the year bike rental count is less and it is increasing month by month till August but from August it starts falling down.

5. Holiday



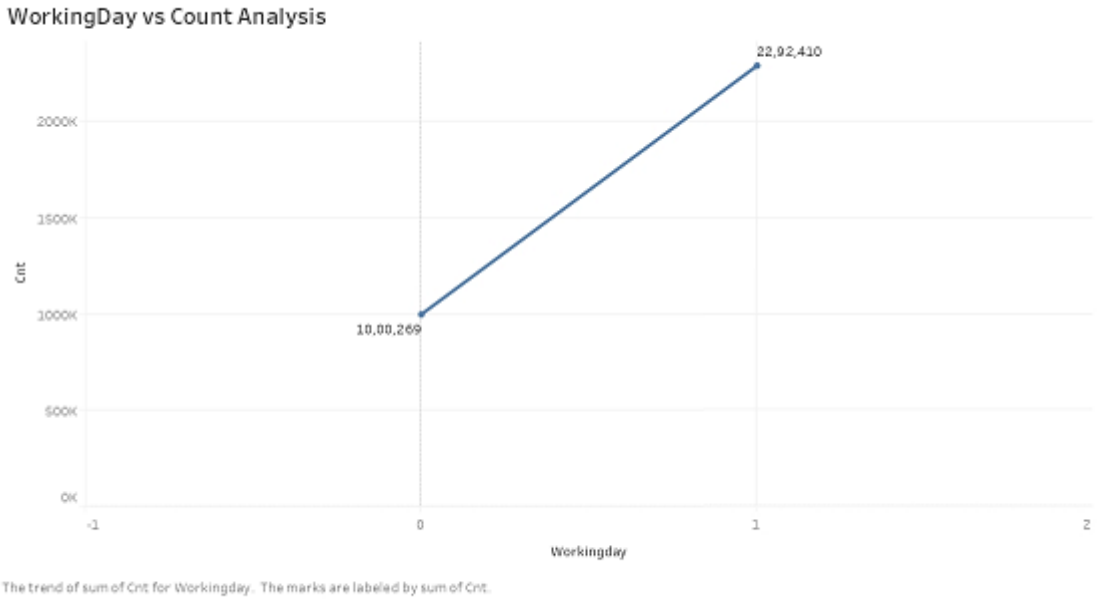
Holiday 0: then more bike rental count

6. Week Day



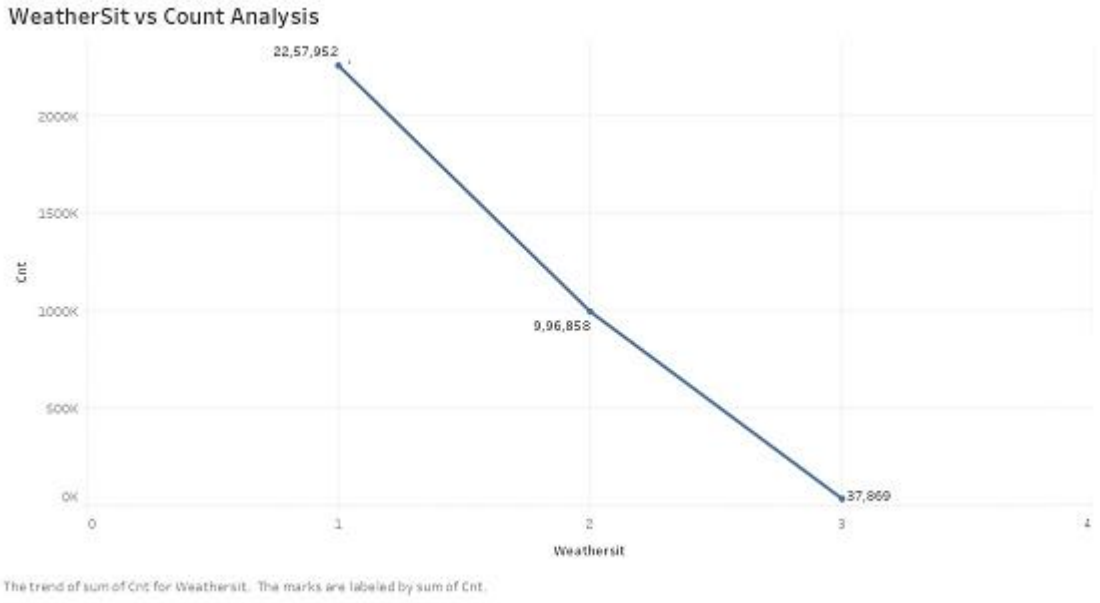
Bike rental count is Uniformly distributed for count.

7. Working Day



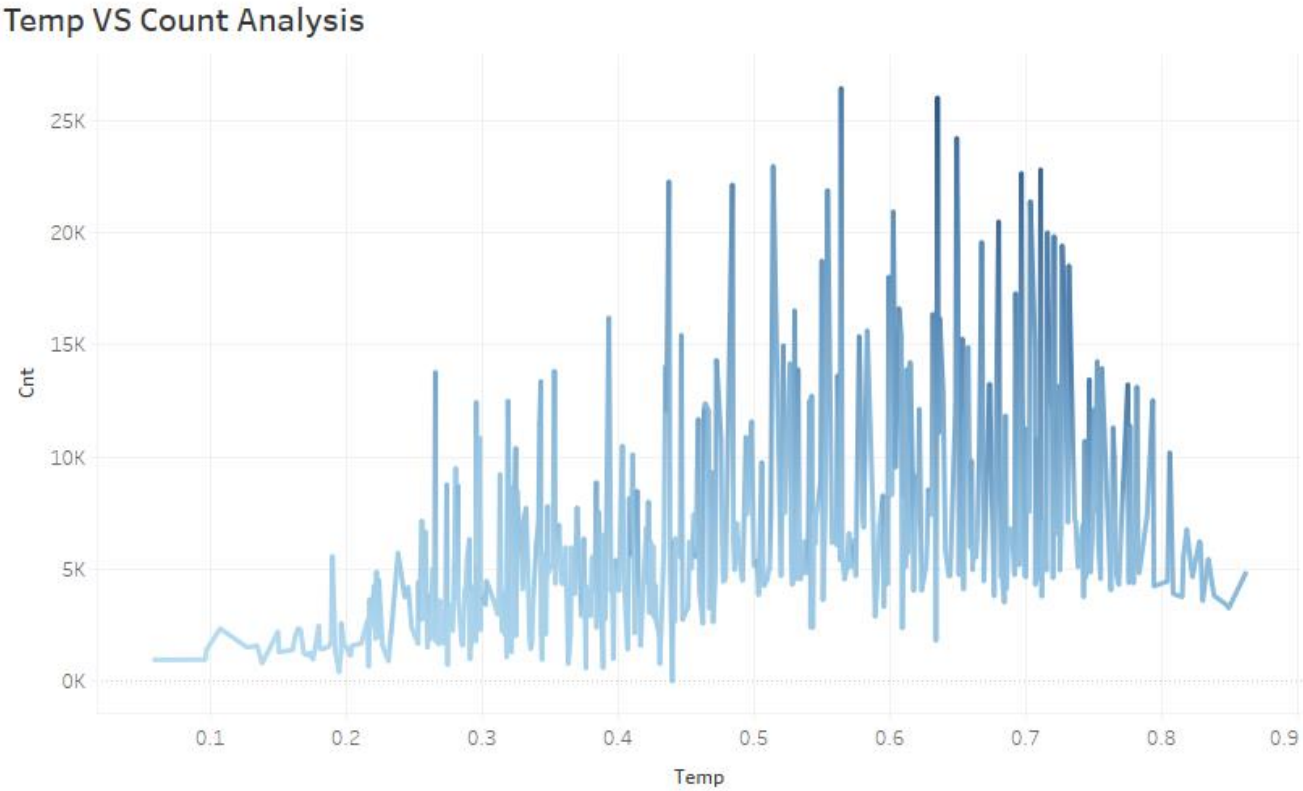
If day is neither weekend nor holiday, then count is high.

8. Weather Sit



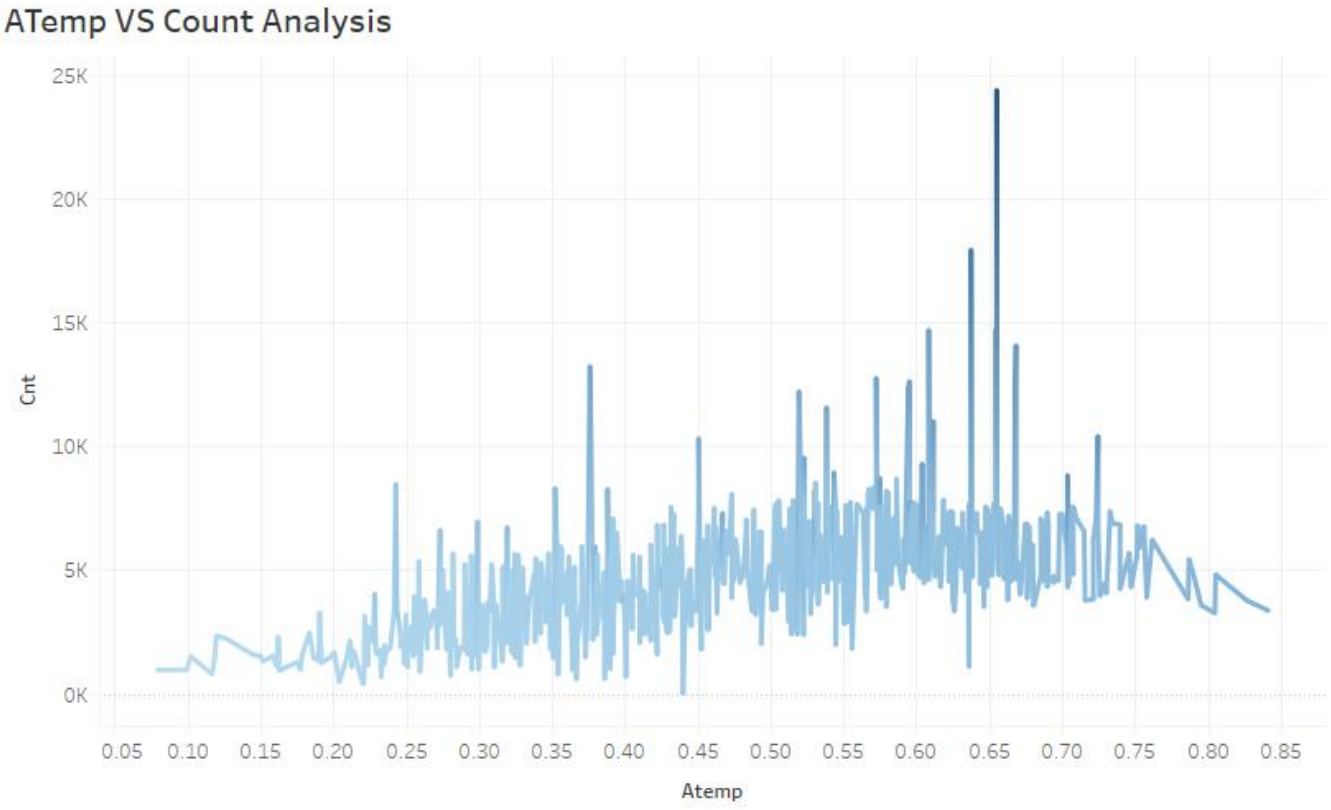
If weather is clear, then count is extremely high.

9. Temperature:

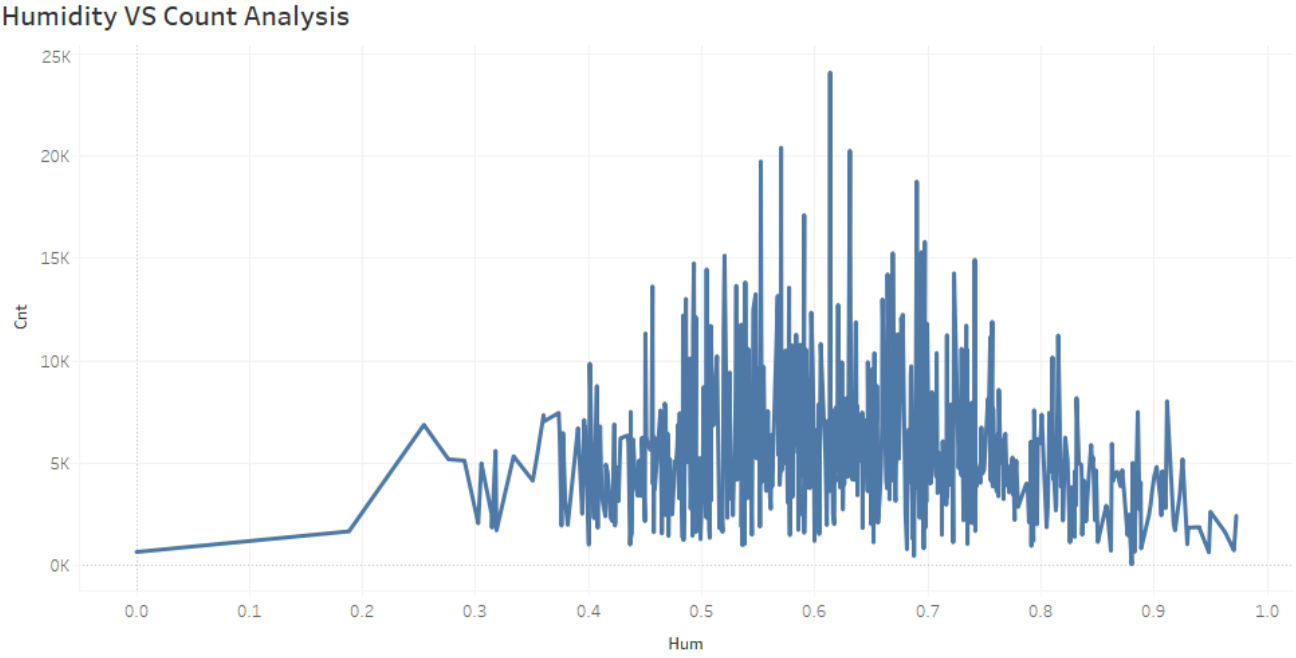


If temp's Normalized value is somewhere between 0.45 to 0.75 then count is good.

10. Atemp:

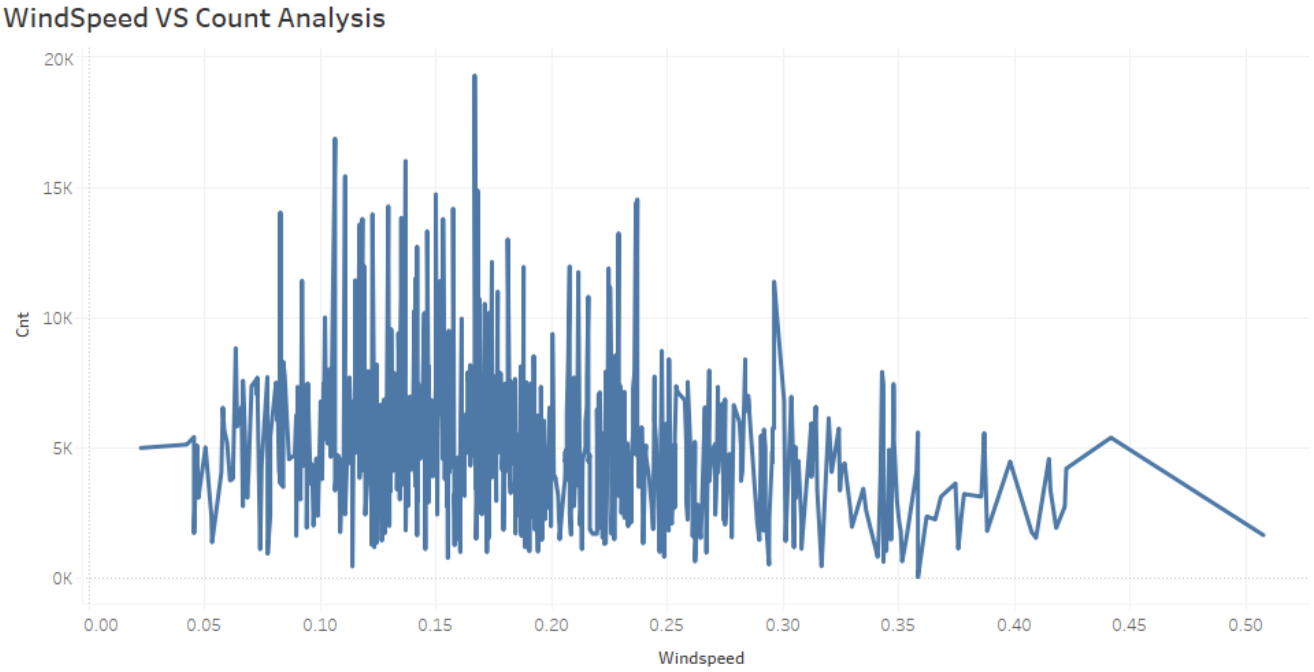


11. Humidity:



Count is good for Humidity value between: 0.45 to 0.75

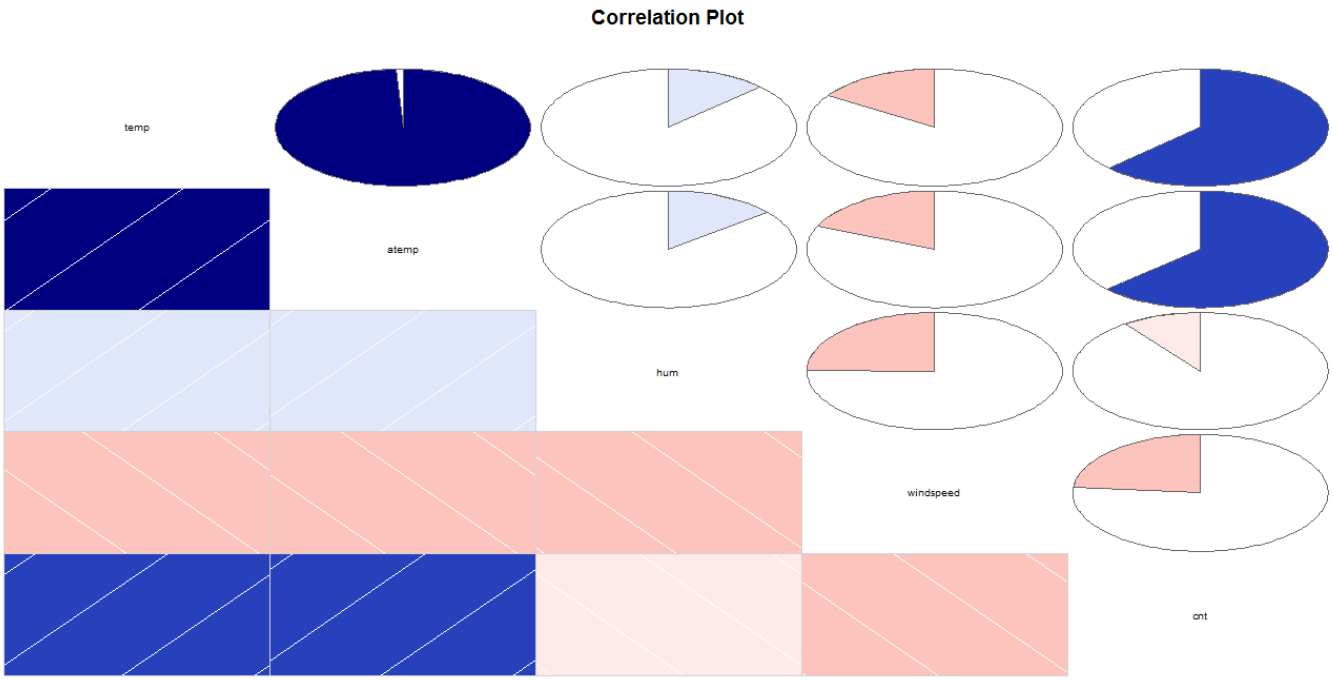
12. Wind Speed:



Lower value of wind speed is good.

Now remove the variable that are not useful for the analysis
1. Instant there is no need to add it as it only explains the row number
2. Dteday there is no need to add date as year and month are already present in 2 different columns and date does not have large impact on the result
3. Casual and Registered: Ideally we should not use these 2 variable because cnt = casual and Registered and we need to calculate the bike count on the basis of environmental and seasonal settings.

Correlation Plot



As we can see in the plot that that temp and atemp are very high +vely correlated then we can drop any 1 So here I am dropping atemp.

2.1.3 Missing Value Analysis

There are no missing values in the dataset so no need to perform missing value analysis.

2.1.4 Feature Scaling

We apply feature scaling only on continuous variable and, In the dataset there are only 3 numeric variables temp, hum, windspeed and the value of these variables are in the range of 0 and 1, so no need to apply feature scaling.

2.2 Modeling

2.2.1 Model Selection:

Our dataset contains 2 types of variable:

1. Continuous
2. Categorical

So identify which machine learning gives us the best result we will apply all the algorithm and then will compare the results with “MAPE”

First we will divide our dataset into 2 parts:

1. Training set
2. Test Set

We train our model using training set and test the model accuracy using test set.

Models

1. Mutiple Linear Regression: Multiple linear regression is the most common form of linear regression analysis. As a predictive analysis, the multiple linear regression is used to explain the relationship between one continuous dependent variable and two or more independent variables.

   Accuracy:
   a. Python :: 80.84%
   b. R :: 79.53%
2. Regression Tree: In computer science, Decision tree learning uses a decision tree to go from observations about an item to conclusions about the item's target value. It is one of the predictive modeling approaches used in statistics, data mining and machine learning. It allows input variables to be a mixture of continuous and categorical variables

   Accuracy:
   a. Python :: 84.09
   b. R :: 78.02%

3. Random Forest: Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees.

Accuracy:

a. Python :: 88.28%
b. R :: 80.42%

4. SVR: This model uses the concept of support vector machines to predict the value of dependent variable.

Accuracy:

a. R :: 83.56%

We calculate the accuracy of these Models using “MAPE”

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 Dataset, 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 target variables, and calculating some average error measure.

3.1.1 Mean Absolute Percentage Error (MAPE)

The mean absolute percentage error (MAPE), also known as mean absolute percentage deviation (MAPD), is a measure of prediction accuracy of a forecasting method in statistics, It used as a loss function for regression problems in machine learning. It usually expresses accuracy as a percentage, and is defined by the formula:

$$\mathrm{M} = \frac{100\%}{n} \sum_{t=1}^{n} \left| \frac{A_t - F_t}{A_t} \right|.$$

Where

where At is the actual value and Ft is the forecast value. The difference between At and Ft is divided by the actual value At again. The absolute value in this calculation is summed for every forecasted point in time and divided by the number of fitted points n. Multiplying by 100% makes it a percentage error.

Mean absolute percentage error is commonly used as a loss function for regression problems and in model evaluation, because of its very intuitive interpretation in terms of relative error.

3.2 Model Selection

As we can see that in Python “Random forest” is giving us the best overall result, so we can use it and, In R “SVR” is giving us the best result

Appendix A- Python Code

```
# -*- coding: utf-8 -*-
"""
Created on Sun Apr 21 00:10:03 2019

@author: vinayak
"""

#Load libraries

import os
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import chi2_contingency
import seaborn as sns
from random import randrange, uniform
from sklearn.cross_validation import train_test_split
from sklearn.tree import DecisionTreeRegressor
from sklearn import tree
from sklearn.metrics import accuracy_score
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestRegressor

#Set working directory
os.chdir("C:\Users\vinayak\Desktop\Bike Renting Project")

#Load data
dataset = pd.read_csv("day.csv")


####################################Missing value
analysis#############################################################
missing_val = pd.DataFrame(dataset.isnull().sum())
#There are no missing values in the dataset so no need to perform missing value analysis.

####################################Feature
Selection#############################################################
#Remove the variable that are not useful for the analysis
#1. Instant there is no need to add it as it only explains the row number
#2. Dteday there is no need to add date as year and month are already present in 2 different columns and
date
#   does not have large impact on the result
#3. Casual and Registered: Ideally we should not use these 2 variable because cnt = casual and Registered
#   and we need to calculate the bike count on the basis of environmental and seasonal settings.
```

```
dataset = dataset.drop(["instant","dteday","casual","registered"], axis=1)

#save numeric names
cnames =  ["temp","atemp","hum","windspeed"]

df_corr = dataset.loc[:,cnames]
#Set the width and hieght of the plot
f, ax = plt.subplots(figsize=(7, 5))
#Generate correlation matrix
corr = df_corr.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)

#As we can see in the plot that that temp and atemp are very high +vely correlated then we can drop any 1
#So here I am dropping atemp.
dataset = dataset.drop(['atemp'],axis = 1)


####################################Feature
scaling#########################################################
# In the dataset there are only 3 numeric variables temp, hum, windspeed and the value of these variable
# are in the range of 0 and 1, so no need to apply feature scaling.

#######################################Model
Development#############################################

#Divide data into train and test set
train, test = train_test_split(dataset, test_size=0.2)

#Calculate MAPE
def MAPE(y_true, y_pred):
   mape = np.mean(np.abs((y_true - y_pred) / y_true))*100
   return mape

#Decision tree for regression
regressor = DecisionTreeRegressor(max_depth =6, random_state = 0)
regressor.fit(train.iloc[:,0:10], train.iloc[:,10])          #Fit the model on training data
y_pred = regressor.predict(test.iloc[:,0:10])                 #predicting the test set results
MAPE(test.iloc[:,10], y_pred)                          #Compare with actual

#Error Rate: 15.91
#Accuracy: 84.09

#Linear Regression
regressor = LinearRegression()
regressor.fit(train.iloc[:,0:10], train.iloc[:,10])          #Fit the model on training data
```

```
y_pred = regressor.predict(test.iloc[:,0:10])              #predicting the test set results
MAPE(test.iloc[:,10], y_pred)                    #Compare with actual

#Error Rate: 19.16
#Accuracy: 80.84

#Random Forest
regressor = RandomForestRegressor(n_estimators = 300, random_state = 0)
regressor.fit(train.iloc[:,0:10], train.iloc[:,10])          #Fit the model on training data
y_pred = regressor.predict(test.iloc[:,0:10])              #predicting the test set results
MAPE(test.iloc[:,10], y_pred)                    #Compare with actual

#Error Rate: 11.72
#Accuracy: 88.28

#NOTE: If you run the Model again and again then you might see some slight difference in the accuracy because we are
# creating the test and training set randomly, because of this, accuracy difference would be there after every run.
```

Appendix B- R Code

```
#Clear the environment
rm(list=ls())

#set the working directory
setwd(dir = "C:/Users/vinayak/Desktop/Bike Renting Project")

#Load Libraries
x = c("ggplot2", "corrgram", "DMwR", "caret", "randomForest", "unbalanced", "C50", "dummies", "e1071",
"Information",
    "MASS", "rpart", "gbm", "ROSE", 'sampling', 'DataCombine',
'inTrees',"createDataPartition","usdm","randomForest","e1071")

install.packages(x)
lapply(x, require, character.only = TRUE)
rm(x)

#Load the dataset
dataset = read.csv("day.csv")

#Check the structure of the dataset
str(dataset)

#There are no missing values in the dataset so no need to perform missing value analysis.

#Univariate Analysis and Variable Consolidation
dataset$season = as.factor(dataset$season)
dataset$yr = as.factor(dataset$yr)
dataset$mnth = as.factor(dataset$mnth)
dataset$holiday = as.factor(dataset$holiday)
dataset$weekday = as.factor(dataset$weekday)
dataset$workingday = as.factor(dataset$workingday)
dataset$weathersit = as.factor(dataset$weathersit)

# Remove the variable that are not useful for the analysis
# 1. Instant there is no need to add it as it only explains the row number
# 2. Dteday there is no need to add date as year and month are already present in 2 different columns and
date
#   does not have large impact on the result
# 3. Casual and Registered: Ideally we should not use these 2 variable because cnt = casual and Registered
#   and we need to calculate the bike count on the basis of environmental and seasonal settings.

c = c(-1,-2,-14,-15)
dataset = dataset[c]
```

```
###################################Outlier
Analysis#################################################

numeric_index = sapply(dataset,is.numeric) #selecting only numeric
numeric_data = dataset[,numeric_index]

factor_index = sapply(dataset,is.factor)  #selecting only factor
factor_data = dataset[,factor_index]


cnames = colnames(numeric_data)
#
for (i in 1:length(cnames))
{
  assign(paste0("gn",i), ggplot(aes_string(y = (cnames[i]), x = "cnt"), data = subset(dataset))+
         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],x="cnt")+
         ggtitle(paste("Box plot of cnt for",cnames[i])))
}
#
## Plotting plots together
gridExtra::grid.arrange(gn1,gn2,gn3,ncol=3)
gridExtra::grid.arrange(gn4,ncol=3)

for(i in cnames){
  val = dataset[,i][dataset[,i] %in% boxplot.stats(dataset[,i])$out]
  #print(length(val))
  dataset[,i][dataset[,i] %in% val] = NA
}
sum(is.na(dataset))

#Impute NA using KNN impute
dataset = knnImputation(dataset, k = 3)

###################################Feature
Selection#################################################
## Correlation Plot
corrgram(dataset[,numeric_index], order = F,
         upper.panel=panel.pie, text.panel=panel.txt, main = "Correlation Plot")


#As we can see in the plot that that temp and atemp are very high +vely correlated then we can drop any 1
#So here I am dropping atemp.
dataset = dataset[,-9]
```

```
####################################Feature
scaling#########################################################
# In the dataset there are only 3 numeric variables temp, hum, windspeed and the value of these variable
# are in the range of 0 and 1, so no need to apply feature scaling.

#####################################Model
Development#############################################
#Clean the environment
rmExcept("dataset")


#Divide data into train and test set
set.seed(1234)
train_index = sample(1:nrow(dataset), 0.8 * nrow(dataset))
train = dataset[train_index,]
test = dataset[-train_index,]

#MAPE
#calculate MAPE
MAPE = function(y, yhat){
  mean(abs((y - yhat)/y))*100
}


#rpart for regression
fit = rpart(cnt ~ ., data = train, method = "anova")
predictions_DT = predict(fit, test[,-11])     #Predict for new test cases
MAPE(test[,11], predictions_DT)

#Error Rate: 21.98337
#Accuracy: 78.01663

#run regression model
lm_model = lm(cnt ~., data = train)
predictions_LR = predict(lm_model, test[,-11])    #Predict
MAPE(test[,11], predictions_LR)                #Calculate MAPE

#Error Rate: 20.46906
#acuracy: 79.53094%

# Fitting Random Forest Regression to the dataset
set.seed(1234)
regressor = randomForest(x = train[-11], y = train$cnt,ntree = 500)
predictions_regressor = predict(regressor, test[,-11])
MAPE(test[,11], predictions_regressor)

#Error Rate: 19.58
#acuracy: 80.42%
```

```
# Fitting SVR to the dataset
regressor = svm(formula = cnt ~ .,data = train,type = 'eps-regression',kernel = 'radial')
predictions_regressor = predict(regressor, test[,-11])
MAPE(test[,11], predictions_regressor)

#Error Rate: 16.44
#acuracy: 83.56%
```

Reference:

1. https://www.wikipedia.org/
2. https://edwisor.com/home