Predicting Bike Rental Count

Harshita Prasad

10 February 2020

Table of Contents

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

We need to build a statistical model that will predict the bike rental count on daily basis based on different environmental and seasonal settings. Given below is the details of data attributes in the dataset. Table 1.1 and Table 1.2 contains a sample of the dataset and table 1.3 contains all the independent variables whereas table 1.4 contains the dependent variables -

- 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)
- holiday: Whether 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 from Freemeteo)
    - 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

Table 1.1: Bike Rental Count Sample Data (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
6	06/01/2011	1	0	1	0	4	1

Table 1.2: Bike Rental Count Sample Data (Columns: 8-16)

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
1	0.204348	0.233209	0.518261	0.089565	88	1518	1606

Table 1.3: Independent 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

Table 1.4: Dependent Variables

S.No.	Response
1	Casual
2	Registered
3	Cnt

Cnt = Casual + Registered

Chapter 2. Methodology

2.1. Pre-processing

The first step in building a statistical model starts with exploratory data analysis. It includes data mining which transforms the raw messy data into clean understandable data. Pre-processing of data mills the data through a series of steps like data cleaning, data transformations and data reduction and includes data visualization thorough graphs and plots. Going further, we will discuss every step in detail and build up the methodology.

2.1.1. Data Exploration

The first step in pre-processing the dataset is to explore it in detail. In doing so, we get a good understanding of the dataset and the observations gathered from this exploration helps us further to analyse it. In order to do that, we need to look at various dimensions of the data. This includes identifying the distribution of the variables and comparing them with the help of various visualizations.

But before we do that, we need to make some changes in the dataset in order to understand it better:

- deleting instant variable since it is an index
- Feature engineering – changing dteday variable's date format into just date value to create a categorical variable with values ranging from 1 to 31 possible dates
- Changing variable names to make it easier to understand. Dteday to date, yr to year, mnth to month, weathersit to weather, hum to humidity and cnt to count
- Converting season, year, month, holiday, weekday, workingday and weather variables into categorical variables
- Omitting casual and registered variable and taking only count variable as target variable since count is nothing but sum of both casual and registered variables

Next, we need to visualize the data by various methods in order to make it easier to understand, compare and analyse.

Fig 2.1 shows scatter plots in tandem with regression line of each independent continuous variable and the target variable. It is used to show the existing type of correlation between the variables. If the plot shows a rising regression line, it means that there is a positive relationship between the variables i.e. rise in one will increase the other. Similarly, a decreasing regression line shows a negative relation and a regression line with no tilt shows no correlation between the variables.

Fig 2.2 shows bar graphs of each independent categorical variable and the target variable. It is used to establish a relationship between the two variables. It clearly shows the number of bikes for each value in the categorical variable.

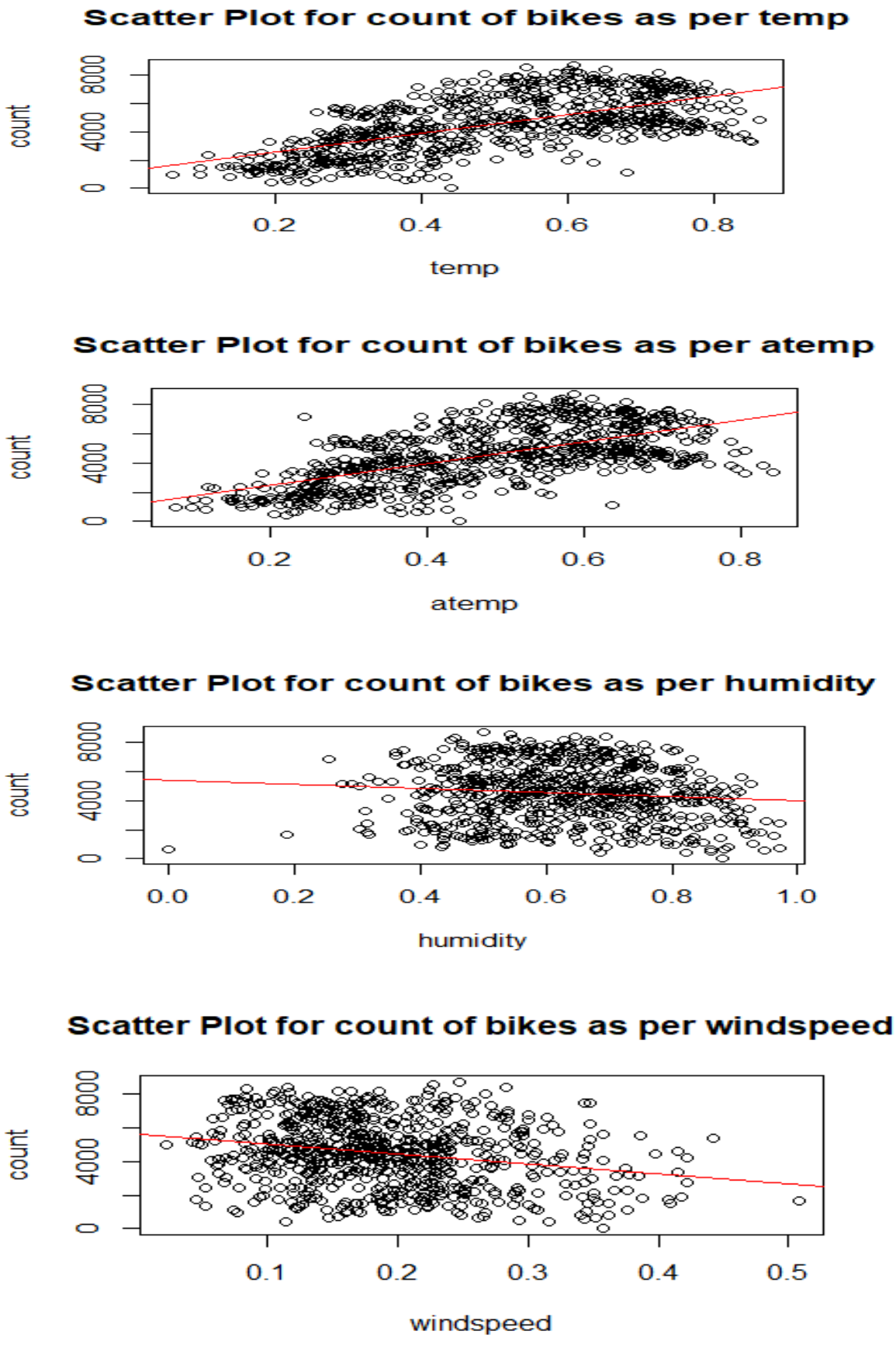
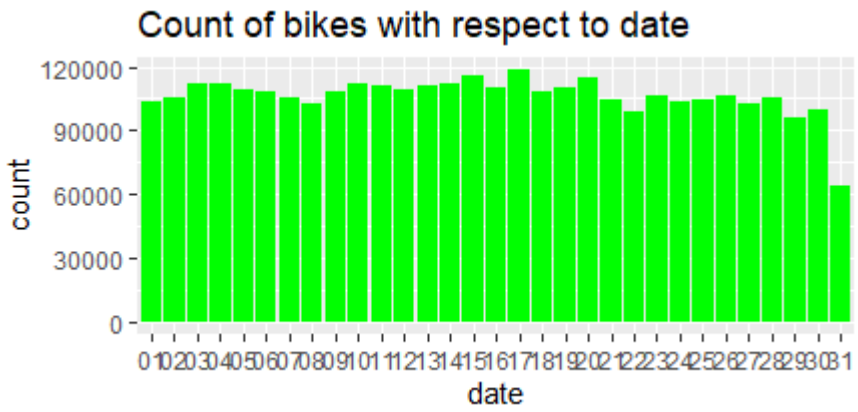
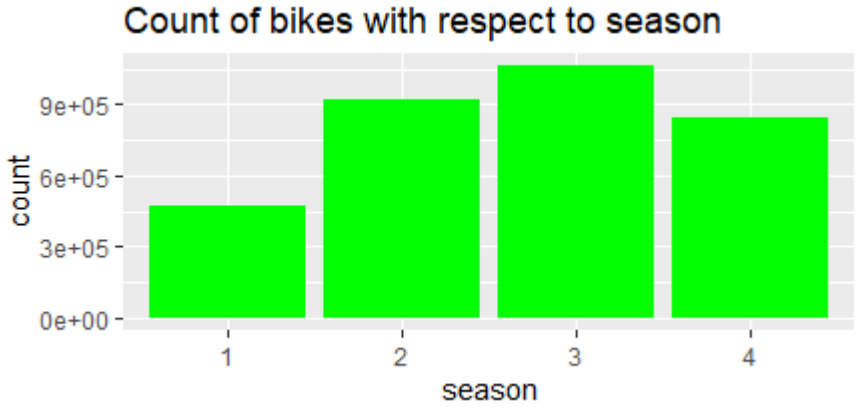
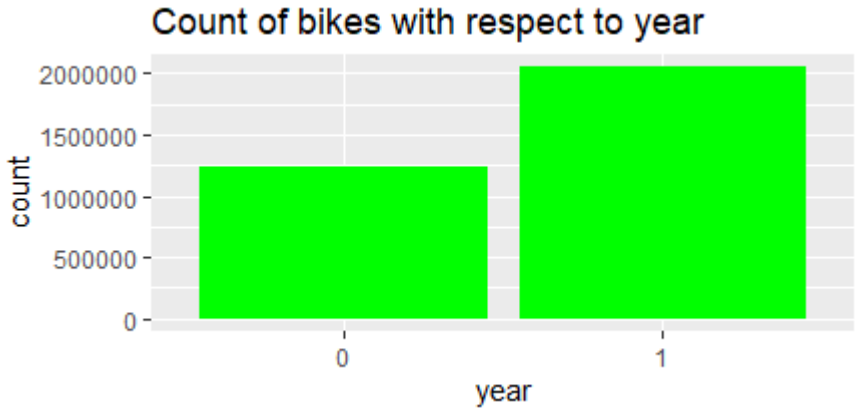
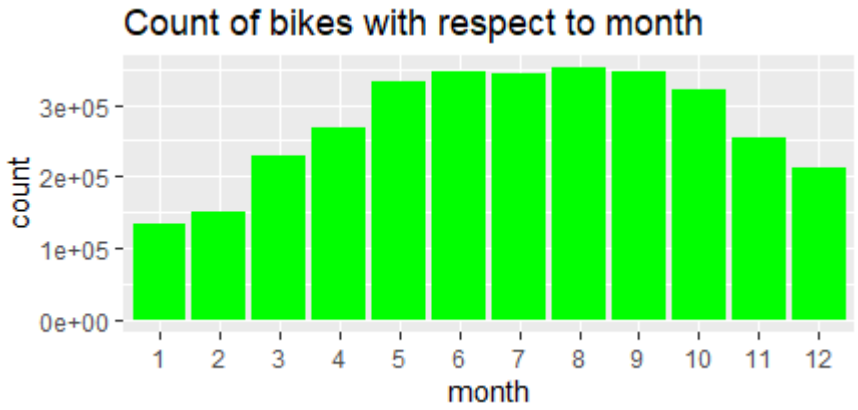


Figure 2.1: Scatter plot and regression line to compare each independent continuous variable and target variable


Count of bikes with respect to date
count
120000
90000
60000
30000
0
date


Count of bikes with respect to season
count
9e+05
6e+05
3e+05
0e+00
1
2
3
4
season


Count of bikes with respect to year
count
2000000
1500000
1000000
500000
0
0
1
year


Count of bikes with respect to month
count
3e+05
2e+05
1e+05
0e+00
1
2
3
4
5
6
7
8
9
10
11
12
month

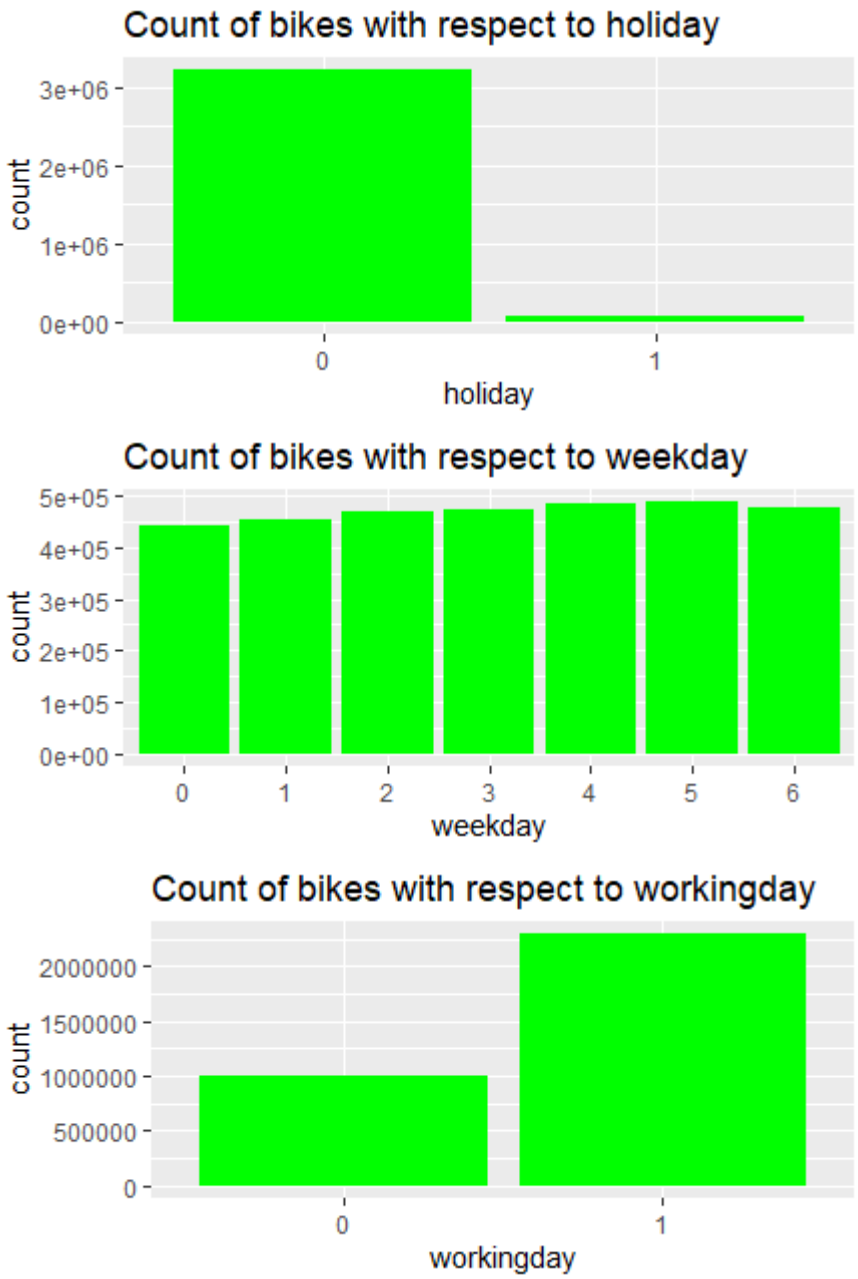


Figure 2.2: Bar graph to compare each independent categorical variable and target variable

Let's also talk about the distribution of the continuous variables. In order to predict a variable accurately, we need to know their behaviour. Once we determine the outcomes of the variables and assign a probability to them, we can then plot them and get a curve which is known as probability distribution curve. And, the likelihood of the target variable getting a value is the probability distribution or probability density function of the variable.

Normal Distribution is the most widely used probability distribution which comprises of the bell-shaped curve and the equality of mean, median and mode. The best way to showcase this is in the form of histograms combined with normal curve and Kernel Density Estimations. The variables that exhibit normal distribution is feasible to be forecasted with higher accuracy.

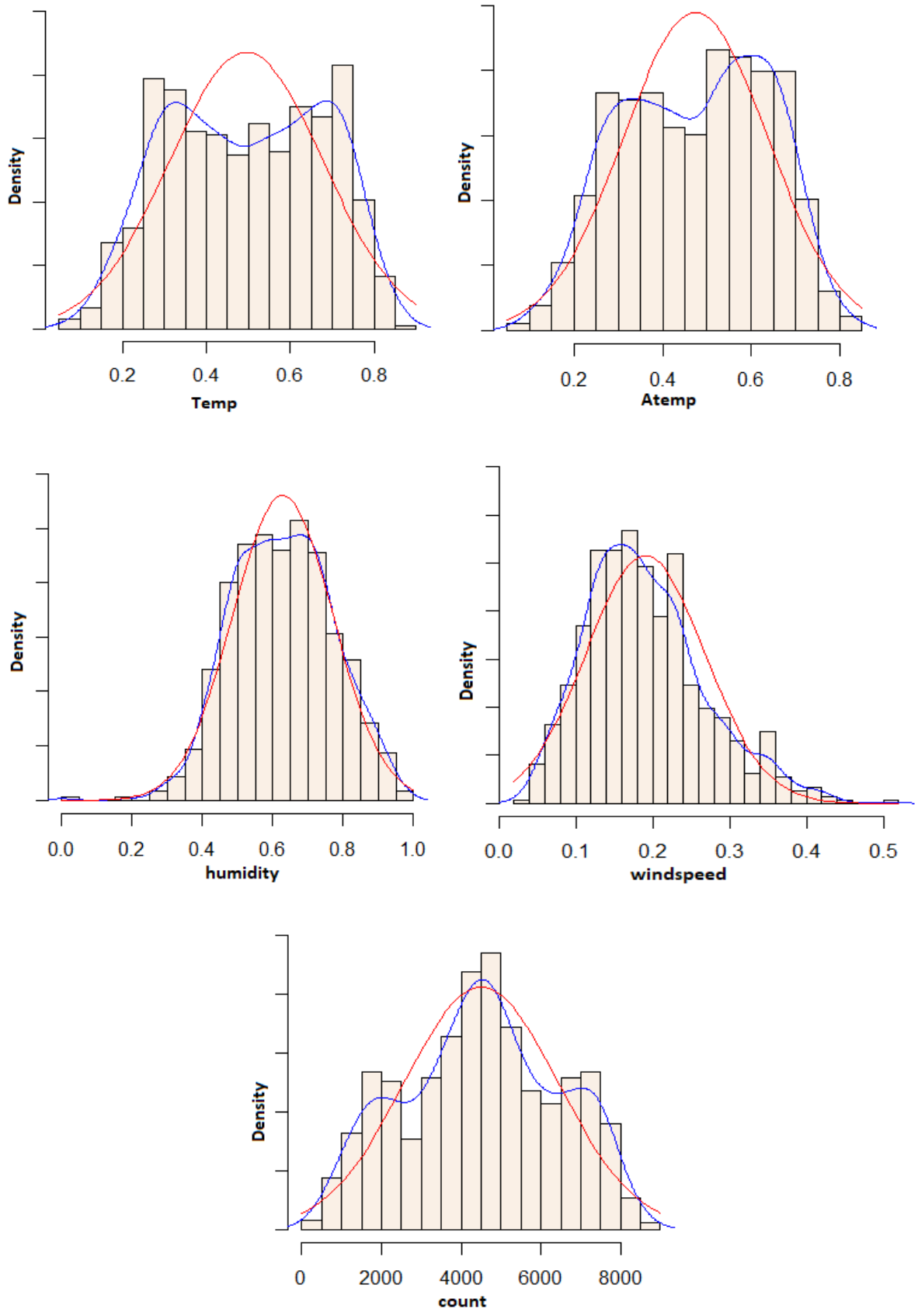


Figure 2.3: Probability Density Functions of continuous variables of bike count data

2.1.2. Missing Values

Missing value analysis is the second step to data pre-processing. The values that are missing from the dataset needs to be taken care of. It may happen due to human error, optional questions and refusal to answer survey questions. In this case, if the missing data is above 30%, it either needs to be removed or calculated using one of the methods like central statistics method, distance-based methods using KNN, Euclidean distance, etc or prediction method. We can see from fig 2.4 that there are no missing values in the dataset.

```
season        0
year          0
month         0
holiday       0
weekday       0
workingday    0
weather       0
temp          0
atemp         0
humidity      0
windspeed     0
count         0
```

Figure 2.4: Missing Value Analysis

2.1.3. Outlier Analysis

After dealing with the missing values, we proceed with the outlier analysis. This analysis is done to handle the observations which are inconsistent with the rest of the dataset. And, it can only be done on continuous variables.

Outliers can happen due to poor data quality, malfunctioning, manual error, exceptional data, etc. In order to detect the outliers and deal with them, we can utilise various methods. We can follow these steps:

- Detect the outliers using the Box Plot graphs.
- Take any one route:
  - Remove the outliers using box plot method
  - Impute it using central statistics like mean, median or KNN method

We can see from the box plots (fig 2.5) that only 2 variables – humidity and windspeed has outliers. But as we can see from the data, we not only have a class imbalance problem, but these variables define the environmental condition as mentioned in problem statement. Hence, we will choose to not deal with the outliers.

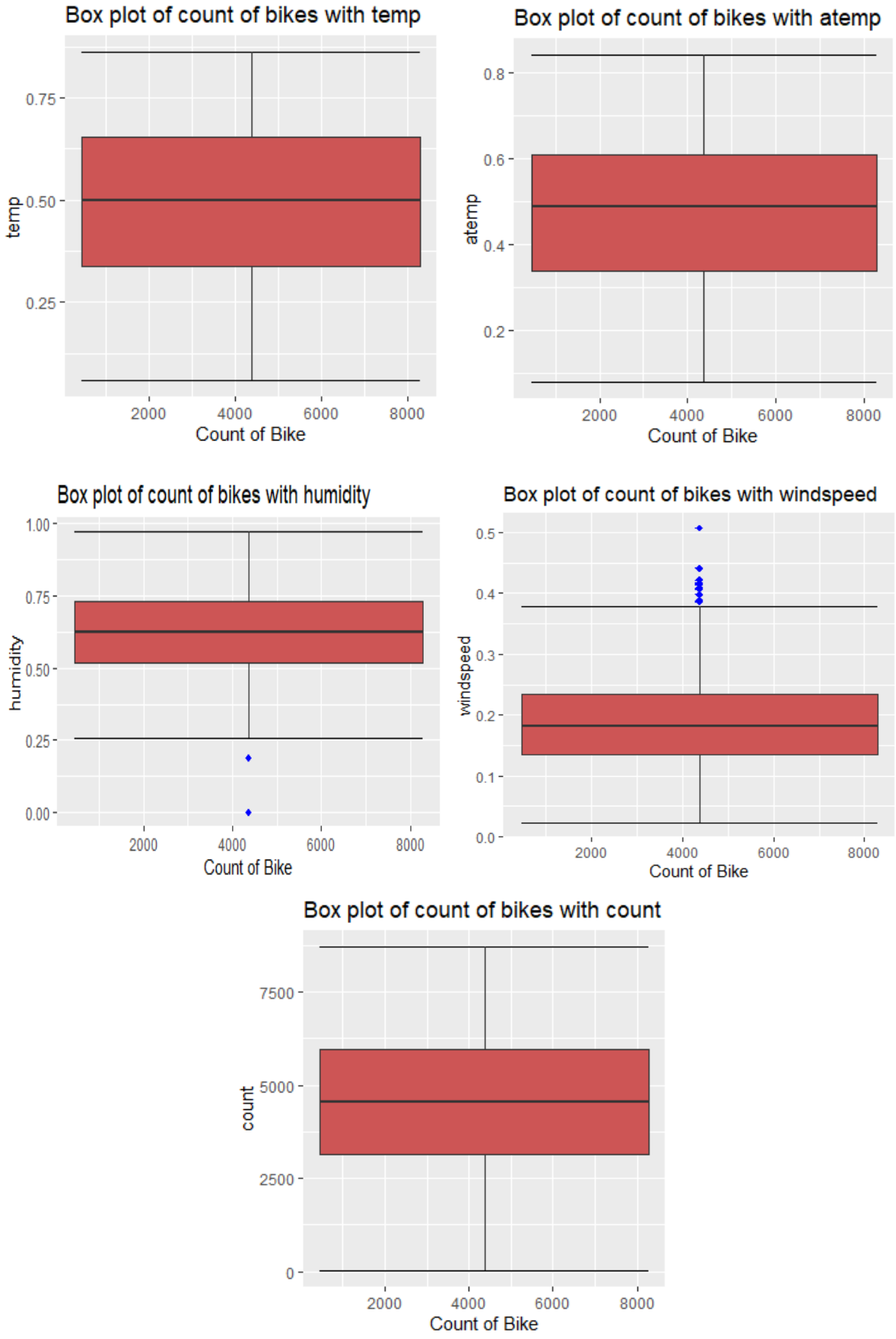


Figure 2.5: Box Plots of continuous variables

2.1.4. Feature Selection

The next step to pre-processing is feature selection analysis. It is done to extract relevant or meaningful features by selecting subsets of relevant features (variables and predictors) out of the data to be considered for model construction. Its also called dimensionality reduction. We look for redundant variables which will not be helpful in predicting the target variable and remove them from our dataset.

There are many methods that helps in feature selection analysis. Let's see some of them in detail:

1. **Correlation Analysis** – It tells you the association between two continuous variables. For two independent variables to be considered for the model, there should be no correlation between them. And for one independent and one dependent variable to be considered, they should be highly correlated. In order to do this analysis, we filter out the numerical variables and do a correlation analysis on them. We do it by plotting a correlation plot and studying it. We can see from the fig x that dark blue colour indicates a high correlation and light pink colour shows less or no correlation.

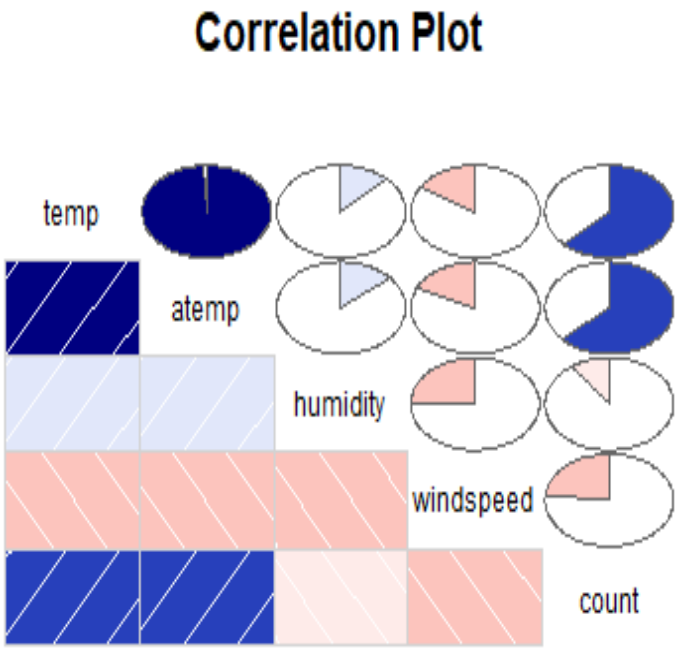


Figure 2.6: Correlation Plot

We can see that variables temp and atemp have high correlation. Since they both are independent variable, we can choose to remove one of them.

2. **Chi-Square Test of Independence** – It compares two categorical variables in a contingency table to check their association. For two independent variables to be considered, there should be no dependency. And, for one independent and dependent variable to be considered, they should have high dependency. In order to do a chi squared test we take the following steps:

a. Establish Hypothesis testing –
   Null hypothesis (H_0) – two variables are independent
   Alternate hypothesis (H_1) - two variables are not independent
b. To understand it -
   i. Chi squared statistics > critical value, reject Null hypothesis (H_0)
      p-value < 0.05, reject H_0 and remove the redundant variable
   ii. Chi squared statistics < critical value, accept Null hypothesis (H_0)
      p-value > 0.05, accept H_0 and keep the variables

After doing the analysis, we can choose to remove variables date, holiday, weekday and workingday as their p-value is less than 0.05.

3. **ANOVA (Analysis of Variance) Test** – It is used test association between one categorical variable and one continuous variable. It compares the means of two or more categories in a variable and tries to find a significance.
   a. Hypothesis testing -
      i. Null hypothesis (H_0) – means of all categories in a variable are same
      ii. Alternate hypothesis (H_1) - mean of at least one category in a variable is different
   b. To understand it -
      i. p-value < 0.05, reject H_0
      ii. p-value > 0.05, accept H_0

4. **Multicollinearity –** It is the occurrence of several independent variables in a regression model which are closely related. It can be dealt with in couple of ways. Either we choose to ignore it, which happens when prediction of dependent variable is the objective. Or, get rid of the redundant variable by using variable selection technique. And for this we need to check the VIF (variance inflation factor):
   a. If VIF = 1, not correlated to any of the variables
   b. If VIF is between 1 to 5, moderately correlated
   c. If VIF > 5, highly correlated

   In the case of high correlation in multiple variables, remove the one with highest VIF.

2.1.5. Feature Scaling

This next step to pre-processing is used to limit the range of variables so that they can be compared on common grounds. It is performed only on continuous variables.

- Normalization – It is the process of reducing unwanted variation either within or between variables to bring all the variables into proportion with one another. It gives a common scale to variables by converting the values in the range of 0 to 1.
- Standardization – this method is used when the data is normally/uniformly distributed. To convert the values, the formula subtracts mean from individual points and divides it by standard deviation.

Since most of our data is not normally distributed, we will choose normalization. We have also checked for variance before applying normalization. And, we will do it on variable count.

3. Modelling

3.1. Model Selection

The next step in analysing the data is selecting the appropriate model and developing it for further process. For this, we need to split the data into train and test data. Then build a model using the train data to predict the output using the test data. Model having less error rate and more accuracy will be chosen as our final model.

For our data, we have divided the data into 80% as train data and 20% as test data. And, the 7th variable new_count is our target variable. Since our data has a continuous target variable, we will use regression models. Following are the few popular models used for this purpose:

3.1.1. Decision Tree

It builds both regression and classification models based on a branching series of Boolean tests i.e. tree like graph. The motive is to create a training model used to predict the class/value of target variable by learning the decision roots inferred from historical data. Each node/ branch represents an attribute and each leaf represents a class label. It can handle both categorical and numerical data. There are different types of algorithms for decision trees like C5.0, C4, CART, etc.

3.1.2. Random Forest

It is an ensemble that consists of 'n' number of decision trees which combines Breiman "bragging" idea and the random selection of features. The idea is to combine multiple decision trees in determining the final output rather than using individual decision tree which will increase accuracy and decrease misclassification error rate. It can handle large number o independent variables without deletion and gives an estimate about importance of variables. It can build both regression and classification models. CART algorithm in tandem with Gini Index can be used for building a random forest model.

3.1.3. Linear Regression

It is the simplest and most common model for predictive analysis. It calculates weights/coefficients which shows the amount of variance it carries to explain the variable. The idea is to know how well the predictor variables predicts the target variable and establish a relationship between the variables. Also, which variables are significant predictors of the target variable, and in what way do they–indicated by the magnitude and sign of the beta estimates–impact the target variable? We can use simple liner regression in case of one independent variable which is the case for our data, and we can use multiple linear regression model for more than one independent variables.

4. Conclusion

4.1. Model Evaluation

In order to conclude our analysis, we need to evaluate the models and choose one. For this, we have few metrics out there that we can apply on our models:

4.1.1. MAPE

Mean absolute percentage error is a measure of prediction accuracy of a forecasting method. It measures accuracy in terms of percentage and lower the MAPE, better the fit.

4.1.2. MAE/MAD

Mean absolute error/deviation shows the average of the absolute errors.

4.1.3. RMSE/RMSD

Root mean squared error/deviation is a time-based measure. It squares the errors, find their average and takes the square root for calculations. Lower the RMSE, better the fit.

4.1.4. MSE

Mean squared error

4.2. Model Selection

After calculation all the methods for each model, we can see from fig 2.7 that we can choose Random Forest model since it gives the least error of 19.83% and accuracy of 80.17%.

(in percent)

	MAPE	MAE	RMSE	MSE
Decision Tree	28.25	8.66	11.84	1.41
Random Forest	19.83	6.01	8.34	0.69
Linear Regression	21.87	7.11	9.88	0.98

Figure 2.7: Error Metrices

5. R Code

```
rm(list = ls())
setwd("C:/Users/Harshita Prasad/Desktop/predicting bike rental counts")
x = c("ggplot2", "corrgram", "DMwR", "caret", "randomForest", "unbalanced", "C50",
"dummies", "e1071", "Information", "MASS", "rpart", "gbm", "ROSE", 'sampling',
'DataCombine', 'inTrees', "psych", "usdm")
lapply(x, require, character.only = TRUE)
rm(x)
original_data = read.csv('day.csv',header = T,na.strings = c("","  ","NA"))
df = original_data

#data exploration
str(df)
summary(df)
df$dteday <- format(as.Date(df$dteday,format="%Y-%m-%d"), "%d")
df$instant <- NULL
df$casual <- NULL
df$registered <- NULL
apply(df, 2,function(x) length(table(x)))
names(df)[names(df) == 'dteday'] <- 'date'
names(df)[names(df) == 'yr'] <- 'year'
names(df)[names(df) == 'mnth'] <- 'month'
names(df)[names(df) == 'hum'] <- 'humidity'
names(df)[names(df) == 'weathersit'] <- 'weather'
names(df)[names(df) == 'cnt'] <- 'count'
categorical_var = c('date', 'season', 'year', 'month', 'holiday', 'weekday', 'workingday','weather')
numerical_var = c('temp', 'atemp', 'humidity', 'windspeed','count')

#data type conversion
typ_conv = function(df,var,type){
  df[var] = lapply(df[var], type)
  return(df)
}
df = typ_conv(df,categorical_var, factor)
str(df)

#visualizations
multi.hist(df$temp, main = NA, dcol = c("blue", "red"), title(main = NULL,xlab =
"temp",ylab = "density"),
         dlty = c("solid", "solid"), bcol = "linen")
multi.hist(df$atemp, main = NA, dcol = c("blue", "red"), title(main = NULL,xlab =
"atemp",ylab = "density"),
         dlty = c("solid", "solid"), bcol = "linen")
multi.hist(df$humidity, main = NA, dcol = c("blue", "red"), title(main = NULL,xlab =
"humidity",ylab = "density"),
         dlty = c("solid", "solid"), bcol = "linen")
multi.hist(df$windspeed, main = NA, dcol = c("blue", "red"), title(main = NULL,xlab =
"windspeed",ylab = "density"),
```

```
        dlty = c("solid", "solid"), bcol = "linen")
multi.hist(df$count, main = NA, dcol = c("blue", "red"), title(main = NULL,xlab =
"count",ylab = "density"),
        dlty = c("solid", "solid"), bcol = "linen")
```

#scatter plots
#bivariate analysis
#lets check distribution between target and continuous variables

```
plot(df$temp,df$count,main = 'Scatter Plot for count of bikes as per temp',
    xlab = 'temp', ylab = 'count')
abline(lm(count ~ temp, data = df), col = 'red')
plot(df$atemp,df$count,main = 'Scatter Plot for count of bikes as per atemp',
    xlab = 'atemp', ylab = 'count')
abline(lm(count ~ atemp, data = df), col = 'red')
plot(df$humidity,df$count,main = 'Scatter Plot for count of bikes as per humidity',
    xlab = 'humidity', ylab = 'count')
abline(lm(count ~ humidity, data = df), col = 'red')
plot(df$windspeed,df$count,main = 'Scatter Plot for count of bikes as per windspeed',
    xlab = 'windspeed', ylab = 'count')
abline(lm(count ~ windspeed, data = df), col = 'red')
```

#lets check categorical variables

```
for(i in 1:length(categorical_var))
{
  assign(paste0("b",i),ggplot(aes_string(y='count',x = (categorical_var[i])),
                        data=subset(df))+
         geom_bar(stat = "identity",fill = "green") +
         ggtitle(paste("Count of bikes with respect to",categorical_var[i])))+
    theme(axis.text.x = element_text( color="red", size=8))+
    theme(plot.title = element_text(face = "old"))
}
#lets plot between categorical and target variables
gridExtra::grid.arrange(b1,ncol=1)
gridExtra::grid.arrange(b2,ncol=1)
gridExtra::grid.arrange(b3,ncol=1)
gridExtra::grid.arrange(b4,ncol=1)
gridExtra::grid.arrange(b5,ncol=1)
gridExtra::grid.arrange(b6,ncol=1)
gridExtra::grid.arrange(b7,ncol=1)
gridExtra::grid.arrange(b8,ncol=1)
```

#missing value analysis

```
apply(df, 2, function(x) {sum(is.na(x))})
```

#outlier analysis

```
for (i in 1:length(numerical_var))
{
  assign(paste0("bp",i), ggplot(aes_string(x="count",y = (numerical_var[i])), d=df)+
         stat_boxplot(geom = "errorbar", width = 0.5) +
         geom_boxplot(outlier.colour="blue", fill = "indianred3" ,outlier.shape=18,
```

```
                outlier.size=2, notch=FALSE) +
        theme(legend.position="bottom")+
        labs(x="Count of Bike", y=numerical_var[i])+
        ggtitle(paste("Box plot of count of bikes with",numerical_var[i])))
}
gridExtra::grid.arrange(bp1,ncol=1)
gridExtra::grid.arrange(bp2,ncol=1)
gridExtra::grid.arrange(bp3,ncol=1)
gridExtra::grid.arrange(bp4,ncol=1)
gridExtra::grid.arrange(bp5,ncol=1)

#correlation
corrgram(df[,numerical_var], order = F,
       upper.panel=panel.pie, text.panel=panel.txt, main = "Correlation Plot")
df = subset(df, select=-c(atemp))

#chi-squared test
categorical_df = df[,categorical_var]
for (i in categorical_var){
  for (j in categorical_var){
    print(i)
    print(j)
    print(chisq.test(table(categorical_df[,i], categorical_df[,j]))$p.value)
  }
}

#ANOVA
anova_date =(lm(count ~ date, data = df))
summary(anova_date)
anova_season =(lm(count ~ season, data = df))
summary(anova_season)
anova_year =(lm(count ~ year, data = df))
summary(anova_year)
anova_month =(lm(count ~ month, data = df))
summary(anova_month)
anova_holiday =(lm(count ~ holiday, data = df))
summary(anova_holiday)
anova_weekday =(lm(count ~ weekday, data = df))
summary(anova_weekday)
anova_workingday =(lm(count ~ workingday, data = df))
summary(anova_workingday)
anova_weather =(lm(count ~ weather, data = df))
summary(anova_weather)

df = subset(df, select=-c(date, holiday, weekday, workingday))

#multicollinearity
vif(df)

#Normalization of count
```

```
min(df$count)
max(df$count)
hist(df$count)
df$new_count = (df$count - min(df$count)) / (max(df$count) - min(df$count))

df = subset(df, select=-c(count))

#clean environment
rmExcept(c("original_data",'df'))

#Divide the data intro train and test
set.seed(123)
train_index = sample(1:nrow(df), 0.8 * nrow(df))
train = df[train_index,]
test = df[-train_index,]

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

#Decision Tree Regression
fit = rpart(new_count ~ ., data = train, method = "anova")
DT_predictions = predict(fit, test[,-8])
MAPE(test[,8], DT_predictions)  #error = 28.25% or accuracy = 71.75%

#Random Forest
RF_model = randomForest(new_count ~ ., train, importance = TRUE, ntree = 100)
RF_predictions = predict(RF_model, test[,-8])
MAPE(test[,8], RF_predictions) #Error = 20.30% and Accuracy = 79.70%

RF_model = randomForest(new_count ~ ., train, importance = TRUE, ntree = 500)
RF_predictions = predict(RF_model, test[,-8])
MAPE(test[,8], RF_predictions) #Error = 19.86% and Accuracy = 80.14%

#Linear Regression
vif(df[,-8])
vifcor(df[,c(5,6,7)])
lm_model = lm(new_count ~., data = train)
summary(lm_model)
LR_predictions = predict(lm_model, test[,1:7])
MAPE(test[,8], LR_predictions) #Error = 21.87% and Accuracy = 78.13%

regr.eval(test[,8], DT_predictions, stats = c('mae', 'rmse', 'mape', 'mse'))
regr.eval(test[,8], RF_predictions, stats = c('mae', 'rmse', 'mape', 'mse'))
regr.eval(test[,8], LR_predictions, stats = c('mae', 'rmse', 'mape', 'mse'))

write.csv(df, "df.csv", row.names = F)
```

6. Python Code

```
import os
import pandas as pd
import numpy as np
import matplotlib as mlt
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeRegressor
from random import randrange, uniform

os.chdir ('C:/Users/Harshita Prasad/Desktop/predicting bike rental counts')

origina_data = pd.read_csv("day.csv", sep = ',')
df = origina_data

df.shape
df.columns

df = df.rename(columns =
{'instant':'index','dteday':'date','yr':'year','mnth':'month','weathersit':'weather',
'hum':'humidity','cnt':'count'})

df.dtypes
df.describe()
df = df.drop(['casual','registered','index','date'],axis=1)

numerical_var = ['temp', 'atemp', 'humidity', 'windspeed', 'count']
categorical_var=['season', 'year', 'month', 'holiday', 'weekday', 'workingday','weather']

df.dtypes

#Data type conversion
df['season'] = df['season'].astype(object)
df['year'] = df['year'].astype(object)
df['month'] = df['month'].astype(object)
df['holiday'] = df['holiday'].astype(object)
df['weekday'] = df['weekday'].astype(object)
df['workingday'] = df['workingday'].astype(object)
df['weather'] = df['weather'].astype(object)

#Pairplot for all numerical variables
pairwise_plot = sns.pairplot(data=df[numerical_var],kind='scatter')
pairwise_plot.fig.suptitle('Pairwise plot of all numerical variables')

#Missing value analysis
Missing_val = df.isnull().sum()
Missing_val
```

```
#Box plot
for i in numerical_var:
    sns.boxplot(y=i,data=df)
    plt.title('Boxplot of '+i)
    plt.savefig('bp'+str(i)+'.png')
    plt.show()

# Univariate Analysis
# temperature
sns.FacetGrid(df , height = 5).map(sns.distplot,'temp').add_legend()

#atemp
sns.FacetGrid(df , height = 5).map(sns.distplot,'atemp').add_legend()

# humidity
sns.FacetGrid(df , height = 5).map(sns.distplot,'humidity').add_legend()

# windspeed
sns.FacetGrid(df , height = 5).map(sns.distplot,'windspeed').add_legend()

# count
sns.FacetGrid(df , height = 5).map(sns.distplot,'count').add_legend()

#impact of continous variables on target variable
#count vs temperature
sns.regplot(x=df["count"], y=df["temp"])

#count vs atemp
sns.regplot(x=df["count"], y=df["atemp"])

#count vs humidity
sns.regplot(x=df["count"], y=df["humidity"])

#count vs windspeed
sns.regplot(x=df["count"], y=df["windspeed"])

# Correlation matrix continuous variables
plt.figure(figsize=(15,15))
_ =
sns.heatmap(df[['temp','atemp','humidity','windspeed','count']].corr(),linewidths=0.5,linecolor
='w',square=True,annot=True)
plt.title('Correlation matrix of all numerical variables')
plt.savefig('corrmap.png')
plt.show()

#chi square analysis
from scipy.stats import chi2_contingency
for i in categorical_var:
    print(i)
    chi2, p, dof, ex = chi2_contingency(pd.crosstab(df['count'], df[i]))
```

```
    print(p)

or,
for i in categorical_var:
    for j in categorical_var:
        if(i != j):
            chi2, p, dof, ex = chi2_contingency(pd.crosstab(df[i], df[j]))
            if(p < 0.05):
                print(i,"and",j,"are dependent on each other with",p,'----Remove')
            else:
                print(i,"and",j,"are independent on each other with",p,'----Keep')

df = df.drop(['atemp','holiday','weekday','workingday'],axis = 1)

df.shape

# numerical variable
num_var = ['temp','humidity', 'windspeed', 'count']
# Categorical variables
cat_var = ['season', 'year', 'month','weather']

#Nomalisation
df['new_count'] = (df['count'] - min(df['count']))/(max(df['count']) - min(df['count']))

df = df.drop(['count'],axis = 1)
df.columns

##Decision tree
#Divide data into train and test
train, test = train_test_split(df, test_size=0.2)

#Decision tree for regression
DT_fit = DecisionTreeRegressor(max_depth=2).fit(train.iloc[:,0:7], train.iloc[:,7])

#Apply model on test data
DT_predictions = DT_fit.predict(test.iloc[:,0:7])

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

MAPE(test.iloc[:,7], DT_predictions)

#Random Forest
from sklearn.ensemble import RandomForestRegressor
RF_model = RandomForestRegressor(n_estimators = 100).fit(train.iloc[:,0:7], train.iloc[:,7])
RF_Predictions = RF_model.predict(test.iloc[:,0:7])
MAPE(test.iloc[:,7], RF_Predictions)
```

```
RF_model1 = RandomForestRegressor(n_estimators = 500).fit(train.iloc[:,0:7],
train.iloc[:,7])
RF_Predictions1 = RF_model1.predict(test.iloc[:,0:7])
MAPE(test.iloc[:,7], RF_Predictions1)
```

#Linear Regression

```
import statsmodels.api as sm
LR_model = sm.OLS(train.iloc[:,7], train.iloc[:,0:7].astype(float)).fit()
LR_predictions = LR_model.predict(test.iloc[:,0:7])
MAPE(test.iloc[:,7], LR_predictions)
```