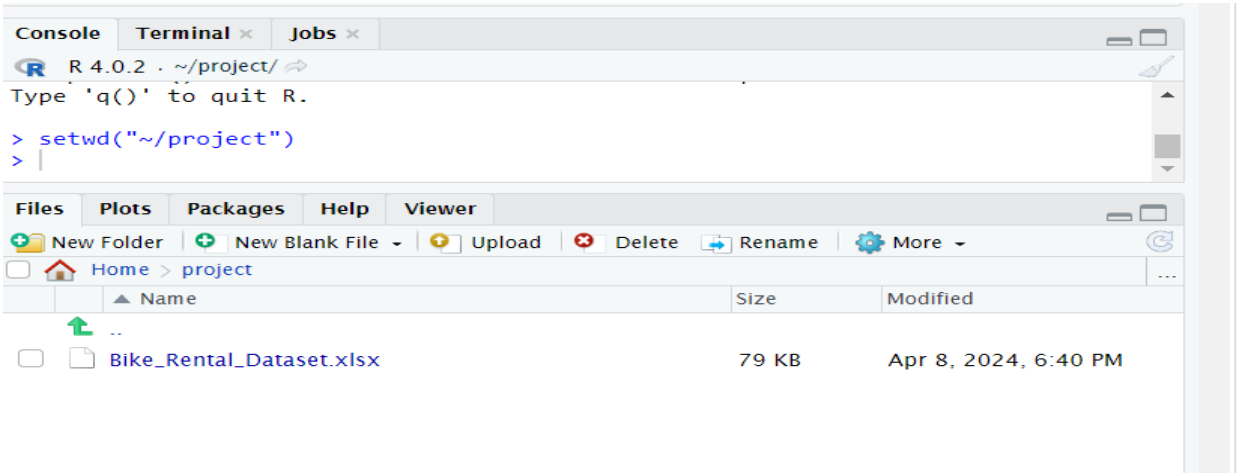
Project Report on Bike_rental_prediction

Step 1:

I. Load the dataset and set the directory

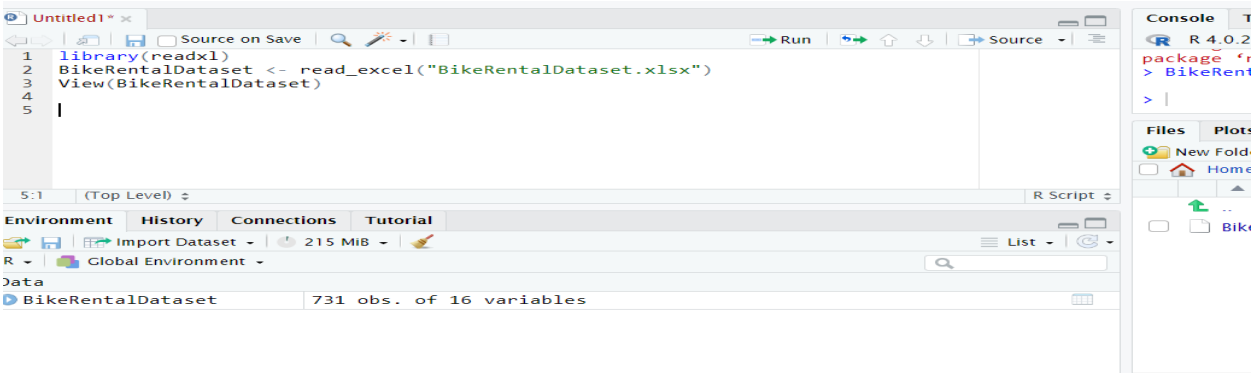


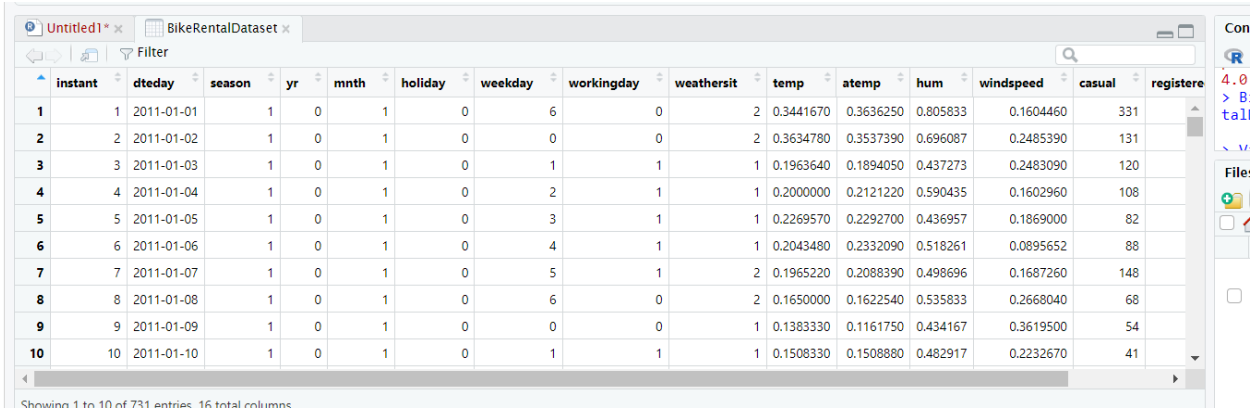
II. Load libraries and view data

library(readxl)

BikeRentalDataset <- read_excel("BikeRentalDataset.xlsx")

View(BikeRentalDataset)

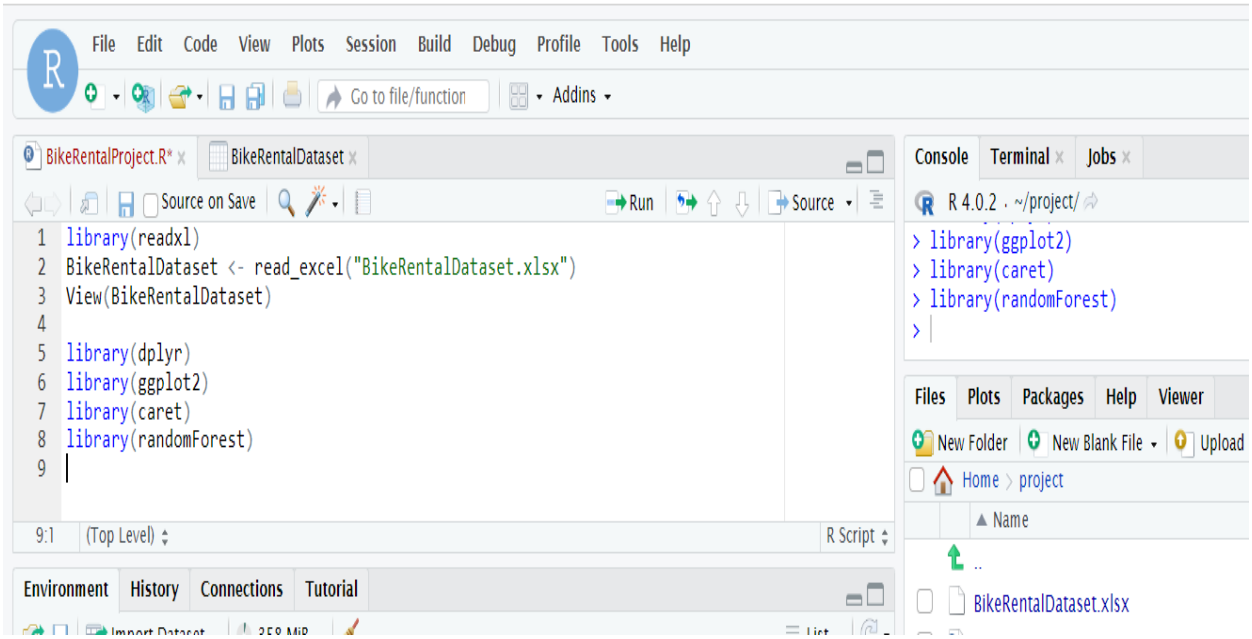




	instant	dteday	season	yr	mnth	holiday	weekday	workingday	weathersit	temp	atemp	hum	windspeed	casual	registere
1	1	2011-01-01	1	0	1	0	6	0	2	0.3441670	0.3636250	0.805833	0.1604460	331	
2	2	2011-01-02	1	0	1	0	0	0	2	0.3634780	0.3537390	0.696087	0.2485390	131	
3	3	2011-01-03	1	0	1	0	1	1	1	0.1963640	0.1894050	0.437273	0.2483090	120	
4	4	2011-01-04	1	0	1	0	2	1	1	0.2000000	0.2121220	0.590435	0.1602960	108	
5	5	2011-01-05	1	0	1	0	3	1	1	0.2269570	0.2292700	0.436957	0.1869000	82	
6	6	2011-01-06	1	0	1	0	4	1	1	0.2043480	0.2332090	0.518261	0.0895652	88	
7	7	2011-01-07	1	0	1	0	5	1	2	0.1965220	0.2088390	0.498696	0.1687260	148	
8	8	2011-01-08	1	0	1	0	6	0	2	0.1650000	0.1622540	0.535833	0.2668040	68	
9	9	2011-01-09	1	0	1	0	0	0	1	0.1383330	0.1161750	0.434167	0.3619500	54	
10	10	2011-01-10	1	0	1	0	1	1	1	0.1508330	0.1508880	0.482917	0.2232670	41	

Showing 1 to 10 of 731 entries, 16 total columns

```
library(dplyr)

library(ggplot2)

library(caret)

library(randomForest)
```



III. Summary and structure of data

summary(BikeRentalDataset)

```
> summary(BikeRentalDataset)
    instant          dteday                       season
 Min.   :  1.0   Min.   :2011-01-01 00:00:00   Min.   :1.000
 1st Qu.:183.5   1st Qu.:2011-07-02 12:00:00   1st Qu.:2.000
 Median :366.0   Median :2012-01-01 00:00:00   Median :3.000
 Mean   :366.0   Mean   :2012-01-01 00:00:00   Mean   :2.497
 3rd Qu.:548.5   3rd Qu.:2012-07-01 12:00:00   3rd Qu.:3.000
 Max.   :731.0   Max.   :2012-12-31 00:00:00   Max.   :4.000
       yr              mnth          holiday           weekday
 Min.   :0.0000   Min.   : 1.00   Min.   :0.00000   Min.   :0.000
 1st Qu.:0.0000   1st Qu.: 4.00   1st Qu.:0.00000   1st Qu.:1.000
 Median :1.0000   Median : 7.00   Median :0.00000   Median :3.000
 Mean   :0.5007   Mean   : 6.52   Mean   :0.02873   Mean   :2.997
 3rd Qu.:1.0000   3rd Qu.:10.00   3rd Qu.:0.00000   3rd Qu.:5.000
 Max.   :1.0000   Max.   :12.00   Max.   :1.00000   Max.   :6.000
   workingday      weathersit         temp             atemp
 Min.   :0.000   Min.   :1.000   Min.   :0.05913   Min.   :0.07907
 1st Qu.:0.000   1st Qu.:1.000   1st Qu.:0.33708   1st Qu.:0.33784
 Median :1.000   Median :1.000   Median :0.49833   Median :0.48673
 Mean   :0.684   Mean   :1.395   Mean   :0.49538   Mean   :0.47435
 3rd Qu.:1.000   3rd Qu.:2.000   3rd Qu.:0.65542   3rd Qu.:0.60860
 Max.   :1.000   Max.   :3.000   Max.   :0.86167   Max.   :0.84090
```

Files | Plots | Packages | Help | Viewer

Tuesday, April 9, 2024

str(BikeRentalDataset)

```
> str(BikeRentalDataset)
tibble [731 × 16] (S3: tbl_df/tbl/data.frame)
 $ instant   : num [1:731] 1 2 3 4 5 6 7 8 9 10 ...
 $ dteday    : POSIXct[1:731], format: "2011-01-01" "2011-01-02" ...
 $ season    : num [1:731] 1 1 1 1 1 1 1 1 1 ...
 $ yr        : num [1:731] 0 0 0 0 0 0 0 0 0 ...
 $ mnth      : num [1:731] 1 1 1 1 1 1 1 1 1 ...
 $ holiday   : num [1:731] 0 0 0 0 0 0 0 0 0 ...
 $ weekday   : num [1:731] 6 0 1 2 3 4 5 6 0 1 ...
 $ workingday: num [1:731] 0 0 1 1 1 1 1 0 0 1 ...
 $ weathersit: num [1:731] 2 2 1 1 1 1 2 2 1 1 ...
 $ temp      : num [1:731] 0.344 0.363 0.196 0.2 0.227 ...
 $ atemp     : num [1:731] 0.364 0.354 0.189 0.212 0.229 ...
 $ hum       : num [1:731] 0.806 0.696 0.437 0.59 0.437 ...
 $ windspeed : num [1:731] 0.16 0.249 0.248 0.16 0.187 ...
 $ casual    : num [1:731] 331 131 120 108 82 88 148 68 54 41 ...
 $ registered: num [1:731] 654 670 1229 1454 1518 ...
 $ cnt       : num [1:731] 985 801 1349 1562 1600 ...
>
```

IV. Data Type conversion of the Attributes

```
BikeRentalDataset_1 <- BikeRentalDataset %>%
  mutate(
    instant = as.integer(instant),
    dteday =as.Date(dteday),
    season =as.factor(season),
    yr =as.factor(yr),
    mnth =as.factor(mnth),
    holiday =as.factor(holiday),
    weekday =as.factor(weekday),
    workingday =as.factor(workingday),
    weathersit =as.factor(weathersit)
  )
str(BikeRentalDataset_1)
```

```
> str(BikeRentalDataset_1)
tibble [731 × 16] (S3: tbl_df/tbl/data.frame)
 $ instant   : int [1:731] 1 2 3 4 5 6 7 8 9 10 ...
 $ dteday    : Date[1:731], format: "2011-01-01" "2011-01-02" ...
 $ season    : Factor w/ 4 levels "1","2","3","4": 1 1 1 1 1 1 1 1 1 1 ...
 $ yr        : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
 $ mnth      : Factor w/ 12 levels "1","2","3","4",..: 1 1 1 1 1 1 1 1 1 1 ...
 $ holiday   : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
 $ weekday   : Factor w/ 7 levels "0","1","2","3",..: 7 1 2 3 4 5 6 7 1 2 ...
 $ workingday: Factor w/ 2 levels "0","1": 1 1 2 2 2 2 2 1 1 2 ...
 $ weathersit: Factor w/ 3 levels "1","2","3": 2 2 1 1 1 1 2 2 1 1 ...
 $ temp      : num [1:731] 0.344 0.363 0.196 0.2 0.227 ...
 $ atemp     : num [1:731] 0.364 0.354 0.189 0.212 0.229 ...
 $ hum       : num [1:731] 0.806 0.696 0.437 0.59 0.437 ...
 $ windspeed : num [1:731] 0.16 0.249 0.248 0.16 0.187 ...
 $ casual    : num [1:731] 331 131 120 108 82 88 148 68 54 41 ...
 $ registered: num [1:731] 654 670 1229 1454 1518 ...
 $ cnt       : num [1:731] 985 801 1349 1562 1600 ...
>
```

V. Missing Value Analysis

```
missing_values <- BikeRentalDataset_1%>%

 summarise_all(~sum(is.na(.)))

print(missing_values)
```

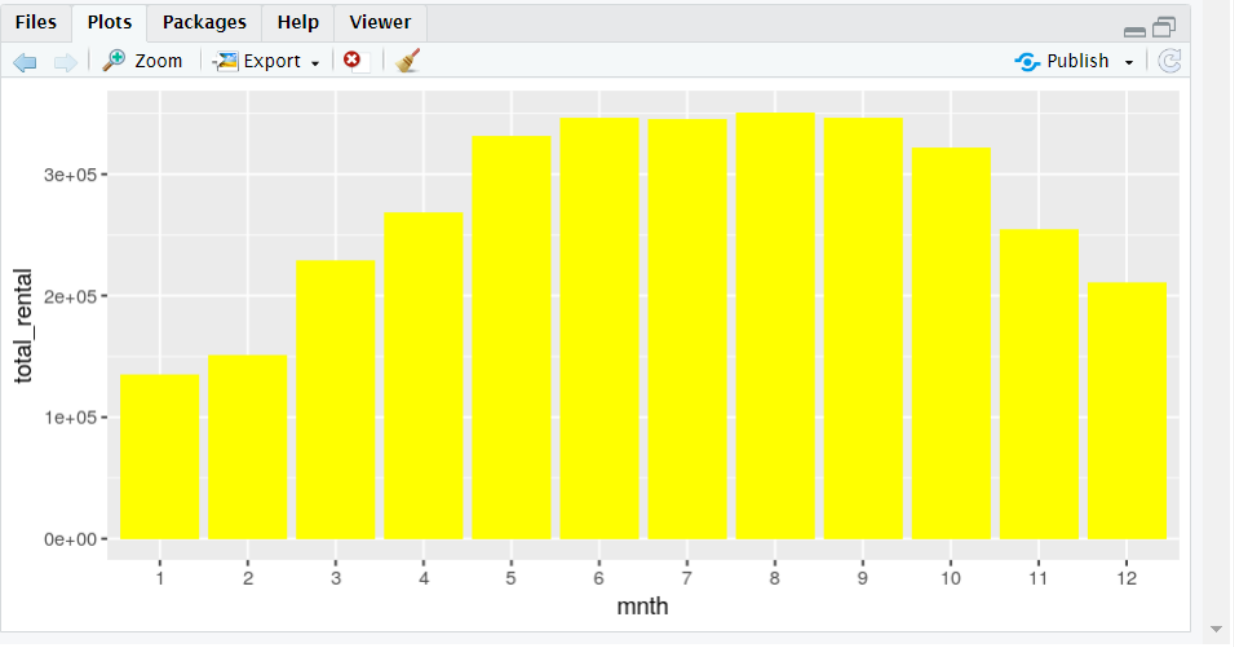
```
missing_values <- BikeRentalDataset_1%>%
  summarise_all(~sum(is.na(.)))
print(missing_values)
A tibble: 1 × 16
instant dteday season    yr  mnth holiday weekday workingday weathersit
  <int>  <int>  <int> <int> <int>   <int>   <int>      <int>      <int>
      0      0      0     0     0       0       0          0          0
… with 7 more variables: temp <int>, atemp <int>, hum <int>,
  windspeed <int>, casual <int>, registered <int>, cnt <int>
```

Step 2: Attributes Distribution and Trends

I. Plot Month wise Distribution by using HISTOGRAM

```
monthly_rental <- BikeRentalDataset_1 %>%

 group_by(mnth) %>%

 summarise(total_rental = sum(cnt))


ggplot(monthly_rental,

      aes(x = mnth ,

        y = total_rental)) +
```

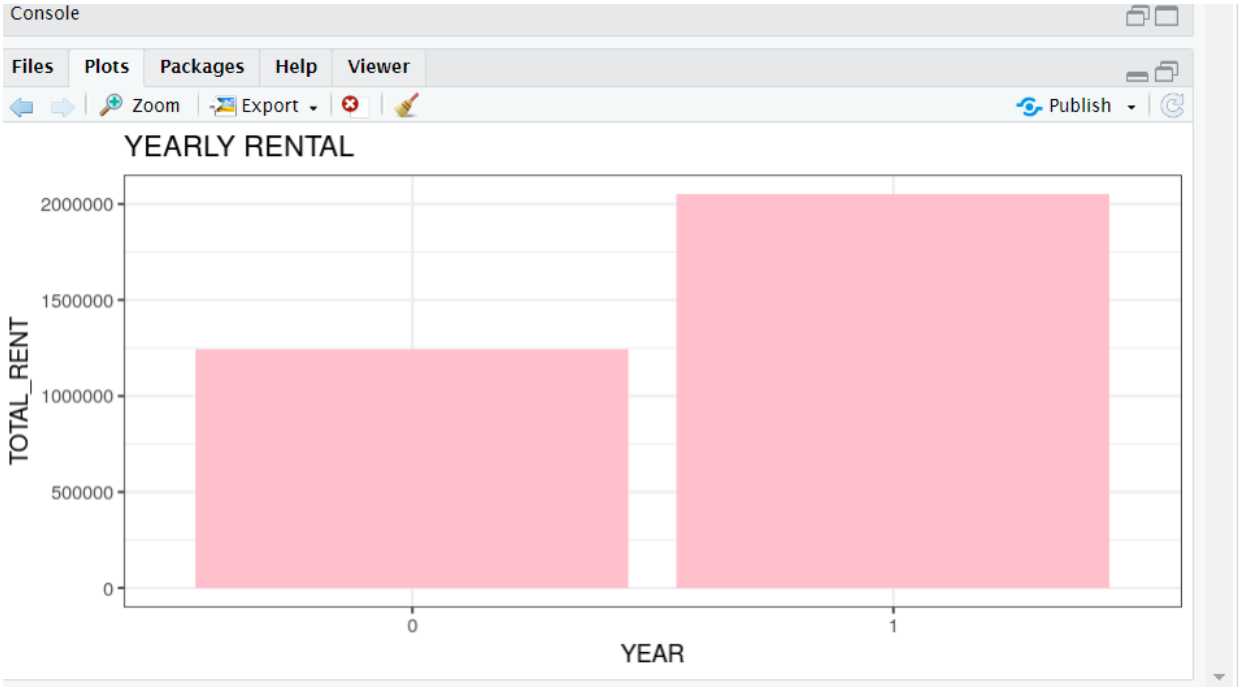
```
  geom_col() +
  geom_bar(stat = "identity", fill = "yellow")
  theme_bw()+
  labs(x = "MONTH", y = "TOTAL_RENT", title ="MONTHLY RENTAL")
```



II. Plot Year wise Distribution by using HISTOGRAM

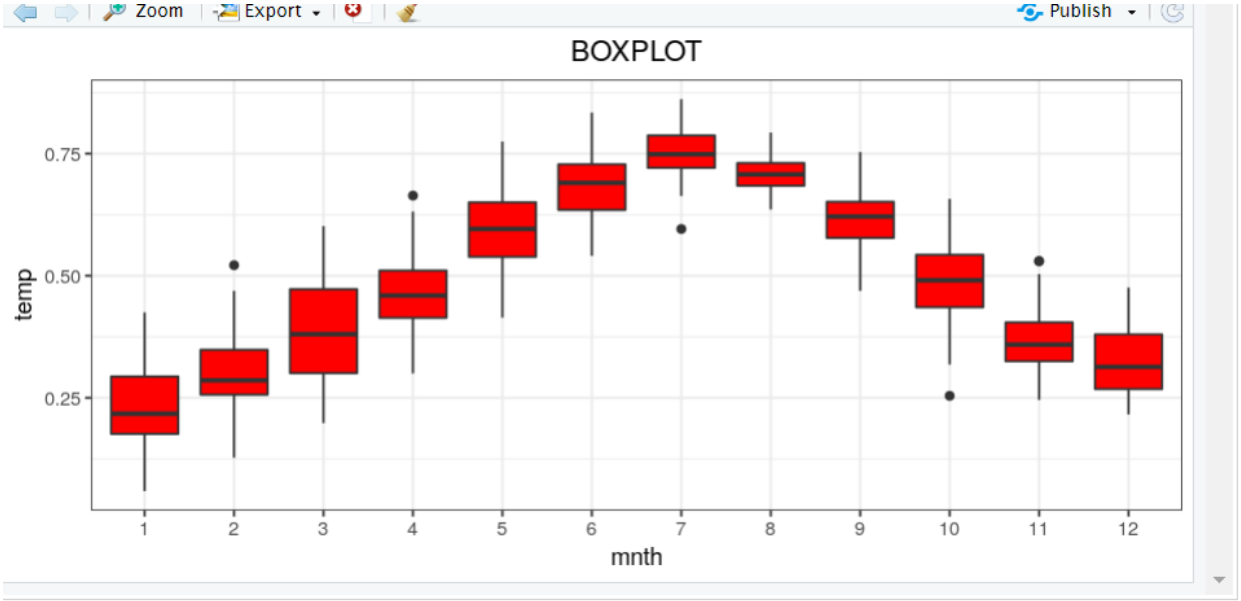
```
BikeRentalDataset_1 <- BikeRentalDataset_1 %>%
  mutate(yr =as.numeric(yr))
yearly_rental <- BikeRentalDataset_1 %>%
  group_by(yr) %>%
  summarise(total_rental = sum(cnt))
```

```
ggplot(yearly_rental,
       aes(x = yr ,
           y = total_rental)) +
 geom_col() +
 geom_bar(stat = "identity", fill = "PINK") +
 theme_bw()+
 labs(x = "YEAR", y = "TOTAL_RENT", title ="YEARLY RENTAL")
```



III. Plot Temperature wise Distribution by using BOXPLOT

```
ggplot(BikeRentalDataset_1,
       aes(x = mnth ,
           y = temp)) +
  geom_boxplot(fill = "red") +
  theme_bw()+
  labs(title = "BOXPLOT")+
  theme(plot.title = element_text(hjust = 0.5))
```



Step 3: Split the dataset into train and test dataset

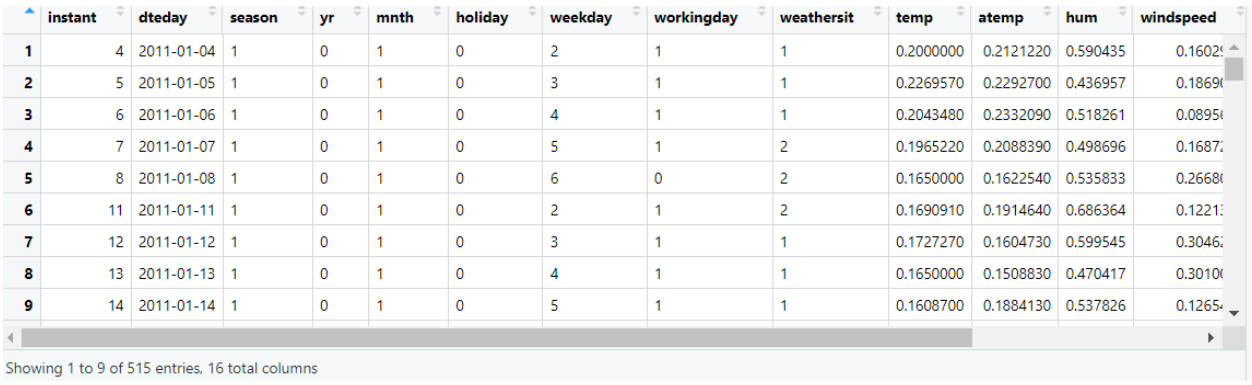
```
set.seed(123)
trainIndex <- createDataPartition(BikeRentalDataset_1$cnt, p = 0.7, list = FALSE)
View(trainIndex)
```

Filter

	Resample1
1	4
2	5
3	6
4	7
5	8
6	11
7	12
8	13
9	14
10	16

Showing 1 to 10 of 515 entries, 1 total columns

```
training_data <- BikeRentalDataset_1[trainIndex, ]
View(training_data)
```

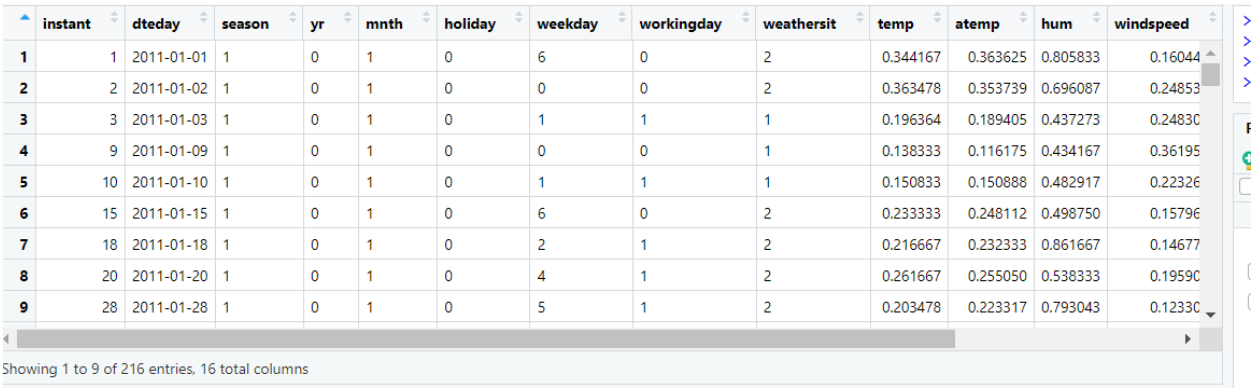


	instant	dteday	season	yr	mnth	holiday	weekday	workingday	weathersit	temp	atemp	hum	windspeed
1	4	2011-01-04	1	0	1	0	2	1	1	0.2000000	0.2121220	0.590435	0.1602
2	5	2011-01-05	1	0	1	0	3	1	1	0.2269570	0.2292700	0.436957	0.1869
3	6	2011-01-06	1	0	1	0	4	1	1	0.2043480	0.2332090	0.518261	0.0895
4	7	2011-01-07	1	0	1	0	5	1	2	0.1965220	0.2088390	0.498696	0.1687
5	8	2011-01-08	1	0	1	0	6	0	2	0.1650000	0.1622540	0.535833	0.2668
6	11	2011-01-11	1	0	1	0	2	1	2	0.1690910	0.1914640	0.686364	0.1221
7	12	2011-01-12	1	0	1	0	3	1	1	0.1727270	0.1604730	0.599545	0.3046
8	13	2011-01-13	1	0	1	0	4	1	1	0.1650000	0.1508830	0.470417	0.3010
9	14	2011-01-14	1	0	1	0	5	1	1	0.1608700	0.1884130	0.537826	0.1265

Showing 1 to 9 of 515 entries, 16 total columns

test_data <- BikeRentalDataset_1[-trainIndex,]

View(test_data)



	instant	dteday	season	yr	mnth	holiday	weekday	workingday	weathersit	temp	atemp	hum	windspeed
1	1	2011-01-01	1	0	1	0	6	0	2	0.344167	0.363625	0.805833	0.16044
2	2	2011-01-02	1	0	1	0	0	0	2	0.363478	0.353739	0.696087	0.24853
3	3	2011-01-03	1	0	1	0	1	1	1	0.196364	0.189405	0.437273	0.24830
4	9	2011-01-09	1	0	1	0	0	0	1	0.138333	0.116175	0.434167	0.36195
5	10	2011-01-10	1	0	1	0	1	1	1	0.150833	0.150888	0.482917	0.22326
6	15	2011-01-15	1	0	1	0	6	0	2	0.233333	0.248112	0.498750	0.15796
7	18	2011-01-18	1	0	1	0	2	1	2	0.216667	0.232333	0.861667	0.14677
8	20	2011-01-20	1	0	1	0	4	1	2	0.261667	0.255050	0.538333	0.19590
9	28	2011-01-28	1	0	1	0	5	1	2	0.203478	0.223317	0.793043	0.12330

Showing 1 to 9 of 216 entries, 16 total columns

Step 4: Create a model using the random forest Algorithm

model <- randomForest(cnt ~ season + yr + mnth + holiday + weekday + workingday + weathersit + temp + atemp + hum + windspeed + casual + registered, data = training_data)

predictions <- predict(model, newdata = test_data)

View(predictions)

Name	Type	Value
predictions	double [216]	1598 1400 1456 1061 1399 1366 ...
1	double [1]	1597.834
2	double [1]	1400.486
3	double [1]	1456.264
4	double [1]	1060.514
5	double [1]	1398.685
6	double [1]	1365.613
7	double [1]	865.658
8	double [1]	1926.18
9	double [1]	1208.544
10	double [1]	1143.21

(No selection)

predictions

```
> predictions
       1        2        3        4        5        6        7
1615.0394 1405.8665 1467.3709 1076.5314 1409.5622 1360.1482  864.3019
       8        9       10       11       12       13       14
1903.5969 1219.4416 1135.0881 1601.4956 2100.4937 2145.4801 2688.1272
      15       16       17       18       19       20       21
3301.3712 1701.0438 1932.4955 2367.1567 1625.1522 2183.7540 1312.6653
      22       23       24       25       26       27       28
1947.9573 1961.1580 2089.6662 2234.1413 1858.3878 1811.7327 2996.6828
      29       30       31       32       33       34       35
2266.7899 1388.5919 4150.8096 4013.1074 4521.1957 2817.5263 4304.6537
      36       37       38       39       40       41       42
4673.5548 4402.4259 4463.0306 4331.0652 4677.4214 4982.8045 4882.1591
      43       44       45       46       47       48       49
4417.3820 4895.2747 4245.7125 4866.7199 5192.1937 5232.9629 5499.7171
      50       51       52       53       54       55       56
4963.4490 4613.0081 4766.6587 4482.8065 4455.9924 3872.9165 3823.5578
      57       58       59       60       61       62       63
4053.7170 4392.1670 4592.6779 4738.0062 4825.1771 4680.1353 4410.4764
      64       65       66       67       68       69       70
5167.8533 4740.8508 5016.9740 3679.4750 2510.0217 4698.5826 3677.0546
      71       72       73       74       75       76       77
2261.3638 3838.1345 4632.9722 4555.0557 3808.2527 4491.0424 5282.4120
```

Files | Plots | Packages | Help | Viewer

model

```
> model

Call:
 randomForest(formula = cnt ~ season + yr + mnth + holiday + weekday +      workin
gday + weathersit + temp + atemp + hum + windspeed +      casual + registered, dat
a = training_data)
               Type of random forest: regression
                     Number of trees: 500
No. of variables tried at each split: 4

          Mean of squared residuals: 77521.91
                    % Var explained: 97.93
>
```

Step 5:Predict the performance of the dataset

I. Calculate Root Mean Squared Error (RMSE)

test_predictions <- predict(model, newdata = test_data)

rmse <- sqrt(mean((test_data$cnt - test_predictions)^2))

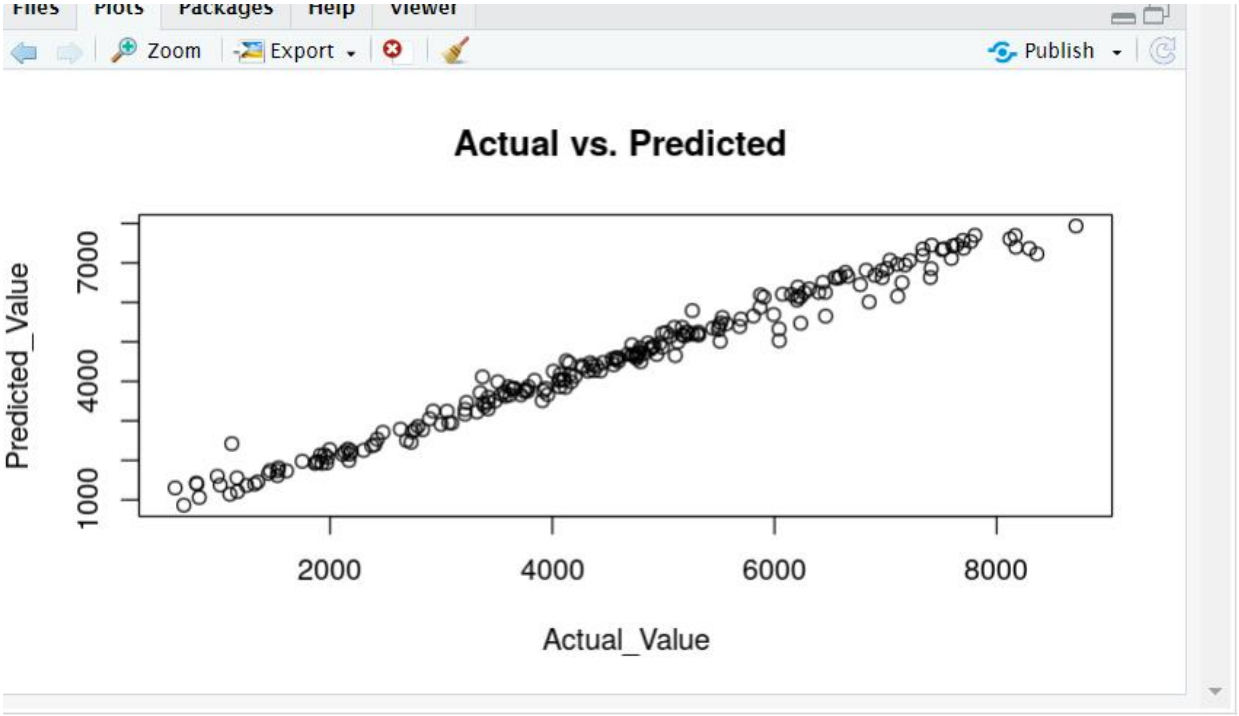
rmse

cat("Root Mean Squared Error (RMSE):", rmse, "\n")

```
> rmse
[1] 299.3603
> cat("Root Mean Squared Error (RMSE):", rmse, "\n")
Root Mean Squared Error (RMSE): 299.3603
>
```

II. Calculate Actual vs. Predicted Value by using Scatterplot

plot(test_data$cnt, test_predictions,xlab = "Actual_Value", ylab = "Predicted_Value", main = "Actual vs. Predicted ")



```
plot(test_data$cnt, test_predictions,xlab = "Actual_Value", ylab =
"Predicted_Value", main = "Actual vs. Predicted ")

abline(0,1, col = "yellow")
```