Prediction of Bike Rental Count

Debayan Chakraborty

21st September 2019

Contents

Chapter 1

Introduction

1.1 Problem Statement

Predicting the accurate number of bikes is a challenge for bike rental companies as keeping less bikes availability will result in revenue loss and keeping more bikes than required will adversely affect the cost of operation. The aim of the project is to predict the bike rental count daily based on the environmental and seasonal settings. This will help the bike rental company accommodate the number of bikes required daily and anticipate peak demand periods.

1.2 Data

Our task is to build regression models (as the target variable is continuous) which will predict the bike rental count daily based on the environmental and seasonal settings. Given below is a sample of the data set that we are using to predict the quality of wine:

Table 1.1: "day.csv" Sample Data (Columns: 1-9)

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

Table 1.2: "day.csv" Sample Data (Columns: 10-16)

temp	atemp	hum	windspeed	casual	registered	cnt
0.3441670	0.3636250	0.805833	0.1604460	331	654	985
0.3634780	0.3537390	0.696087	0.2485390	131	670	801
0.1963640	0.1894050	0.437273	0.2483090	120	1229	1349
0.2000000	0.2121220	0.590435	0.1602960	108	1454	1562

The details of variables in the dataset are mentioned as follows:

The dataset contains continuous as well as categorical variables.

a) **instant**: Serial no. of the dataset

b) **season** :(1 corresponds to spring, 2 corresponds to summer, 3 corresponds to fall, 4 corresponds to winter)

c) **yr** corresponds to Year

d) **mnth**: Months (1 to 12)

e) **holiday**: weather day is holiday or not

f) **weekday**: Day of the week

g) **workingday:** If day is neither weekend nor holiday is 1, otherwise is 0

h) **weathersit**: 4 types of weather situation are depicted in the data viz. 1 to 4

i) **temp:** Temperature in Celsius

j) **atemp:** Actual (feel temperature) source: 'Accuweather'

k) **hum:** Humidity

l) **windspeed:** speed of wind in a particular day

m) **casual:** count of casual users

n) **registered:** count of registered users

o) **count:** Count of total number of bike rentals

```
> str(project_data)
'data.frame':   731 obs. of  16 variables:
 $ instant   : int  1 2 3 4 5 6 7 8 9 10 ...
 $ dteday    : Factor w/ 731 levels "2011-01-01","2011-01-02",..: 1 2 3 4 5 6 7 8
9 10 ...
 $ season    : int  1 1 1 1 1 1 1 1 1 1 ...
 $ yr        : int  0 0 0 0 0 0 0 0 0 0 ...
 $ mnth      : int  1 1 1 1 1 1 1 1 1 1 ...
 $ holiday   : int  0 0 0 0 0 0 0 0 0 0 ...
 $ weekday   : int  6 0 1 2 3 4 5 6 0 1 ...
 $ workingday: int  0 0 1 1 1 1 1 0 0 1 ...
 $ weathersit: int  2 2 1 1 1 1 2 2 1 1 ...
 $ temp      : num  0.344 0.363 0.196 0.2 0.227 ...
 $ atemp     : num  0.364 0.354 0.189 0.212 0.229 ...
 $ hum       : num  0.806 0.696 0.437 0.59 0.437 ...
 $ windspeed : num  0.16 0.249 0.248 0.16 0.187 ...
 $ casual    : int  331 131 120 108 82 88 148 68 54 41 ...
 $ registered: int  654 670 1229 1454 1518 1518 1362 891 768 1280 ...
 $ cnt       : int  985 801 1349 1562 1600 1606 1510 959 822 1321 ...
```

Chapter 2

Methodology

2.1 Pre-Processing

Any predictive modeling requires that we look at the data before we start modelling. 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 convert the data variables into required datatypes and visualize the data of continuous variables using histogram and categorical variables using bar charts. Next feature engineering operations will include steps viz. missing value analysis, outlier analysis and correlation analysis.

2.1.1 Data type conversion

The data variables in the dataset need to be converted to desired variable type so as to successfully carry out statistical analysis which are very much dependent on the variable type.

The data types are successfully converted using proper feature engineering techniques and the structure of the data set post data type conversion can be seen below.

```
> str(project_data)
'data.frame':	731 obs. of  16 variables:
 $ instant   : int  1 2 3 4 5 6 7 8 9 10 ...
 $ dteday    : chr  "2011-01-01" "2011-01-02" "2011-01-03" "2011-01-04" ...
 $ 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  0.344 0.363 0.196 0.2 0.227 ...
 $ atemp     : num  0.364 0.354 0.189 0.212 0.229 ...
 $ hum       : num  0.806 0.696 0.437 0.59 0.437 ...
 $ windspeed : num  0.16 0.249 0.248 0.16 0.187 ...
 $ casual    : num  331 131 120 108 82 88 148 68 54 41 ...
 $ registered: num  654 670 1229 1454 1518 ...
 $ cnt       : int  985 801 1349 1562 1600 1606 1510 959 822 1321 ...
> 
```

©Debayan Chakraborty

2.1.2 Data Visualization

a) Histogram of continuous variables:

In fig. 1.1 we have plotted the data distribution of *continuous variables* related to weather using histogram.

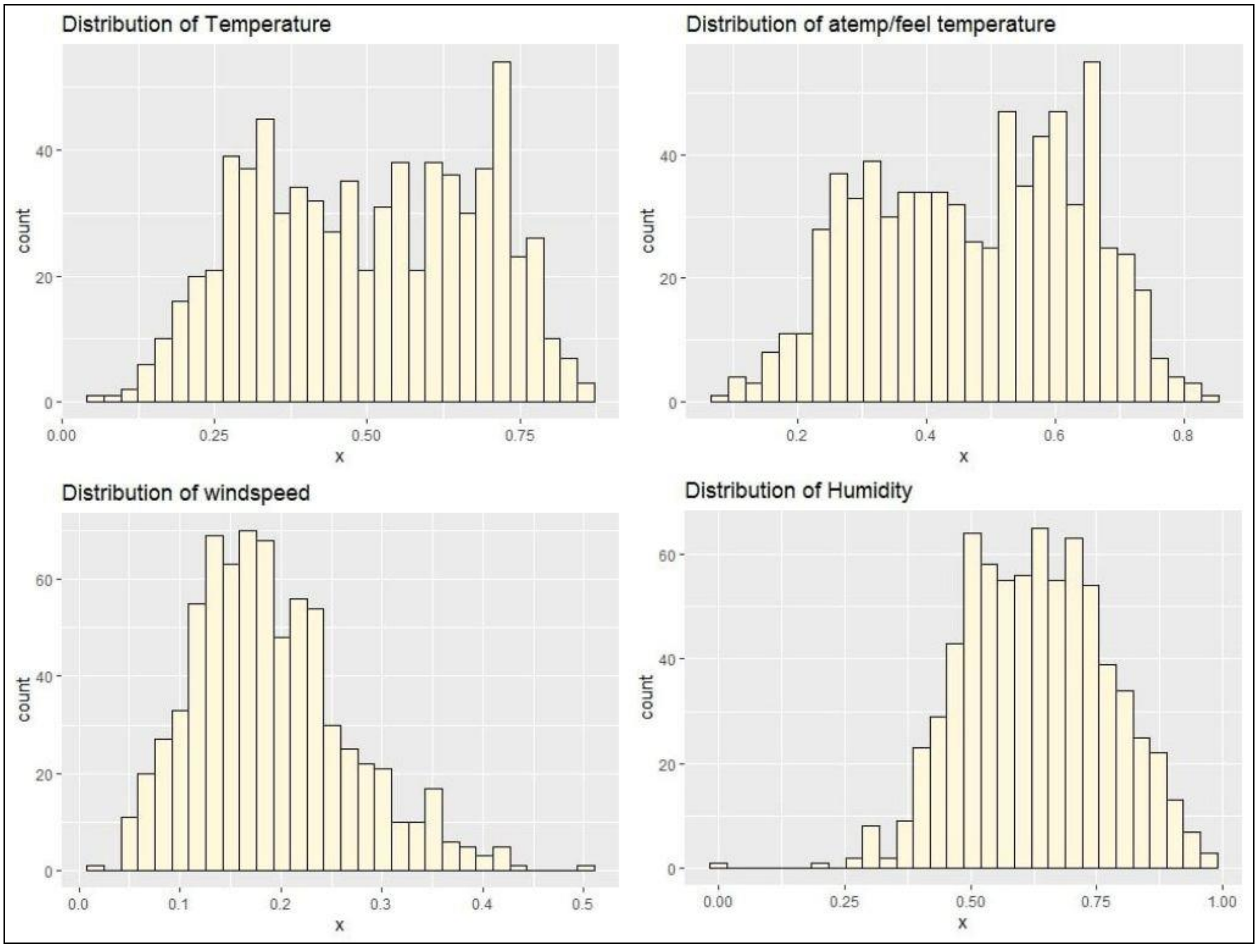
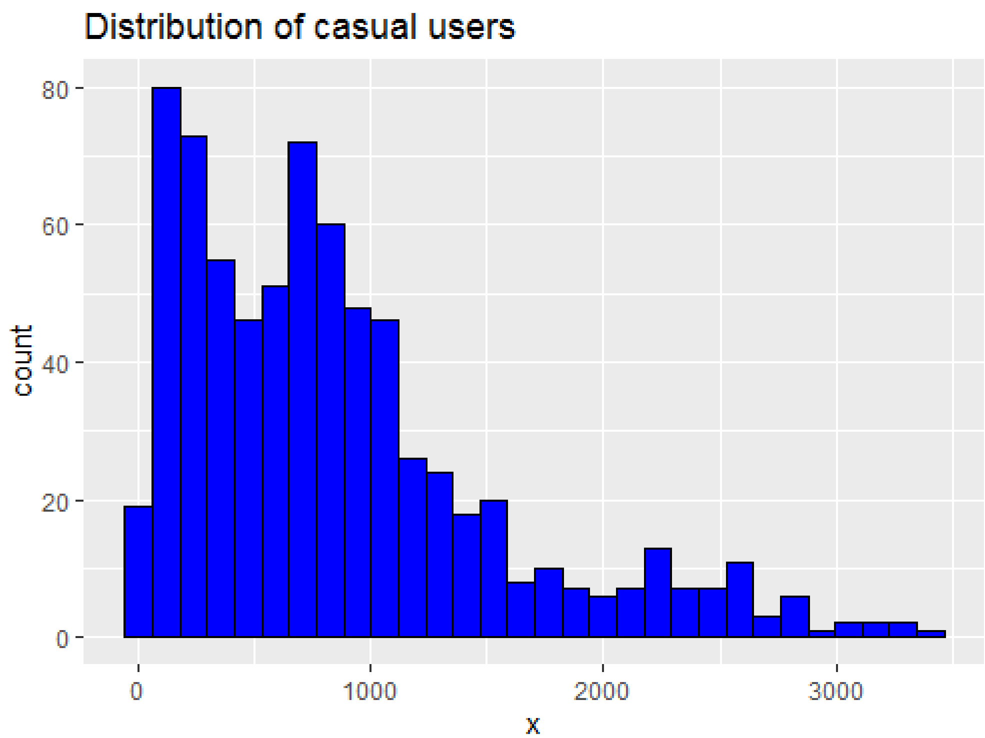


Fig. 1.1

In fig.1.2 we have plotted the data distribution of *continuous variables* related to user type using histogram.



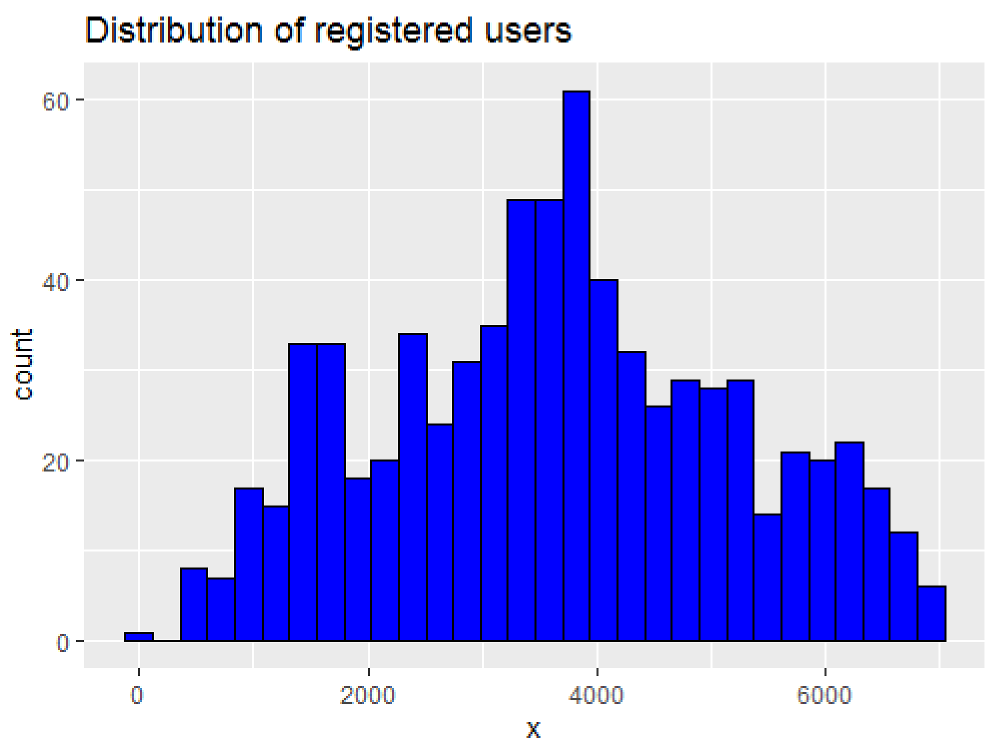


Fig. 1.2

b) Bar charts of categorical variables:

In fig.1.4 we have plotted the data distribution of *categorical variables* related to user type using bar graph.

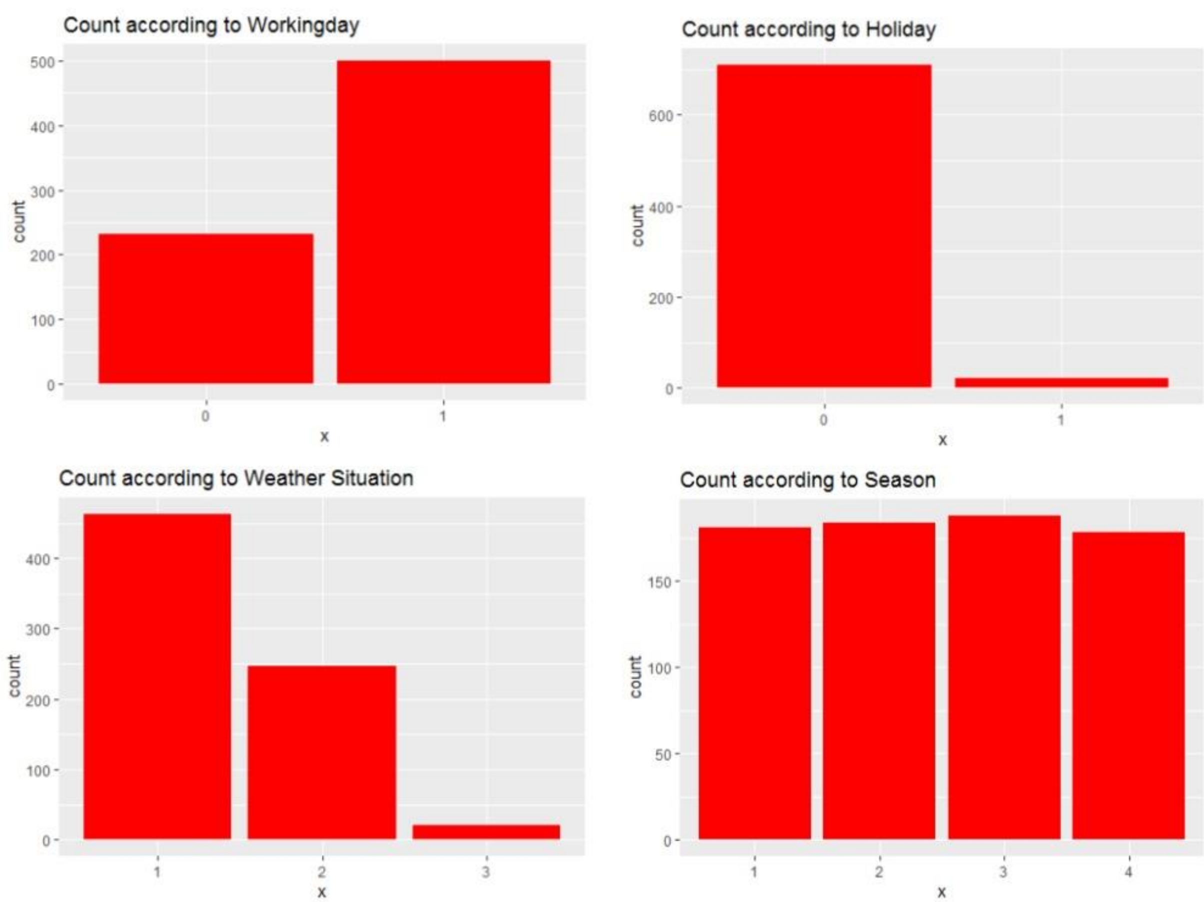


Fig. 1.3

Interpretation:

- It can be inferred from the data distribution of continuous variables related to climate conditions, viz. ‘temp’ (temperature) and ‘atemp’(real feel temperature) are almost normally distributed.
- Skewness in the data distribution has been observed in ‘windspeed’ and humidity. One of the possible reasons for this skewness might be due to the presence of extreme data points and outliers in the data set.
- Data distribution of the continuous variables related to user type informs about the normal distribution of data in ‘registered’ user category, whereas skewness towards LHS can been noticed for ‘casual user category indicating negatively skewed data.
- From the bar charts of categorical variables, it can be perceived that, variables like ‘weathersit’ (weather situation), day of the week viz. holiday, working day/weekend have a huge impact on the target variable and plays a crucial role in deciding the count of bikes.

2.1.3 Outlier Analysis

We can clearly observe from these probability distributions that some variables are skewed, for example, *windspeed*, *humidity* and *casual users*. The skew in these distributions can be most likely explained by the presence of outliers and extreme values in the data.

One of the other steps of **pre-processing** is the analysis of the outliers in the using *boxplots*.

In figure 1.5 we have plotted the boxplots of the four continuous variables with respect to each quality value ranging from 3 to 8. A lot

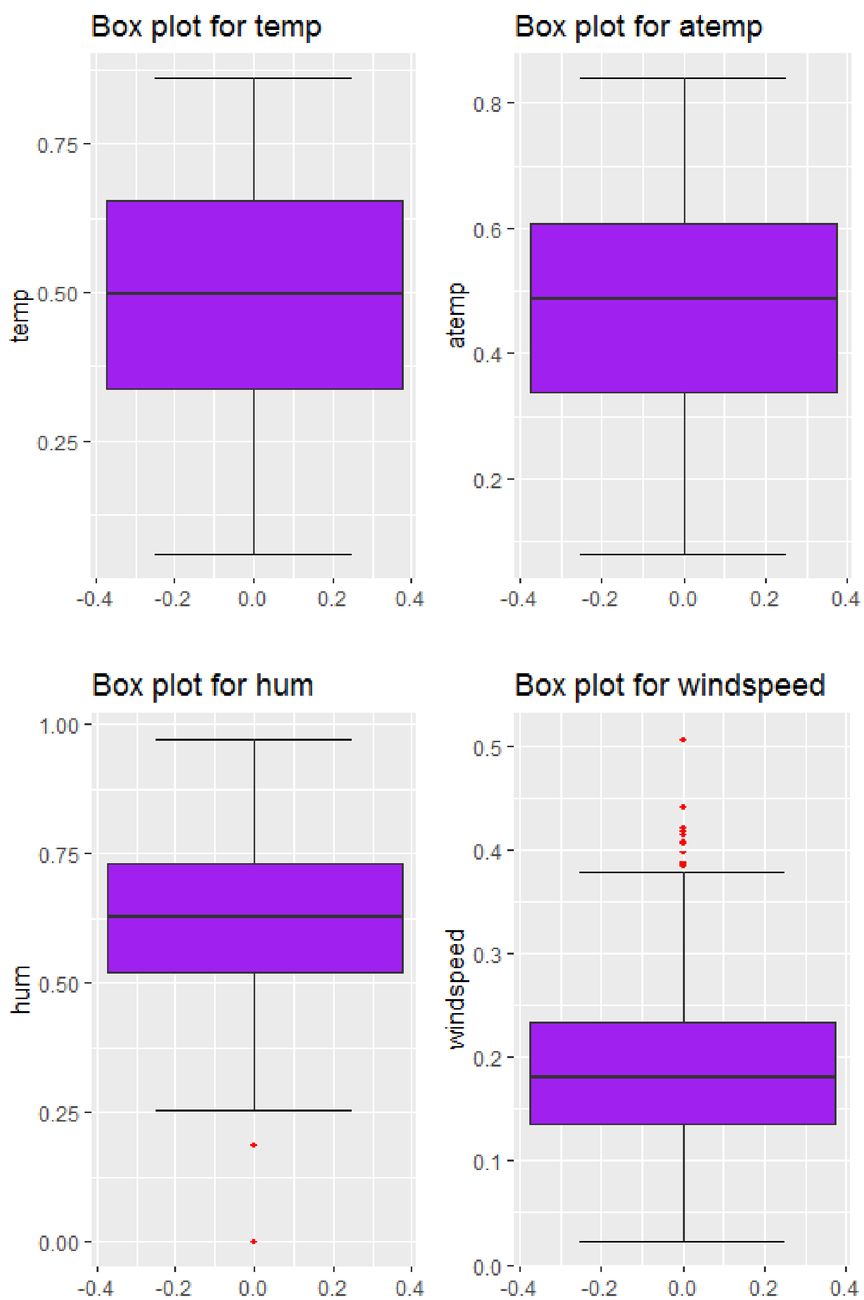


Fig. 1.4

Interpretation:

- Boxplots has been plotted for continuous variables and outliers has been detected mostly in windspeed as depicted in red colour.
- We have calculated the inter quartile range (IQR) after getting maximum and minimum of the variables. Values ranging outside the maximum and minimum are discarded.

Removal of outliers:

We have got the modified dataset as shown below, after removing outliers.

```
project_data    717 obs. of 16 variables
```

2.1.4 Missing value analysis

In the R code we have performed one missing value analysis before and one analysis after the outlier analysis as sometimes missing values are generated after removal of outliers, but in this dataset we haven't found any missing values.

Refer to the below output from R, where no missing values were found against the variables in the dataset.

```
   instant     dteday     season         yr       mnth    holiday    weekday 
         0          0          0          0          0          0          0 
workingday weathersit       temp      atemp        hum  windspeed     casual 
         0          0          0          0          0          0          0 
registered        cnt 
         0          0 
```

2.1.5 Feature Selection

i) Correlation analysis

Before performing any type of modelling, 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 and we need to select only those selective variables which will have very high correlation with the target variable and thereby contribute to predict the dependent variable. Moreover, it will help us detect and eliminate the variables that can cause multicollinearity problem. In this step we drop certain variables which contains the same information and can increase the complexity of the of the model on which we are going to predict the target variable, hence performing feature selection will help us to remove irrelevant features from the dataset.

We have plotted the below correlation plot of continuous variables to determine the possible multicollinearity between variables.

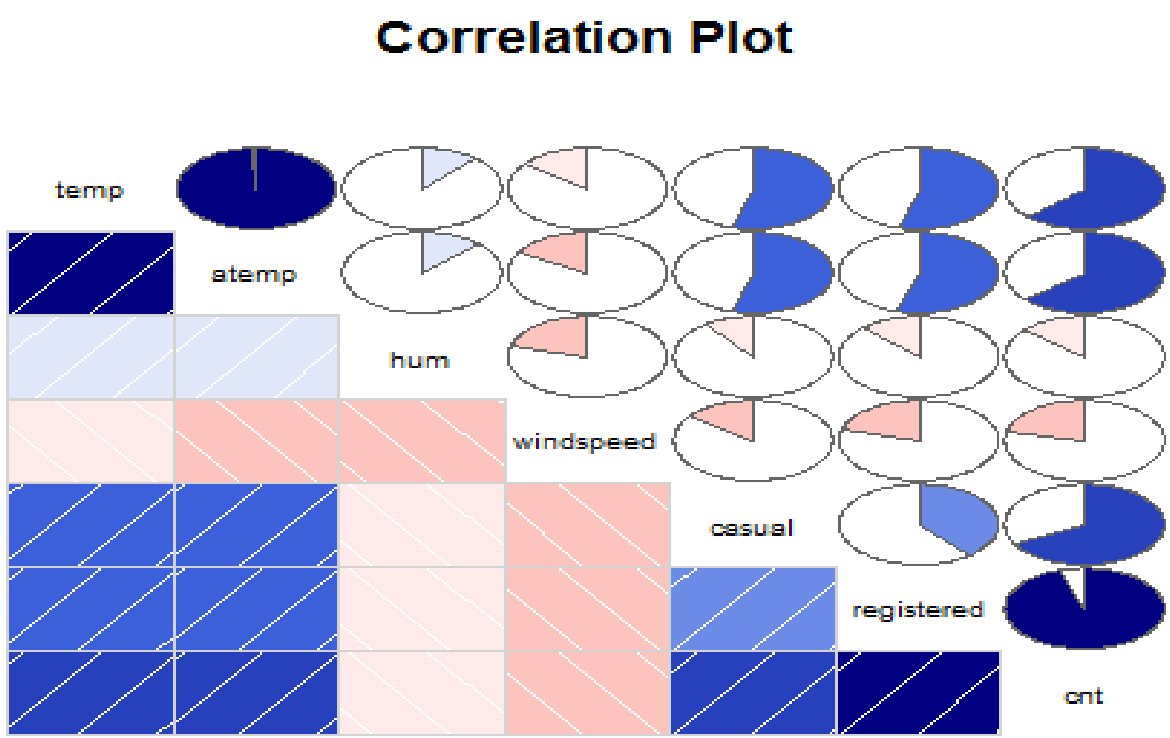


Fig. 1.5

Interpretation:

- Here we have dropped casual and registered as both the variables are resonating the same correlation as shown by the dependent variable with other independent variables.
- Here we are also dropping 'temp' as it is highly correlated with 'atemp', which will cause multicollinearity problem, if not removed.

ii) Chi square test

In this step we have performed chi square test on two set of categorical data where they are tested for collinearity in a contingency table.

While running the test between two categorical variables, we have performed the hypothesis testing, stating the Null hypothesis as two variables are independent i.e. p>0.05 and stating the alternate hypothesis as two variables are not independent i.e. p<0.05.

Hence, if p>0.05, variable is kept and if p<0.05, variable is rejected.

Variables which are highly dependent on each other based on p-values are:

```
[1] "mnth"
[1] "weathersit"
[1] 0.009828299
[1] "season"
[1] "weathersit"
[1] 0.01324758
[1] "holiday"
[1] "weekday"
[1] 5.970403e-11
[1] "mnth"
[1] "season"
[1] 0
[1] "holiday"
[1] "workingday"
[1] 3.657543e-11
```

After analysing p value of all categorical variables, we conclude that, categorical variables viz. *holiday, mnth & season* need to be removed and we shall keep rest of the categorical variables.

Hence, we decided to drop the following categorical as well as numerical variables from our dataset

i) holiday ii) mnth iii) season iv) registered v) casual vi) atemp vii) instant viii) dteday

The total number of variables comes down to 8 post our dimension reduction operation.

```
Global Environment
project_data    717 obs. of 8 variables
```

We have also checked for multicollinearity of continuous data post dimension using VIF method & variable inflation factor was found to be within desired limit and hence any chance of multicollinearity is ruled out here.

```
> vif(project_data[,-8])
   Variables      VIF
1         yr       NA
2    weekday       NA
3 workingday       NA
4 weathersit       NA
5       temp 1.099206
6        hum 1.968452
7  windspeed 1.137032
```

Chapter 3

Modelling

3.1 Model Selection & Evaluation

Since our target variable is continuous and we want to know to know how well the model predicts the new data and we will select the following regression models:

1. Multiple linear regression

2. Decision Tree

3. Random Forest

3.1.1 Multiple linear regression

Multiple linear regression is used to explain the relationship between one dependent carriable with multiple independent variables. Our dependent variable here is a continuous one.

The summary of the multiple linear regression model is given below:

```
lm(formula = cnt ~ ., data = train)

Residuals:
    Min      1Q  Median      3Q     Max 
-3354.9  -638.1     5.9   720.1  2433.5 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  1555.27     321.49   4.838 1.70e-06 ***
yr1          1984.08      83.69  23.708  < 2e-16 ***
weekday1     -365.11     280.87  -1.300  0.19417    
weekday2     -210.07     305.30  -0.688  0.49168    
weekday3     -276.04     306.73  -0.900  0.36854    
weekday4     -156.52     306.02  -0.511  0.60923    
weekday5     -137.85     303.62  -0.454  0.64999    
weekday6      425.97     153.97   2.767  0.00585 ** 
workingday1   530.97     267.78   1.983  0.04787 *  
weathersit2  -518.74     111.91  -4.635 4.44e-06 ***
weathersit3 -1838.30     305.13  -6.025 3.08e-09 ***
temp         5802.24     234.85  24.706  < 2e-16 ***
hum          -564.04     415.00  -1.359  0.17465    
windspeed   -3011.15     615.72  -4.890 1.32e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 987.4 on 559 degrees of freedom
Multiple R-squared:  0.7364,	Adjusted R-squared:  0.7302 
F-statistic: 120.1 on 13 and 559 DF,  p-value: < 2.2e-16
```

From the summary of the above model we can draw the following inferences:

- Value of Adjusted R square is 73%, which means we can explain 73% of the data by linear regression model.
- p value is less than 0.05 which again confirms that all the predictors have influence over target variable

Model Evaluation:

We have calculated the following error metrices, where we find the MAPE and RMSE to be error metrices for the evaluation of the model.

```
         mae          mse         rmse         mape
7.459957e+02 8.108159e+05 9.004531e+02 2.618554e-01
```

Mean of absolute percentage error for this model is 26% and root mean square of errors is 900

3.1.2 Decision Tree

Now we will try and use another regression model known as decision tree to predict the count of bike rentals on a certain day.

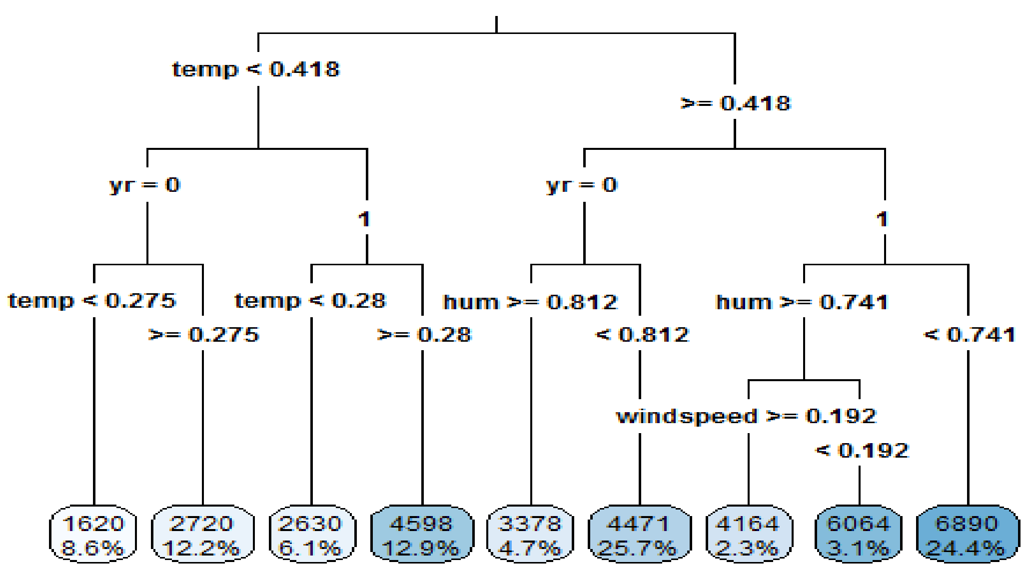


Fig. 1.6

Model Evaluation:

We have calculated the following error metrices, where we find the MAE and RMSE to be error metrices for the evaluation of the model.

```
> regr.eval(trues = test[,8], dtreepreds, stats = c("mae","mse","rmse","mape"))
         mae          mse         rmse         mape 
7.962755e+02 1.034893e+06 1.017297e+03 3.133509e-01
```

Mean of absolute percentage error for this model is 31.3% and root mean square of errors is 1017

3.1.3 Random Forest

Now we will try our final model for regression called random forest. Random forest is an ensemble that consist of many decision trees. In our model we have considered 500 decision trees for prediction in the forest.

Model Evaluation:

We have calculated the following error metrices, where we find the MAPE and RMSE to be error metrices for the evaluation of the model.

```
> regr.eval(trues = test[,8],rforestpreds, stats = c("mae", "mse", "rmse","mape"))
         mae          mse         rmse         mape 
7.031364e+02 7.692421e+05 8.770645e+02 2.876393e-01
```

Mean of absolute percentage error for this model is 28.7% and root mean square of errors is 877.

Chapter 4

Conclusion

Now that we have run three different regression 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 case of prediction of bike count dataset, 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.

In section 3.1 we have evaluated all the three models that were selected for prediction of bike counts. Out of the four error metrices we have selected MAPE and RMSE for model evaluation and selection. As it is a time series data our we will give more importance to the RMSE value.

4.1 MAPE

MAPE is one of the error measures used to calculate the predictive performance of the model.

MAPE calculated for three different models:

Linear Regression Model = 26%
Decision Tree = 31.3%
Random Forest = 28%

4.2 RMSE

MAE is one of the error measures used to calculate the predictive performance of the model.

RMSE calculated for three different models:

Linear Regression Model = 900
Decision Tree = 1017
Random Forest = 877

Based on the above error metrics, Decision tree is the better model for our analysis. Hence Decision tree is chosen as the model for prediction of bike rental count.

APPENDIX A – FIGURES

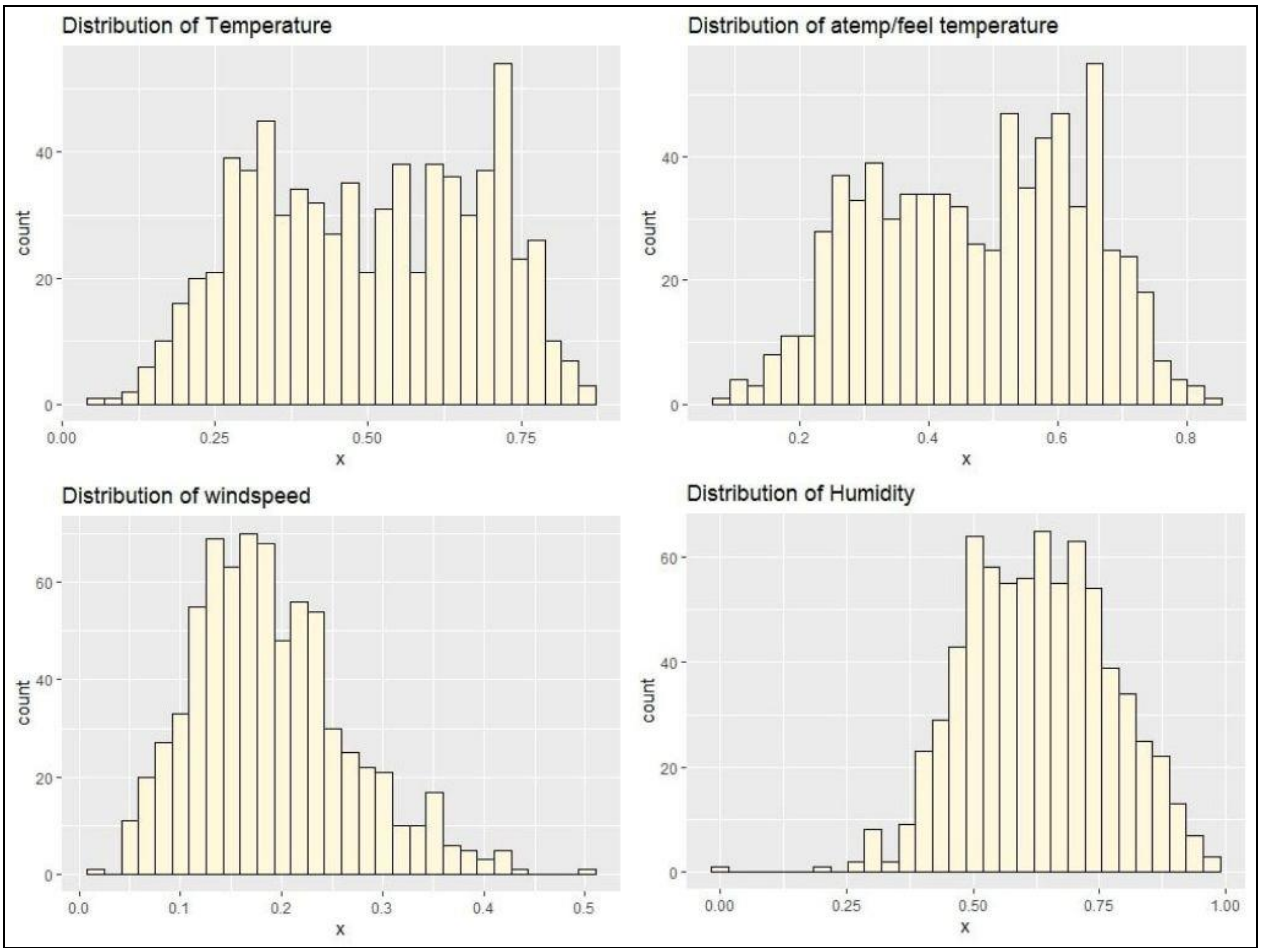


Fig. 1.1

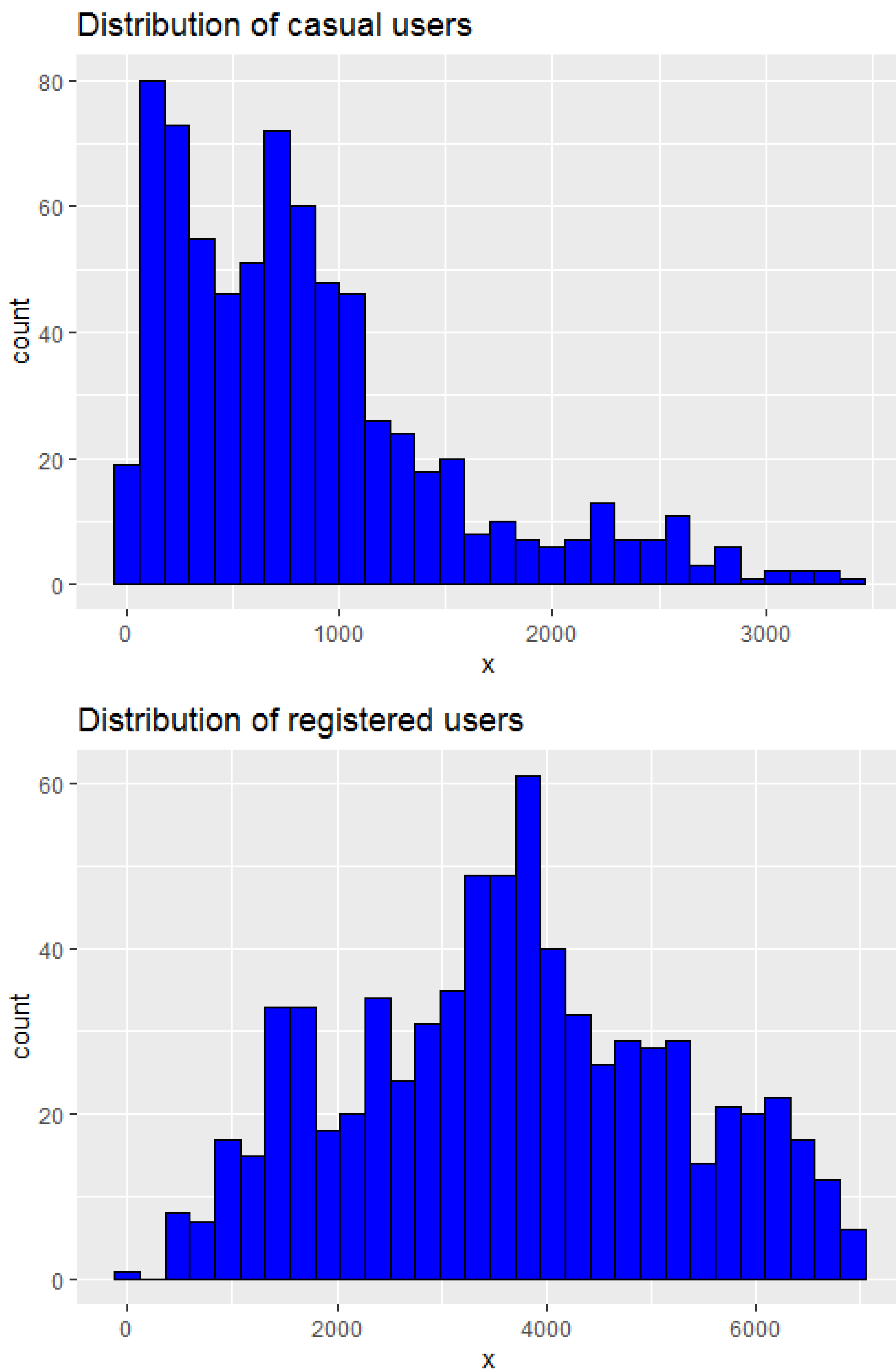

Distribution of casual users
count
80
60
40
20
0
0
1000
2000
3000
x
Distribution of registered users
count
60
40
20
0
0
2000
4000
6000
x

Fig. 1.2

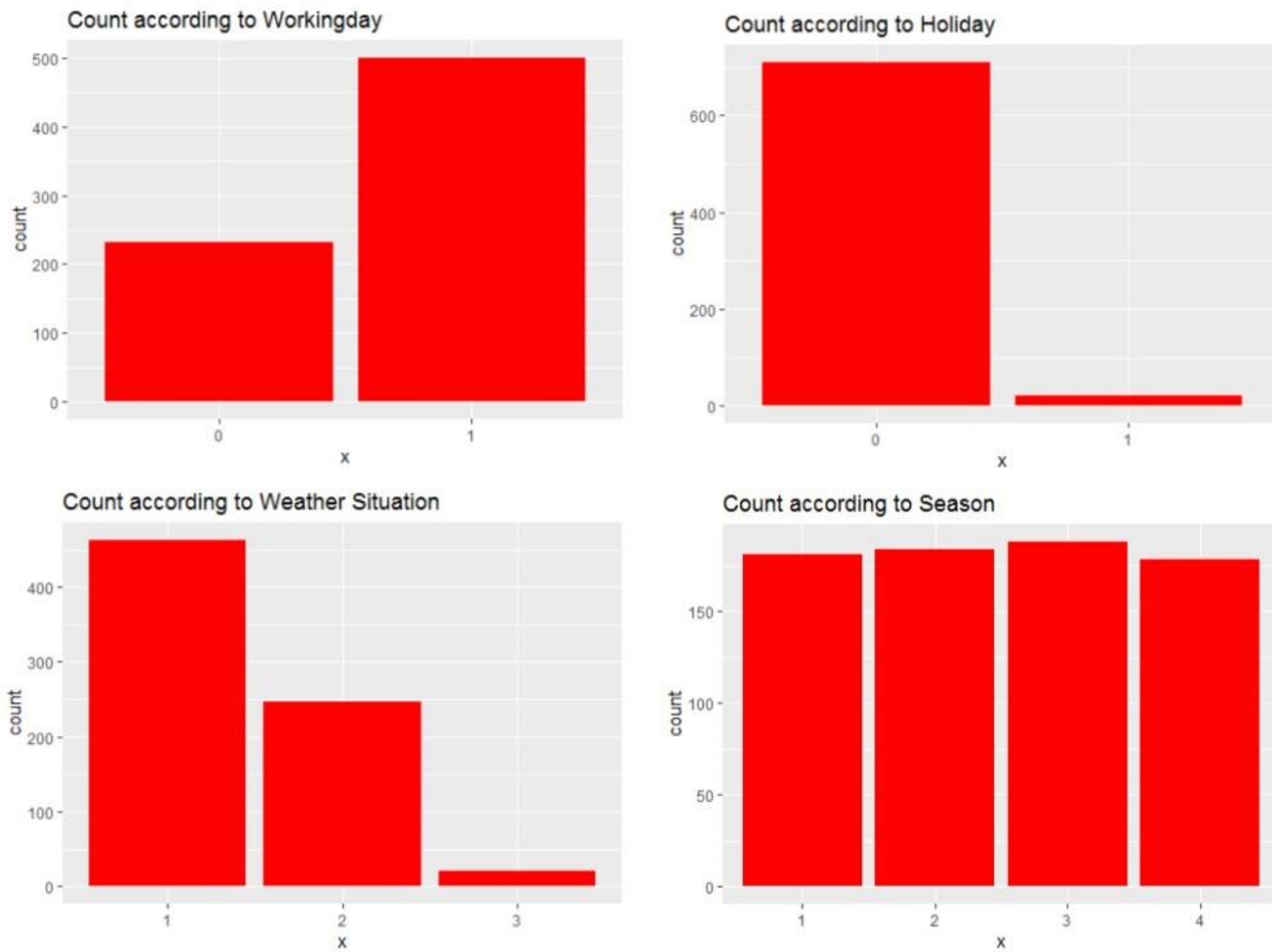
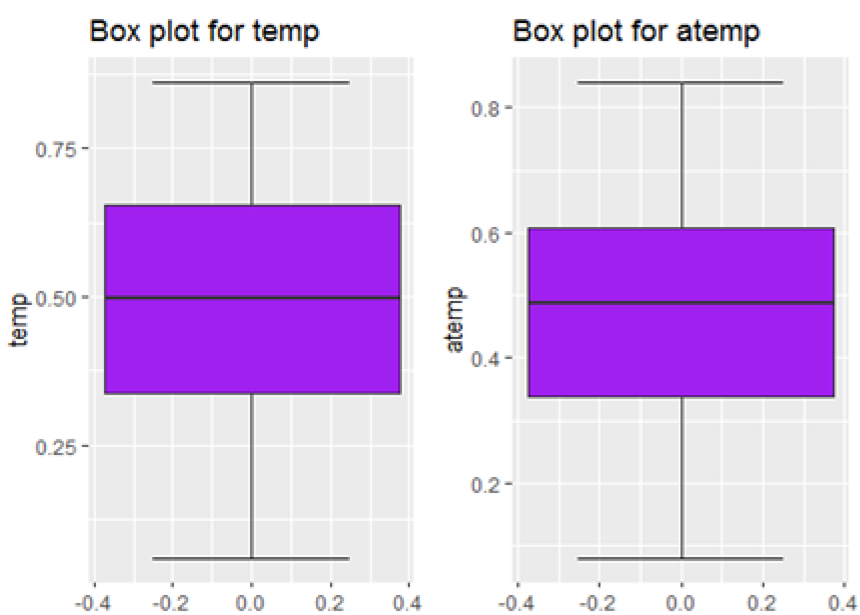

Count according to Workingday
Count according to Holiday
Count according to Weather Situation
Count according to Season
count
x

Fig. 1.3


Box plot for temp
Box plot for atemp
temp
atemp

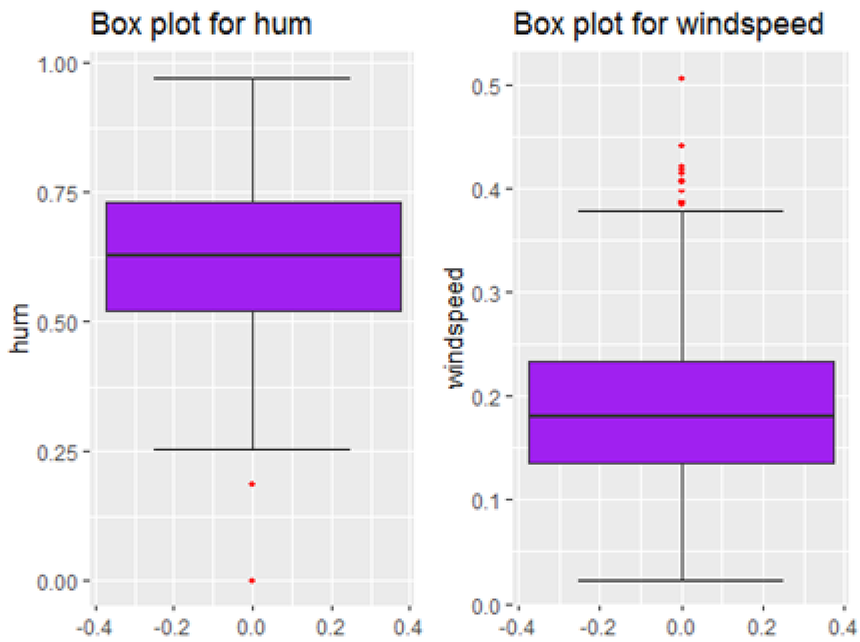

Box plot for hum
Box plot for windspeed
hum
windspeed
1.00
0.75
0.50
0.25
0.00
0.5
0.4
0.3
0.2
0.1
0.0
-0.4
-0.2
0.0
0.2
0.4

Fig. 1.4

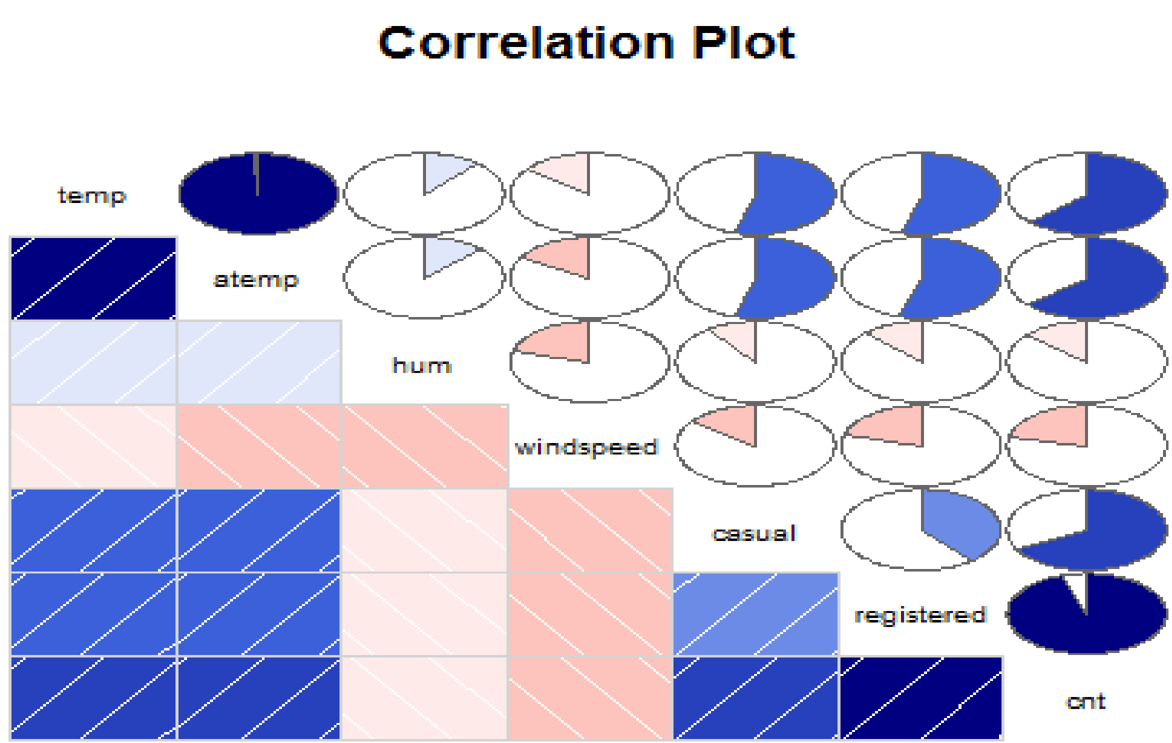

Correlation Plot
temp
atemp
hum
windspeed
casual
registered
cnt

Fig. 1.5

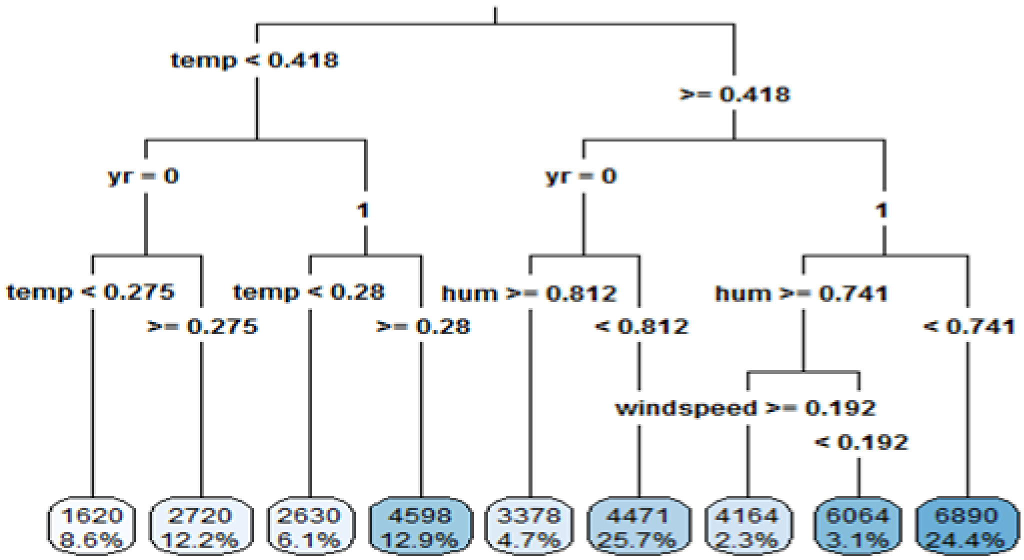


Fig. 1.6

Extra Figures : Error plots of regression and random forest models

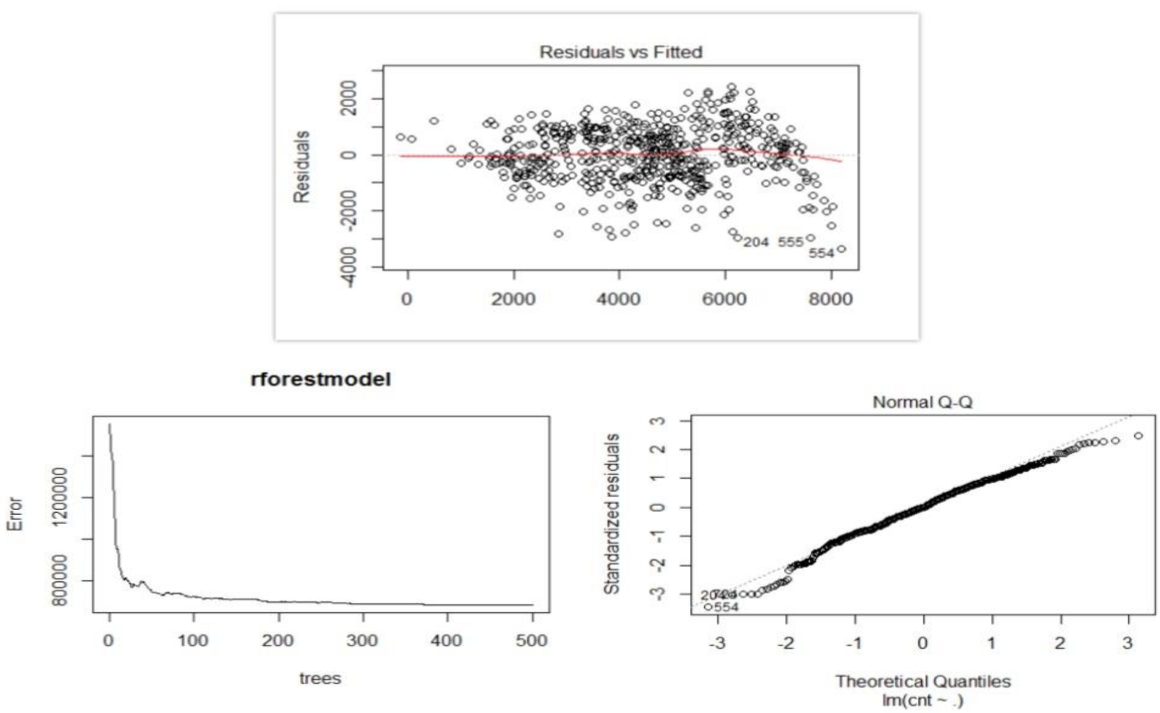


Fig. 1.7

APPENDIX B – R code

```
> rm(list=ls())
> setwd("C:/Users/Debayan Chakraborty/Documents/Edwisor bike project/Project
data")
> getwd()
[1] "C:/Users/Debayan Chakraborty/Documents/Edwisor bike project/Project
data"
> load_lib = c("ggplot2", "corrgram", "DMwR", "usdm", "randomForest", "plyr",
"dplyr", "DataCombine", "intrees", "rpart", "rpart.plot")
> lapply(load_lib, install.packages)
WARNING: Rtools is required to build R packages but is not currently
installed. Please download and install the appropriate version of Rtools
before proceeding:

https://cran.rstudio.com/bin/windows/Rtools/
Installing package into ‘C:/Users/Debayan Chakraborty/Documents/R/win-
library/3.6’
(as ‘lib’ is unspecified)
trying URL
'https://cran.rstudio.com/bin/windows/contrib/3.6/ggplot2_3.2.1.zip'
Content type 'application/zip' length 3975719 bytes (3.8 MB)
downloaded 3.8 MB

package ‘ggplot2’ successfully unpacked and MD5 sums checked

The downloaded binary packages are in
      C:\Users\Debayan
Chakraborty\AppData\Local\Temp\Rtmp695f8o\downloaded_packages
WARNING: Rtools is required to build R packages but is not currently
installed. Please download and install the appropriate version of Rtools
before proceeding:

https://cran.rstudio.com/bin/windows/Rtools/
Installing package into ‘C:/Users/Debayan Chakraborty/Documents/R/win-
library/3.6’
(as ‘lib’ is unspecified)
trying URL
'https://cran.rstudio.com/bin/windows/contrib/3.6/corrgram_1.13.zip'
Content type 'application/zip' length 322406 bytes (314 KB)
downloaded 314 KB

package ‘corrgram’ successfully unpacked and MD5 sums checked

The downloaded binary packages are in
      C:\Users\Debayan
Chakraborty\AppData\Local\Temp\Rtmp695f8o\downloaded_packages
```

```
WARNING: Rtools is required to build R packages but is not currently
installed. Please download and install the appropriate version of Rtools
before proceeding:

https://cran.rstudio.com/bin/windows/Rtools/
Installing package into 'C:/Users/Debayan Chakraborty/Documents/R/win-
library/3.6'
(as 'lib' is unspecified)
trying URL 'https://cran.rstudio.com/bin/windows/contrib/3.6/DMwR_0.4.1.zip'
Content type 'application/zip' length 3184702 bytes (3.0 MB)
downloaded 3.0 MB

package 'DMwR' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
      C:\Users\Debayan
Chakraborty\AppData\Local\Temp\Rtmp695f8o\downloaded_packages
WARNING: Rtools is required to build R packages but is not currently
installed. Please download and install the appropriate version of Rtools
before proceeding:

https://cran.rstudio.com/bin/windows/Rtools/
Installing package into 'C:/Users/Debayan Chakraborty/Documents/R/win-
library/3.6'
(as 'lib' is unspecified)
trying URL 'https://cran.rstudio.com/bin/windows/contrib/3.6/usdm_1.1-18.zip'
Content type 'application/zip' length 558291 bytes (545 KB)
downloaded 545 KB

package 'usdm' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
      C:\Users\Debayan
Chakraborty\AppData\Local\Temp\Rtmp695f8o\downloaded_packages
WARNING: Rtools is required to build R packages but is not currently
installed. Please download and install the appropriate version of Rtools
before proceeding:

https://cran.rstudio.com/bin/windows/Rtools/
Installing package into 'C:/Users/Debayan Chakraborty/Documents/R/win-
library/3.6'
(as 'lib' is unspecified)
trying URL
'https://cran.rstudio.com/bin/windows/contrib/3.6/randomForest_4.6-14.zip'
Content type 'application/zip' length 250260 bytes (244 KB)
downloaded 244 KB

package 'randomForest' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
      C:\Users\Debayan
Chakraborty\AppData\Local\Temp\Rtmp695f8o\downloaded_packages
```

```
WARNING: Rtools is required to build R packages but is not currently
installed. Please download and install the appropriate version of Rtools
before proceeding:

https://cran.rstudio.com/bin/windows/Rtools/
Installing package into ‘C:/Users/Debayan Chakraborty/Documents/R/win-
library/3.6’
(as ‘lib’ is unspecified)
trying URL 'https://cran.rstudio.com/bin/windows/contrib/3.6/plyr_1.8.4.zip'
Content type 'application/zip' length 1302686 bytes (1.2 MB)
downloaded 1.2 MB

package ‘plyr’ successfully unpacked and MD5 sums checked

The downloaded binary packages are in
      C:\Users\Debayan
Chakraborty\AppData\Local\Temp\Rtmp695f8o\downloaded_packages
WARNING: Rtools is required to build R packages but is not currently
installed. Please download and install the appropriate version of Rtools
before proceeding:

https://cran.rstudio.com/bin/windows/Rtools/
Installing package into ‘C:/Users/Debayan Chakraborty/Documents/R/win-
library/3.6’
(as ‘lib’ is unspecified)
trying URL 'https://cran.rstudio.com/bin/windows/contrib/3.6/dplyr_0.8.3.zip'
Content type 'application/zip' length 3264868 bytes (3.1 MB)
downloaded 3.1 MB

package ‘dplyr’ successfully unpacked and MD5 sums checked

The downloaded binary packages are in
      C:\Users\Debayan
Chakraborty\AppData\Local\Temp\Rtmp695f8o\downloaded_packages
WARNING: Rtools is required to build R packages but is not currently
installed. Please download and install the appropriate version of Rtools
before proceeding:

https://cran.rstudio.com/bin/windows/Rtools/
Installing package into ‘C:/Users/Debayan Chakraborty/Documents/R/win-
library/3.6’
(as ‘lib’ is unspecified)
trying URL
'https://cran.rstudio.com/bin/windows/contrib/3.6/DataCombine_0.2.21.zip'
Content type 'application/zip' length 120094 bytes (117 KB)
downloaded 117 KB

package ‘DataCombine’ successfully unpacked and MD5 sums checked

The downloaded binary packages are in
      C:\Users\Debayan
Chakraborty\AppData\Local\Temp\Rtmp695f8o\downloaded_packages
```

```
WARNING: Rtools is required to build R packages but is not currently
installed. Please download and install the appropriate version of Rtools
before proceeding:

https://cran.rstudio.com/bin/windows/Rtools/
Installing package into ‘C:/Users/Debayan Chakraborty/Documents/R/win-
library/3.6’
(as ‘lib’ is unspecified)
Warning in install.packages :
  package ‘intrees’ is not available (for R version 3.6.0)
Warning in install.packages :
  Perhaps you meant ‘inTrees’ ?
WARNING: Rtools is required to build R packages but is not currently
installed. Please download and install the appropriate version of Rtools
before proceeding:

https://cran.rstudio.com/bin/windows/Rtools/
Installing package into ‘C:/Users/Debayan Chakraborty/Documents/R/win-
library/3.6’
(as ‘lib’ is unspecified)
trying URL 'https://cran.rstudio.com/bin/windows/contrib/3.6/rpart_4.1-
15.zip'
Content type 'application/zip' length 769904 bytes (751 KB)
downloaded 751 KB

package ‘rpart’ successfully unpacked and MD5 sums checked

The downloaded binary packages are in
      C:\Users\Debayan
Chakraborty\AppData\Local\Temp\Rtmp695f8o\downloaded_packages
WARNING: Rtools is required to build R packages but is not currently
installed. Please download and install the appropriate version of Rtools
before proceeding:

https://cran.rstudio.com/bin/windows/Rtools/
Installing package into ‘C:/Users/Debayan Chakraborty/Documents/R/win-
library/3.6’
(as ‘lib’ is unspecified)
trying URL
'https://cran.rstudio.com/bin/windows/contrib/3.6/rpart.plot_3.0.8.zip'
Content type 'application/zip' length 1078231 bytes (1.0 MB)
downloaded 1.0 MB

package ‘rpart.plot’ successfully unpacked and MD5 sums checked

The downloaded binary packages are in
      C:\Users\Debayan
Chakraborty\AppData\Local\Temp\Rtmp695f8o\downloaded_packages
[[1]]
NULL

[[2]]
NULL
```

```
[[3]]
NULL

[[4]]
NULL

[[5]]
NULL

[[6]]
NULL

[[7]]
NULL

[[8]]
NULL

[[9]]
NULL

[[10]]
NULL

[[11]]
NULL

> lapply(load_lib, require, character.only = TRUE)
Loading required package: ggplot2
Loading required package: corrgram
Registered S3 method overwritten by 'seriation':
  method         from
  reorder.hclust gclus
Loading required package: DMwR
Loading required package: lattice

Attaching package: ‘lattice’

The following object is masked from ‘package:corrgram’:

    panel.fill

Loading required package: grid
Registered S3 method overwritten by 'xts':
  method     from
  as.zoo.xts zoo
Registered S3 method overwritten by 'quantmod':
  method            from
  as.zoo.data.frame zoo
Loading required package: usdm
Loading required package: sp
Loading required package: raster
Loading required package: randomForest
randomForest 4.6-14
```

```
Type rfNews() to see new features/changes/bug fixes.

Attaching package: ‘randomForest’

The following object is masked from ‘package:ggplot2’:

    margin

Loading required package: plyr

Attaching package: ‘plyr’

The following object is masked from ‘package:DMwR’:

    join

The following object is masked from ‘package:corrgram’:

    baseball

Loading required package: dplyr

Attaching package: ‘dplyr’

The following objects are masked from ‘package:plyr’:

    arrange, count, desc, failwith, id, mutate, rename, summarise, summarize

The following object is masked from ‘package:randomForest’:

    combine

The following objects are masked from ‘package:raster’:

    intersect, select, union

The following objects are masked from ‘package:stats’:

    filter, lag

The following objects are masked from ‘package:base’:

    intersect, setdiff, setequal, union

Loading required package: DataCombine

Attaching package: ‘DataCombine’

The following object is masked from ‘package:raster’:

    shift

Loading required package: intrees
Loading required package: rpart
```

```
Loading required package: rpart.plot
[[1]]
[1] TRUE

[[2]]
[1] TRUE

[[3]]
[1] TRUE

[[4]]
[1] TRUE

[[5]]
[1] TRUE

[[6]]
[1] TRUE

[[7]]
[1] TRUE

[[8]]
[1] TRUE

[[9]]
[1] FALSE

[[10]]
[1] TRUE

[[11]]
[1] TRUE

There were 13 warnings (use warnings() to see them)
> #In the above codes, firstly we have cleaned the R environment, secondly we
set our working directory and finally installed and loaded the required
libraries. Now we will be extracting the required csv file and perform
exploratory data analysis on it#
>
> project_data = read.csv("day.csv", header = T, sep = ",", na.strings =
c(""," ", "NA"))
>
> #Exploratory data analysis#
>
> View(project_data)
> dim(project_data)
[1] 731  16
> str(project_data)
'data.frame':	731 obs. of  16 variables:
 $ instant   : int  1 2 3 4 5 6 7 8 9 10 ...
 $ dteday    : Factor w/ 731 levels "2011-01-01","2011-01-02",..: 1 2 3 4 5 6
7 8 9 10 ...
 $ season    : int  1 1 1 1 1 1 1 1 1 1 ...
```

```
 $ yr        : int  0 0 0 0 0 0 0 0 0 0 ...
 $ mnth      : int  1 1 1 1 1 1 1 1 1 1 ...
 $ holiday   : int  0 0 0 0 0 0 0 0 0 0 ...
 $ weekday   : int  6 0 1 2 3 4 5 6 0 1 ...
 $ workingday: int  0 0 1 1 1 1 1 0 0 1 ...
 $ weathersit: int  2 2 1 1 1 1 2 2 1 1 ...
 $ temp      : num  0.344 0.363 0.196 0.2 0.227 ...
 $ atemp     : num  0.364 0.354 0.189 0.212 0.229 ...
 $ hum       : num  0.806 0.696 0.437 0.59 0.437 ...
 $ windspeed : num  0.16 0.249 0.248 0.16 0.187 ...
 $ casual    : int  331 131 120 108 82 88 148 68 54 41 ...
 $ registered: int  654 670 1229 1454 1518 1518 1362 891 768 1280 ...
 $ cnt       : int  985 801 1349 1562 1600 1606 1510 959 822 1321 ...
> #Applying necesssary data type conversions#
>
> project_data$season = as.factor(project_data$season)
> project_data$yr = as.factor(project_data$yr)
> project_data$mnth = as.factor(project_data$mnth)
> project_data$holiday = as.factor(project_data$holiday)
> project_data$workingday = as.factor(project_data$workingday)
> project_data$weathersit = as.factor(project_data$weathersit)
> project_data$dteday = as.character(project_data$dteday)
> project_data$casual = as.numeric(project_data$casual)
> project_data$registered = as.numeric(project_data$registered)
> project_data$weekday = as.factor(project_data$weekday)
>
> #Structure after applying data type conversions#
>
> str(project_data)
'data.frame':	731 obs. of  16 variables:
 $ instant   : int  1 2 3 4 5 6 7 8 9 10 ...
 $ dteday    : chr  "2011-01-01" "2011-01-02" "2011-01-03" "2011-01-04" ...
 $ 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  0.344 0.363 0.196 0.2 0.227 ...
 $ atemp     : num  0.364 0.354 0.189 0.212 0.229 ...
 $ hum       : num  0.806 0.696 0.437 0.59 0.437 ...
 $ windspeed : num  0.16 0.249 0.248 0.16 0.187 ...
 $ casual    : num  331 131 120 108 82 88 148 68 54 41 ...
 $ registered: num  654 670 1229 1454 1518 ...
 $ cnt       : int  985 801 1349 1562 1600 1606 1510 959 822 1321 ...
>
> #Applying missing value analysis#
>
> missing_val = sapply(project_data, function(x){sum(is.na(x))})
> missing_val
   instant     dteday     season         yr       mnth    holiday    weekday
         0          0          0          0          0          0          0
```

```
workingday weathersit       temp      atemp        hum  windspeed     casual 
         0          0          0          0          0          0          0 
registered        cnt 
         0          0 
> 
> #Data visualisation using graphs#
> 
> #Data visualisation using histogram to understand the data distribution#
> 
> histo1 = ggplot(project_data, aes_string(x = project_data$temp)) + 
ggtitle("Distribution of Temperature") + geom_histogram(fill= "cornsilk", 
colour = "black")
> View(histo1)
> histo1
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
> 
> histo2 = ggplot(project_data,aes_string(x = project_data$hum)) + 
geom_histogram(fill ="cornsilk", colour ="black", bins = 30) + 
ggtitle("Distribution of Humidity")
> histo2
> histo3 = ggplot(project_data, aes_string(x = project_data$windspeed)) + 
geom_histogram(fill= "cornsilk", colour = "black" , bins = 30) + 
ggtitle("Distribution of windspeed")
> 
> histo3
> 
> histo4 = ggplot(project_data, aes_string(x = project_data$atemp)) + 
geom_histogram(fill = "cornsilk", colour = "black", bins = 30) + 
ggtitle("Distribution of atemp/feel temperature")
> histo4
> 
> histo5 = ggplot(project_data, aes_string(x= project_data$casual)) + 
geom_histogram(fill = "blue", colour = "black", bins = 30) + 
ggtitle("Distribution of casual users")
> histo5
> histo6 = ggplot(project_data, aes_string(x= project_data$registered)) + 
geom_histogram(fill = "blue", colour = "black", bins = 30) + 
ggtitle("Distribution of registered users")
> histo6
> histo3
> 
> #Now we have successfully plotted the distribution of numerical data with 
bike count. Next we will be plotting bar charts explaining the distribution 
of categorical variables#
> 
> barchart1 = ggplot(project_data, aes_string(x = project_data$season)) + 
geom_bar(stat ="count", fill = "red") + ggtitle("Count according to Season")
> barchart1
> barchart2 = ggplot(project_data, aes_string(x = project_data$holiday)) + 
geom_bar(stat ="count", fill = "red") + ggtitle("Count according to Holiday")
> barchart2
> barchart3 = ggplot(project_data, aes_string(x = project_data$workingday)) + 
geom_bar(stat ="count", fill = "red") + ggtitle("Count according to 
Workingday")
```

```
> barchart3
> barchart4 = ggplot(project_data, aes_string(x = project_data$weathersit)) +
geom_bar(stat ="count", fill = "red") + ggtitle("Count according to Weather
Situation")
> barchart4
>
> #In the above codes we have plotted the bar graph of the required
categorical variables. Now we shall proceed for outlier analysis using
boxplots#
>
> #Outlier analysis#
>
> numeric_sort = sapply(project_data, is.numeric)
> numeric_select = project_data[,numeric_sort]
> cnames = colnames(numeric_select)
> for (i in 1:length(cnames))
+
+ { assign(paste0("XY",i), ggplot(aes_string(y = cnames[i]))) }
Error: `data` must be a data frame, or other object coercible by `fortify()`,
not an S3 object with class uneval
Did you accidentally pass `aes()` to the `data` argument?
> numeric_sort = sapply(project_data, is.numeric)
> numeric_select = project_data[,numeric_sort]
> cnames = colnames(numeric_select)
> for (i in 1:length(cnames)) {assign(paste0("XY", i), ggplot(aes_string(y =
cnames[i]), data = project_data) + stat_boxplot(geom = "errorbar", width =
0.5) + geom_boxplot(outlier.color = "red", fill = "purple", outlier.shape =
18, outlier.size = 1, notch = FALSE) + theme(legend.position = "bottom") +
labs(y=cnames[i] + ggtitle(paste("Box plot for", cnames[i]))) )}
Error in cnames[i] + ggtitle(paste("Box plot for", cnames[i])) :
  non-numeric argument to binary operator
> gridExtra::grid.arrange(XY1,XY2, XY3, ncol=3)
Error in arrangeGrob(...) : object 'XY1' not found
> savehistory("~/Edwisor bike project/Project data/project 1.Rhistory")
> for (i in 1:length(cnames))
{assign(paste0("XY", i), ggplot(aes_string(y = cnames[i]), data =
project_data) + stat_boxplot(geom = "errorbar", width = 0.5) +
geom_boxplot(outlier.color = "red", fill = "purple", outlier.shape = 18,
outlier.size = 1, notch = FALSE) + theme(legend.position = "bottom") +
labs(y=cnames[i]) + ggtitle(paste("Box plot for", cnames[i])))
}
> ##There was an error in first trial of the code as i forgot to close ")"
labs(). Now it has been resolved and run successfully. We will now plot the
boxplots ##
>
> gridExtra::grid.arrange(XY1,XY2,XY3,ncol =3)
> #Here we have generated a boplot but it has considered "instant" which is
not required, hence we need to modify the code and plot it again considering
only the required variables#
> gridExtra::grid.arrange(XY2,XY3,ncol =2)
> gridExtra::grid.arrange(XY4,XY5,ncol =2)
>
> #we have found some outliers in humidity and windspeed which needs removal#
>
```

```
> outval1 = project_data$windspeed[project_data$windspeed %in%
boxplot.stats(project_data$windspeed)$out]
> project_data = project_data[which(!project_data$windspeed %in% outval1 ),]
> outval2 = project_data$hum[project_data$hum %in%
boxplot.stats(project_data$hum)$out]
> project_data = project_data[which(!project_data$hum %in% outval2 ),]
> View(project_data)
> gridExtra::grid.arrange(XY4,XY5,ncol =2)
> View(project_data)
> sum(is.na(project_data))
[1] 0
> savehistory("~/Edwisor bike project/Project data/project 1.Rhistory")
>
>
> #Feature Selection#
>
> corrgram(project_data[,num_data], order = F, upper.panel = panel.pie,
text.panel = panel.txt, main = "Correlation Plot")
> #As we have plotted the correlation among numerical variables, same way we
will perform chi sq test on categorical variables and finally remove the
unwanted variables of both kinds#
>
> #chi square test#
> cat_index = c("dteday", "season", "yr", "mnth", "holiday", "weekday",
"workingday", "weathersit")
> cat_data = project_data[,cat_index]
> ##since the target variable is a continuous variable so chi sq test
dependency with target variable and categorical variable is not applicable,
hence we may use chi sq test to check dependency among categorical
variables##

> for (i in cat_index) { for (j in cat_index) { print(i) print(j)
print(chisq.test(table(cat_data[,i], cat_data[,j]))$p.value)} }
> chians = chisq.test(table(cat_index))
Warning message:
In chisq.test(table(cat_index)) :
  Chi-squared approximation may be incorrect
> chians

	Chi-squared test for given probabilities

data:  table(cat_index)
X-squared = 0, df = 7, p-value = 1

> for (i in cat_index) {
+     for (j in cat_index) {
+     print(i)
+     print(j)
+     print(chisq.test(table(cat_data[,i], cat_data[,j]))$p.value)
+     }
+     }
[1] "dteday"
[1] "dteday"
[1] 0.2396477
```

```
[1] "dteday"
[1] "season"
[1] 0.4777028
[1] "dteday"
[1] "yr"
[1] 0.4824403
[1] "dteday"
[1] "mnth"
[1] 0.4629856
[1] "dteday"
[1] "holiday"
[1] 0.4824403
[1] "dteday"
[1] "weekday"
[1] 0.4713443
[1] "dteday"
[1] "workingday"
[1] 0.4824403
[1] "dteday"
[1] "weathersit"
[1] 0.4801367
[1] "season"
[1] "dteday"
[1] 0.4777028
[1] "season"
[1] "season"
[1] 0
[1] "season"
[1] "yr"
[1] 0.9988045
[1] "season"
[1] "mnth"
[1] 0
[1] "season"
[1] "holiday"
[1] 0.6405518
[1] "season"
[1] "weekday"
[1] 1
[1] "season"
[1] "workingday"
[1] 0.9463399
[1] "season"
[1] "weathersit"
[1] 0.01324758
[1] "yr"
[1] "dteday"
[1] 0.4824403
[1] "yr"
[1] "season"
[1] 0.9988045
[1] "yr"
[1] "yr"
[1] 4.442159e-157
```

```
[1] "yr"
[1] "mnth"
[1] 1
[1] "yr"
[1] "holiday"
[1] 0.9948241
[1] "yr"
[1] "weekday"
[1] 0.9999605
[1] "yr"
[1] "workingday"
[1] 0.9561017
[1] "yr"
[1] "weathersit"
[1] 0.1832495
[1] "mnth"
[1] "dteday"
[1] 0.4629856
[1] "mnth"
[1] "season"
[1] 0
[1] "mnth"
[1] "yr"
[1] 1
[1] "mnth"
[1] "mnth"
[1] 0
[1] "mnth"
[1] "holiday"
[1] 0.5712387
[1] "mnth"
[1] "weekday"
[1] 1
[1] "mnth"
[1] "workingday"
[1] 0.9927346
[1] "mnth"
[1] "weathersit"
[1] 0.009828299
[1] "holiday"
[1] "dteday"
[1] 0.4824403
[1] "holiday"
[1] "season"
[1] 0.6405518
[1] "holiday"
[1] "yr"
[1] 0.9948241
[1] "holiday"
[1] "mnth"
[1] 0.5712387
[1] "holiday"
[1] "holiday"
[1] 2.156836e-150
```

```
[1] "holiday"
[1] "weekday"
[1] 5.970403e-11
[1] "holiday"
[1] "workingday"
[1] 3.657543e-11
[1] "holiday"
[1] "weathersit"
[1] 0.5987296
[1] "weekday"
[1] "dteday"
[1] 0.4713443
[1] "weekday"
[1] "season"
[1] 1
[1] "weekday"
[1] "yr"
[1] 0.9999605
[1] "weekday"
[1] "mnth"
[1] 1
[1] "weekday"
[1] "holiday"
[1] 5.970403e-11
[1] "weekday"
[1] "weekday"
[1] 0
[1] "weekday"
[1] "workingday"
[1] 6.500488e-133
[1] "weekday"
[1] "weathersit"
[1] 0.2490378
[1] "workingday"
[1] "dteday"
[1] 0.4824403
[1] "workingday"
[1] "season"
[1] 0.9463399
[1] "workingday"
[1] "yr"
[1] 0.9561017
[1] "workingday"
[1] "mnth"
[1] 0.9927346
[1] "workingday"
[1] "holiday"
[1] 3.657543e-11
[1] "workingday"
[1] "weekday"
[1] 6.500488e-133
[1] "workingday"
[1] "workingday"
[1] 6.092456e-157
```

```
[1] "workingday"
[1] "weathersit"
[1] 0.2937806
[1] "weathersit"
[1] "dteday"
[1] 0.4801367
[1] "weathersit"
[1] "season"
[1] 0.01324758
[1] "weathersit"
[1] "yr"
[1] 0.1832495
[1] "weathersit"
[1] "mnth"
[1] 0.009828299
[1] "weathersit"
[1] "holiday"
[1] 0.5987296
[1] "weathersit"
[1] "weekday"
[1] 0.2490378
[1] "weathersit"
[1] "workingday"
[1] 0.2937806
[1] "weathersit"
[1] "weathersit"
[1] 2.930764e-309
There were 30 warnings (use warnings() to see them)
> #performing dimension reduction#
> project_data = subset(project_data, select = -c(registered, casual, atemp,
dteday, holiday, mnth, season))
> View(project_data)
> project_data = subset(project_data, select = -c(instant))
> View(project_data)
>
> #Data modelling#
>
> #Divide the data into test and train#
>
> set.seed(123)
> train_index = sample(1:nrow(project_data), 0.8*nrow(project_data))
> train = project_data[train_index,]
> test = project_data[-train_index,]
>
> #Running Linear regression model#
>
> lrgmodel = lm(cnt~.,data = train)
> summary(lrgmodel)

Call:
lm(formula = cnt ~ ., data = train)

Residuals:
    Min      1Q  Median      3Q     Max
```

```
-3354.9  -638.1     5.9   720.1  2433.5

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  1555.27     321.49   4.838 1.70e-06 ***
yr1          1984.08      83.69  23.708  < 2e-16 ***
weekday1     -365.11     280.87  -1.300  0.19417    
weekday2     -210.07     305.30  -0.688  0.49168    
weekday3     -276.04     306.73  -0.900  0.36854    
weekday4     -156.52     306.02  -0.511  0.60923    
weekday5     -137.85     303.62  -0.454  0.64999    
weekday6      425.97     153.97   2.767  0.00585 ** 
workingday1   530.97     267.78   1.983  0.04787 *  
weathersit2  -518.74     111.91  -4.635 4.44e-06 ***
weathersit3 -1838.30     305.13  -6.025 3.08e-09 ***
temp         5802.24     234.85  24.706  < 2e-16 ***
hum          -564.04     415.00  -1.359  0.17465    
windspeed   -3011.15     615.72  -4.890 1.32e-06 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 987.4 on 559 degrees of freedom
Multiple R-squared:  0.7364,  Adjusted R-squared:  0.7302 
F-statistic: 120.1 on 13 and 559 DF,  p-value: < 2.2e-16

> lrgprediction = predict(lrgmodel, test[,-10])
> lrgprediction
       1        3        9       18       22       27       32       42 
43 
2521.796 1866.153 1023.142 1686.607 1580.891 2330.602 1844.329 2433.076 
2352.943 
      44       55       58       60       64       65       73       80 
85 
2348.265 1978.790 2787.527 2469.986 2487.991 1652.658 2915.515 2414.397 
2670.297 
      89       97      102      104      106      128      131      147 
149 
2146.803 3639.388 3028.434 4004.119 1115.458 4039.757 4237.661 4810.127 
4322.260 
     150      151      152      154      155      159      178      181 
187 
4663.634 5678.571 4719.861 4583.833 5037.525 5548.920 4467.135 5169.062 
5117.230 
     188      213      219      221      225      236      250      252 
261 
5434.696 5415.942 4832.052 5596.119 4393.639 4613.657 2638.999 4320.124 
3570.462 
     265      269      271      277      279      280      281      288 
290 
4162.107 4157.190 4049.617 3665.096 4040.457 4459.047 4475.830 3871.303 
3965.500 
     299      304      319      320      326      327      328      331 
338
```

```
3246.794 2976.954 3443.009 1685.026 1554.878 2410.572 2751.391 3196.390
2784.433
     342      348      357      367      371      372      375      389
394
2422.156 2575.743 2901.693 3550.767 5060.039 5421.434 4721.967 5003.392
4280.103
     402      407      414      419      425      426      428      443
446
4530.936 2142.620 5100.329 5661.315 4280.260 5702.112 5014.240 4923.236
5627.517
     458      460      461      465      472      476      482      483
486
4998.514 6127.364 5565.699 5285.817 5853.437 6145.266 5327.877 5314.139
5041.355
     488      493      498      499      507      509      511      514
522
6202.420 5235.829 6597.012 6090.336 5498.359 6138.576 7033.395 6335.156
5502.547
     527      530      532      533      539      541      542      559
564
7049.354 6237.606 6776.706 6844.179 7569.533 7143.883 6669.783 7326.571
7978.541
     569      573      577      580      581      583      586      588
589
6180.719 7226.053 7105.919 7518.453 6845.103 6759.707 6914.342 6444.309
6429.503
     592      594      596      606      609      612      613      621
623
7179.878 7366.783 7025.288 7161.604 7527.502 6378.221 6934.258 6548.293
6915.160
     634      642      646      656      663      665      666      669
680
6018.381 6442.435 4608.789 5743.064 6625.084 5726.683 5405.960 4080.687
5685.142
     684      693      700      702      703      705      709      713
728
4559.607 5303.120 5121.019 4197.976 5648.744 5088.367 4263.265 4831.940
4082.704
> #Calculating MSE,MAPE,RMSE and MAE for evaluation of model#
>
> regr.eval(trues = test[,8], lrgprediction, stats = c("mae","mse", "rmse",
"mape"))
          mae          mse         rmse         mape
7.459957e+02 8.108159e+05 9.004531e+02 2.618554e-01
>
> lrgprediction = predict(lrgmodel, test[,-8])
> lrgprediction
       1        3        9       18       22       27       32       42
43
2521.796 1866.153 1023.142 1686.607 1580.891 2330.602 1844.329 2433.076
2352.943
      44       55       58       60       64       65       73       80
85
```

```
2348.265 1978.790 2787.527 2469.986 2487.991 1652.658 2915.515 2414.397
2670.297
      89       97      102      104      106      128      131      147
149
2146.803 3639.388 3028.434 4004.119 1115.458 4039.757 4237.661 4810.127
4322.260
     150      151      152      154      155      159      178      181
187
4663.634 5678.571 4719.861 4583.833 5037.525 5548.920 4467.135 5169.062
5117.230
     188      213      219      221      225      236      250      252
261
5434.696 5415.942 4832.052 5596.119 4393.639 4613.657 2638.999 4320.124
3570.462
     265      269      271      277      279      280      281      288
290
4162.107 4157.190 4049.617 3665.096 4040.457 4459.047 4475.830 3871.303
3965.500
     299      304      319      320      326      327      328      331
338
3246.794 2976.954 3443.009 1685.026 1554.878 2410.572 2751.391 3196.390
2784.433
     342      348      357      367      371      372      375      389
394
2422.156 2575.743 2901.693 3550.767 5060.039 5421.434 4721.967 5003.392
4280.103
     402      407      414      419      425      426      428      443
446
4530.936 2142.620 5100.329 5661.315 4280.260 5702.112 5014.240 4923.236
5627.517
     458      460      461      465      472      476      482      483
486
4998.514 6127.364 5565.699 5285.817 5853.437 6145.266 5327.877 5314.139
5041.355
     488      493      498      499      507      509      511      514
522
6202.420 5235.829 6597.012 6090.336 5498.359 6138.576 7033.395 6335.156
5502.547
     527      530      532      533      539      541      542      559
564
7049.354 6237.606 6776.706 6844.179 7569.533 7143.883 6669.783 7326.571
7978.541
     569      573      577      580      581      583      586      588
589
6180.719 7226.053 7105.919 7518.453 6845.103 6759.707 6914.342 6444.309
6429.503
     592      594      596      606      609      612      613      621
623
7179.878 7366.783 7025.288 7161.604 7527.502 6378.221 6934.258 6548.293
6915.160
     634      642      646      656      663      665      666      669
680
6018.381 6442.435 4608.789 5743.064 6625.084 5726.683 5405.960 4080.687
5685.142
```

```
     684      693      700      702      703      705      709      713
728
4559.607 5303.120 5121.019 4197.976 5648.744 5088.367 4263.265 4831.940
4082.704
>
> #Decision Tree#
>
> dtreemodel = rpart(cnt~.,train, method = "anova")
> dtreepreds = predict(dtreemodel, test[,-8])
> dtreepreds
       1        3        9       18       22       27       32       42
43
2719.543 1619.980 1619.980 1619.980 1619.980 1619.980 1619.980 1619.980
1619.980
      44       55       58       60       64       65       73       80
85
2719.543 2719.543 2719.543 1619.980 2719.543 2719.543 2719.543 4471.061
1619.980
      89       97      102      104      106      128      131      147
149
2719.543 4471.061 4471.061 4471.061 3377.630 4471.061 4471.061 4471.061
3377.630
     150      151      152      154      155      159      178      181
187
4471.061 4471.061 4471.061 4471.061 4471.061 4471.061 4471.061 4471.061
4471.061
     188      213      219      221      225      236      250      252
261
4471.061 4471.061 4471.061 4471.061 4471.061 4471.061 3377.630 3377.630
4471.061
     265      269      271      277      279      280      281      288
290
3377.630 3377.630 3377.630 4471.061 4471.061 4471.061 4471.061 4471.061
4471.061
     299      304      319      320      326      327      328      331
338
4471.061 2719.543 4471.061 3377.630 2719.543 4471.061 2719.543 4471.061
2719.543
     342      348      357      367      371      372      375      389
394
1619.980 2719.543 2719.543 2630.486 4598.243 4598.243 4598.243 4598.243
4598.243
     402      407      414      419      425      426      428      443
446
4598.243 2630.486 4598.243 6889.607 4598.243 6889.607 4598.243 6063.722
6063.722
     458      460      461      465      472      476      482      483
486
6889.607 6889.607 6889.607 6889.607 6889.607 6889.607 6063.722 6889.607
6889.607
     488      493      498      499      507      509      511      514
522
6063.722 6889.607 6889.607 6889.607 4164.462 6063.722 6063.722 6889.607
6889.607
```

```
     527      530      532      533      539      541      542      559 
564 
6889.607 6889.607 6889.607 6889.607 6889.607 6889.607 6889.607 6889.607 
6889.607 
     569      573      577      580      581      583      586      588 
589 
6063.722 6889.607 6889.607 6889.607 6889.607 6889.607 6889.607 6889.607 
6889.607 
     592      594      596      606      609      612      613      621 
623 
6889.607 6889.607 6889.607 6889.607 6889.607 6063.722 4164.462 6889.607 
6889.607 
     634      642      646      656      663      665      666      669 
680 
6889.607 6063.722 4598.243 6889.607 6889.607 6063.722 6889.607 4598.243 
4598.243 
     684      693      700      702      703      705      709      713 
728 
4598.243 4598.243 4598.243 4598.243 6063.722 6889.607 4598.243 4598.243 
2630.486 
> regr.eval(trues = test[,8], dtreepreds, stats = 
c("mae","mse","rmse","mape"))
         mae          mse         rmse         mape 
7.962755e+02 1.034893e+06 1.017297e+03 3.133509e-01 
> rpart.plot(dtreemodel,type = 3, digits = 3, fallen.leaves = TRUE)
> 
> #Random Forest#
> 
> rforestmodel = randomForest(cnt~., train, ntree = 500)
> rforestpreds = predict(rforestmodel, test[,-8])
> 
> regr.eval(trues = test[,8],rforestpreds, stats = c("mae", "mse", 
"rmse","mape"))
         mae          mse         rmse         mape 
7.031364e+02 7.692421e+05 8.770645e+02 2.876393e-01 
> savehistory("~/Edwisor bike project/Project data/project 1.Rhistory")
> save.image("~/Edwisor bike project/Project data/project1 workspace.RData")
> plot(rforestmodel)
> plot(dtreemodel)
> plot(lrgmodel)
Hit <Return> to see next plot: 
Hit <Return> to see next plot: 
Hit <Return> to see next plot: 
Hit <Return> to see next plot: 
> 
> 
> savehistory("~/Edwisor bike project/Project data/project 1.Rhistory")
> save.image("~/Edwisor bike project/Project data/project1 workspace.RData")
```

---X---