Bike Renting Project

By

Siddharth Saxena

Chapter 1

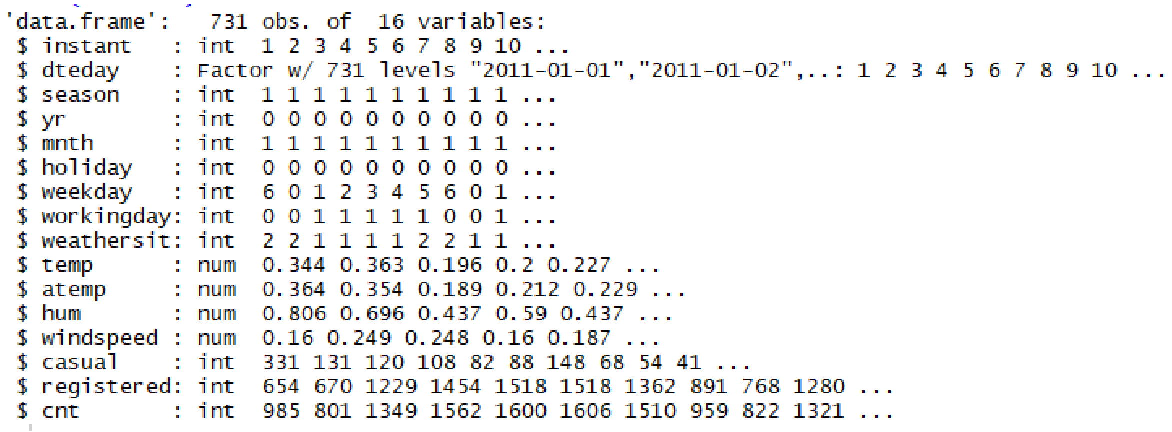
1. Introduction

1.1 Problem Statement

The main objective of this project is the prediction of numbers of rented bikes on a daily basis. The predictions are based on the environmental and seasonal settings. There are many predictors in the dataset. The project tries to predict the number of bikes which could be rented by assessing the parameters given in the dataset.

1.2 Data

As our target dataset is numeric, we need to apply regression models to accurately predict the number of bikes. There are 16 variables and 731 observations in the dataset.



```
'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

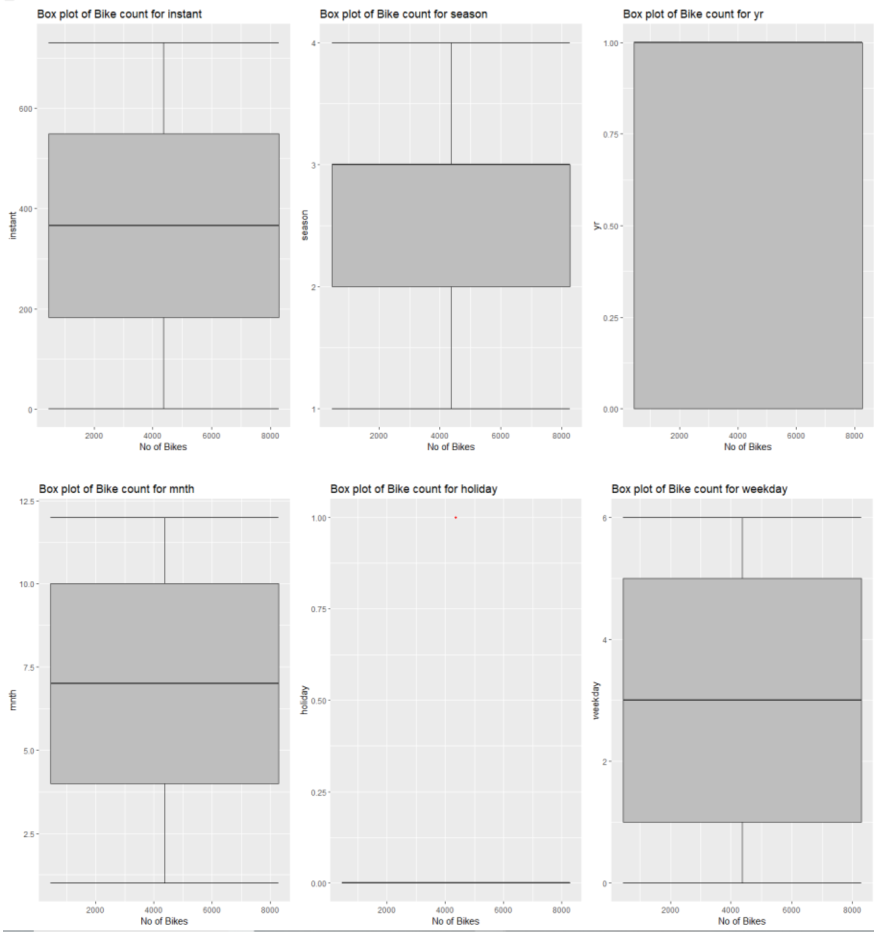
2. Methodology

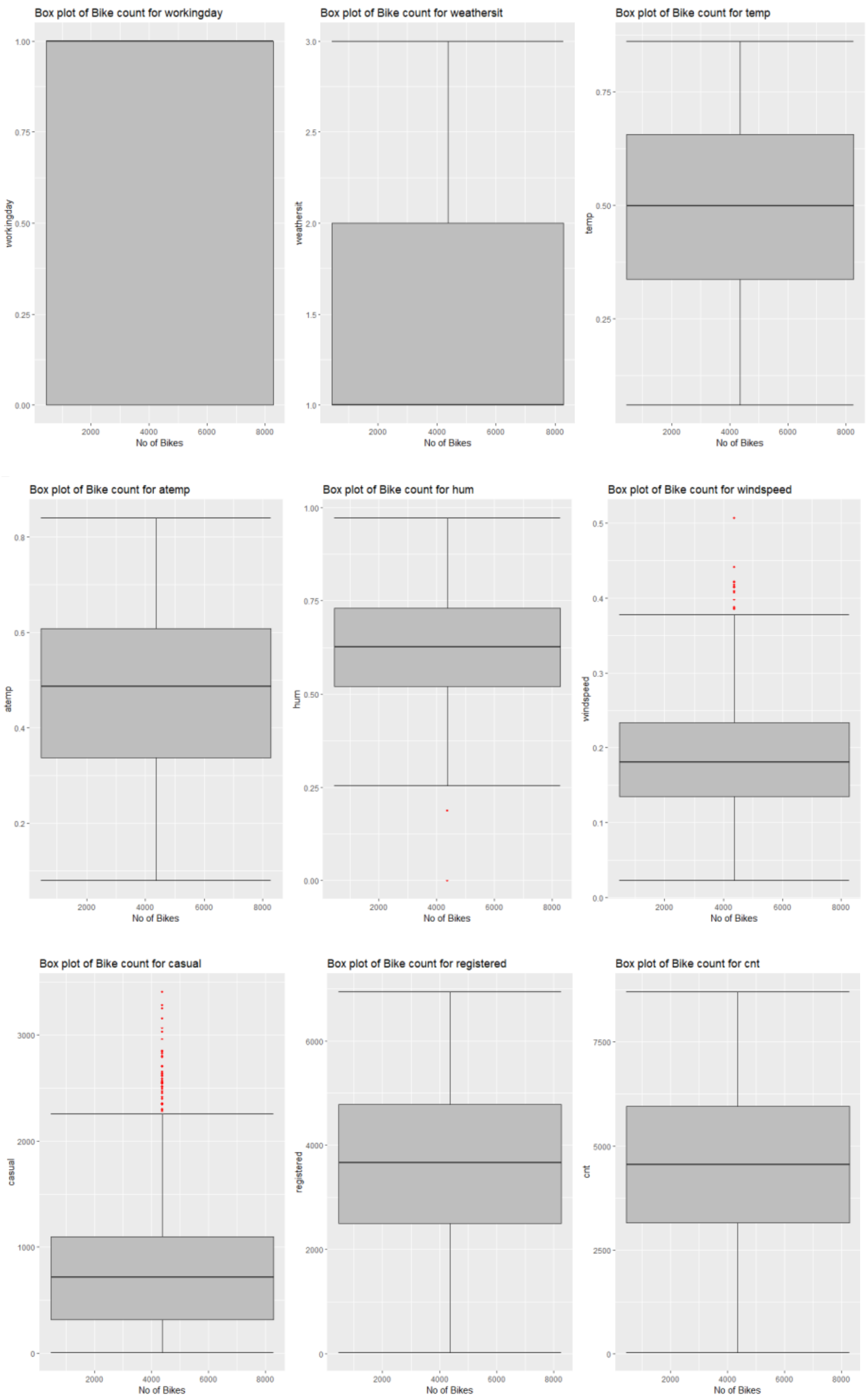
2.1 Data Pre-Processing

Data Pre-Processing is divided in four parts. These are Outlier Analysis, Missing Value Analysis, Feature Selection and Feature Scaling.

2.2 Outlier Analysis

In Outlier Analysis, we used the boxplot method to determine the values which are insignificant for our model. We first created and array of numerical variable and checked each boxplot one by one to find out the outliers.





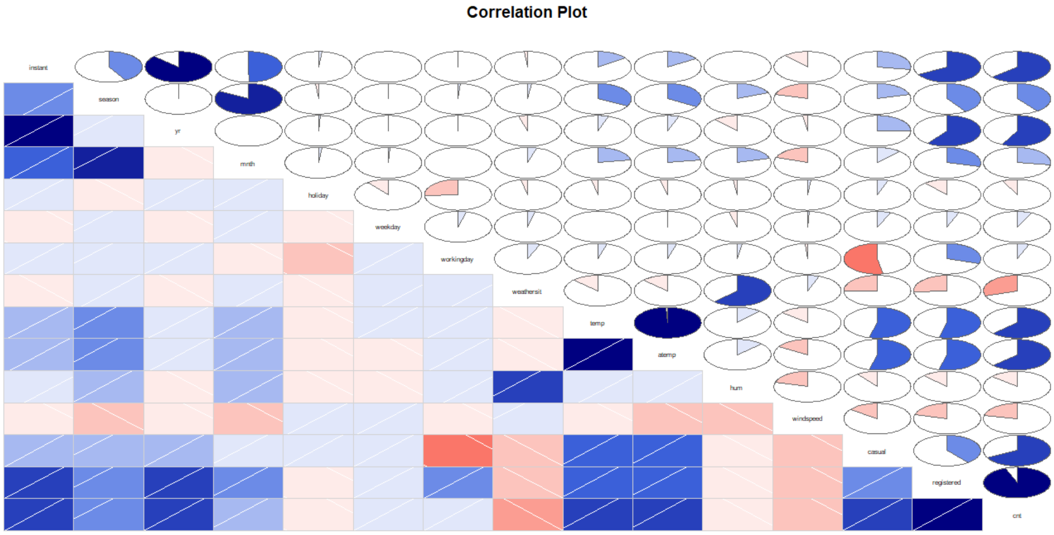
As we can see in boxplots, Humidity, windspeed and casual variables have outliers. As the number of outliers are low, we'll remove these outliers. Casual will not be treated in outlier removal as it is just a part of the target variable.

2.3 Missing Value Analysis

We will skip the missing value process as there is no missing value in the dataset after the removal of outliers in the above part.

2.4 Feature Selection

For better model development, we need to remove those independent variables which are highly correlated to other independent variables. We also need to remove those features which have no effect on our project.



According to this correlation plot, the variables temp and atemp are highly correlated. We removed atemp variable as there is no use of using it in model development.

Instant variable is removed because it is just index number and doesn’t represent anything. Casual and registered are removed because they are just a part of target variable. Dteday is also removed because it is a factor variable with not much significance.

2.5 Feature Scaling

We’ve scaled the whole data between 0 and 1 by using Normalization. Standard scaling was important in this project as it helped in turning data into same proportions which will further help in predicting the outcomes.

Chapter 3

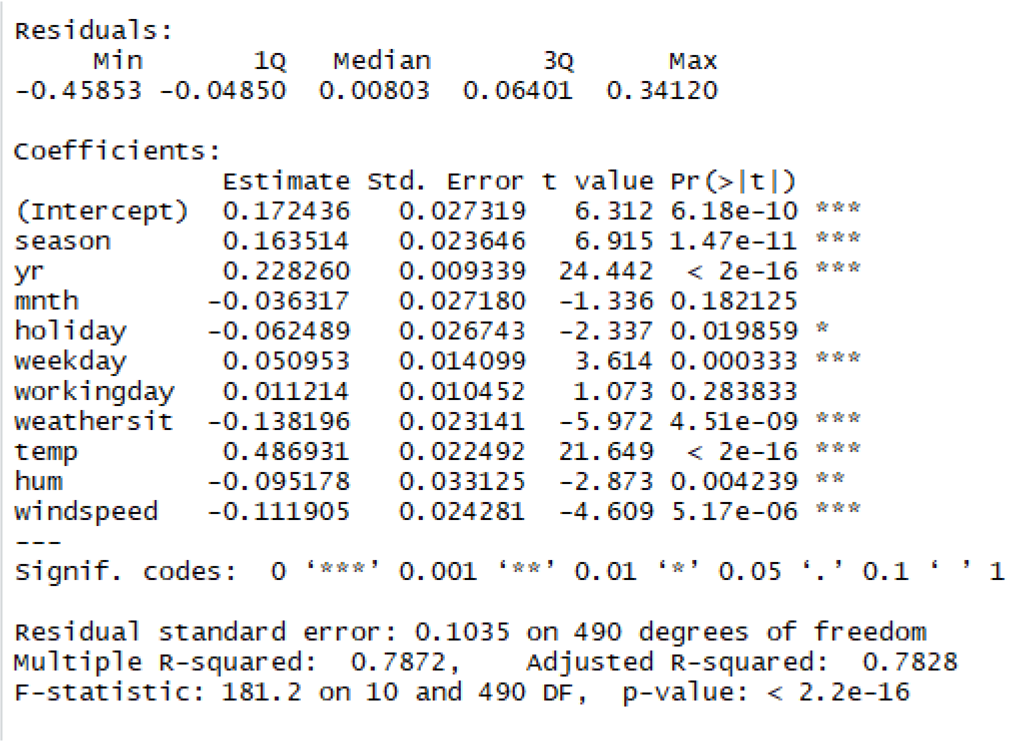
3. Model Development

3.1 Linear Regression

Model development starts with checking the variance inflation factor as it tells about multicollinearity. We checked and found that no variable out of 10 variables have collinearity problem.

After that we divided the data into train and test datasets. The data is chosen randomly and there is no order followed in dividing the data.

After training the data we find out these statistics



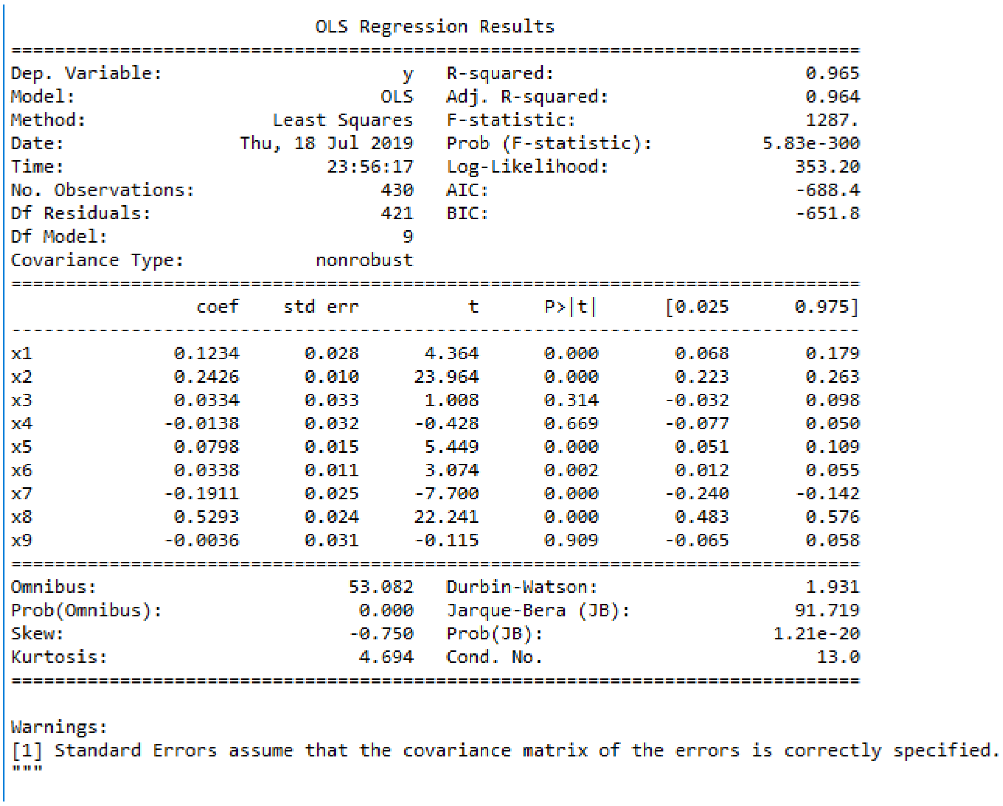
```
Residuals:
     Min       1Q   Median       3Q      Max 
-0.45853 -0.04850  0.00803  0.06401  0.34120 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept)  0.172436   0.027319   6.312 6.18e-10 ***
season       0.163514   0.023646   6.915 1.47e-11 ***
yr           0.228260   0.009339  24.442  < 2e-16 ***
mnth        -0.036317   0.027180  -1.336 0.182125    
holiday     -0.062489   0.026743  -2.337 0.019859 *  
weekday      0.050953   0.014099   3.614 0.000333 ***
workingday   0.011214   0.010452   1.073 0.283833    
weathersit  -0.138196   0.023141  -5.972 4.51e-09 ***
temp         0.486931   0.022492  21.649  < 2e-16 ***
hum         -0.095178   0.033125  -2.873 0.004239 ** 
windspeed   -0.111905   0.024281  -4.609 5.17e-06 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.1035 on 490 degrees of freedom
Multiple R-squared:  0.7872,	Adjusted R-squared:  0.7828 
F-statistic: 181.2 on 10 and 490 DF,  p-value: < 2.2e-16
```

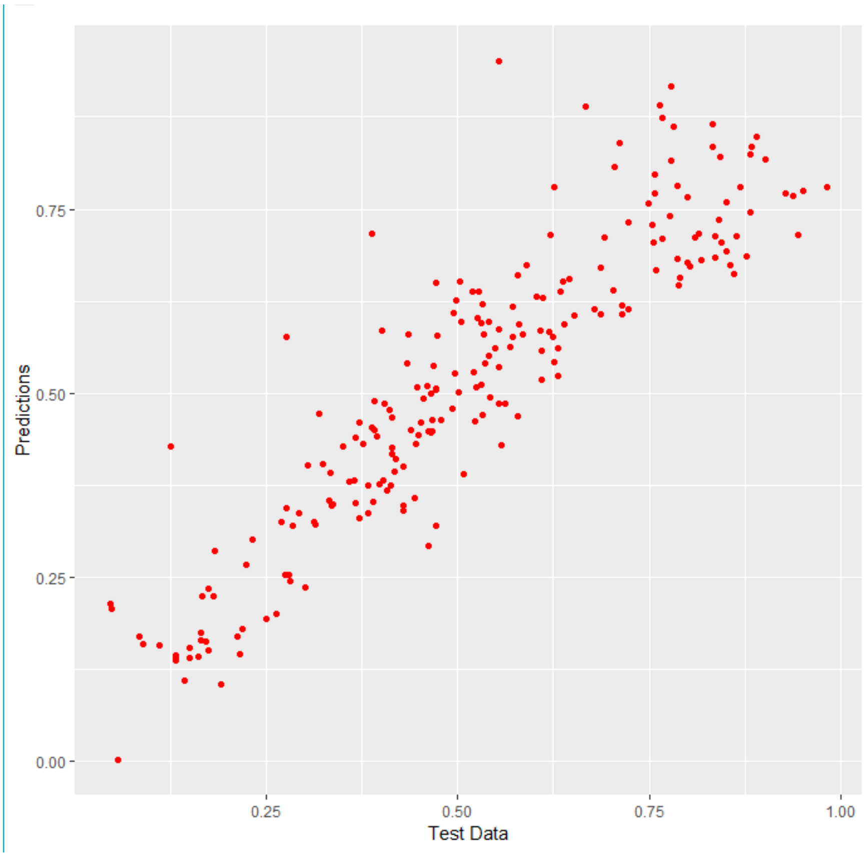
We can see that 78 percent of variation in our output is explained by our input variables. Also, the p-value is less than 0.05 which rejects our null hypothesis that there is no significance between count of bicycles and our input variables.

We used Ordinary Least Squared method in python and got R-squared and adjusted R-squared value of around 96 percent which is great. Our input variable can explain 96 percent of variation in our output variables.



```
                            OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.965
Model:                            OLS   Adj. R-squared:                  0.964
Method:                 Least Squares   F-statistic:                     1287.
Date:                Thu, 18 Jul 2019   Prob (F-statistic):          5.83e-300
Time:                        23:56:17   Log-Likelihood:                 353.20
No. Observations:                 430   AIC:                            -688.4
Df Residuals:                     421   BIC:                            -651.8
Df Model:                           9
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
x1             0.1234      0.028      4.364      0.000       0.068       0.179
x2             0.2426      0.010     23.964      0.000       0.223       0.263
x3             0.0334      0.033      1.008      0.314      -0.032       0.098
x4            -0.0138      0.032     -0.428      0.669      -0.077       0.050
x5             0.0798      0.015      5.449      0.000       0.051       0.109
x6             0.0338      0.011      3.074      0.002       0.012       0.055
x7            -0.1911      0.025     -7.700      0.000      -0.240      -0.142
x8             0.5293      0.024     22.241      0.000       0.483       0.576
x9            -0.0036      0.031     -0.115      0.909      -0.065       0.058
==============================================================================
Omnibus:                       53.082   Durbin-Watson:                   1.931
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               91.719
Skew:                          -0.750   Prob(JB):                     1.21e-20
Kurtosis:                       4.694   Cond. No.                         13.0
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
"""
```



At the end we used mean absolute percentage error method and find out that:

In R the accuracy of model is 80.73 percent.

In Python the accuracy of model is 80.77 percent.

3.2 Decision Tree

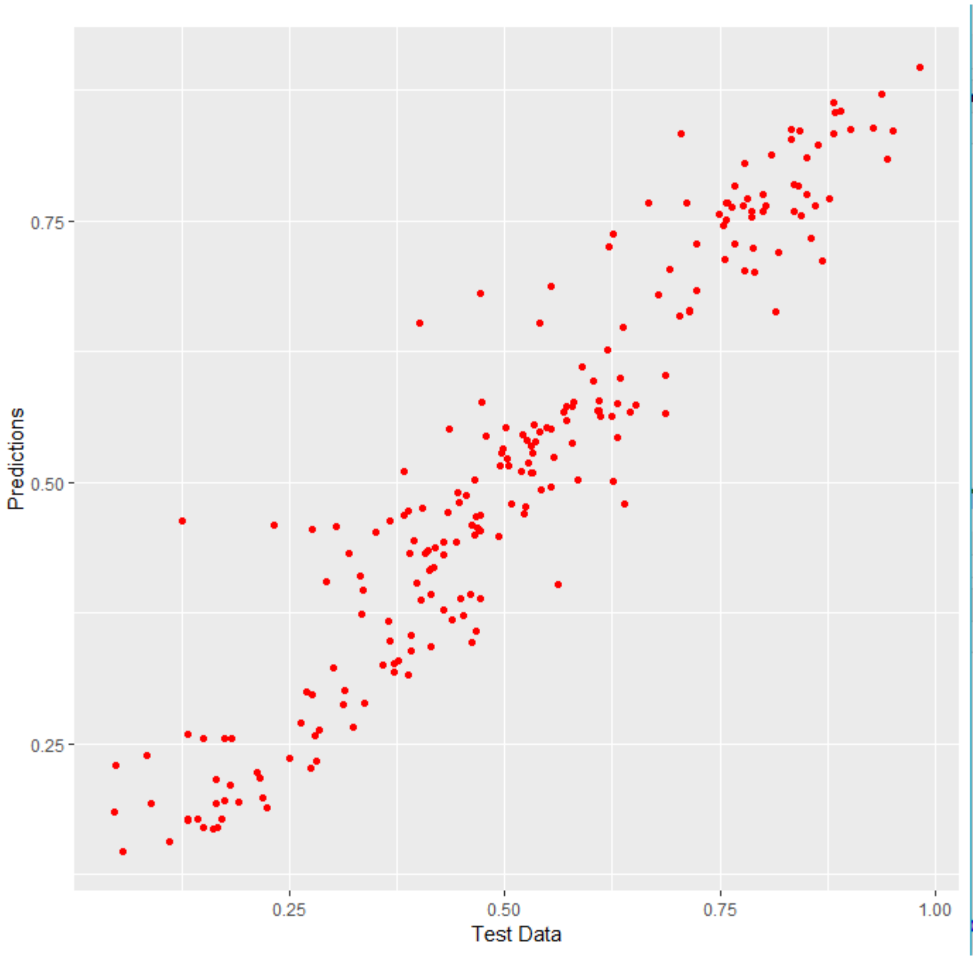
We used recursive partitioning and regression trees for model development in R. CART model is used via rpart. Our method is anova as we are using tree for regression. In Python we used Decision Tree Regressor which is a part of sklearn library.

In R the accuracy of Decision Tree model is 75.66 percent.

In Python the accuracy of Decision Tree model is 72.54 percent.

3.3 Random Forest

As we didn’t move ahead in accuracy, we tried Random Forest for a better prediction. Random Forest is a tree-based algorithm which involves building several decision trees and then combining their outputs to improve the model.



In R the accuracy of Random Forest model is 82.17 percent.

In Python the accuracy of Random Forest model is 84.56 percent.

3.4 Conclusion

We can see that Decision Tree didn’t performed very well as compared to linear regression model. Both Linear Regression model and Random Forest model performed well with Random Forest giving slight better performance. The model is able to predict nearly 85 percent of data correctly.