Bike Rent

Deepak Jaiswal
17/12/2019

Contents

1. INTRODUCTION

1.1 Problem statement

The objective of this Case is to predict the bike rental count based on the environmental and seasonal settings, So that required bikes would be arranged and managed by the shops according to environmental and seasonal conditions.

1.2 Data Set

Our task is to build regression models which will predict the count of bike rented depending on various environmental and seasonal conditions. Given below is a sample of the data set that we are using to predict the count of bike rents:

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

instant	dteday	season	yr	mnth	holiday	weekday	workingday
1	1/1/2011	1	0	1	0	6	0
2	1/2/2011	1	0	1	0	0	0
3	1/3/2011	1	0	1	0	1	1
4	1/4/2011	1	0	1	0	2	1
5	1/5/2011	1	0	1	0	3	1
6	1/6/2011	1	0	1	0	4	1

Table 1.2: Sample Data (Columns: 7-16)

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	0.0895652	88	1518	1606

Variables present in given dataset are instant, dteday, season, yr, mnth, holiday, weekday, workingday, weathersit, temp, atemp, hum, windspeed, casual, registered, cnt

The details of variable present in the dataset are as follows - 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)

hr: Hour (0 to 23)

holiday: weather day is holiday or not (extracted fromHoliday Schedule) weekday:

Day of the week

workingday: If day is neither weekend nor holiday is 1, otherwise is 0.

weathersit: (extracted fromFreemeteo)

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

2. Methodology

2.1 Pre Processing

Any predictive modeling requires that we look at the data before we start modeling.However, in data mining terms *looking at data* refers to so much more than just looking. Looking at data refers to exploring the data, cleaning the data as well as visualizing the data through graphs and plots. This is often called as Exploratory Data Analysis.

2.1.1 Exploratory Data Analysis

In exploring the data we have

- Converted season, mnth, workingday, weathersit into categorical variables
- Feature Engineering : Changed deday variables's date value to day of date and converted to categorical variable having 31 levels as a month has 31 days.
- Deleted instant variable as it is nothing but an index.
- Omitted registered and casual variable as sum of registered and casual is the total count that is what we have to predict.

2.1.2 Missing Value Analysis

Missing value analysis is done to check is there any missing value present in given dataset. Missing values can be easily treated using various methods like mean, median method, knn method to impute missing value.

In R -**function(x) {sum(is.na(x))}** is the function used to check the sum of missing values.
In python - **bike_train.isnull().sum()** is used to detect any missing value

dteday	0
season	0
yr	0
mnth	0
holiday	0
weekday	0
workingday	0
weathersit	0
temp	0
hum	0
windspeed	0
cnt	0

There is no missing value found in given dataset.

2.1.3 Outlier Analysis

Outlier analysis is done to handle all inconsistent observations present in given dataset. As outlier analysis can only be done on continuousvariable.
Figure 2.1 and 2.2 are visualization of numeric variable present in our dataset to detect outliers using boxplot. Outliers will be detected with redcolor

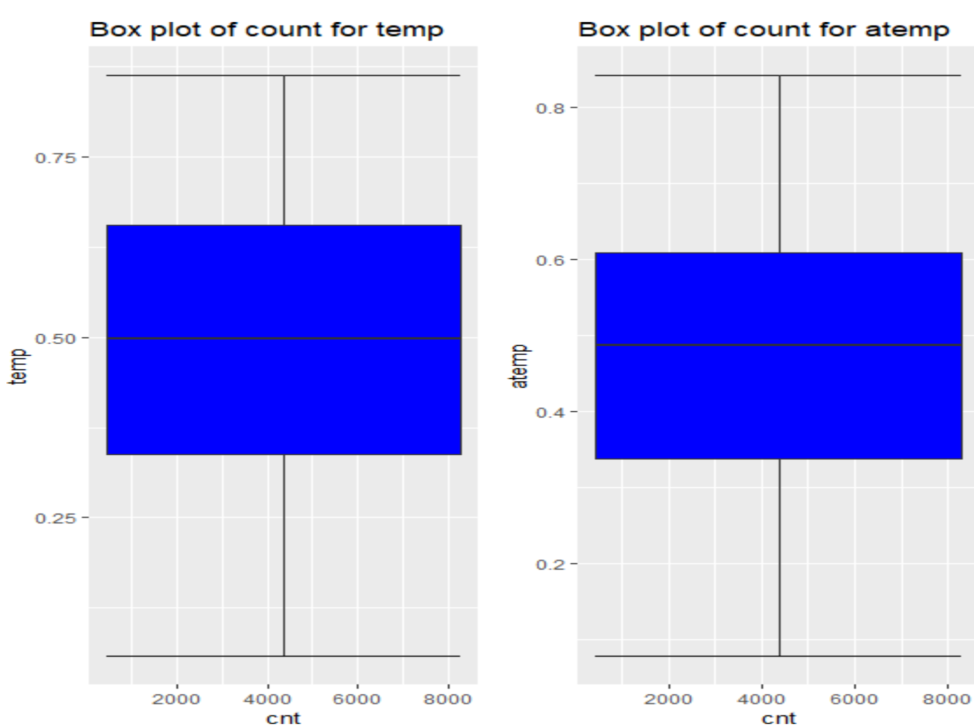


Figure 2.1 Boxplot graph of temp and atemp variables

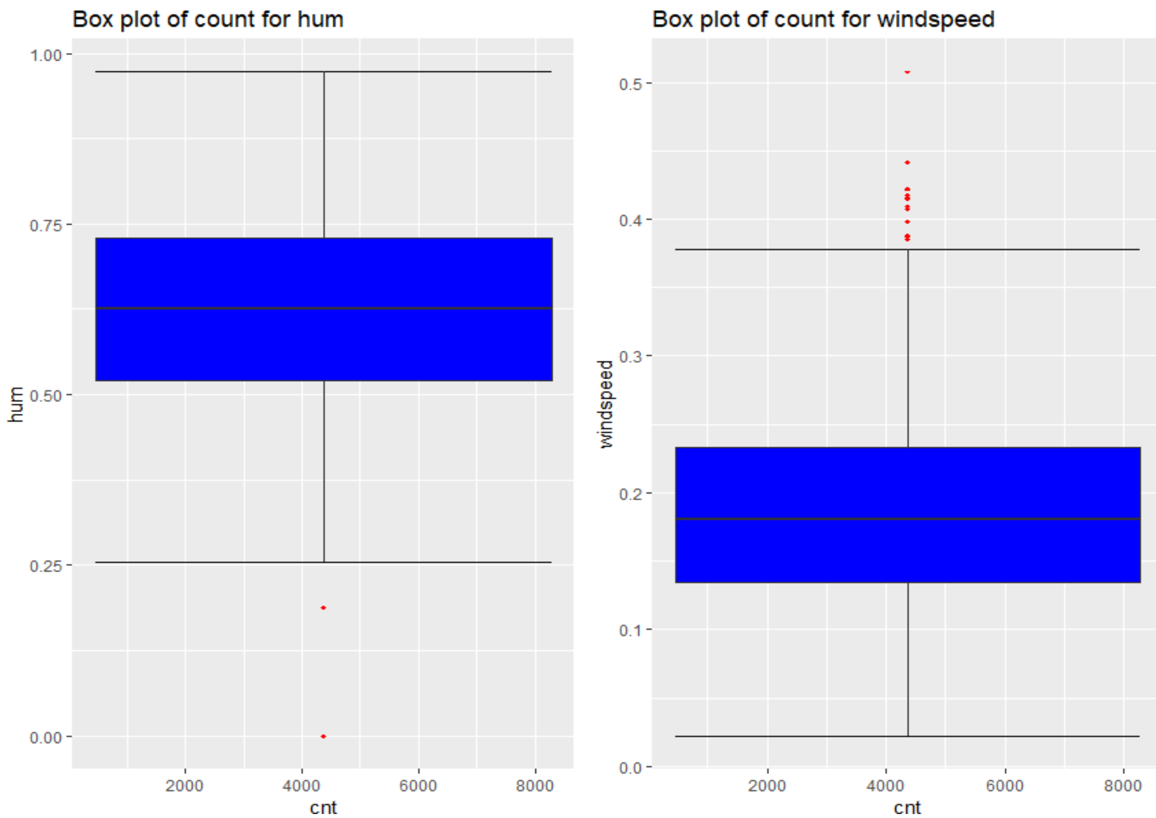


Figure 2.2 Boxplot graph of hum and windspeed variables

According to above visualizations there is no outlier found in temp and atemp variable but there are few outliers found in windspeed and hum variable.
As windspeed variable defines the windspeed on a particular day and hum defines the humidity of that day so we can neglect these outliers because both these variable define environmental condition. Due to drastic change in weather like strome, heavy rain condition.

2.1.4 Feature Selection

Feature selection analysis is done to Select subsets of relevant features (variables, predictors) to be in model construction.
As our target variable is continuous so we can only go for correlation check. As chi-square test is only for categorical variable.

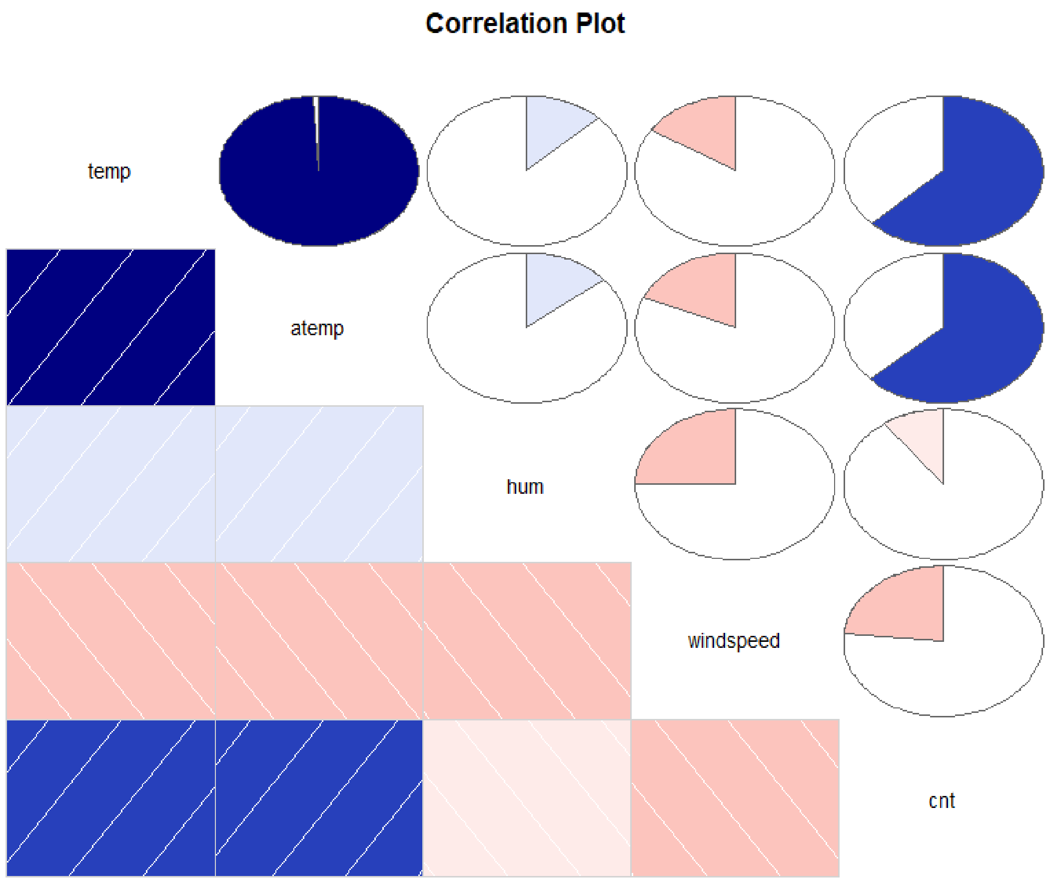


Figure 2.4 correlation plot

In above visualization we can see that only 2 variables are highly correlated with each other. Dark blue color represent highly correlated and light color represent very less correlated so we have a choice to remove either temp or atemp because these variables contains nearly equal information.
So I have removed atemp variable from dataset.

2.1.4 Feature Scaling

Feature scaling includes two functions normalization and standardization. It is done reduce unwanted variation either within or between variables and to bring all of the variables into proportion with one another.
In given dataset all numeric values are already present in normalized form.

2.2 Modeling

2.2.1 Model Selection

In this case we have to predict the count of bike renting according to environmental and seasonal condition. So the target variable here is a continuous variable. For Continuous we can use various Regression models. Model having less error rate and more accuracy will be our final model.

Models built are

1. c50 (Decision tree for regression target variable)
2. Random Forest (with 200 trees)
3. Linear regression

2.2.2 C50

This model is also known a Decision tree for regression target variable.
For this model we have divided the dataset into train and test part using random sampling. Where train contains 80% data of data set and test contains 20% data and contains 12 variable where 12th variable is the target variable.

Creating Model In
R

```
> ###########Decision tree regression  ######
> fit = rpart(cnt ~ ., data = train, method = "anova")
> predictions_DT = predict(fit, test[,-12])
>
```

In python

```
######c50#######
fit_DT = DecisionTreeRegressor(max_depth=2).fit(train.iloc[:,0:11], train.iloc[:,11])
predictions_DT = fit_DT.predict(test.iloc[:,0:11])
```

2.2.3 Random Forest

In Random forest we have divided the dataset into train and test part using random sampling. For this model we have divided the dataset into train and test part using random sampling Where train contains 80% data of data set and test contains 20% data and contains 12 variable where 12th variable is the target variable.

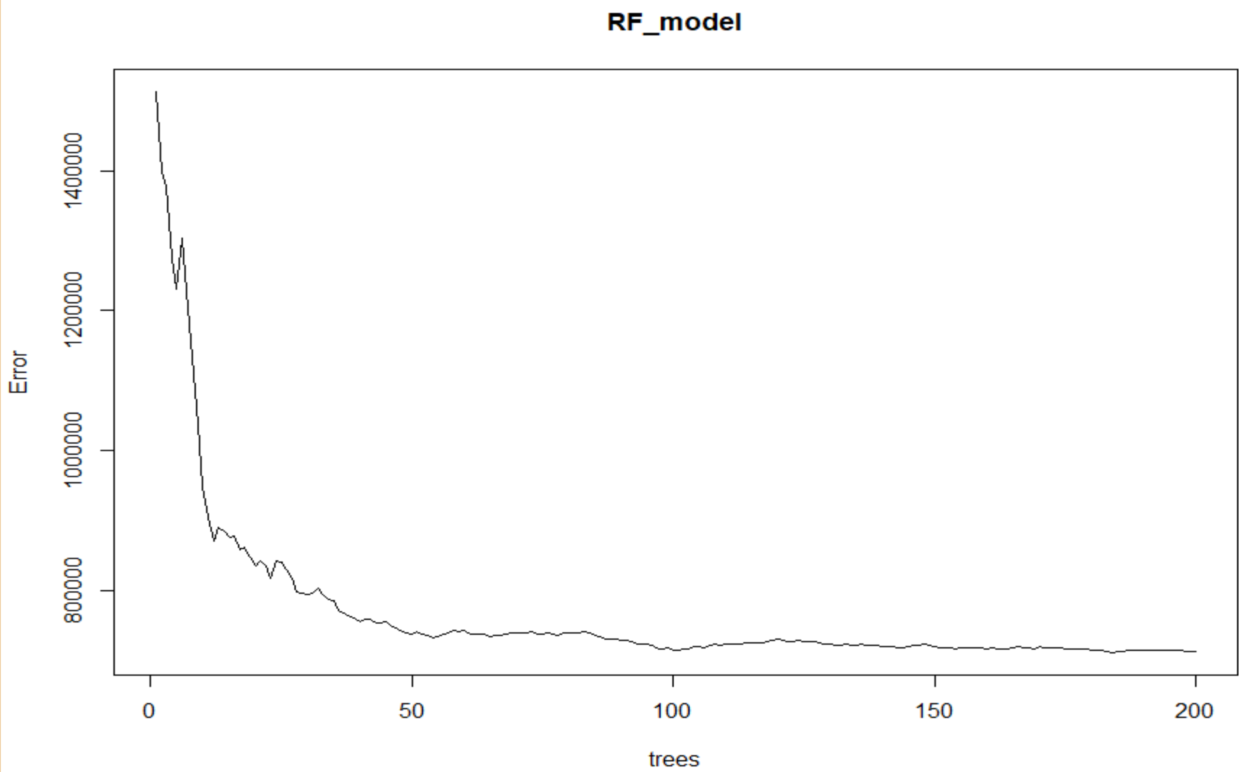


Figure2.2.3

Above Figure2.2.3 represents the curve of error rate as the number of trees increases. After 200 trees the error rate reaches to beconstant.

In this model we are using 200 trees to predict the target variable.

Creating Model In
Python

```
#random forest
RFmodel = RandomForestRegressor(n_estimators = 200).fit(train.iloc[:,0:11], train.iloc[:,11])
RF_Predictions = RFmodel.predict(test.iloc[:,0:11])
```

In R

```
> #############Random Forest Model##########################
> RF_model = randomForest(cnt ~ ., train, importance = TRUE, ntree = 200)
> predictions_RF = predict(RF_model, test[,-12])
```

2.2.4 Linear Regression

For linear regression model we have divided the categorical containing more than 2 classes into dummy variable. So that all categorical variable should be in binary classes form. On creating dummy variable there are 64 variable in both R and Python. Where 64^{th} is the target variable.

Further the data is again divided into train and test with 80 % train data and 20 % test data using random sampling.
Creating Model In
R

```
> #model making
> lm_model = lm(cnt ~., data = train_lr)
> predictions_LR = predict(lm_model,test_lr[,-64])
```

In python

```
trainlr, testlr = train_test_split(data_lr, test_size=0.2)
model = sm.OLS(trainlr.iloc[:,63], trainlr.iloc[:,0:63]).fit()
predictions_LR = model.predict(testlr.iloc[:,0:63])
```

Model summary
Call:
lm(formula = cnt ~ ., data = train_lr)

Residuals:

Min	1Q	Median	3Q	Max
-2770.29	-387.31	54.31	431.11	2036.17

Coefficients: (6 not defined because of singularities)

	Estimate	Std. Error	t value	Pr(>|t|)	
(Intercept)	1639.102	436.406	3.756	0.000192	***
dteday_01	-340.952	290.187	-1.175	0.240551	
dteday_02	-322.218	281.341	-1.145	0.252609	
dteday_03	-135.178	290.237	-0.466	0.641586	
dteday_04	-222.522	278.793	-0.798	0.425138	
dteday_05	-323.017	287.728	-1.123	0.262100	
dteday_06	-139.706	273.329	-0.511	0.609475	
dteday_07	-390.488	286.884	-1.361	0.174054	
dteday_08	-266.207	281.082	-0.947	0.344033	
dteday_09	-308.336	274.417	-1.124	0.261694	
dteday_10	-152.902	278.304	-0.549	0.582960	
dteday_11	-153.288	279.017	-0.549	0.582974	
dteday_12	-299.836	280.427	-1.069	0.285464	
dteday_13	-241.322	276.954	-0.871	0.383964	
dteday_14	-309.660	280.062	-1.106	0.269368	
dteday_15	-9.398	275.923	-0.034	0.972841	
dteday_16	-141.620	284.685	-0.497	0.619072	
dteday_17	-158.191	290.124	-0.545	0.585810	
dteday_18	-428.554	279.247	-1.535	0.125464	
dteday_19	-49.009	290.536	-0.169	0.866111	
dteday_20	5.073	278.903	0.018	0.985496	
dteday_21	-186.402	274.066	-0.680	0.496718	
dteday_22	-623.919	275.694	-2.263	0.024037	*
dteday_23	-313.725	280.278	-1.119	0.263508	

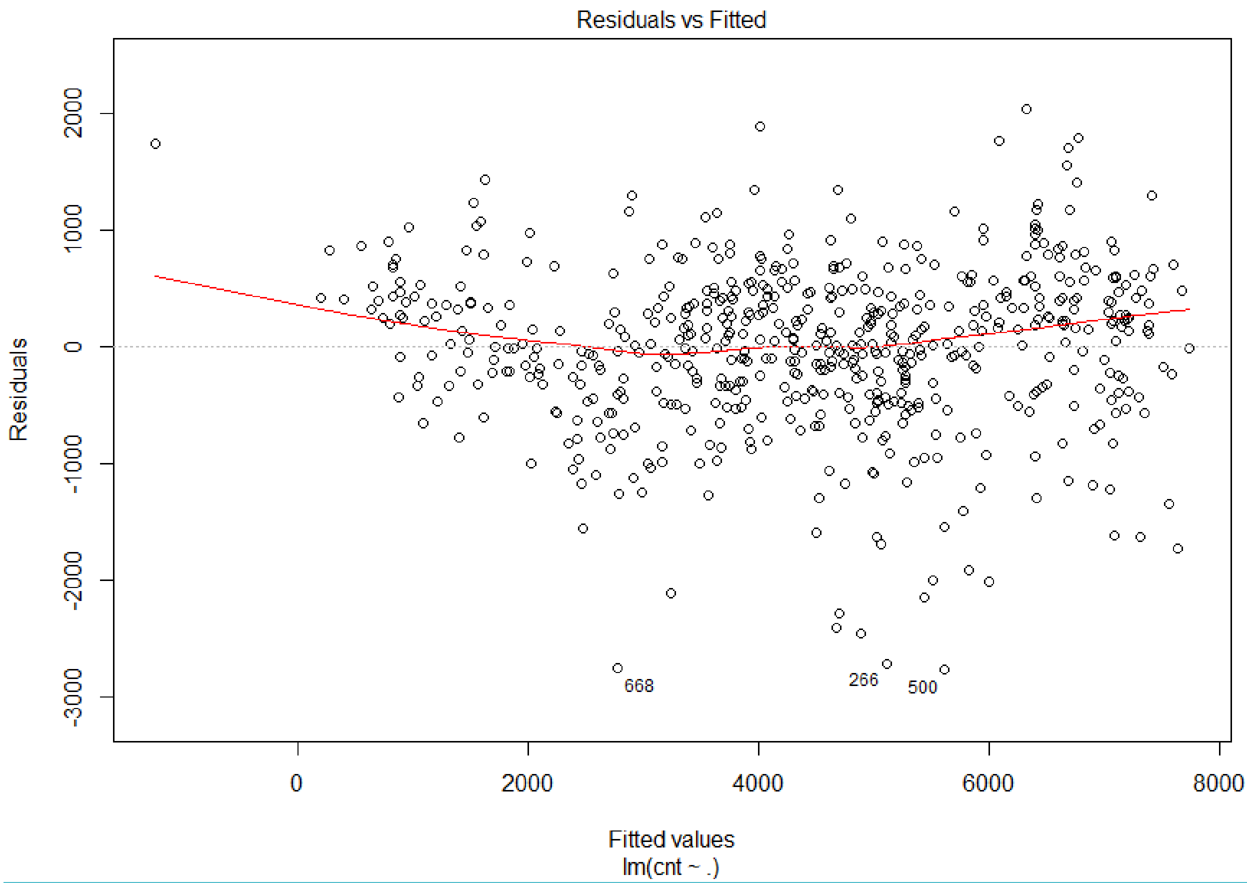
dteday_24	-619.680	278.453	-2.225	0.026475	*
dteday_25	-349.580	292.075	-1.197	0.231890	
dteday_26	-272.698	279.281	-0.976	0.329300	
dteday_27	-519.036	281.177	-1.846	0.065463	.
dteday_28	-491.710	277.258	-1.773	0.076729	.
dteday_29	-1015.756	287.422	-3.534	0.000445	***

dteday_30	-303.037	281.340	-1.077	0.281921	
dteday_31	NA	NA	NA	NA	
season_1	-1567.752	190.689	-8.222	1.58e-15	***
season_2	-617.154	226.081	-2.730	0.006550	**
season_3	-672.991	196.398	-3.427	0.000659	***
season_4	NA	NA	NA	NA	
mnth_1	5.487	191.308	0.029	0.977129	
mnth_2	161.340	193.242	0.835	0.404146	
mnth_3	544.971	198.260	2.749	0.006188	**
mnth_4	347.533	260.233	1.335	0.182300	
mnth_5	652.997	274.921	2.375	0.017896	*
mnth_6	329.008	281.124	1.170	0.242398	
mnth_7	-153.445	296.283	-0.518	0.604745	
mnth_8	288.104	281.614	1.023	0.306756	
mnth_9	960.011	224.978	4.267	2.35e-05	***
mnth_10	602.777	176.477	3.416	0.000686	***
mnth_11	-77.503	160.151	-0.484	0.628634	
mnth_12	NA	NA	NA	NA	
weekday_6	-17.625	116.183	-0.152	0.879485	
weekday_0	-463.112	118.205	-3.918	0.000101	***
weekday_1	-296.695	114.012	-2.602	0.009521	**
weekday_2	-150.171	112.824	-1.331	0.183759	
weekday_3	-52.437	115.545	-0.454	0.650145	
weekday_4	-161.957	115.105	-1.407	0.160006	
weekday_5	NA	NA	NA	NA	
weathersit_2	1550.221	195.182	7.942	1.21e-14	***
weathersit_1	2047.928	207.080	9.890	< 2e-16	***
weathersit_3	NA	NA	NA	NA	
yr1	2000.192	62.438	32.035	< 2e-16	***
holiday1	-755.129	209.588	-3.603	0.000345	***
workingday1	NA	NA	NA	NA	
temp	4661.163	447.522	10.416	< 2e-16	***
hum	-1389.554	315.234	-4.408	1.27e-05	***
windspeed	-2731.396	435.368	-6.274	7.37e-10	***

Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 728.6 on 526 degrees of freedom Multiple R-squared: 0.8717, Adjusted R-squared: 0.8578
F-statistic: 62.72 on 57 and 526 DF, p-value: < 2.2e-16

Visualization of Linear regression model



In above figure red line represent the predicted values and small circle are actual values

3. Conclusion

3.1 Model Evaluation

Now that we have a few models for predicting the target variable, we need to decide which one to choose. There are several criteria that exist for evaluating and comparing models. We can compare the models using any of the following criteria:

1. Predictive Performance
2. Interpretability
3. Computational Efficiency

In our case of Bike Renting, the latter two, *Interpretability* and *Computation Efficiency*, do not hold much significance. Therefore we will use *Predictive performance* as the criteria to compare and evaluate models.

Predictive performance can be measured by comparing Predictions of the models with real values of the target variables, and calculating some average error measure.

3.1.1 Mean Absolute Percentage Error (MAPE)

MAPE is one of the error measures used to calculate the predictive performance of the model. We will apply this measure to our models that we have generated in the previous sections

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

In above function y_true is the actual value and y_pred is the predicted value. It will provide the error percentage of model.

MAPE value in Python are as follow

```
#MAPE for decision tree regression
MAPE(test.iloc[:,11], predictions_DT)
```

```
27.737837701228408
```

```
#MAPE for random forest regression
MAPE(test.iloc[:,11],RF_Predictions)
```

```
14.923072236915019
```

```
#MAPE for linear regression
MAPE(testlr.iloc[:,63], predictions_LR)
```

```
18.137949688224342
```

MAPE values is R are as follow

```
> MAPE(test[,12], predictions_DT)
[1] 19.35408
>
> MAPE(test[,12], predictions_RF)
[1] 17.36805
>
> MAPE(test_lr[,64],  predictions_LR)
[1] 19.24258
>
```

Where predictions_DT are predicted values from C50 model. predictions_RF are predicted values from random forest model predictions_LR are predicted values from linear regression model

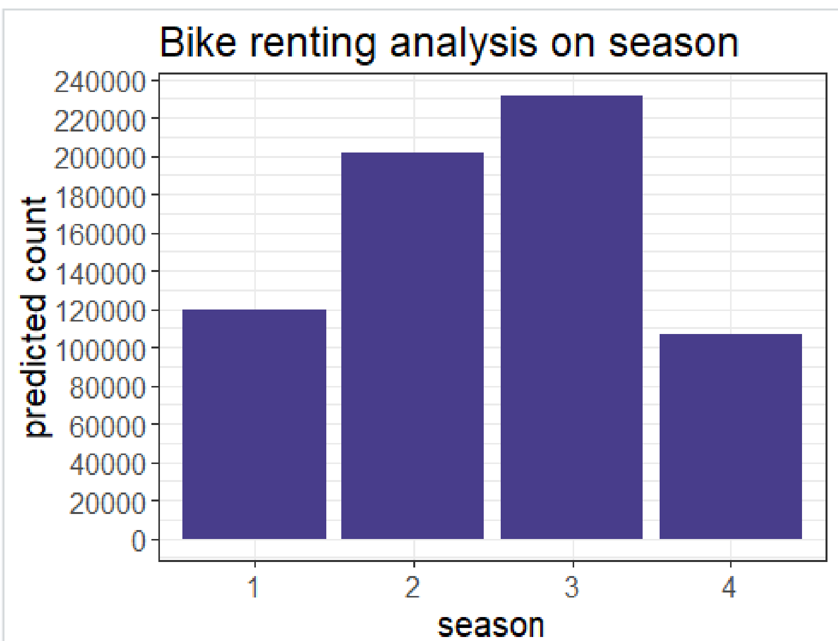
3.2 Model Selection

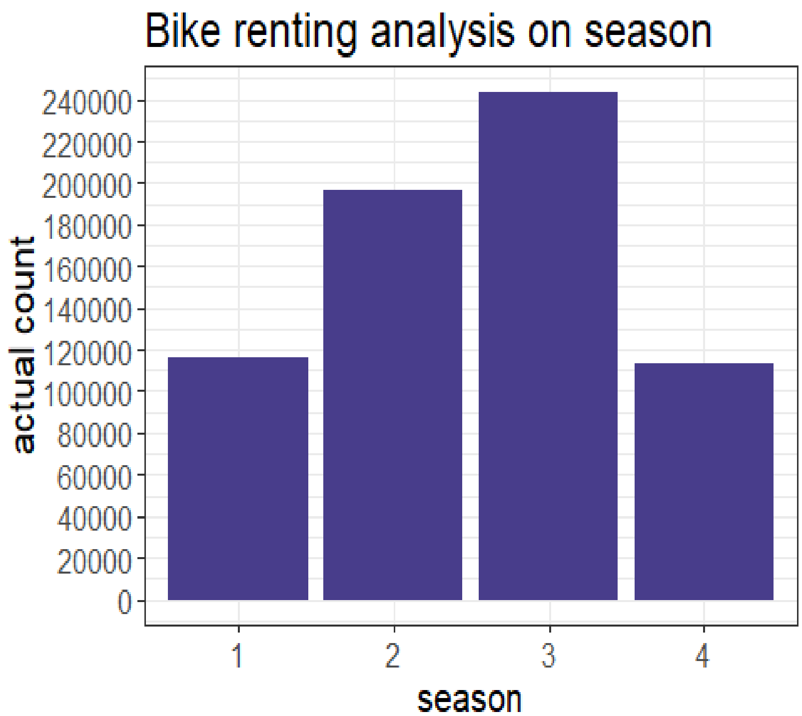
We can see that from both R and Python Random forest model performs best out of c50 and linear regression. So random forest model is selected with 83% accuracy in R and with 85% accuracy in python.

Extracted predicted value of random forest model are saved with .csv file format.

4. Visualizations

4.1 Visualization on result stored on seasonal settings

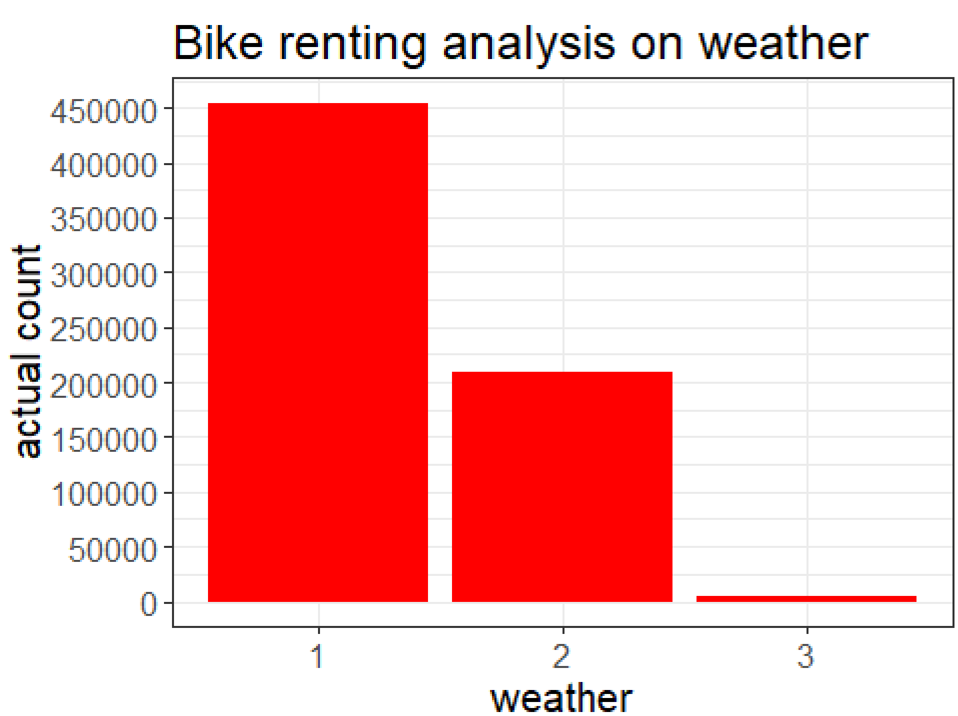


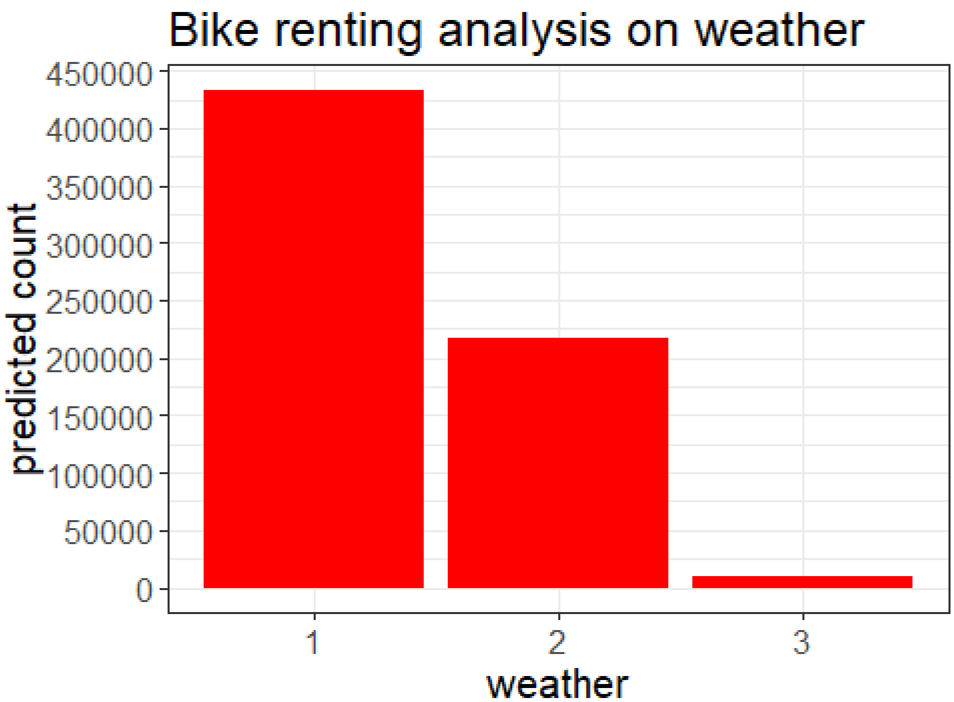


season: Season (1:springer, 2:summer, 3:fall, 4:winter)

Above two bar graph represents the comparison of predicted count value and actual count value based on seasonal condition.

4.2 Visualization on result stored on weather conditions





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

Above bar graph shows predicted count and actual count based on weather conditions

According to Seasonal and weather condition bar graph we can clearly notice that fall season that is autumn and where weather conditions are clear, few or partly cloudy on these conditions bike rent count is quite high than any other condition.