**PROJECT 1**

**Prediction of Bike Rental**

**Rishabh Jain**

**Data Analyst, Home Credit India Pvt. Ltd.**

**Content**

1. Introduction
   1. Problem Statement
   2. Data
2. Methodology
   1. Data Checks and Modification
   2. Data Distribution
   3. Relationship between Independent and Dependent Variable
   4. Outlier Analysis
   5. Feature Selection
   6. Final Input DataFrame
3. Modelling
   1. Sampling
   2. Multivariate Linear Regression
   3. Decision Tree
   4. Random Forest
   5. XGBoost with Python
4. Conclusion
5. **Introduction**
   1. Problem Statement

The objective of this Case is to Prediction of bike rental count on daily based on the environmental and seasonal settings. The details of data attributes 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\_max- t\_min), t\_min=-16, t\_max=+50 (only in hourly scale)

hum: Normalized humidity.

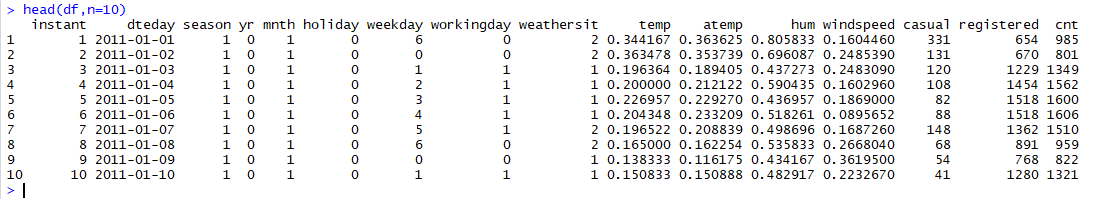
The values are divided to 100 (max)

windspeed: Normalized wind speed. The values are divided to 67 (max) casual: count of casual users.

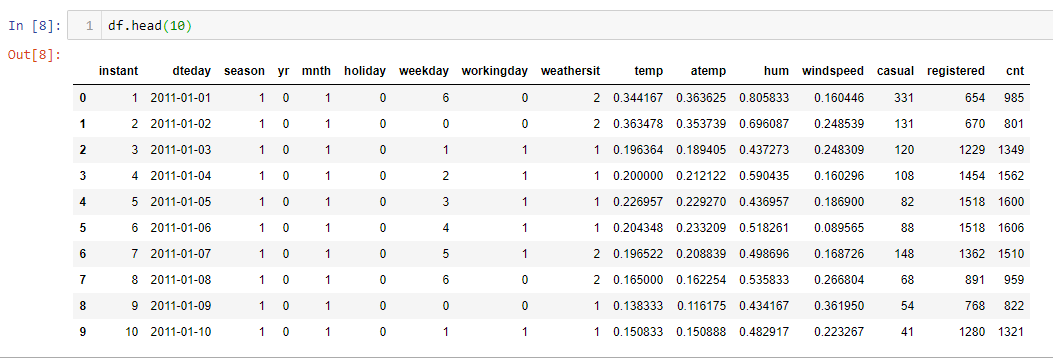
* 1. Data

The day.csv file is used for the analysis and modelling. To view some rows of the data, the following code is used:

# R



# Python



The data contain 16 variables or attributes as mentioned in the problem statement. Out of these variables, “cnt” variable is target variable.

1. **Methodology**
   1. Data Checks and Modification

To initialize the process of data modelling, certain checks are performed so that the model doesn’t get biased and predict wrong value of target variable for the certain input variables.

First check is check for NULL values in the dataset. If so, then certain procedures are to be used to fill that NULL values, for example

1. Removing NULL value containing row
2. Mean method i.e., filling NULL value with the mean for the values in that column
3. Median method i.e., fiulling NULL values with median value
4. KNN Imputation (K-Nearest Neighbour mehtod)

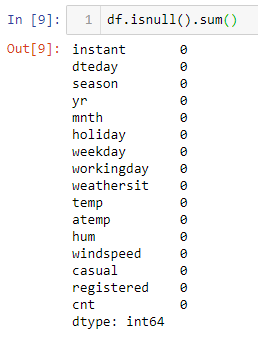
Below code is used in R and Python to check for NULL values:

# R



This means there is no NULL value in dataframe. Analysis can be continued on this data frame.

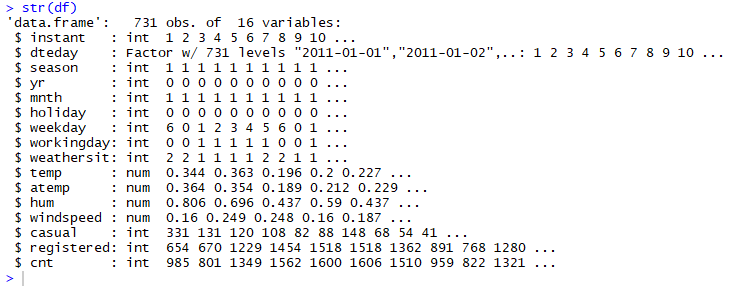
# Python



The above is used in python to check for NULL values in each column of the dataframe.

Second check is to check for the datatypes of the variables in the loaded dataframe. Below code in R and Python gives a feel how to do it in both languages.

# R

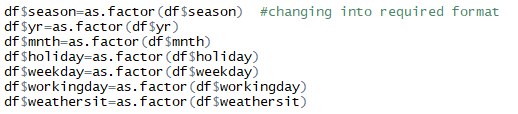


The independent variable “instant” is of no use as it is just representing the index number for the dataset so drop the column by using the following code:

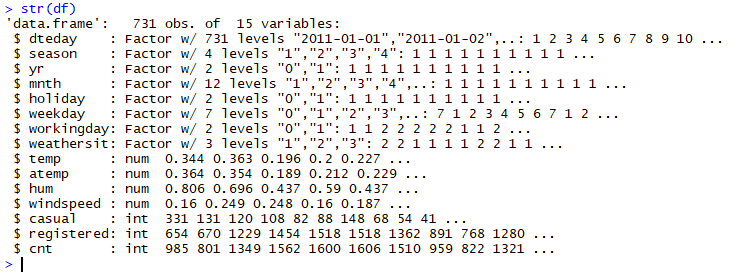
df = subset(df, select = -c(instant))

Now, looking at the variable type and values in the variables from the problem statement, the data type for certain variables are changed to correct type.

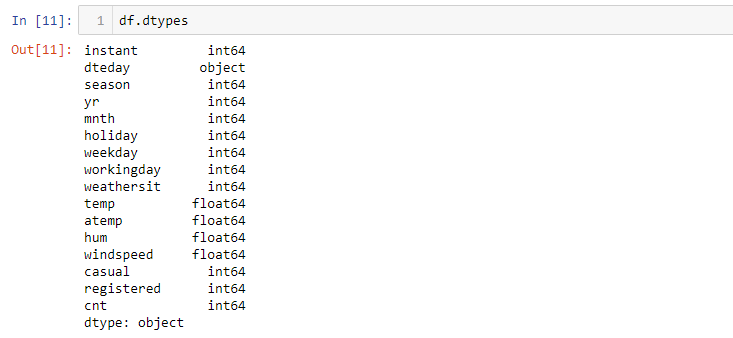
“season”, “yr”, “mnth”, “holiday”, “weekday”, “workingday”,”weathersit” are converted to categorical variable(or factors in R) from integer data type using the below mentioned code:



Now, the datatypes look like in this way,



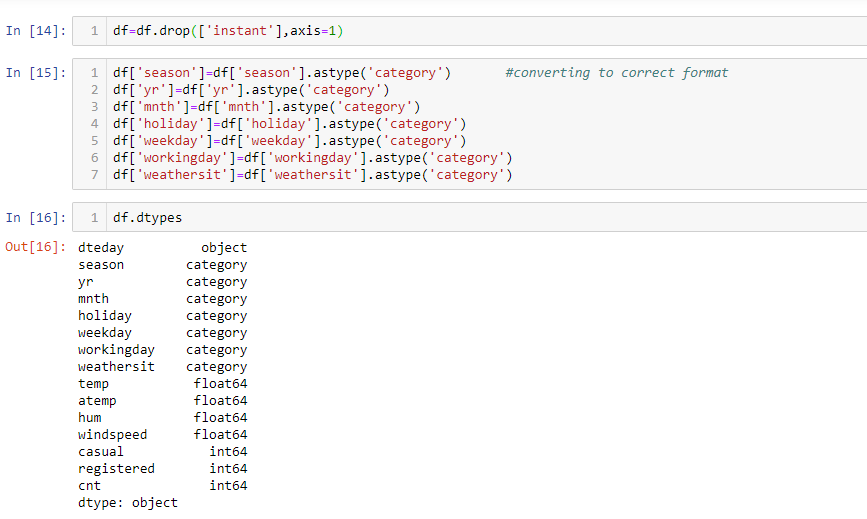
# Python



Just like in R, the similar process in python looks like this:

df=df.drop([‘instant’], axis=1)

Then, converting the data types will be done as below:



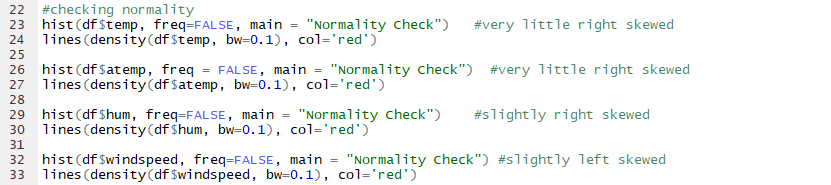
* 1. Data Distribution

The model is greatly impacted by the data distribution because on the basis of that the model makes rules to predict the target variable. The data distribution for continuous variable should be normalised to develop a better model. So to have a look at the distribution of the data, Histograms are used with density curves to know how the data is distributed among the dataframe.

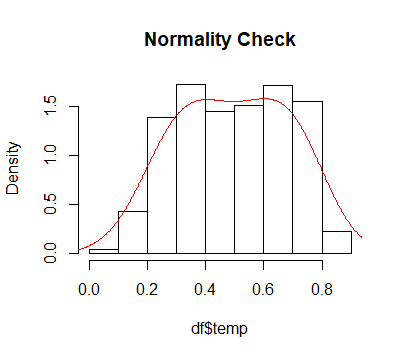
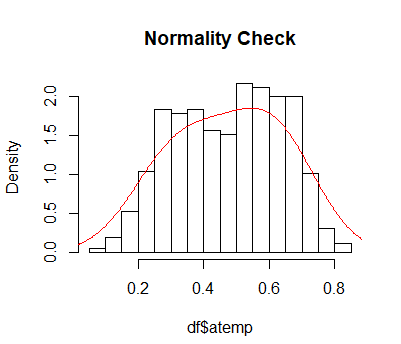
For Continuous variables: “temp”, “atemp”, “hum” & “windspeed”

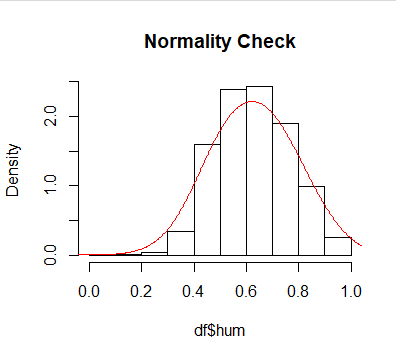
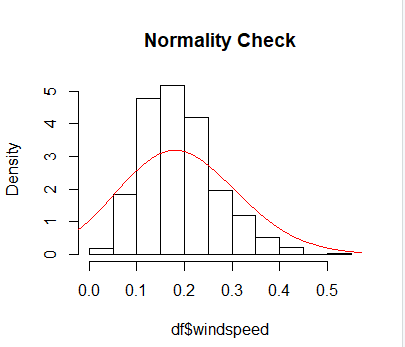
Below are the graph and plot produced in R studio and Jupyter notebook for visualization:

# R



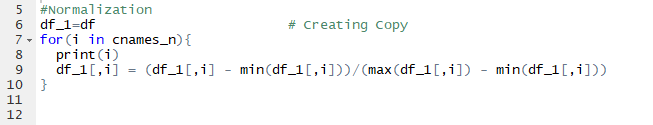
The plots are given on the next page:

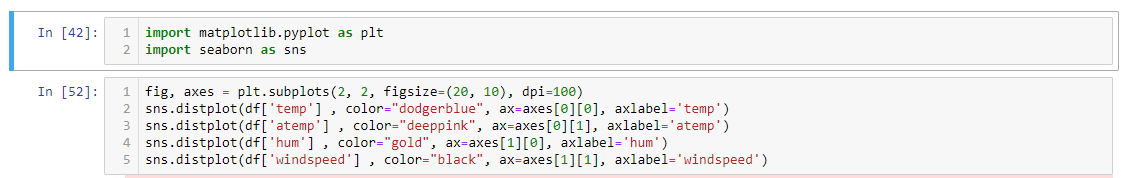
 

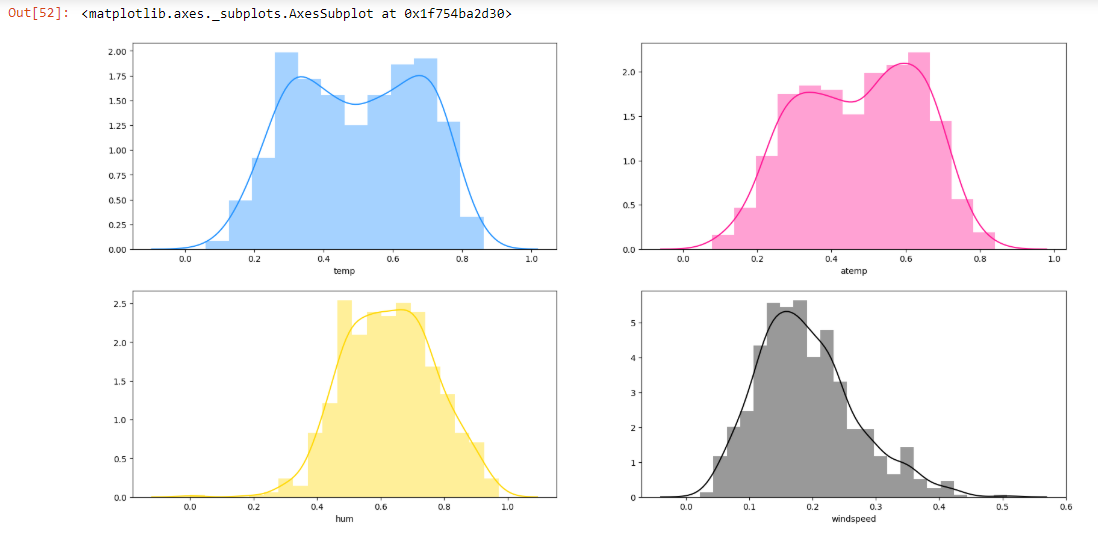
From above two things can be inferred:

1. The variables “temp” and “atemp” are almost normalized with very little skewness towards the right.
2. For “hum” and “windspeed”, the data is slightly right skewed and left skewed respectively. But the data is already being normalised so the code to normalise the data would return same results.



# Python

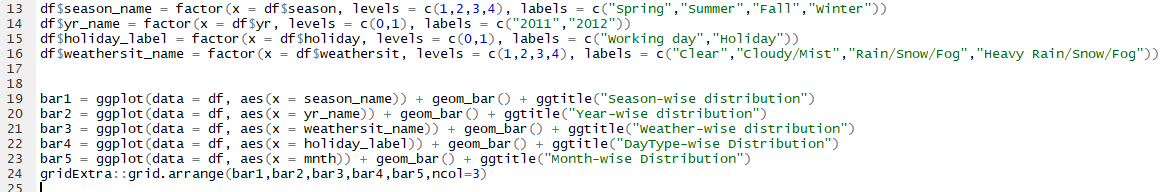




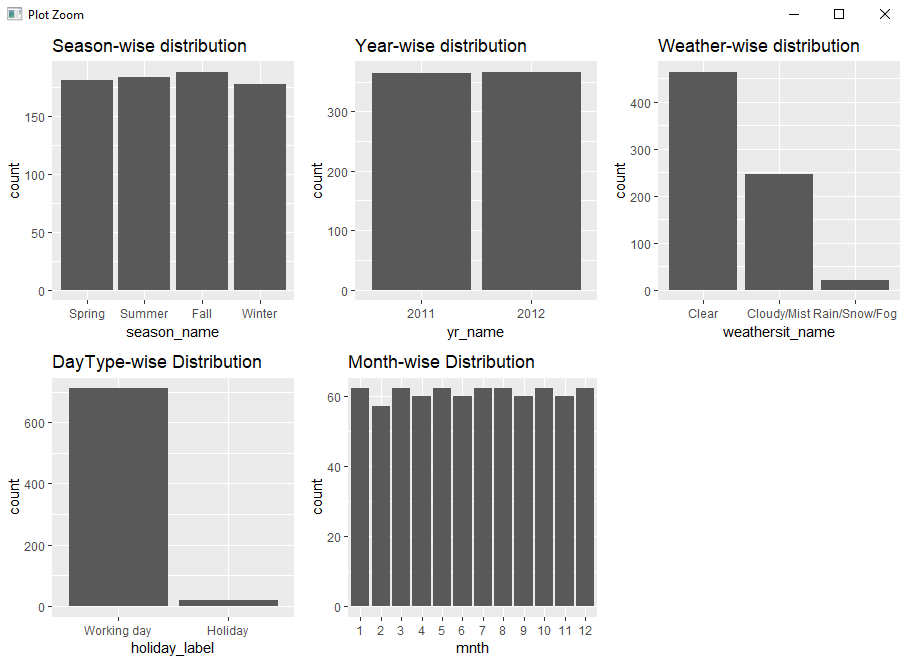
For Categorical Variables: “season”, “yr”, “mnth”, “weekday” & “holiday”

Below are the graphs and code to visualize the data distribution for each category in the data column:

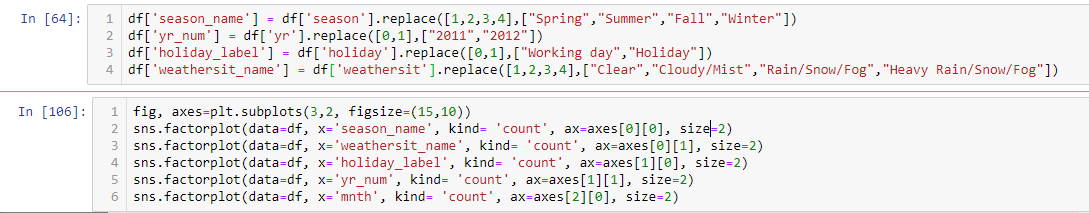
# R

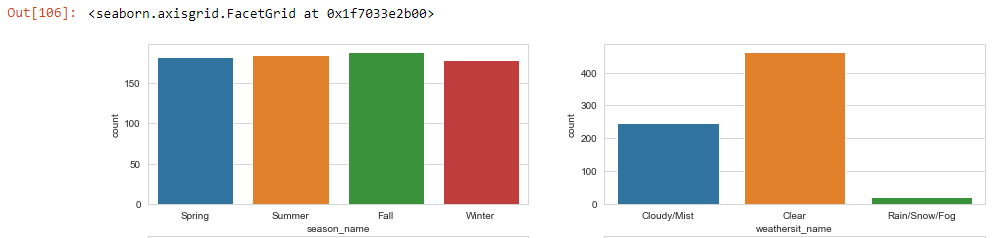


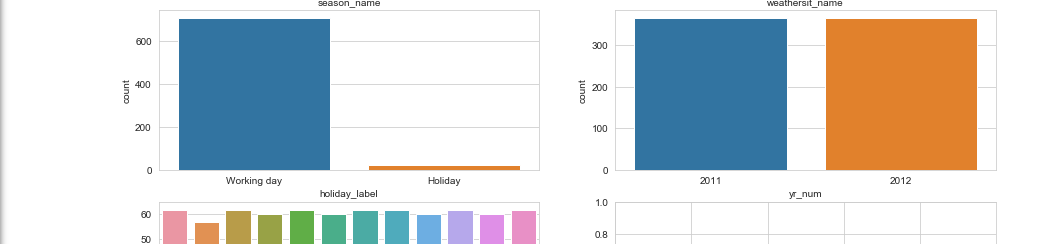
The plots are given on the next page.

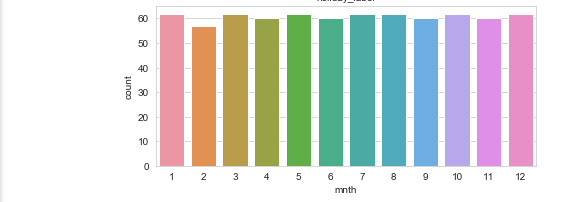


# Python

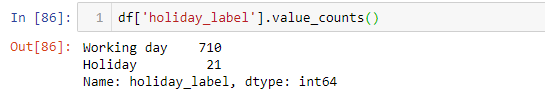








From the above graphs, it is clear that over the month, year and season the data is equally distributed. But for the clear weather the number of bike rentals are more than the other weather conditions and for Rainy weather there is very less data points which seems logical also. Also, on holidays there is very less number of bike rentals (shown in the below image) which is because people travel through bigger transport medium like bus, car and train most probably rather than travelling through bike/bicycle.



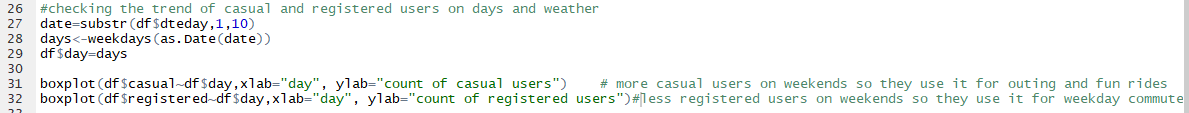
* 1. Relationship between Independent and Dependent Variable

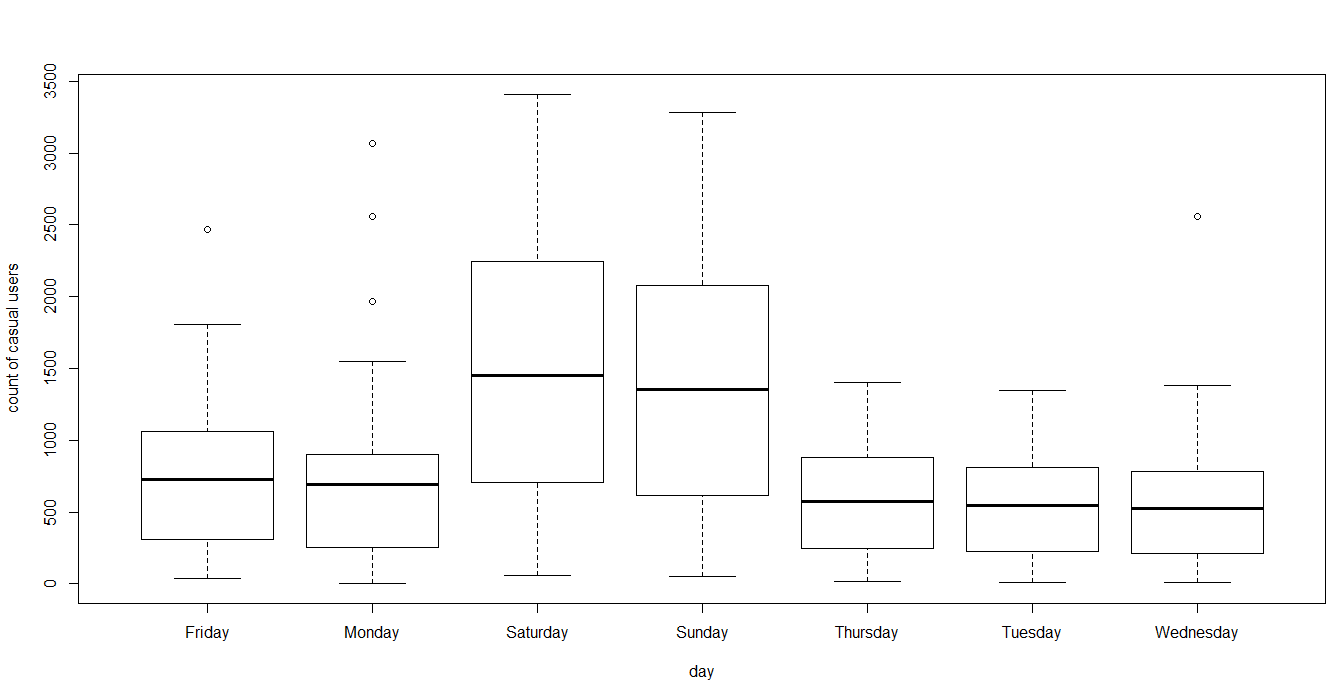
The relationship of the independent variables vs dependent variable is our main purpose for the analysis as this will tell what is the trend of bike rental count over these parameters.

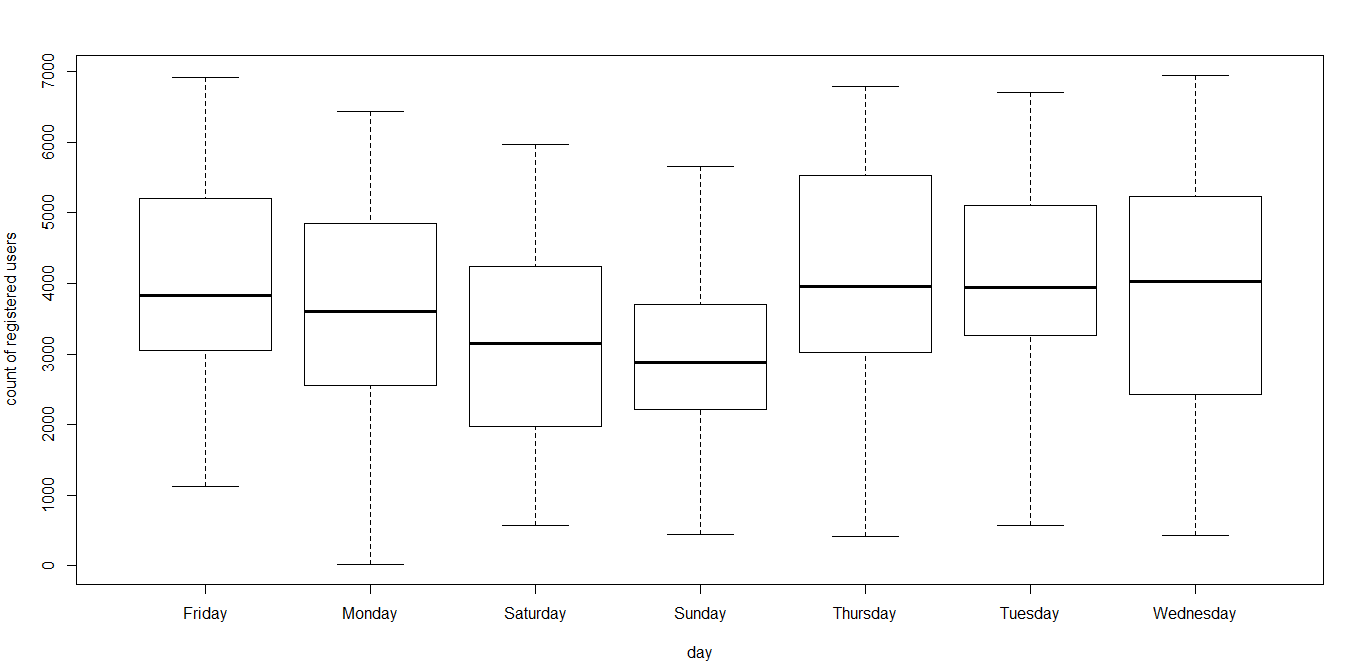
There are two types of users for renting a bike i.e., casual and registered, so to analyse how their counts are varying with the data below code is used in R and Python.

Day Type vs Casual/Registered users: There are two variables in the dataset that are describing the day and day type i.e., “workingday” & “weekday”. So to check if “weekday” alone would do the part in modelling of the data or both of them are to be incorporated.

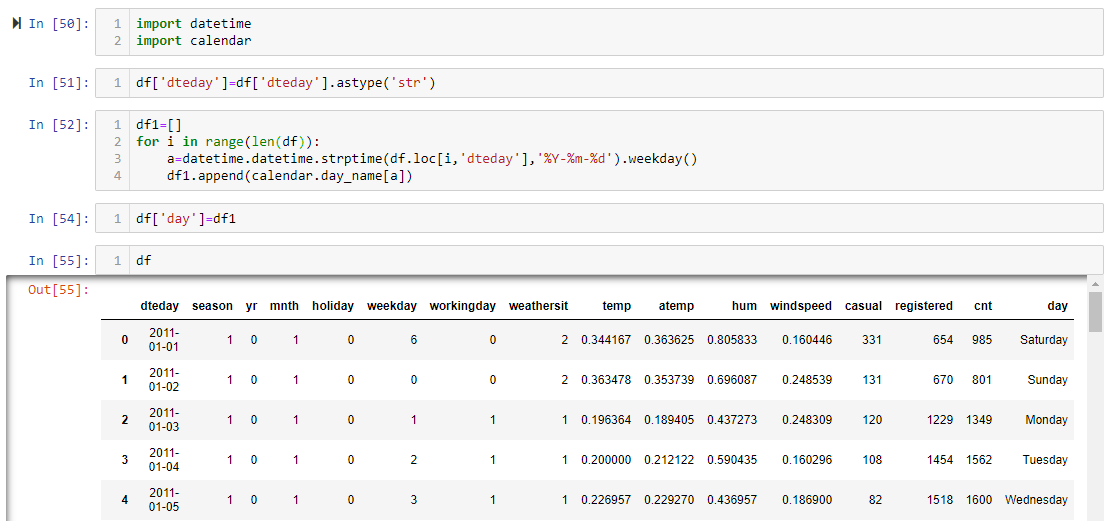
# R







# Python





From above plots it can be observed that the number of bike rentals varies on the type of day and majorly on the type weekday or weekends as

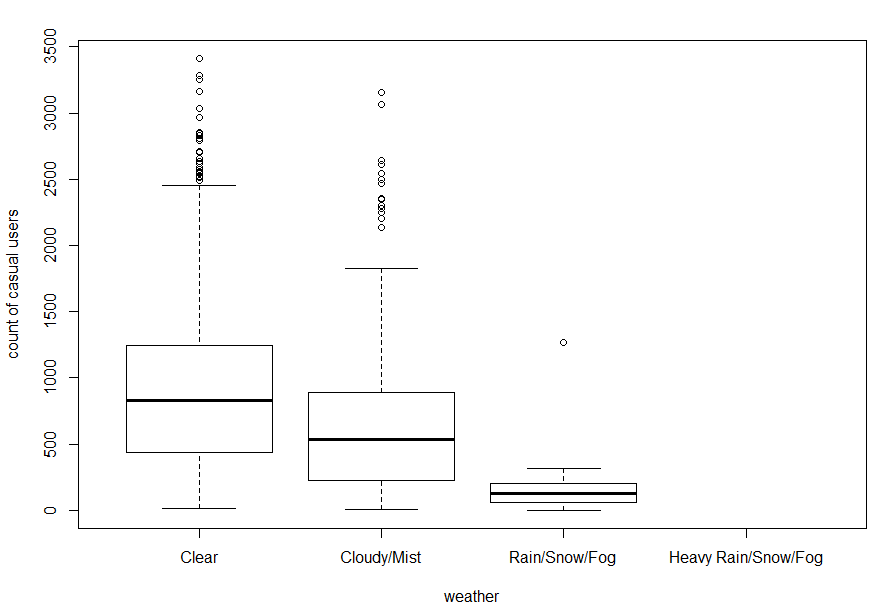
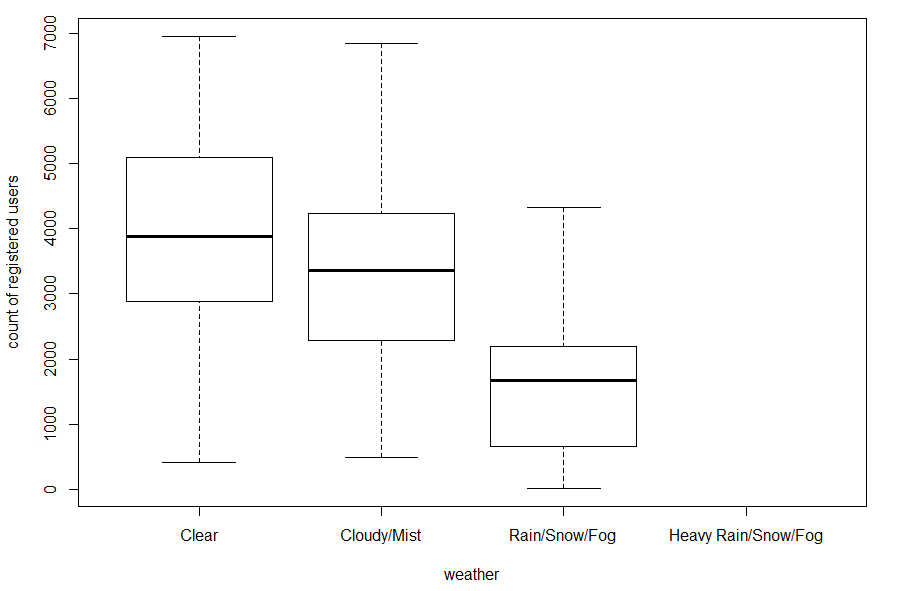
1. Casual bike rentals are increased during weekends (as the upper whisker of the plot for weekends are above than weekdays) probably because people rent bike for fun trips and casual outings whereas,
2. Registered bike rentals are less on weekends as compared to weekdays (as the upper whisker of the plot for weekends are lower than weekdays) which is probably because the customers who have registered to the company use it for daily commutes.

Therefore, both variable “workingday” and “weekday” plays an important role in modelling.

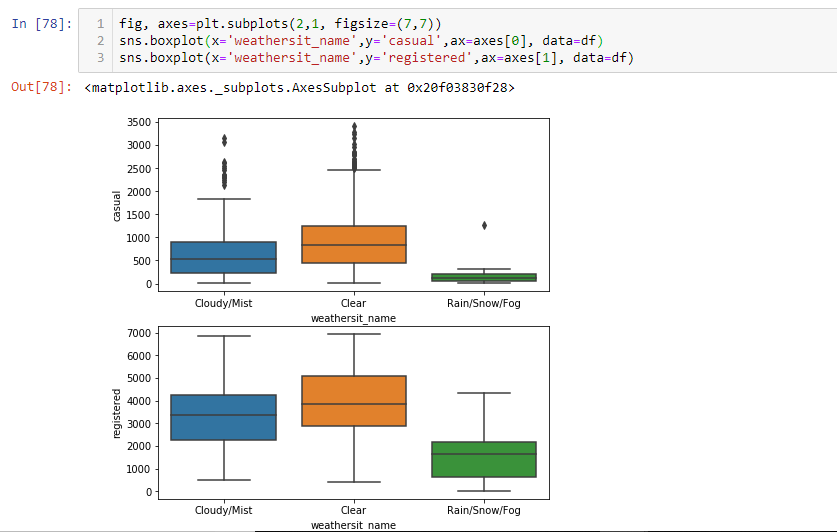
Season vs Casual/Registered users: For rain/snow/fog season, the bike rental count should drop drastically as due to the outside weather condition are not favourable. From data distribution it is clear that the rain/snow/fog season has less rows in dataframe than others but to see that what is the number of bike rentals the following code is used.

# R



# Python

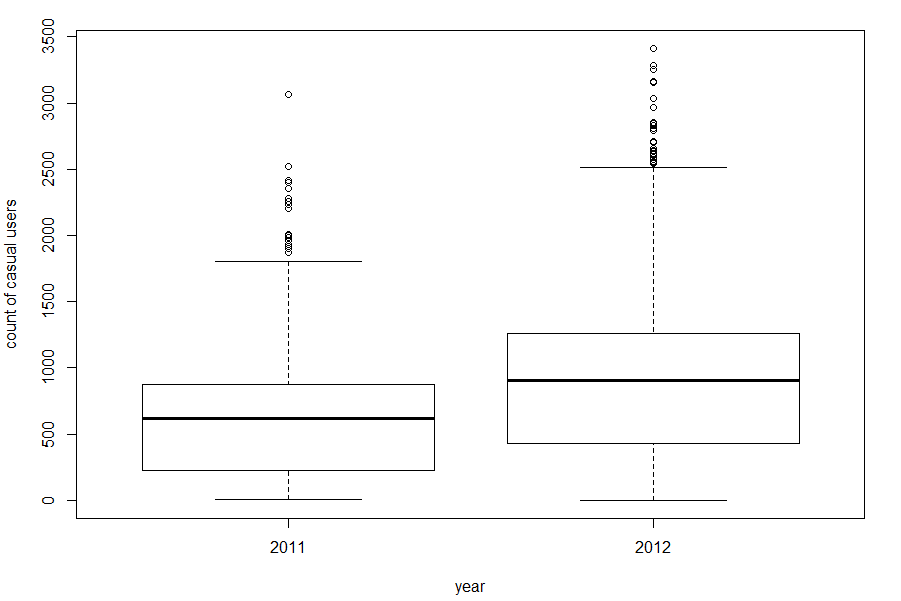
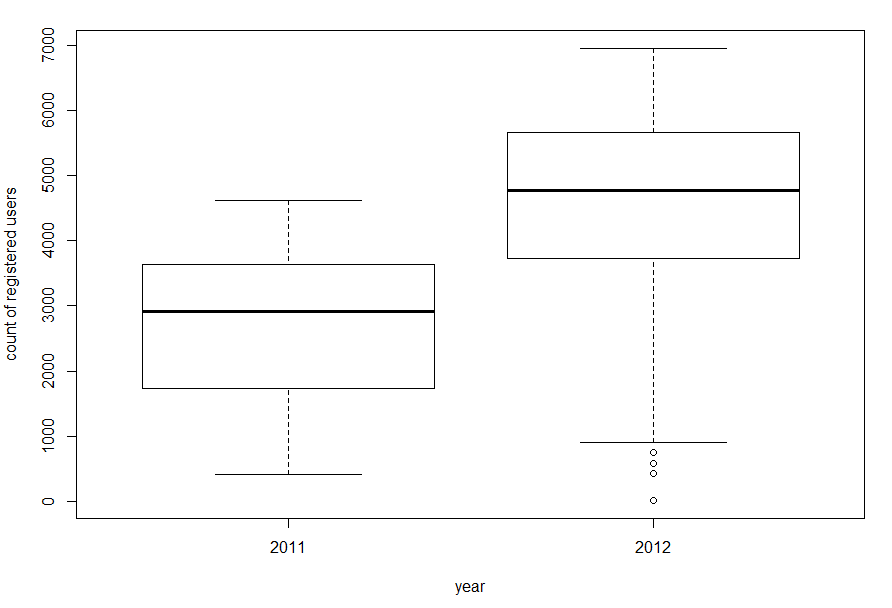


Above plots shows that for the rain/snow/fog season the number of casual and registered bike rental count decreases.

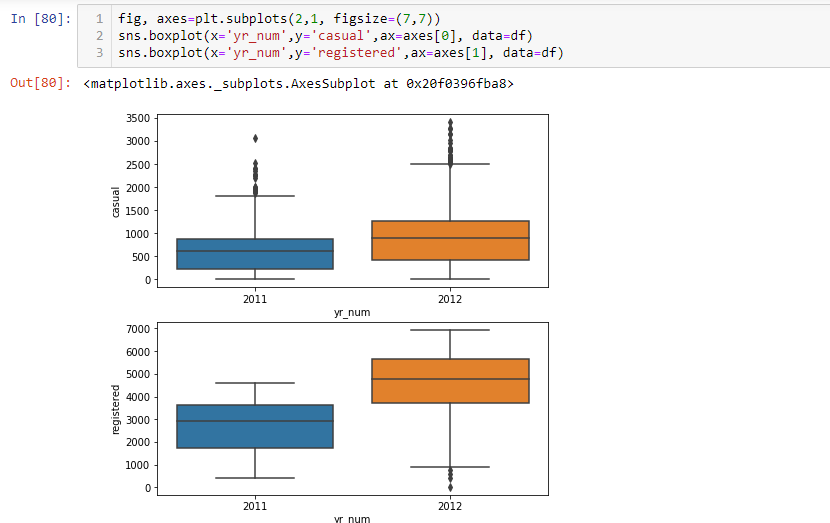
Year vs Casual/Registered users: To see the yearly trend for the casual and registered customers over the years.

# R



# Python

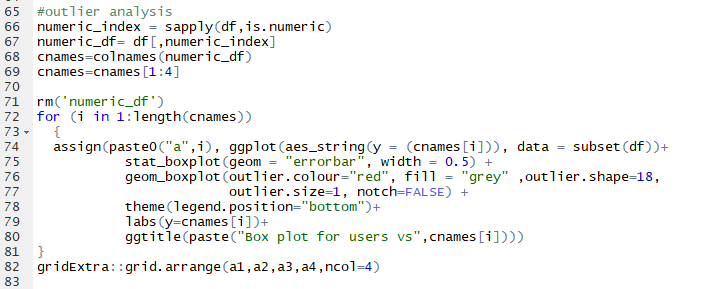


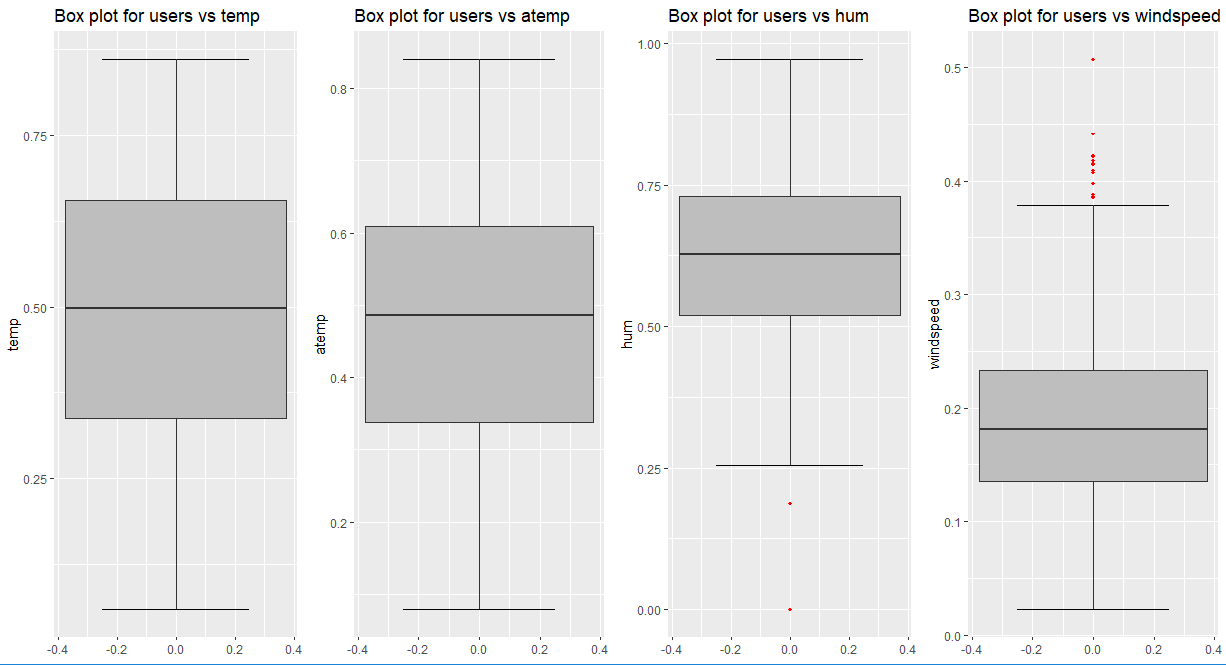
For both casual and registered bike rental count there is an incremental growth over the years 2011 and 2012 as the upper whisker i.e., 100 percentile of the data, for the 2012 boxplot is above than 2011 boxplot.

* 1. Outlier Analysis

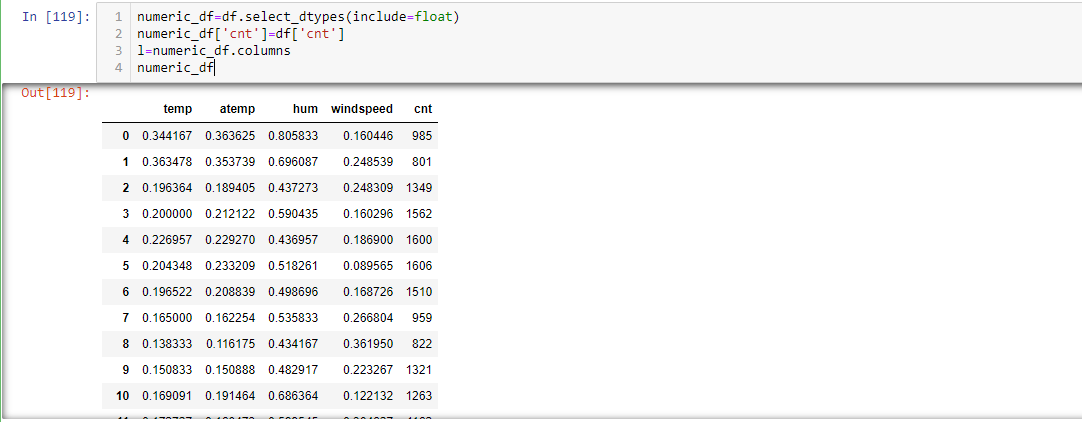
There are some continuous independent variables in the dataframe like “temp”, “atemp”, “hum” & “windspeed” on which the prediction variable depends, these variables can have some outliers due to which the machine learning model will become biased and produce wrong predictions. To analyse the outliers in these variable below mentioned code is used in R and Python,

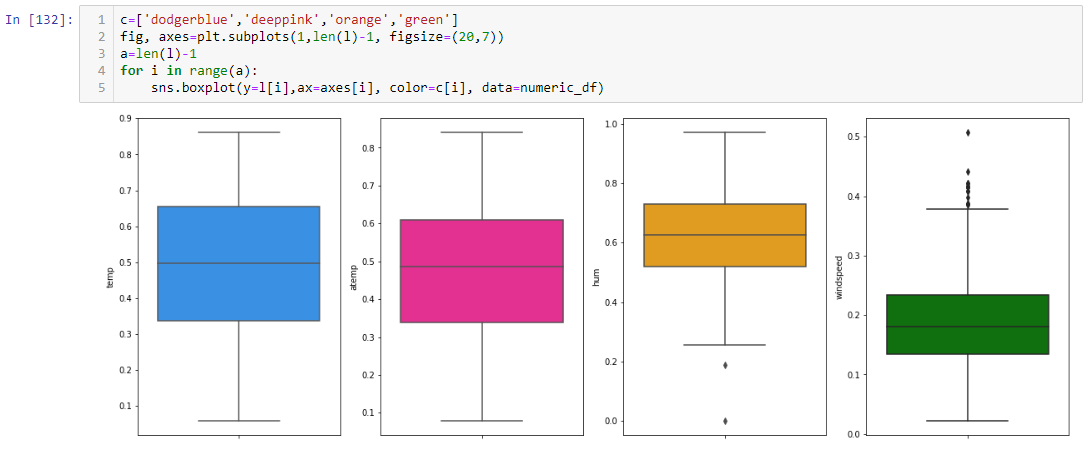
# R





# Python

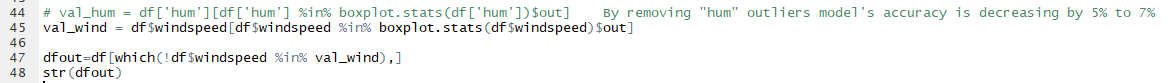


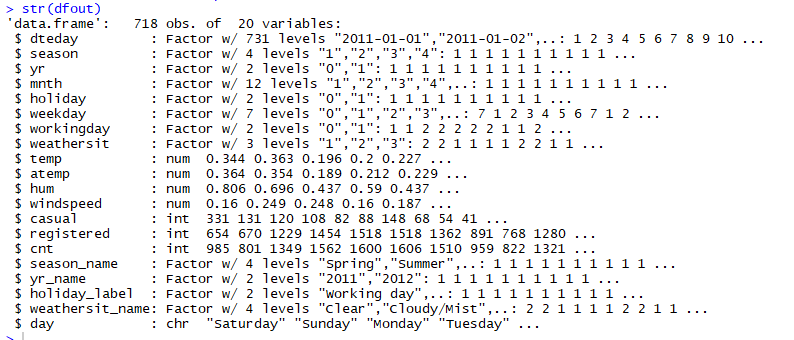


From above plot it can be observed that the variables “hum” and “windspeed” has outliers present. For “hum” there are very less outliers present below the bottom whiskers in comparison with outliers present above the “windspeed” variable.

The outliers are to be removed so as to avoid any prediction of biased results affected because of these outliers i.e., done by below code:

# R





Code for Python is given on the next page.

#Python



Now the dataframe has reduced to only 718 observations as 13 observations which are detected during outlier analysis are removed.

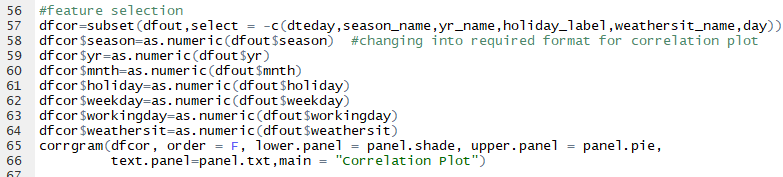
Note: Outliers for the variable “hum” are not removed because after removal of them the machine learning model is producing less accurate results. For the decision tree and random forest model there is decrease in 5% and 7% accuracy rate respectively.

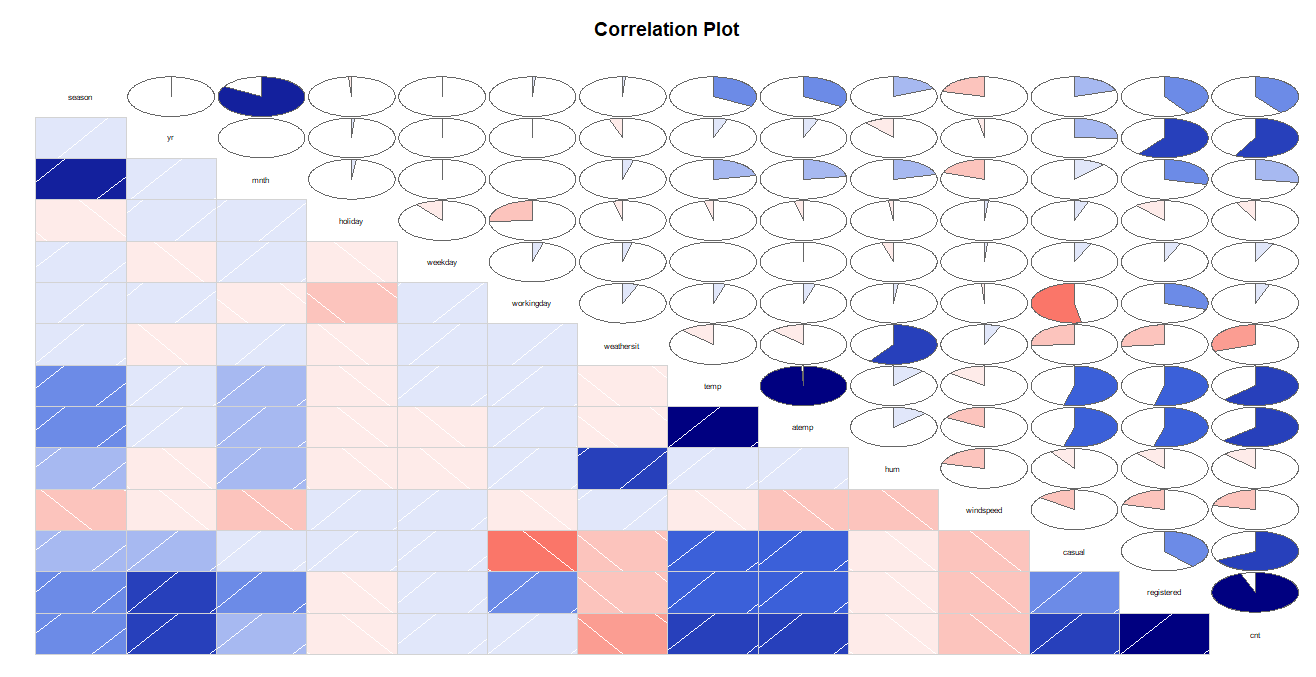
* 1. Feature Selection

The independent variables might have some correlation between one and another i.e., the so called independent variable will have some dependency over each other. The machine learning model considers all predictor variable to be totally independent to one another and if there exists some correlation between them, then one of those variable is totally redundant thus affecting the prediction results. So to check the inter-dependency of the predictor variables Correlation analysis is carried out.

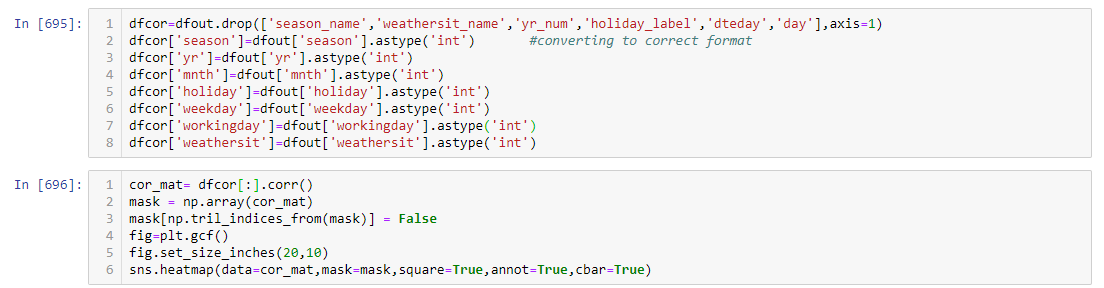
Below is given the codes and plot for Correlation analysis:

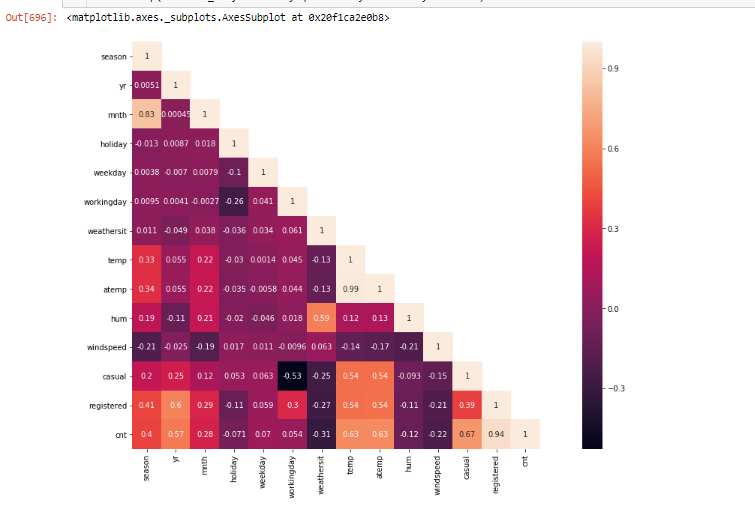
# R





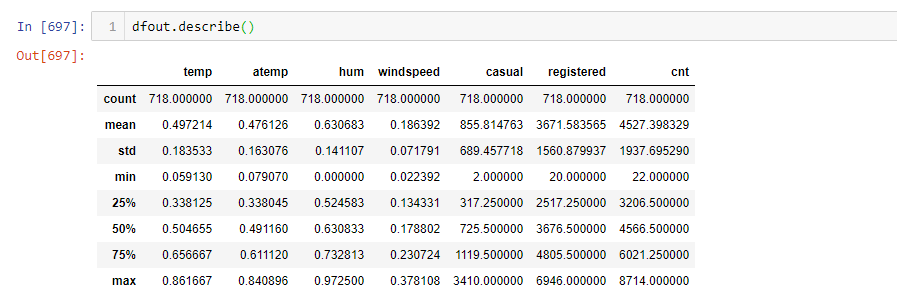
#Python





From the plots few things are inferred given as below:

1. Variable “temp” and “atemp” are highly positively correlated (correlation value=0.99) with each other thus choosing one of them as an input in the model would be good.
2. “windspeed” is slightly negatively correlated with “temp” and “hum” whereas “temp” and “hum” are slightly positively correlated.



Further it can be seen from the above image that for the variables “temp” and “atemp” all statistic characteristics are similar to each other like mean, min, max, each quartile value and standard deviation.

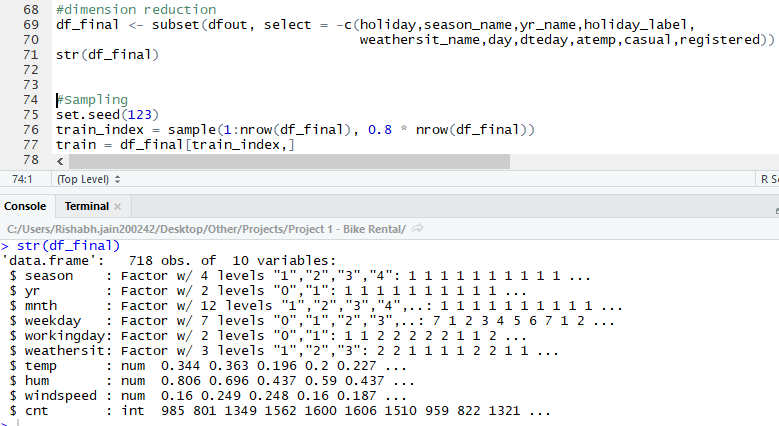
* 1. Final Input DataFrame

The dataframe needs to be modified now so that the input dataframe to the machine learning model produces non-biased and accurate results. The modifications that needs to be done are mentioned as below:

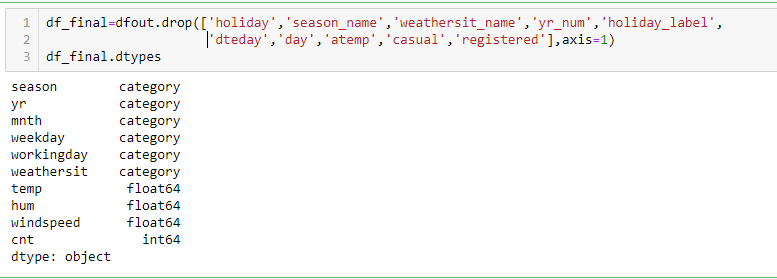
1. Dropping the variable “atemp” as its correlation with “temp” is observed in Correlation analysis
2. As variable “holiday” contains very less percentage (i.e., 2.9%) of observations for the holiday\_label=1, so dropping this will also produce error free results
3. Variable “dteday” does not have any use for now, as “weekday” and “workinday” are taking care for the daytype segregation (Note: this variable has been dropped at the time of correlation analysis)
4. Dropping “casual” and “registered” variables as “cnt” is arithmetic sum of these two variable values.

Below is the code to do so in R and Python,

# R



# Python



Now, the dataframe has the required predictor variables. The final dataframe “df\_final” is of size 718x10.

1. **Modelling**

The type of model needs to be implemented mainly depends upon the target variable data type, predictor variables’ data type and type of relationship they carry with each other.

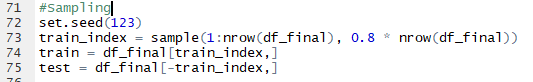
The target variable is continuous variable and the predictor variables have both types of variables i.e., categorical and continuous variables. As the target variable is a continuous variable discussed below machine learning model would be a better prediction model for this problem. But at first sampling is done on the provided dataframe to test the model results with actual values.

* 1. Sampling

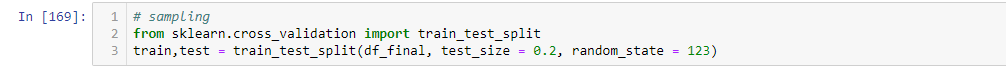
Random sampling is done here as there is no such category in the population data (whole dataframe) on which the data is distributed in a manner that stratified sampling must be used.

To perform sampling below code is used in R and Python:

# R

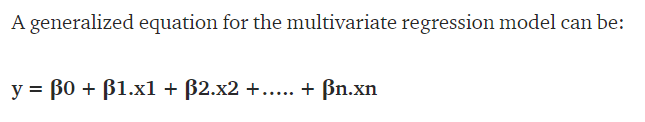


# Python



* 1. Multivariate Linear Regression

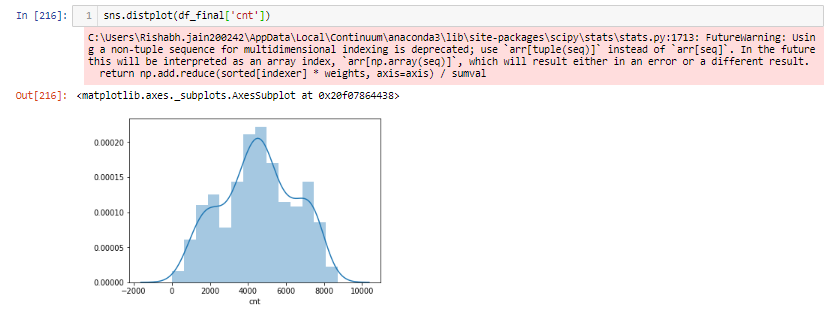
The dataframe consists of 9 predictor variables which plays vital role to predict the target variable. Multivariate linear regression model is used to predict the dependent variable for two or more independent variables. The model got an input dataframe containing all the predictor variables which are used to generate a multi-variable linear relationship mathematical equation with target variable.



The model tries to predict these coefficients (β0, β1,……..., Βn) attached to each variable so as to calculate the target variable value by summing up each one’s product with its respective variable.

Assumptions for the model:

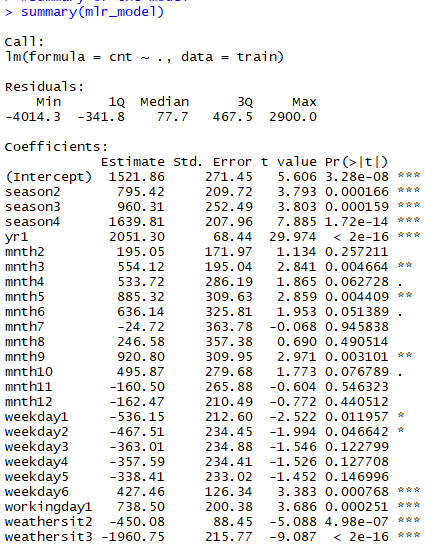
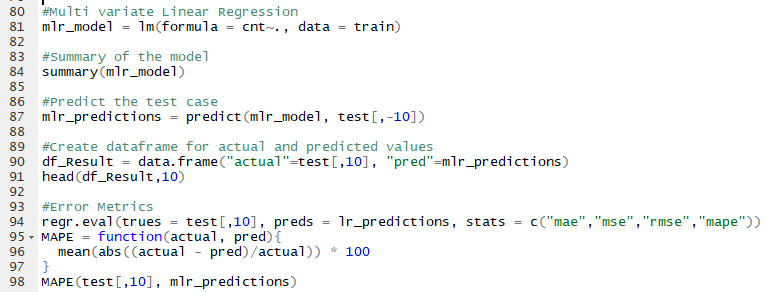
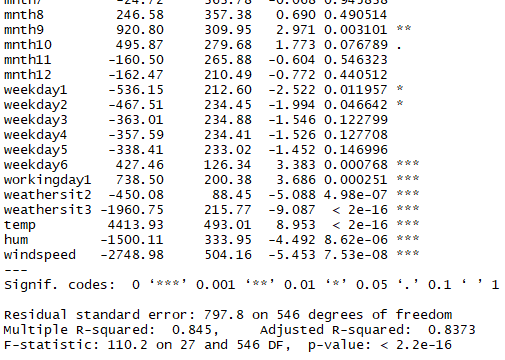
1. Linear relationship between independent variable and dependent variable
2. Multivariate normality i.e., the target variable looks pretty much normally distributed as shown in fig below,

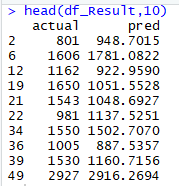


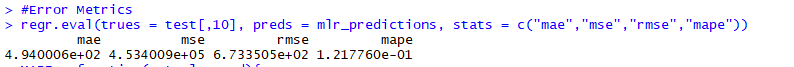
1. No or little multi-collinearity between independent variables i.e., done in correlation analysis
2. No auto correlation i.e., no correlation between residuals (or errors) which can be viewed through the model summary.

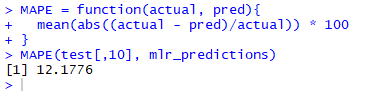
The code to develop a multi variate linear regression model is given below in both R and Python,

# R



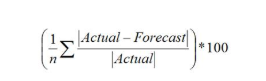




From above results for the model below mentioned points can be inferred about the model performance:

1. Multiple R-squared is 0.845 which means that 84.5% of variance of target variable is explained by the independent variables.
2. Adjusted R-squared is 0.837 which is less than the R-squared.
3. F-statistics value is greater than 1 i.e., 110.2 which tells that the fit is significant for the prediction.

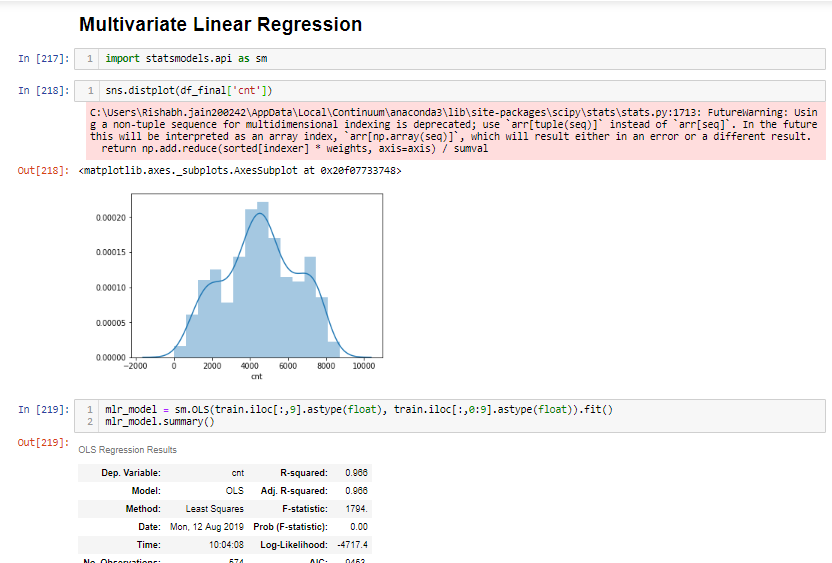
Error metrics used for evaluating the model is Mean Absolute Percentage Error which is calculated as below:

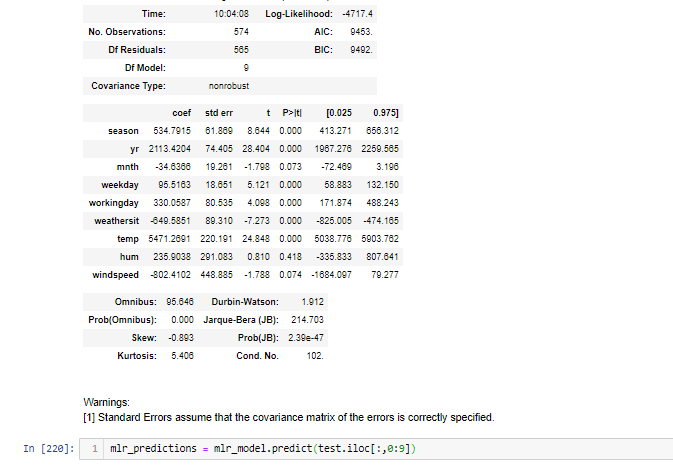


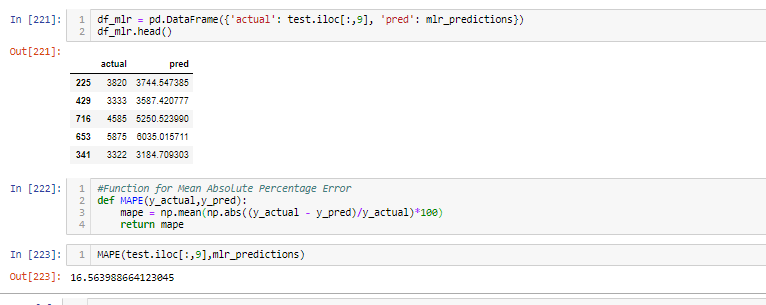
To calculate the error metrics a user defined function MAPE is built in both languages.

The MAPE for this model in R comes out to be 12.17% which means the model accuracy percentage is 87.83% which is pretty good with this small of dataset.

# Python







For the above model coded in python below are the mentioned model parameters that will judge the model performance:

1. Multiple R-squared is 0.966 which means that 96.6% of variance of target variable is explained by the independent variables.
2. Adjusted R-squared is 0.966 which is not greater than the R-squared.
3. F-statistics value is greater than 1 i.e., 1794.

And the error metrics used is MAPE which comes out to be 16.56% that means the model accuracy is 83.44%.

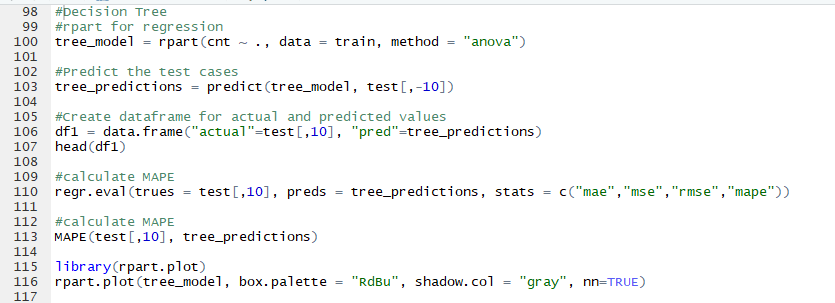
Note: The results produced in R and Python are quite different and that is because during sampling different data rows are selected to produce the training and test set in both R studio and Jupyter notebook.

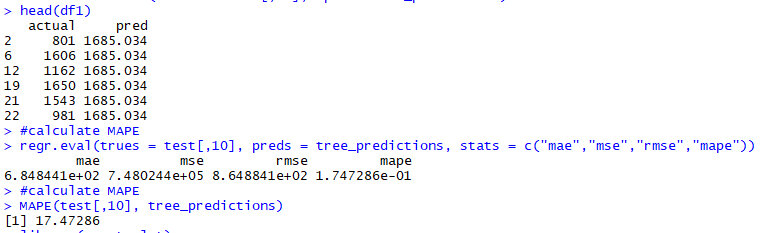
* 1. Decision Tree

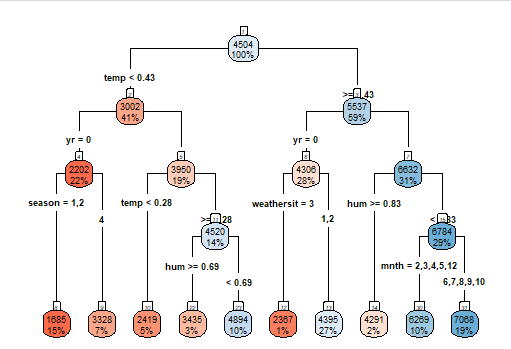
Decision tree ML model is based on branching series of Boolean tests carried out until all right child nodes or leaf nodes are produced. It mainly develops certain sets of rules from the historical data in the form of tree according to which the values are predicted.

Below shown code is used in R and Python to develop this model,

# R







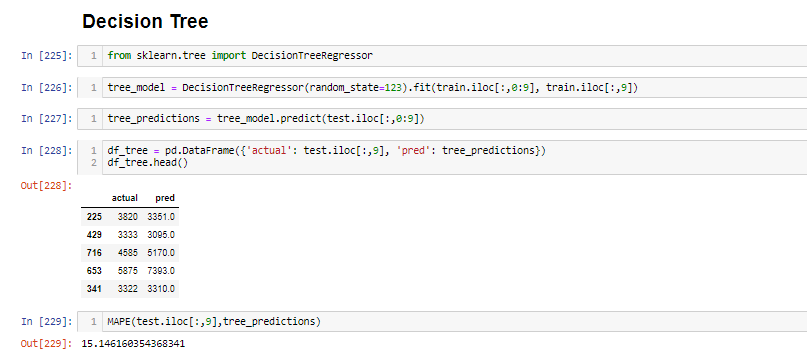
To know the accuracy of the predictions made by the model, MAPE error metrics is used which comes out to be 17.47% which means 82.53% is the model accuracy percentage.

Also, by using regr.eval() function other error metrics can be seen like

MAE (mean absolute error) – 684.4

RMSE (root mean squared error) – 864.8

# Python



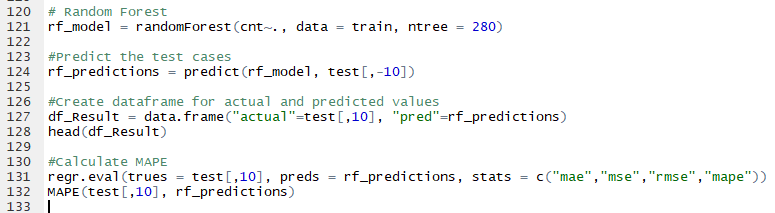
MAPE for decision tree model developed in python language comes out to be 15.14% which means that the accuracy percentage of the model is 84.86% i.e., greater than the linear regression model coded in python.

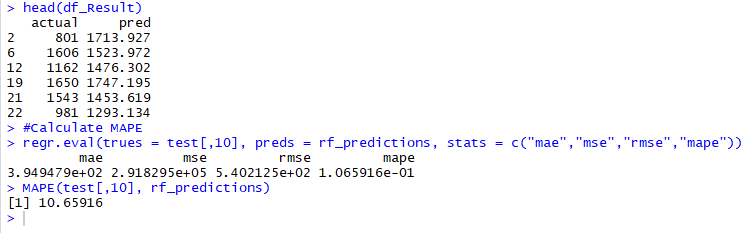
* 1. Random Forest

Random Forest is a type of ensemble learning method used for classification and regression problems. It is a meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting.

The code to develop random forest classifier in R and Python are shown below in the figures,

# R





For the above developed Random Forest classifier, MAPE is calculated as 10.65% giving the model accuracy percentage as 89.35% which represents very effective predictions made by the model. Also, as calculated in decision tree classifier model, MAE is 394.9 and RMSE is 402.1

# Python



Here also MAPE comes out to be pretty lower than the two models discussed earlier. MAPE is equals to 11.34% which means the model is predicting 88.66% accurate values for target variable.

* 1. XGBoost with Python

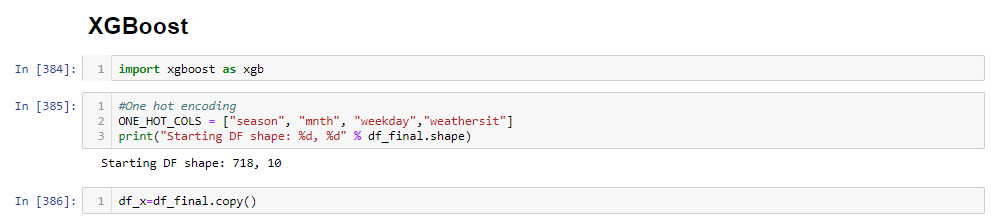
XGBoost stands for eXtreme Gradient Boosting algorithm that is one of the sequential boosting method that uses the ensemble learning technique to predict accurate results fastly. This is an adaptive learning algorithm that takes up the weak learners and over the iterations it calculates the loss in the predicted outcomes. This loss is analysed and then the algorithm adds up the weak learners to form a strong learner than before and reduce that loss calculated i.e., regularizing the loss function. The xgboost algorithm is fast in comparison with other boosting methods which are Adaptive boosting and Gradient boosting.

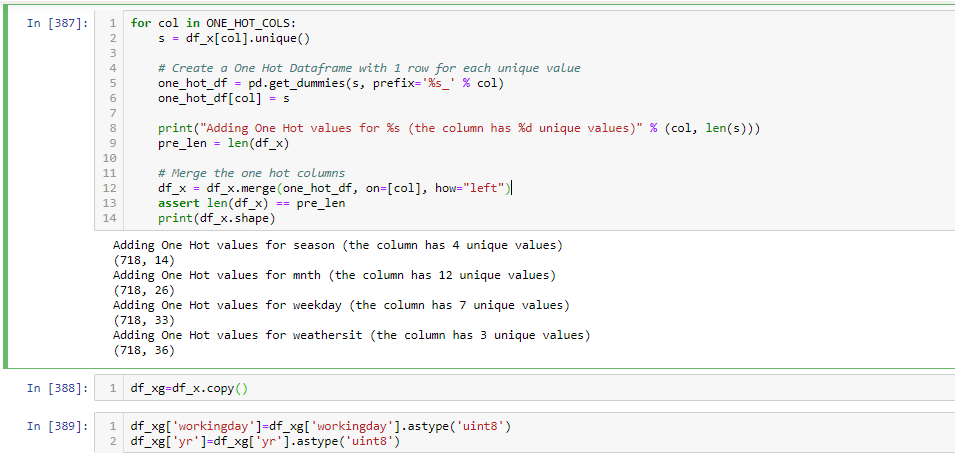
To implement it in python below mentioned is the code,

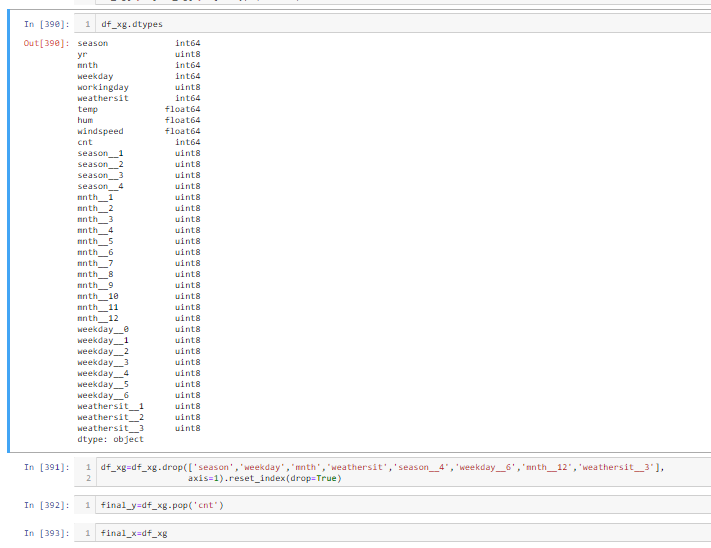
There are two classes which can be used to implement xgboost in python i.e.,

1. xgb.train
2. XGBRegressor

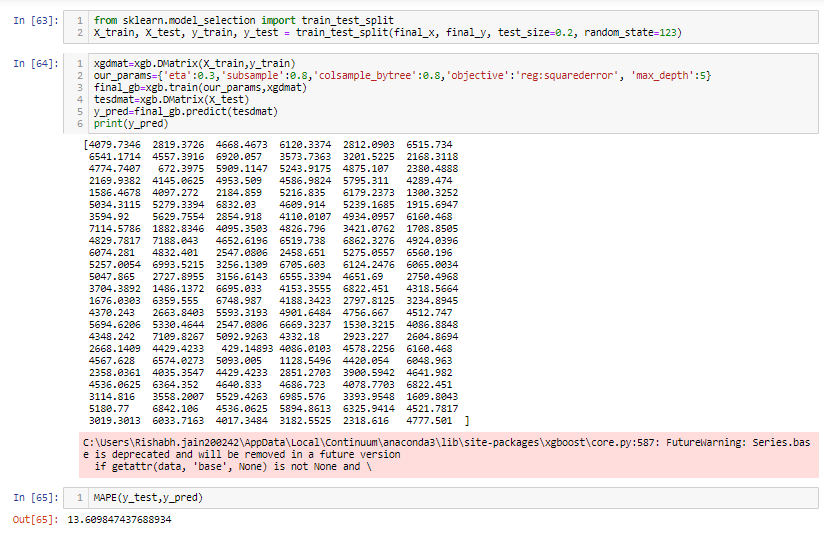
Both work in the similar manner, the only difference is in xgb.train class the number of iterations are 100 by default whereas in XGBRegressor class the number of iterations can be set by providing value to the parameter named as “n\_estimators”.



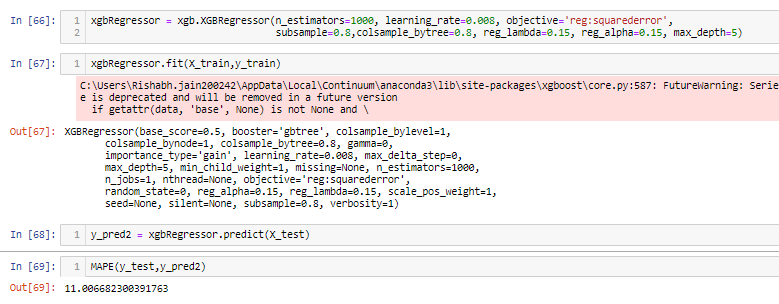




#xgb.train



#XGBRegressor



The parameters are tuned to give optimized training model and less MAPE. For xgb.train class the best model developed have MAPE equals to 13.60% i.e., 86.4% accurate model whereas for XGBRegressor class the best model developed have MAPE 11% i.e., 89% accurate model.

1. **Conclusion**

After getting the MAPE values for each of the model that are discussed above it is quite clear that the Random forest model and XGBoost model performed much better than others. It can be shown in the below table:

|  |  |  |
| --- | --- | --- |
| Model\_Name | R | Python |
| Multivariate Linear Regression | MAE= 12.17%  Accuracy= 87.83% | MAE= 16.56%  Accuracy= 85.44% |
| Decision Tree Classifier | MAE= 17.47%  Accuracy= 82.53% | MAE= 15.14%  Accuracy= 84.86% |
| Random Forest | MAE= 10.65%  Accuracy= 89.35% | MAE= 11.34%  Accuracy= 88.66% |
| XGBoost | - | MAE= 11%  Accuracy= 89% |