Project: **BIKE RENTALS PREDICTION**

September 11, 2019

Author : Aashit Singh

Index

Chapter 1

Introduction

1.1 Problem Statement

Finding the number of bikes to be rented on a given day, in advance can be extremely useful. Not only for the company renting the bikes, but also to many other sectors. It can help out in planning traffic flow, intimating customers beforehand about a day of rush and much more. Not having a knowledge of this affects the business and causes loss of potential revenue. I would like to predict the number of bike rentals that can be expected on a given day, using the provided features of that day.

1.2 Data

Our task is to build regression models which will predict the count of bike rentals depending on multiple factors of that particular day - meteorological, chronological & more. Given below is a sample of the data set that we are using to predict the **count of bike rentals**:

Table 1.1 Bike Rentals Sample Data (Column 1 - 10)

instant	**dteday**	**season**	**yr**	**mnth**	**holiday**	**weekday**	**workingday**	**weathersit**	**temp**
1	2011-01-01	1	0	1	0	6	0	2	0.344167
2	2011-01-02	1	0	1	0	0	0	2	0.363478
3	2011-01-03	1	0	1	0	1	1	1	0.196364
4	2011-01-04	1	0	1	0	2	1	1	0.2

Table 1.2 Bike Rentals Sample Data (Column 11 - 16)

atemp	**hum**	**windspeed**	**casual**	**registered**	**cnt**
0.363625	0.805833	0.160446	321	654	985
0.353739	0.696087	0.248539	131	670	801
0.189405	0.437273	0.248309	120	1229	1349
0.212122	0.590435	0.160296	108	1454	1562

As you can see below we have the following variables:

1. instant
2. dteday
3. season
4. yr
5. mnth
6. holiday
7. weekday
8. workingday
9. weathersit
10. temp
11. atemp
12. hum
13. windspeed
14. casual
15. registered
16. **cnt** - **TARGET VARIABLE**

I have dropped some variables that don't help the prediction, by intuition. Such variables are:

1. instant
2. dteday (features are already extracted from date)

And some variables are a part of the target variable itself. Like:

1. casual
2. registered

So the final list of predictor variables that we have now are:

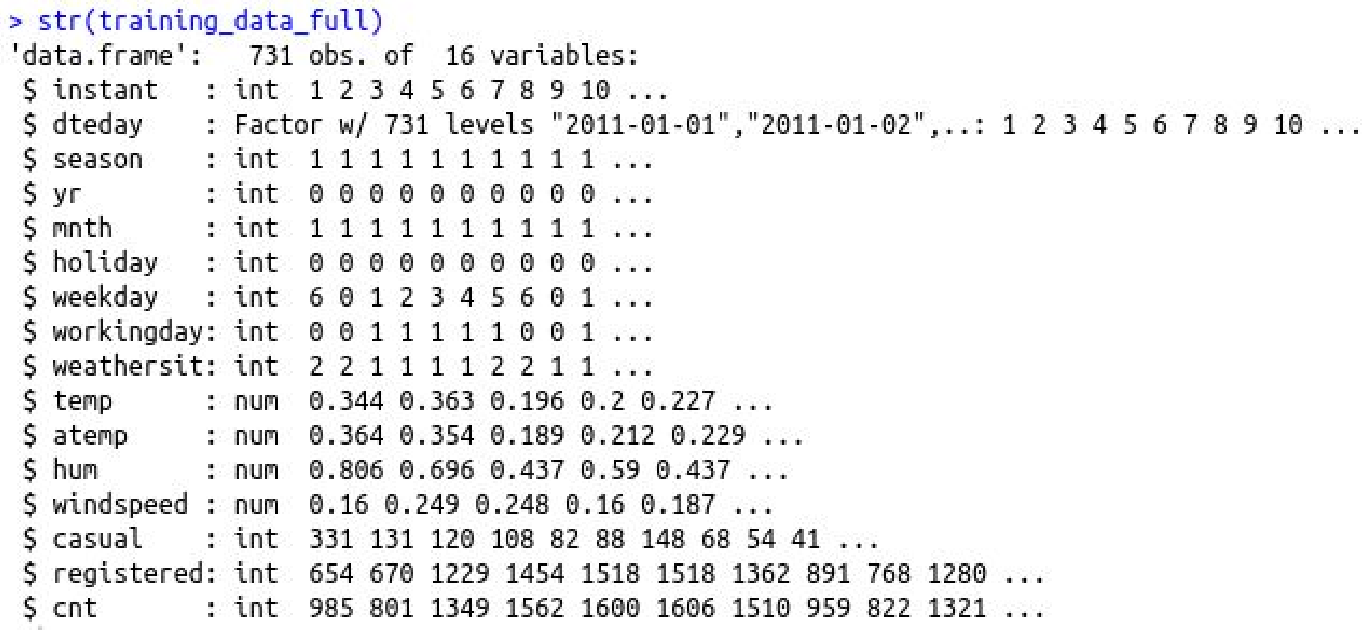
1. season
2. yr
3. mnth
4. holiday
5. weekday
6. workingday
7. weathersit
8. temp
9. atemp
10. hum
11. windspeed

Chapter 2

Methodology

2.1 Pre Processing

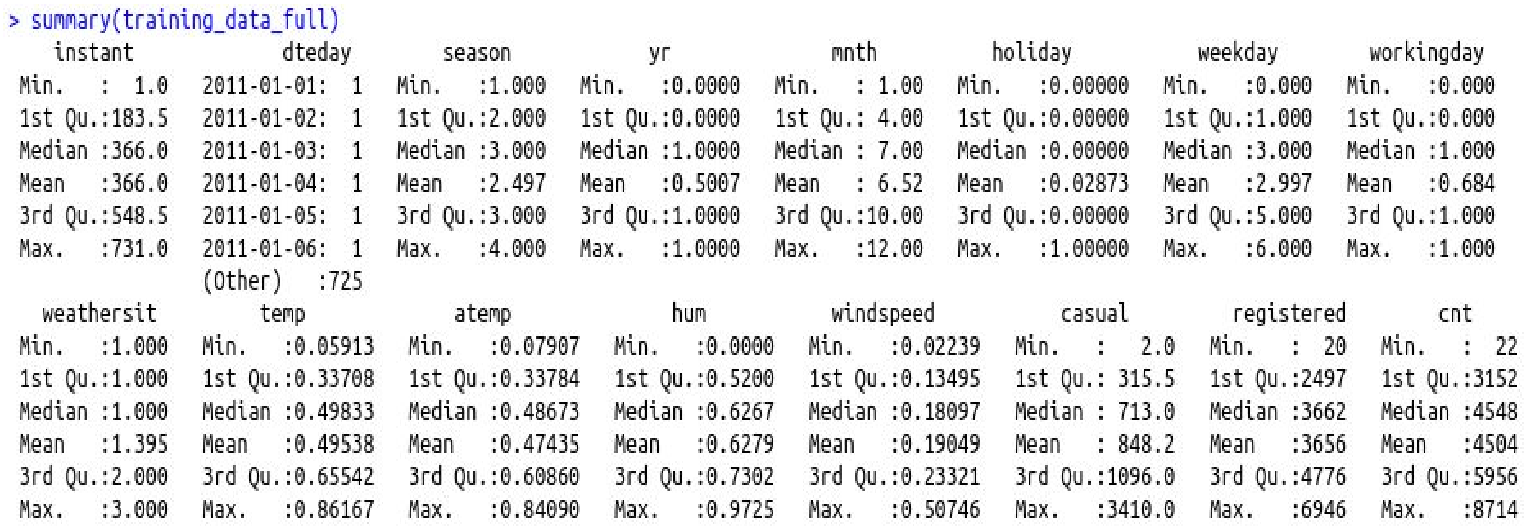
Before we start building & training models on our data, we have to make sure our data is structured & cleansed. This process is called pre-processing. And before we structure and clean the data we have to understand and explore the data as well as visualise the data through graphs. This is called Exploratory Data Analysis (EDA). To start with, we will see the structure of our data (Fig. 2.1).



```
> str(training_data_full)
'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 ...
```

Fig 2.1 Structure of data in RStudio

Summary of the data (Fig. 2.2).



```
> summary(training_data_full)
   instant            dteday        season            yr              mnth          holiday           weekday        workingday   
 Min.   :  1.0   2011-01-01:  1   Min.   :1.000   Min.   :0.0000   Min.   : 1.00   Min.   :0.00000   Min.   :0.000   Min.   :0.000  
 1st Qu.:183.5   2011-01-02:  1   1st Qu.:2.000   1st Qu.:0.0000   1st Qu.: 4.00   1st Qu.:0.00000   1st Qu.:1.000   1st Qu.:0.000  
 Median :366.0   2011-01-03:  1   Median :3.000   Median :1.0000   Median : 7.00   Median :0.00000   Median :3.000   Median :1.000  
 Mean   :366.0   2011-01-04:  1   Mean   :2.497   Mean   :0.5007   Mean   : 6.52   Mean   :0.02873   Mean   :2.997   Mean   :0.684  
 3rd Qu.:548.5   2011-01-05:  1   3rd Qu.:3.000   3rd Qu.:1.0000   3rd Qu.:10.00   3rd Qu.:0.00000   3rd Qu.:5.000   3rd Qu.:1.000  
 Max.   :731.0   2011-01-06:  1   Max.   :4.000   Max.   :1.0000   Max.   :12.00   Max.   :1.00000   Max.   :6.000   Max.   :1.000  
                 (Other)   :725                                                                                                      
   weathersit         temp             atemp              hum           windspeed           casual         registered        cnt      
 Min.   :1.000   Min.   :0.05913   Min.   :0.07907   Min.   :0.0000   Min.   :0.02239   Min.   :   2.0   Min.   :  20   Min.   :  22  
 1st Qu.:1.000   1st Qu.:0.33708   1st Qu.:0.33784   1st Qu.:0.5200   1st Qu.:0.13495   1st Qu.: 315.5   1st Qu.:2497   1st Qu.:3152  
 Median :1.000   Median :0.49833   Median :0.48673   Median :0.6267   Median :0.18097   Median : 713.0   Median :3662   Median :4548  
 Mean   :1.395   Mean   :0.49538   Mean   :0.47435   Mean   :0.6279   Mean   :0.19049   Mean   : 848.2   Mean   :3656   Mean   :4504  
 3rd Qu.:2.000   3rd Qu.:0.65542   3rd Qu.:0.60860   3rd Qu.:0.7302   3rd Qu.:0.23321   3rd Qu.:1096.0   3rd Qu.:4776   3rd Qu.:5956  
 Max.   :3.000   Max.   :0.86167   Max.   :0.84090   Max.   :0.9725   Max.   :0.50746   Max.   :3410.0   Max.   :6946   Max.   :8714  
```

Fig 2.2 6-point summary of all columns in RStudio

I look for missing values. If any, it can be imputed with an appropriate value - mean, median, mode or others. This data has no missing values.

2.2 Outlier Analysis

Most models & analytic functions like regression, require the data to be normally distributed (follow the normal distribution). We can visualize that at once by looking at the histograms of the variable. Outliers are some rare and extreme values in the whole dataset, that cause the data to be skewed & brings some unnecessary changes in descriptive statistics of the data, like mean and median. These outliers can be good for the predictive model that we will be building, as they help in capturing the variability & variety in the population & training set.

It is mentioned in the problem statement document provided that the numerical features like *temp*, *atemp*, *hum* & *windspeed* are normalised already.

Below are some figures of box plots & histogram & box plots combined that explain the distribution of the data.

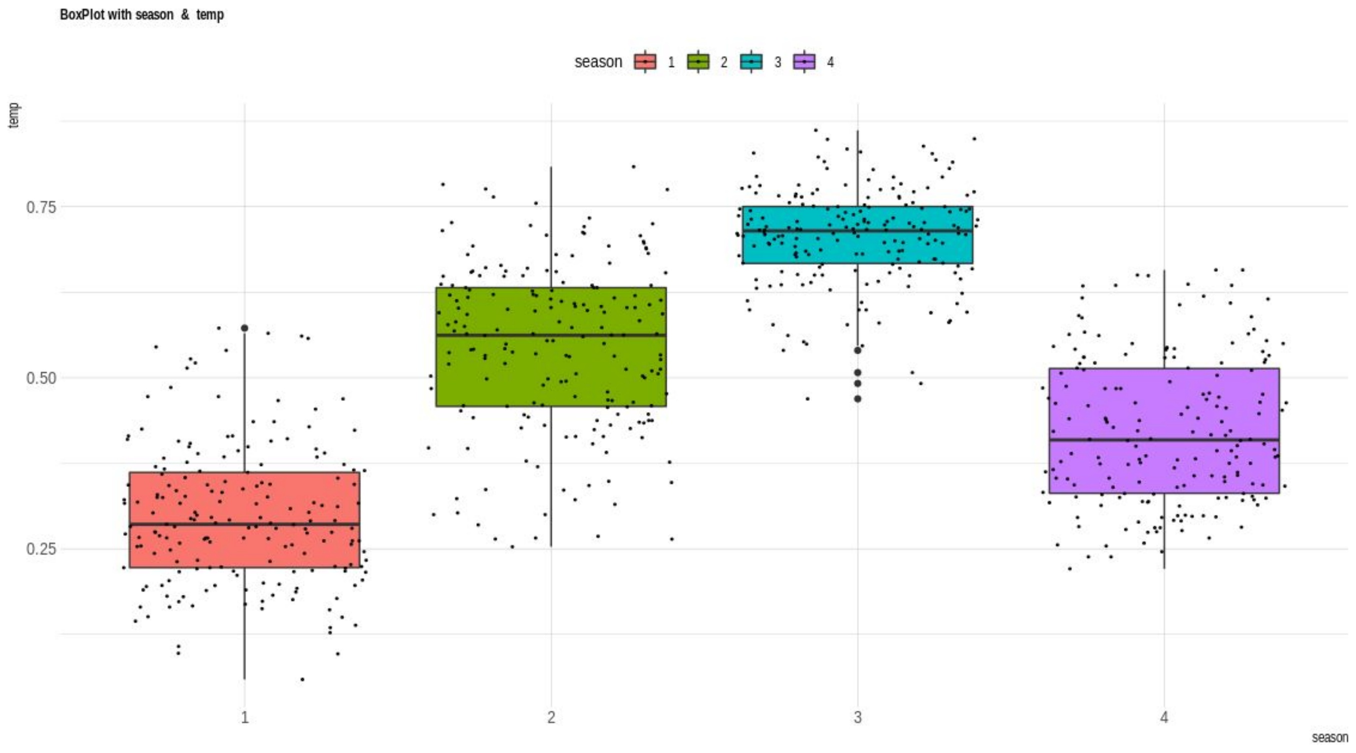
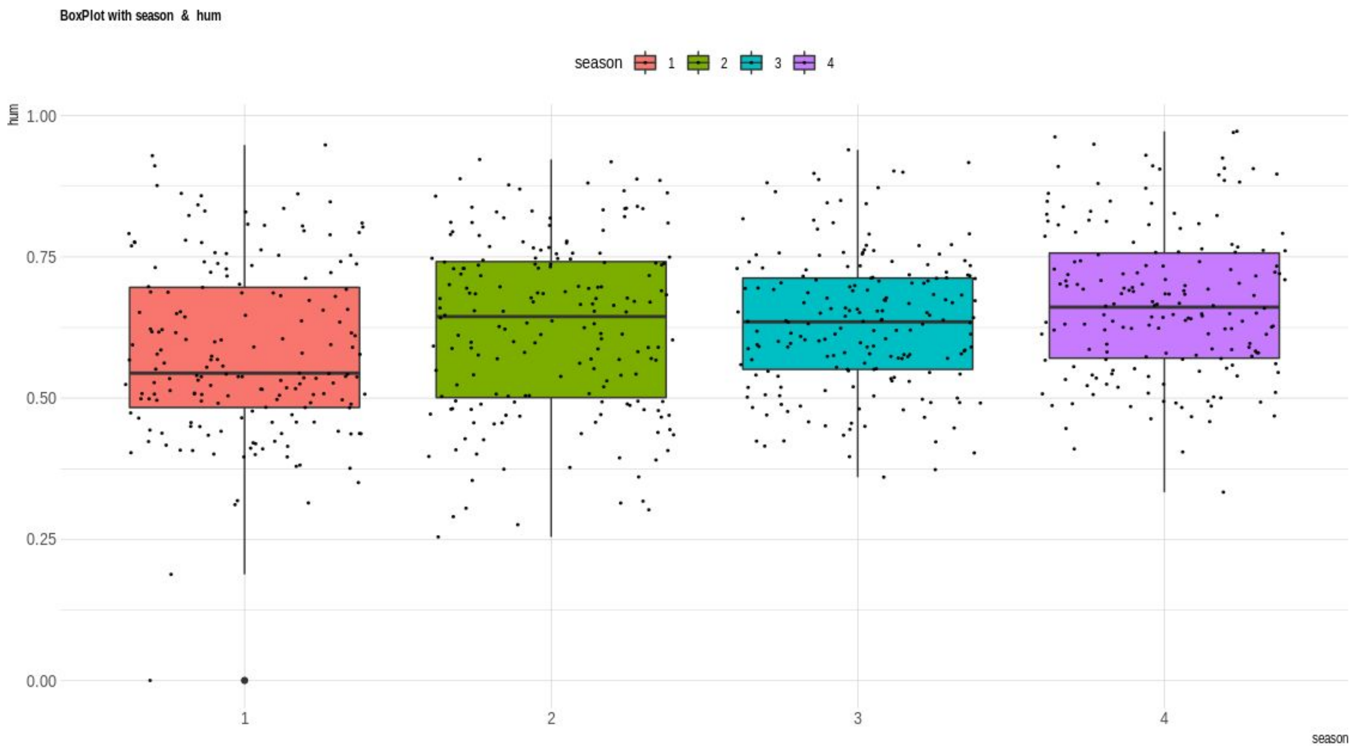


Fig 2.3 Box Plot of **temp** distributed over **season** (above)
Fig 2.4 Box Plot of **hum** distributed over **season** (below)



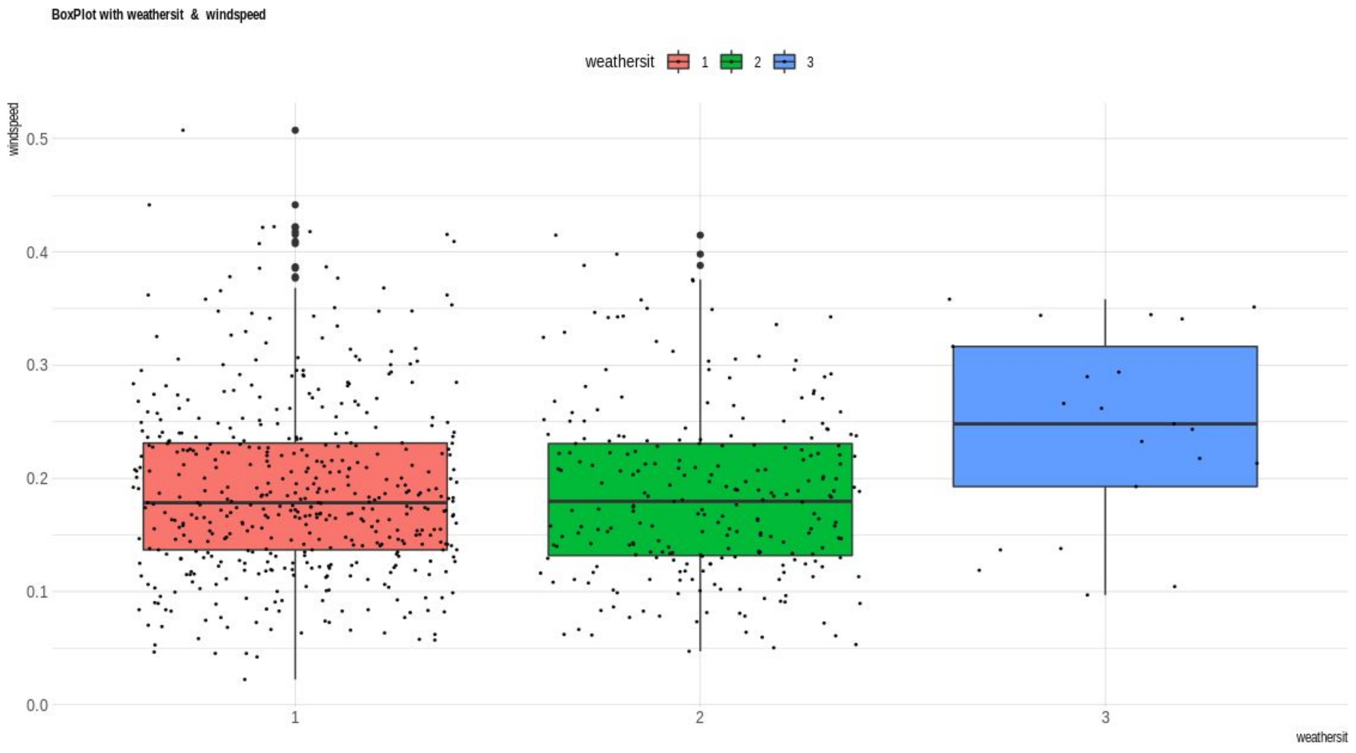
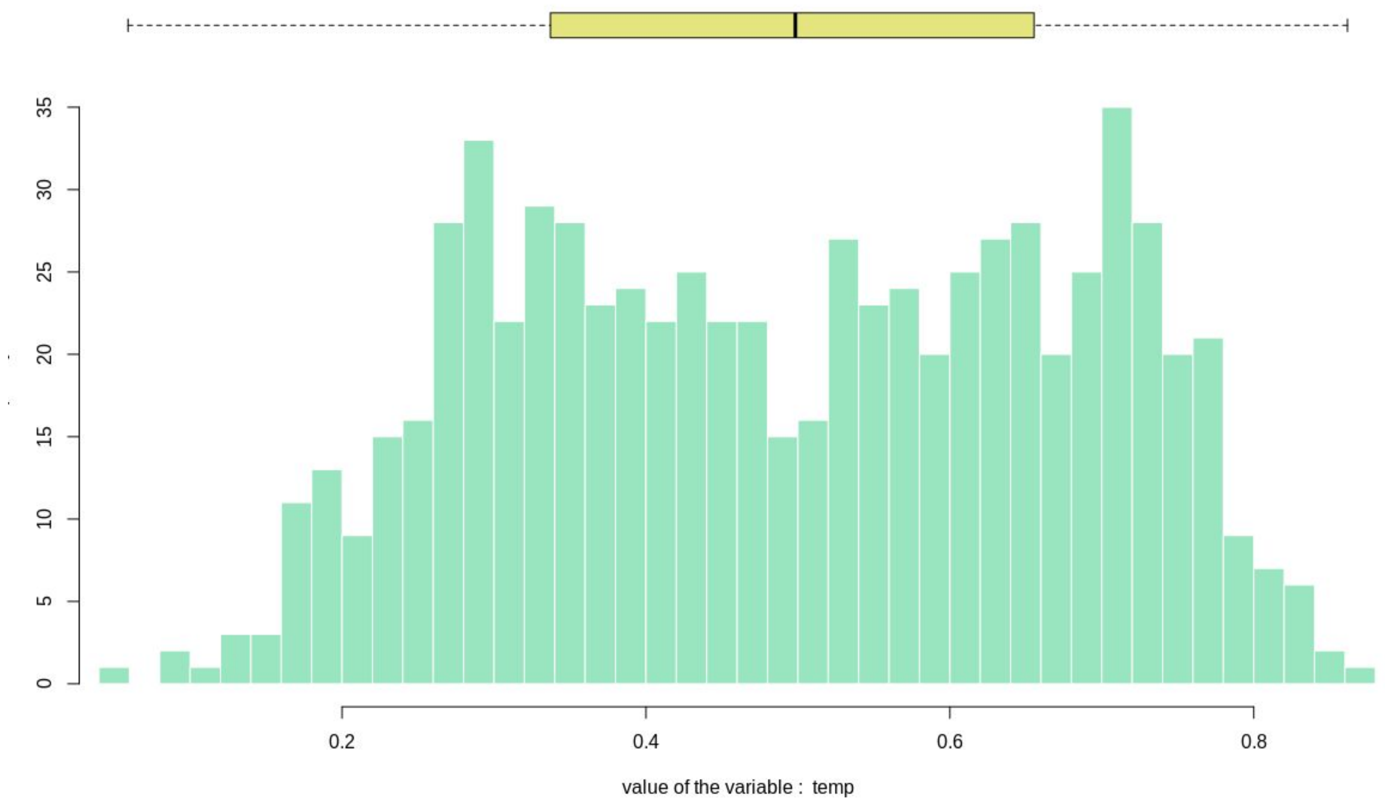


Fig 2.5 Box Plot of **windspeed** distributed over **season** (above)
Fig 2.6 Box & Histogram Plot of **temp** variable (below). No outliers.



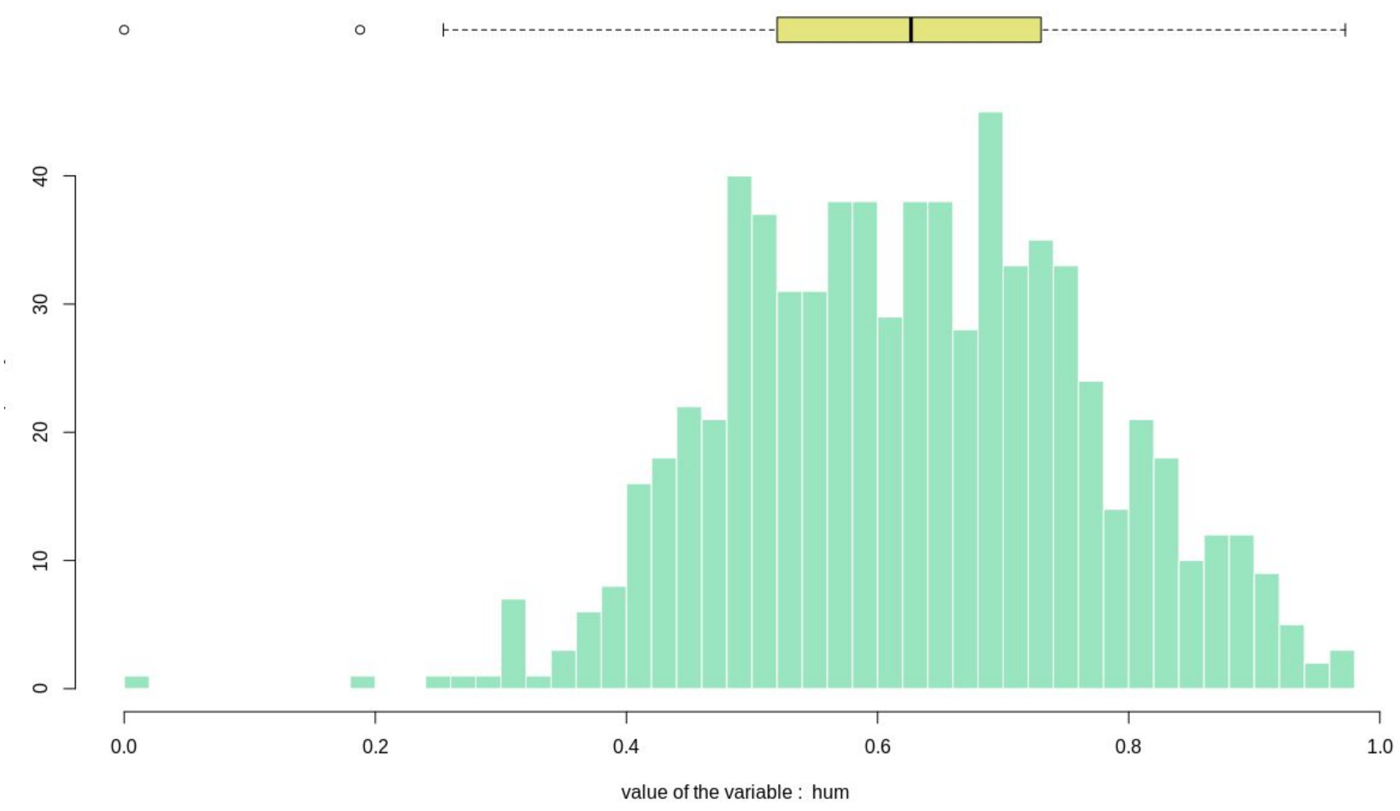
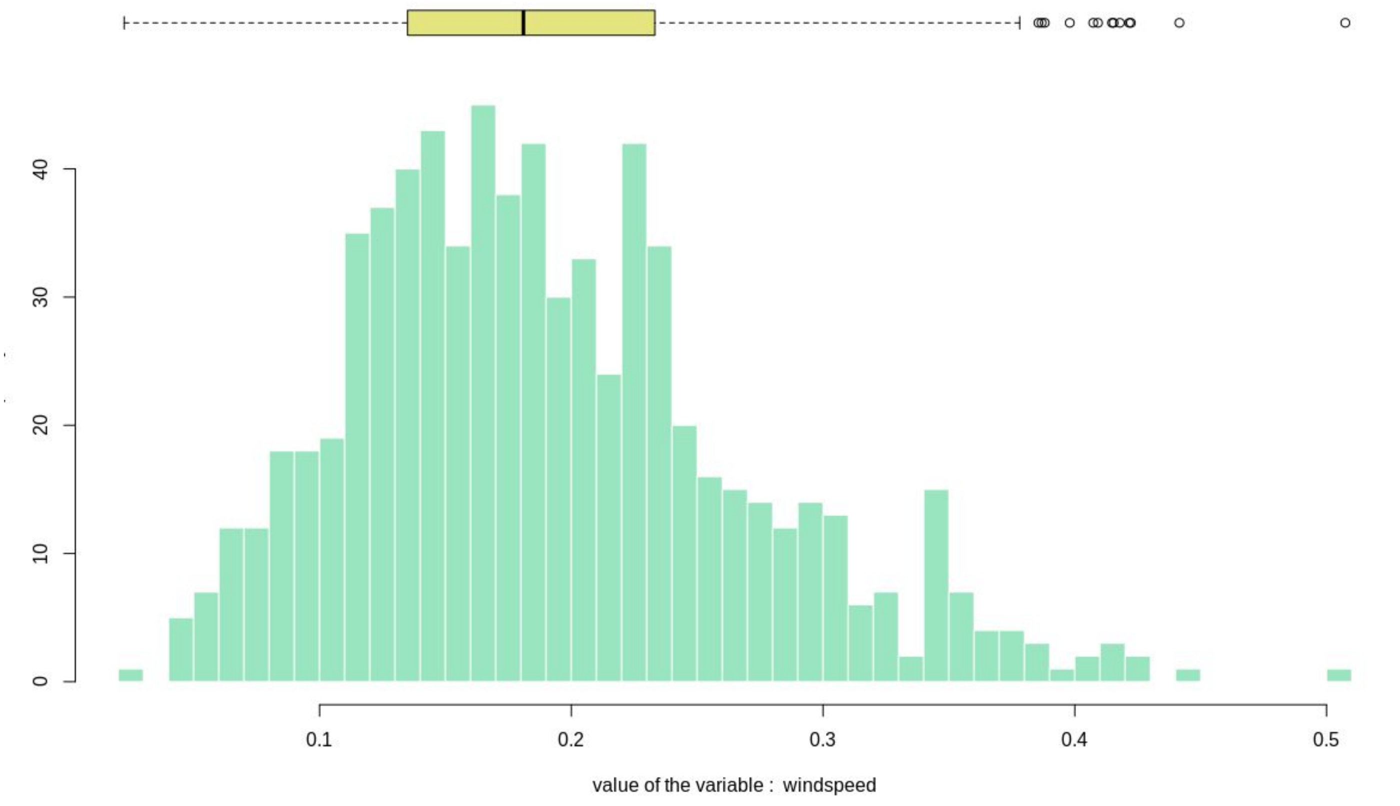


Fig 2.7 Box & Histogram Plot of **hum** variable (above). Few outliers. Left Skewed.
Fig 2.8 Box & Histogram Plot of **windspeed** variable (below). Some outliers. Right Skewed.



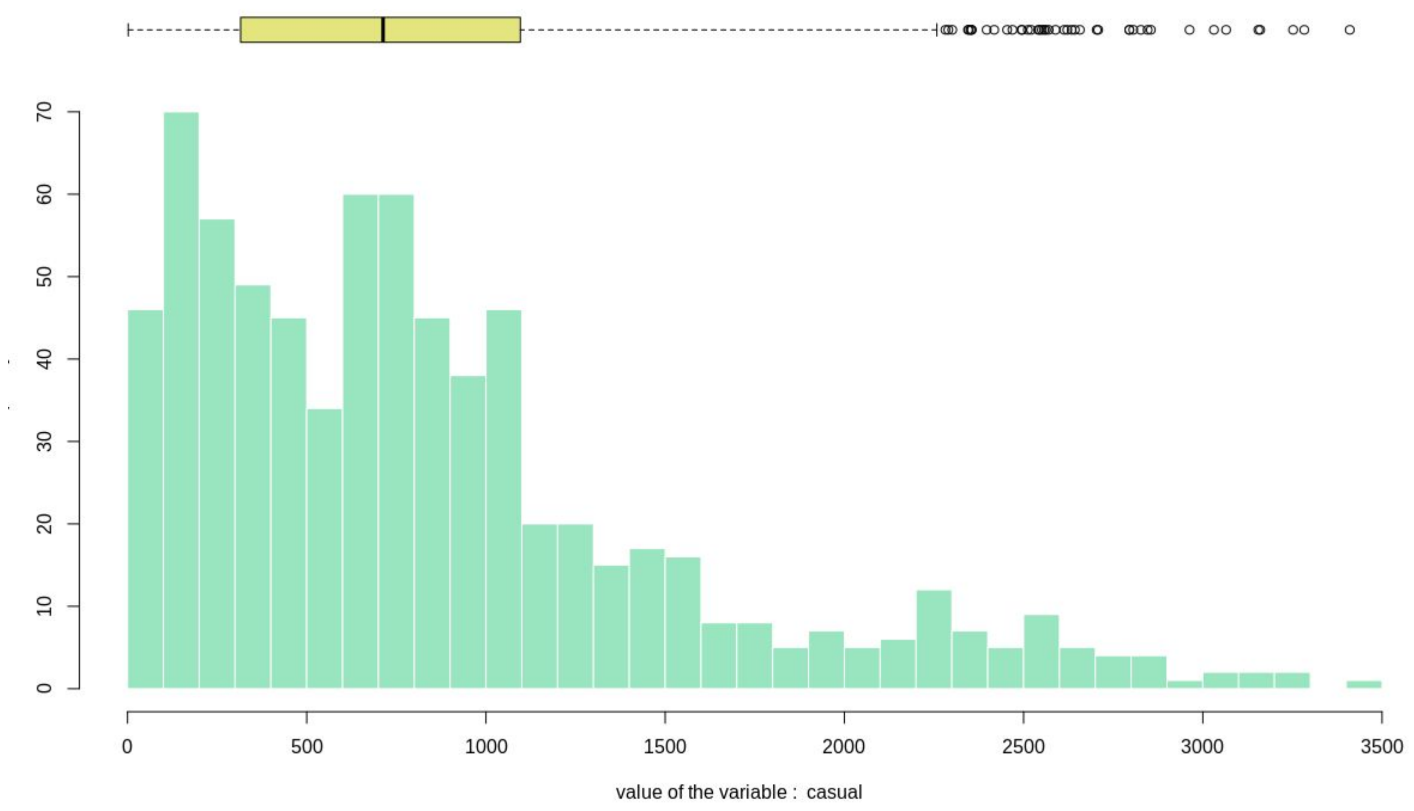
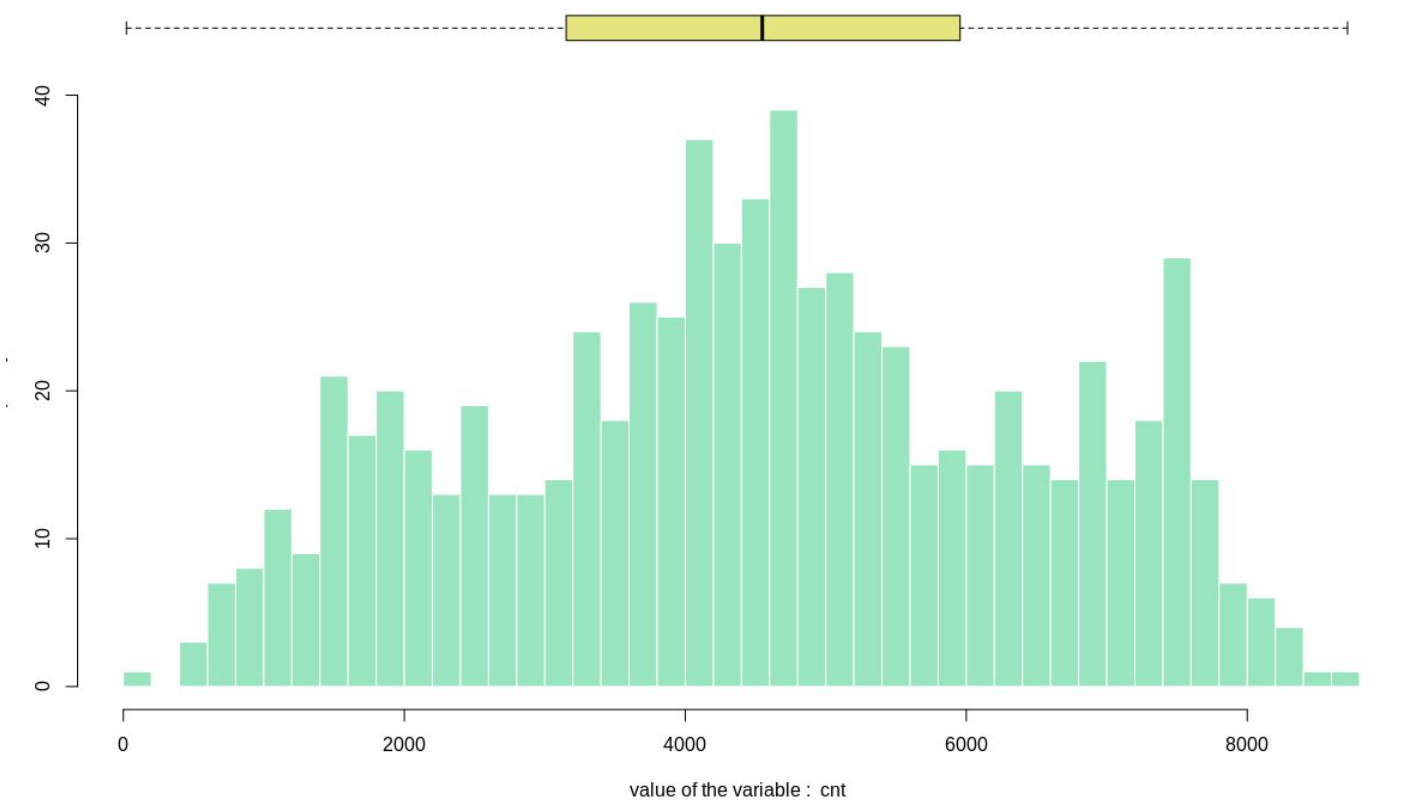


Fig 2.9 Box & Histogram Plot of **casual** variable (above). Many outliers. Right Skewed.
Fig 2.10 Box & Histogram Plot of **cnt** variable (below). No outliers.



2.3 Data Visualisation

Visualising data in the right way can and will give many insights as to how the variety contained in the data and the possible relationships in the data. I have used stacked bar plots to visualise data in a more effective way. A stacked bar plot is more effective because as compared to a usual bar plot which shows the distribution of a continuous variable over 1 categorical variable, a stacked bar plot shows distribution over 2 categorical variables. Below are a few stacked bar plots.

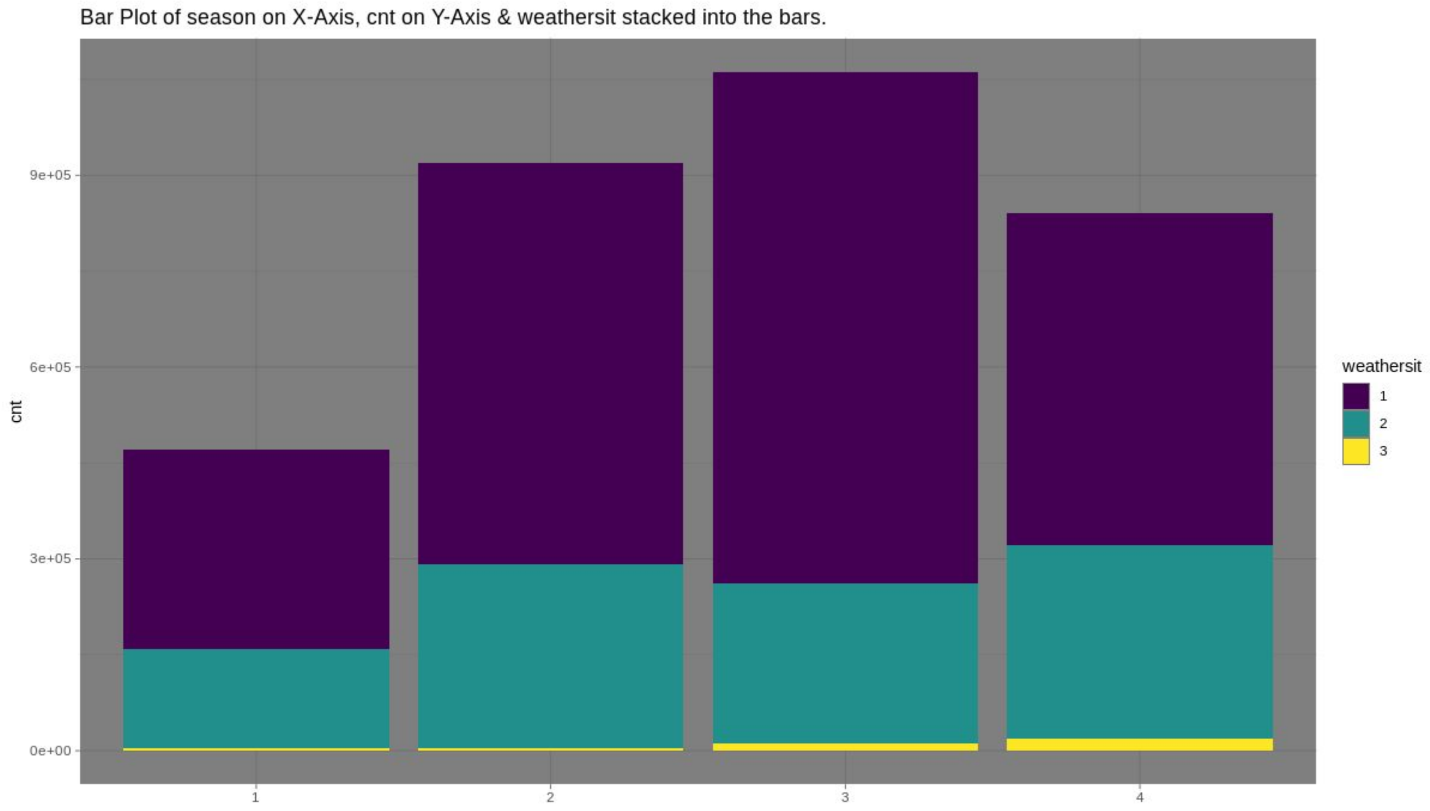


Fig 2.11 Stacked Bar Plot of **cnt** over different **season** & stacked by **weathersit**

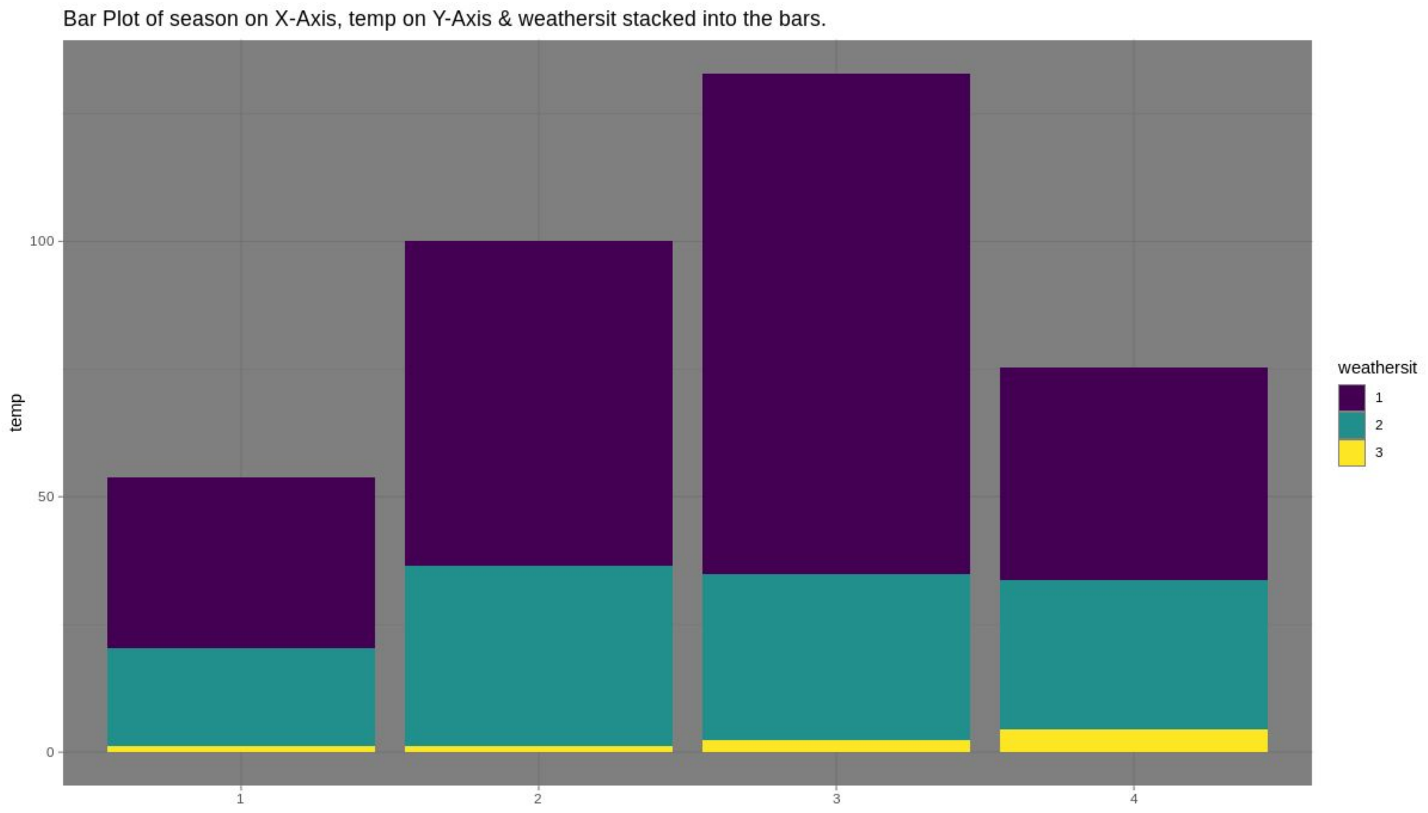
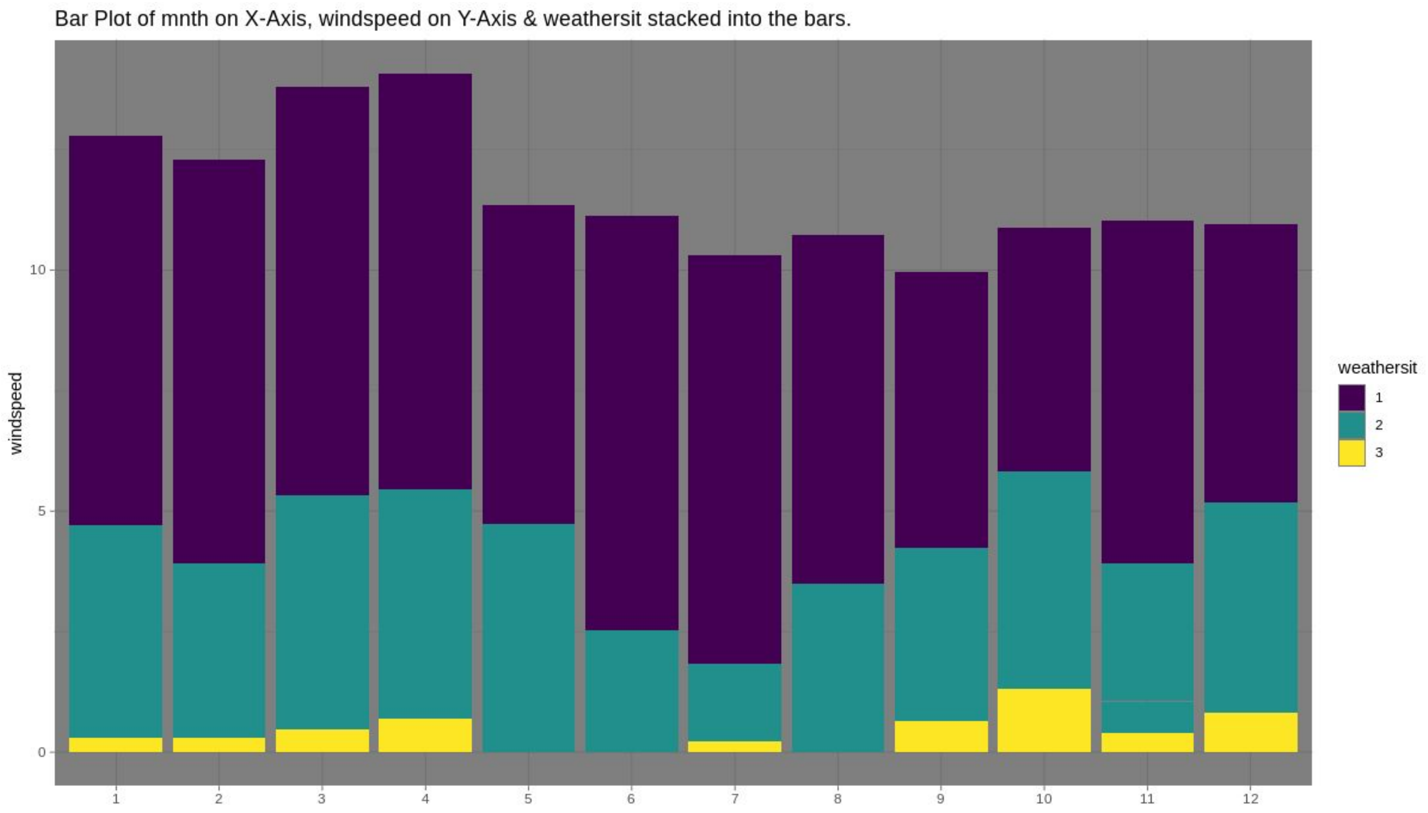


Fig 2.12 Stacked Bar Plot of **temp** distributed over **season** & stacked **weathersit** (above).
Fig 2.13 Stacked Bar Plot of **windspeed** over **season** & stacked **weathersit** (below).



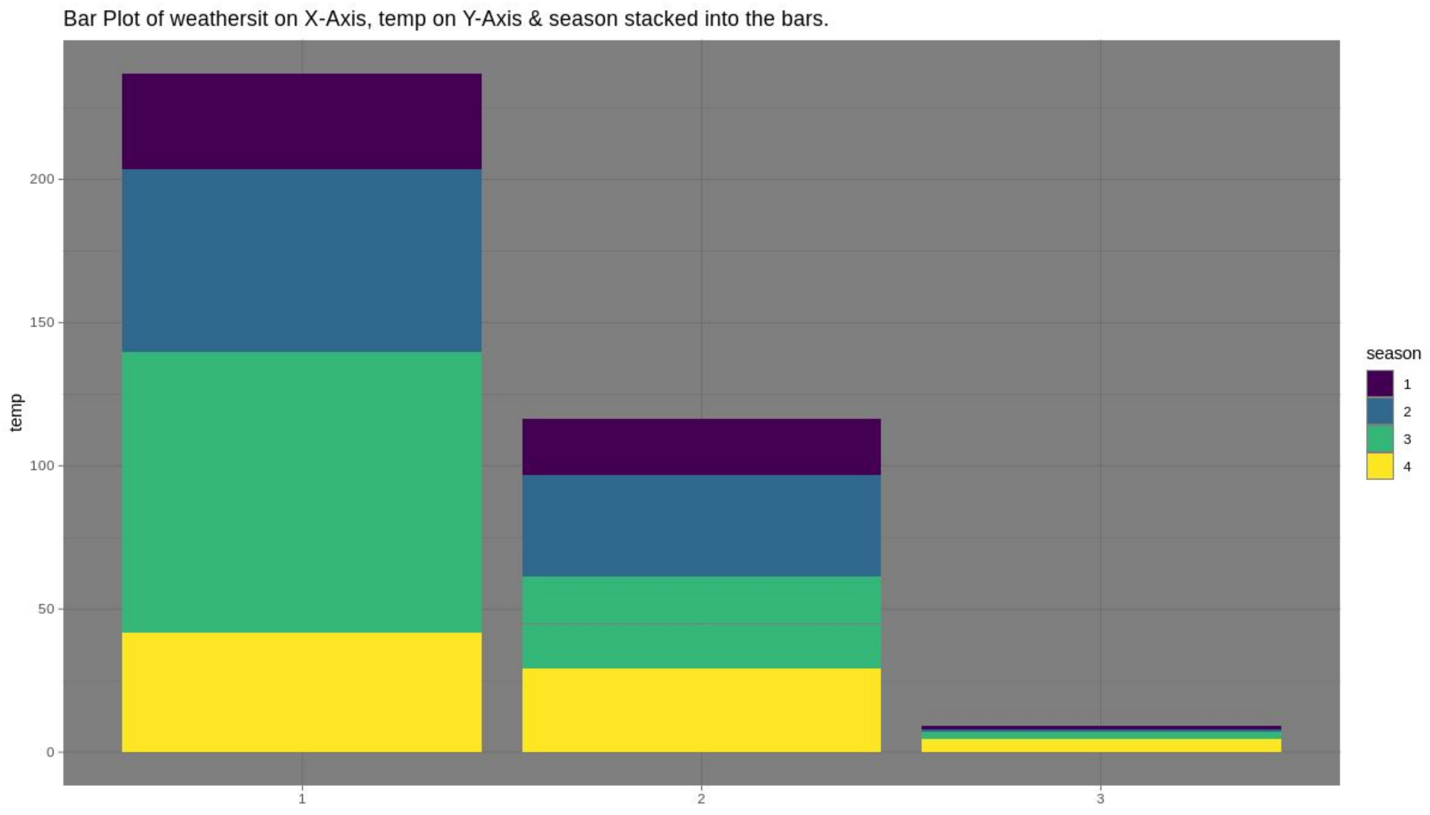


Fig 2.14 Stacked Bar Plot of **temp** over **weathersit** & stacked **season** (above).

2.4 Feature Engineering & Selection

Before putting my cleaned data into training models, we have to make sure that the features in the data are selected properly. There are few things to take care of, like:

- multicollinearity of few predictor variables
- encoding of the categorical variables
- standardising or normalising the numerical variables

Multicollinearity is checked for:

- 2 numerical variables using VIF (Variance Inflation Factor) score & a correlation plot or a heatmap
- 2 categorical variables using the Chi-Square test & hypothesis testing (threshold p-value = 0.05)

Below are some correlation plots & scatterplots for the numerical variables.

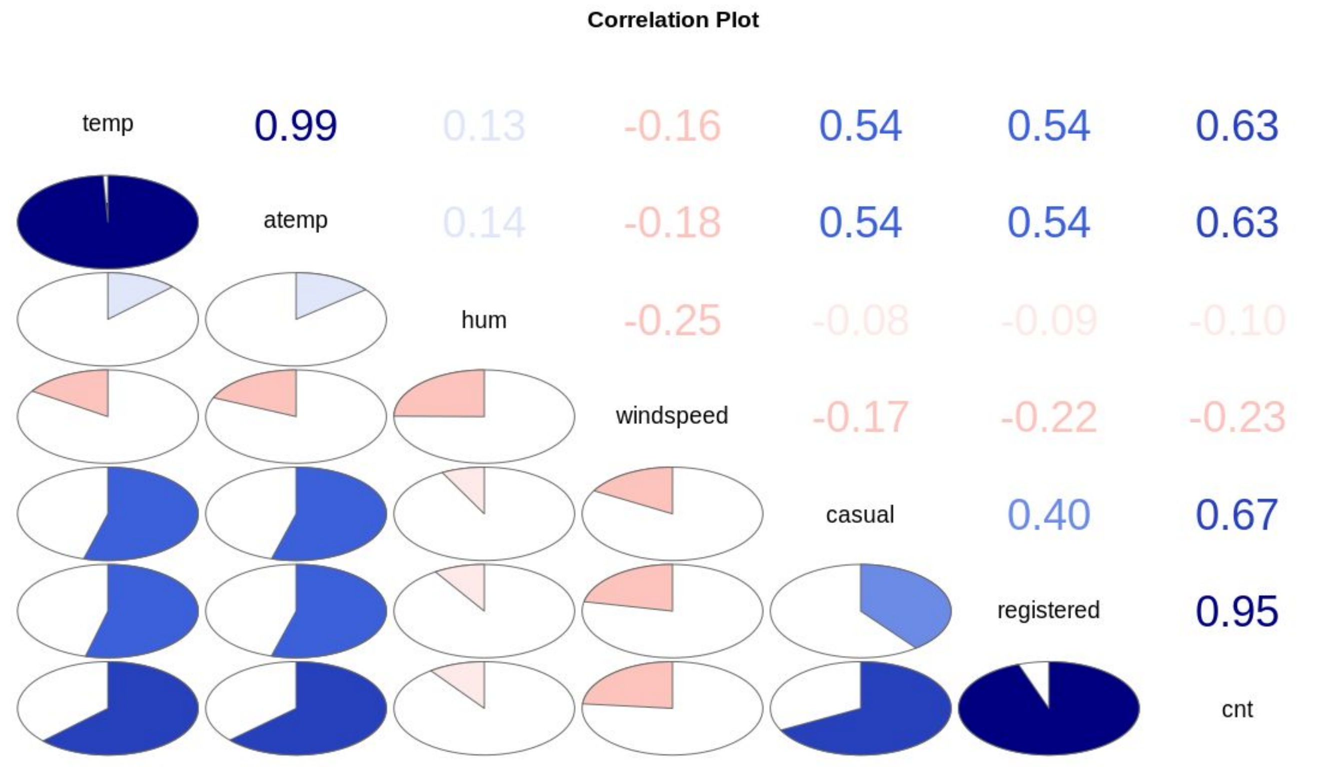
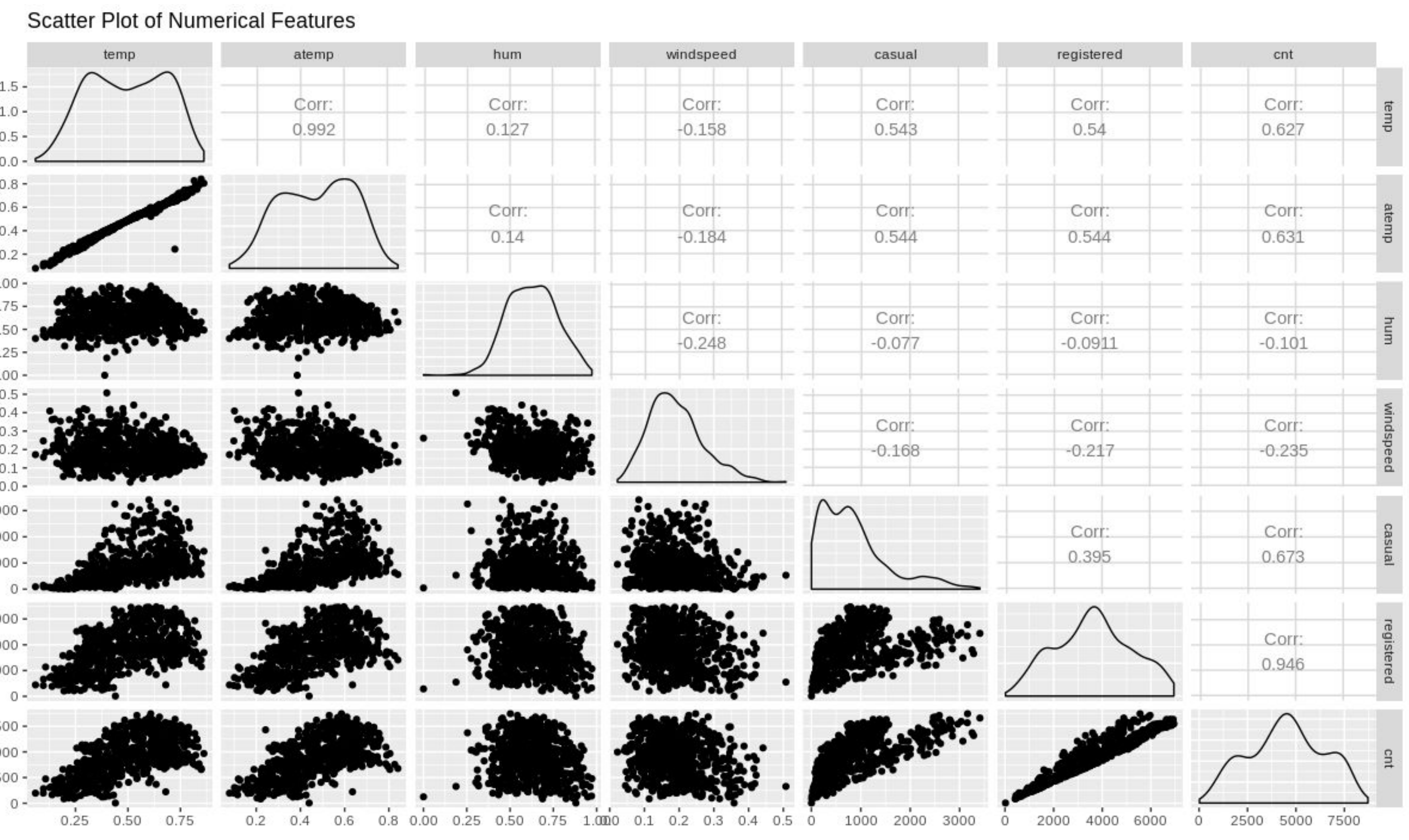


Fig 2.15 Correlation Plot between numerical variables of training dataset (above).
Fig 2.16 Scatterplot & Correlation plot of numerical features (below).



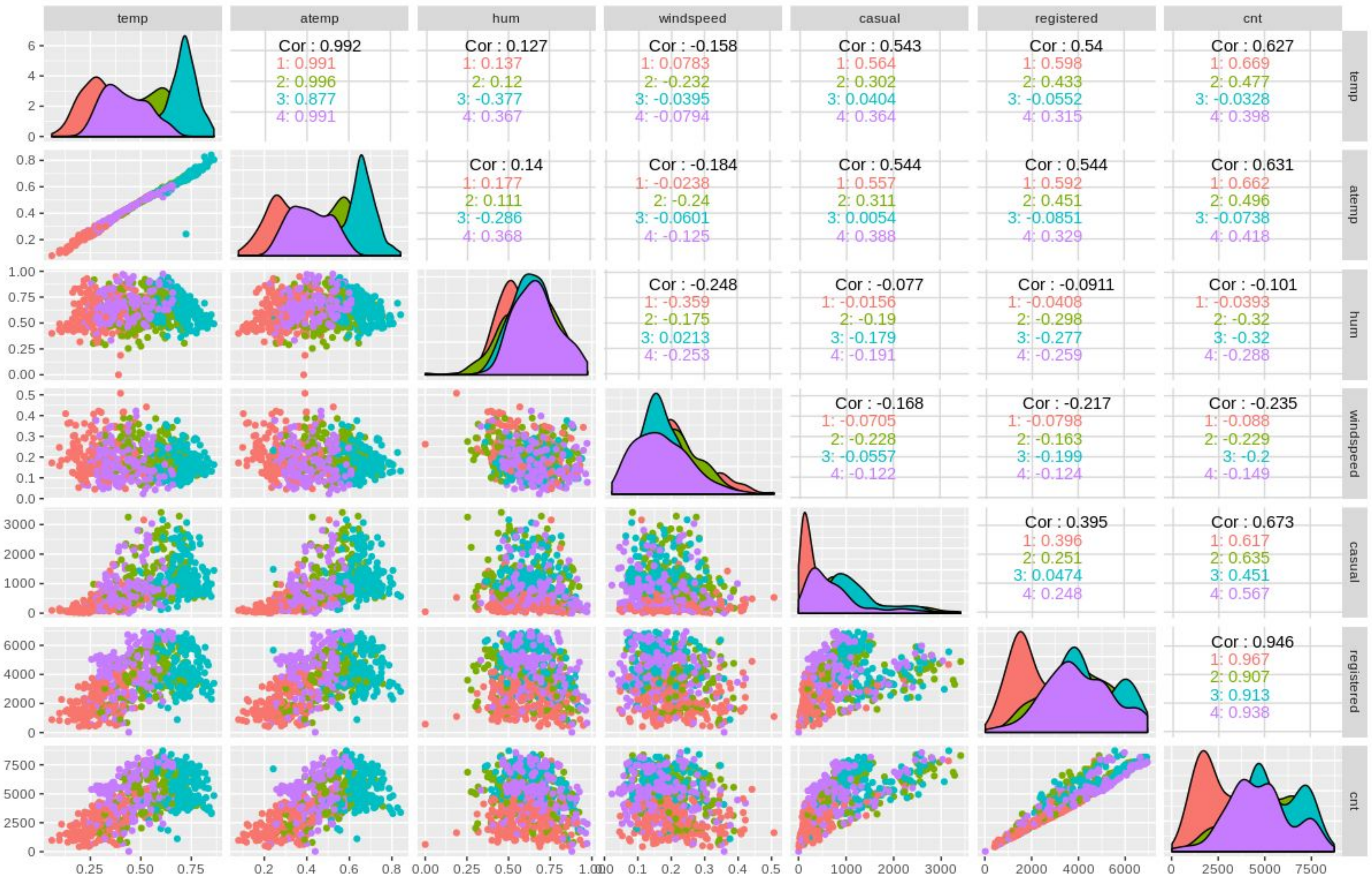


Fig 2.17 Scatterplot & Correlation plot of numerical features of training dataset, color coded using the categorical variable **season**. Overall correlation score & individual correlation score for every season is mentioned in each graph for each pair (above).

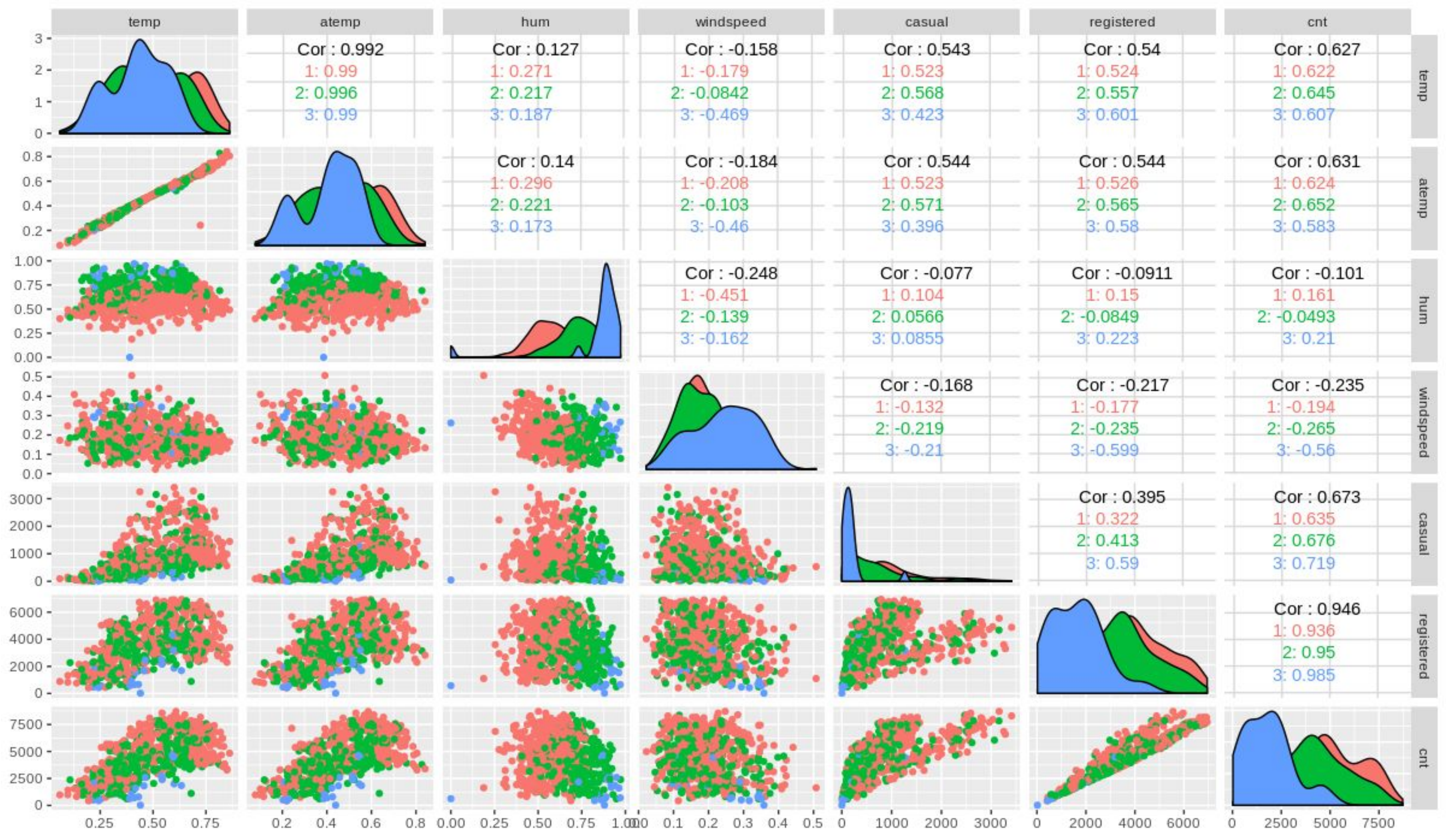


Fig 2.18 Scatterplot & Correlation plot of numerical features of training dataset, color coded using the categorical variable **weathersit**. Overall correlation score & individual correlation score for every season is mentioned in each graph for each pair (above).

After checking the VIF score and correlation plots, it was noticed that the variance was being inflated by 3 highly correlated numerical variables:

- *atemp*
- *casual*
- *registered*

These features will be dropped.

mnth & **weekday** are replaced by dummy variables (0 or 1) to represent:

- *mnth* is a winter_month or not (0 for winter_month, 1 otherwise). Rationale: number of bike rentals in winter season is less, being cold weather.
- *weekday* is a weekday or not (1 for weekday, 0 otherwise).

season & **weathersit** categorical variables are one-hot encoded, i.e. the **n** - levels in them are replaced by **n-1** dummy variables (0 or 1) representing each level. One level is dropped to avoid the ***dummy trap***. So **season** & **weathersit** will be dropped.

Chi-Square Test results show some correlated categorical variables that will be dropped:

- *holiday*: given weekday & workingday, holiday feature can be figured out. So dropping it is fine. Just like avoiding dummy trap.

Final predictor variables are:

- *temp (num)*
- *hum (num)*
- *windspeed (num)*
- *yr (cat)*
- *mnth (cat)*
- *weekday (cat)*
- *workingday (cat)*
- *season.2 (season dummy 2)*
- *season.3 (season dummy 3)*
- *season.4 (season dummy 4)*
- *weathersit.2 (weathersit dummy 2)*
- *weathersit.3 (weathersit dummy 3)*

Target variable is -> ***cnt***
Some plots after removing redundant columns are given below.

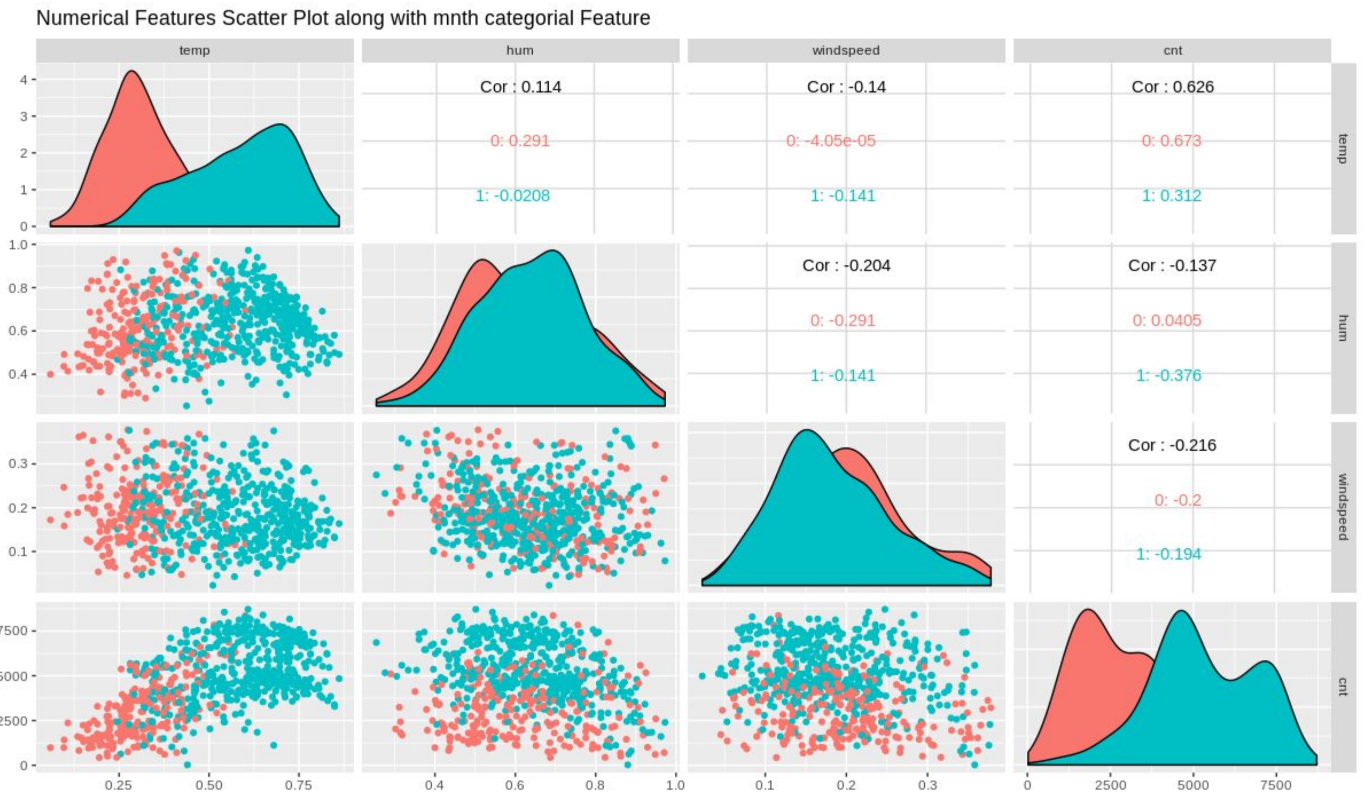
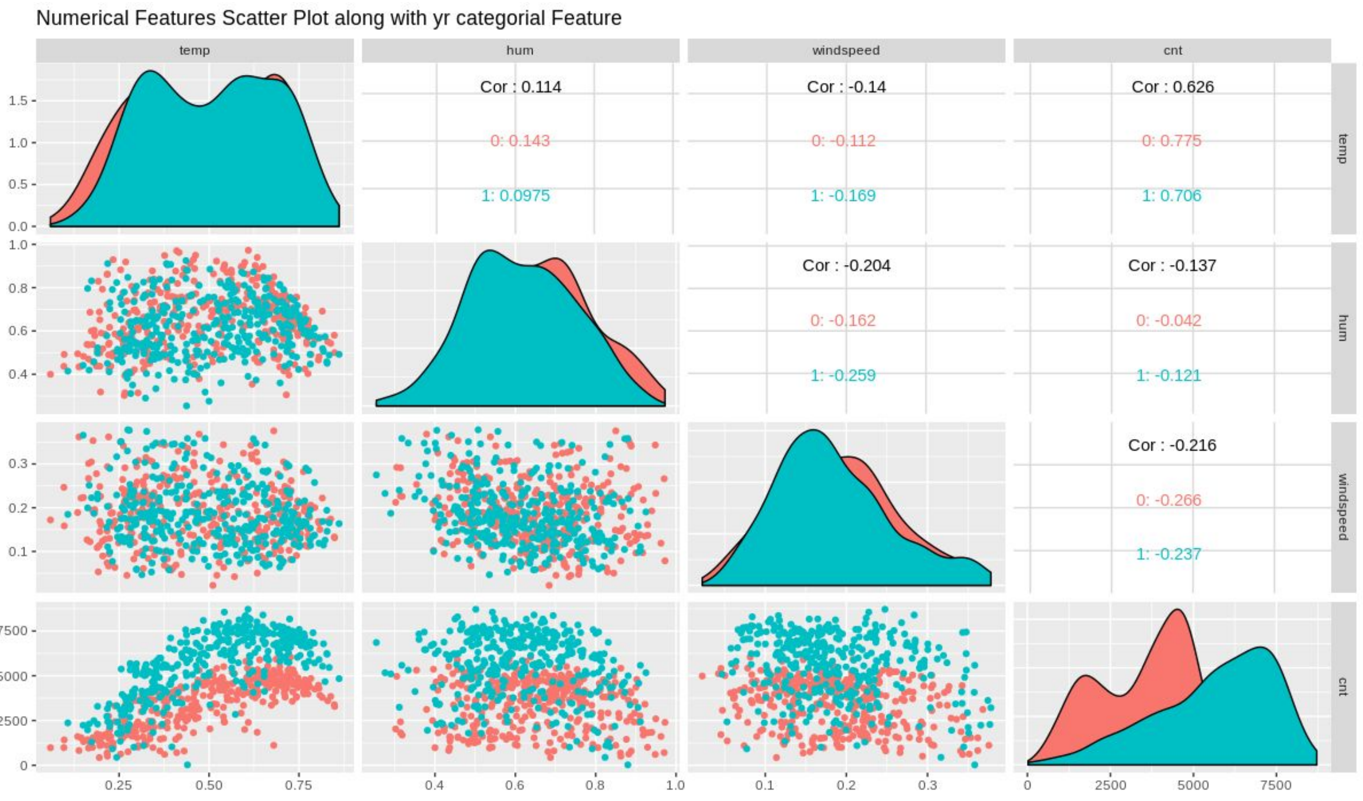


Fig 2.19 Correlation & Scatterplot of selected numerical features color coded by **mnth** (above).

Fig 2.20 Correlation & scatterplot of selected numerical features color coded by **yr** (below).



2.5 Modelling

2.5.1 Model Selection

Depending on what kind of value we have to predict, i.e. what kind of data our dependent variable is, we will choose what models to use. Data can be of the following types:

1. Nominal
2. Ordinal
3. Interval
4. Ratio

If we have to predict a nominal data, we will build **Classification** models. If we have to predict data of interval or ratio type, we build **Regression** models. Ordinal data can be predicted using both kinds of models. But, because our dependent variable is a **ratio** variable, we will be building Regression Models.

There are over 50 regression models that exist, but for this project, I have decided to focus on these Regression Algorithms & models:

1. **Linear Regression**
   Boosted Linear Model
   Simple Linear Model
2. **K-Nearest Neighbor**
3. **Decision Trees**
   CART - Classification And Regression Trees - 3 models
4. **Random Forest**
   Quantile Random Forest
   Parallel Random Forest
   Random Forest
   EXTremely RAndom Trees (EXTRA Trees)
5. **XGBoost**
   XGBDart
   XGBLinear
   XGBTree
   Stochastic Gradient Boosting

6. **Neural Network** (trial)

In total, 16 models have been built. The best performing model is then used to train on data without outliers & those 2 resulting models are used for predicting results of unseen data.

2.5.2 Model Evaluation Metric & CrossValidation

Commonly used metrics to evaluate predictions of a regression models are

1. RMSE - Root Mean Squared Error (lower is better)
2. MAE - Mean Absolute Error (lower is better)
3. R-Squared & Adjusted R-Squared - Explained Variability (higher is better)

For this dataset, I will be using **MAE** as the benchmark metric.
CrossValidation is performed in **repeatedcv** manner with 10 folds on data.

2.5.3 Hyperparameters Tuning

Every model built has been upgraded to perform optimally by tuning the hyperparameters. Hyperparameters are some of the parameters of the model that are set by the programmer & that affect the performance & output of the models & are used to find other parameters of the model.
The hyperparameters are specified in a **grid**, and models of all the values of hyperparameters in that grid is built & the ***best_tuned*** model is selected.

For the best performing model, extreme hyperparameter tuning is applied to increase the performance to the maximum, and by spending enough time it can be tuned to perform a lot better than right now.

Chapter 3

Conclusion

3.1 Models' Results

The results of the models are given below:

1. Linear Regression - lowest MAE achieved : ***601.28***
2. K-Nearest Neighbor - lowest MAE : ***557.68***
3. Decision Trees - lowest MAE : ***625.05***
4. Random Forest - lowest MAE : ***482.44***
5. XGBoost - lowest MAE : ***507.07***

3.2 Best Model Tuning & Plotting Predictions

The best performing model is **Random Forest** ("rf"), with hyperparameter value - *mtry* = 5, trained on data with outliers.

This model is tuned even more & trained on data within Quartile Ranges (i.e. without outliers), to achieve the **lowest MAE** : *283* on **unseen test data without outliers**.

This model is then stored to disk & reloaded to predict sample output on unseen data.

Below are some figures show the performance.

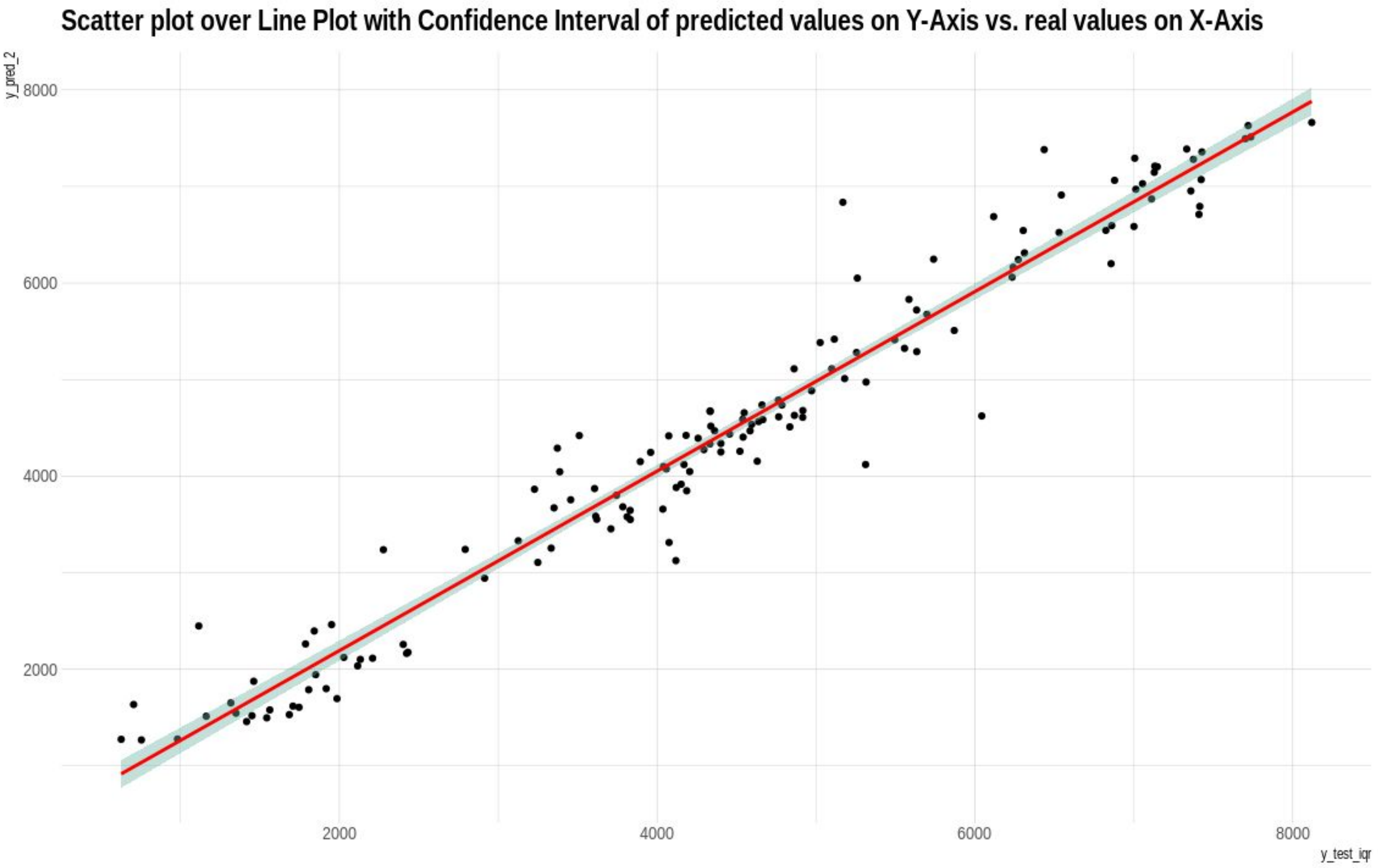


Fig 3.1 Expected Output (red line with confidence interval) & Predicted Output (black dots)

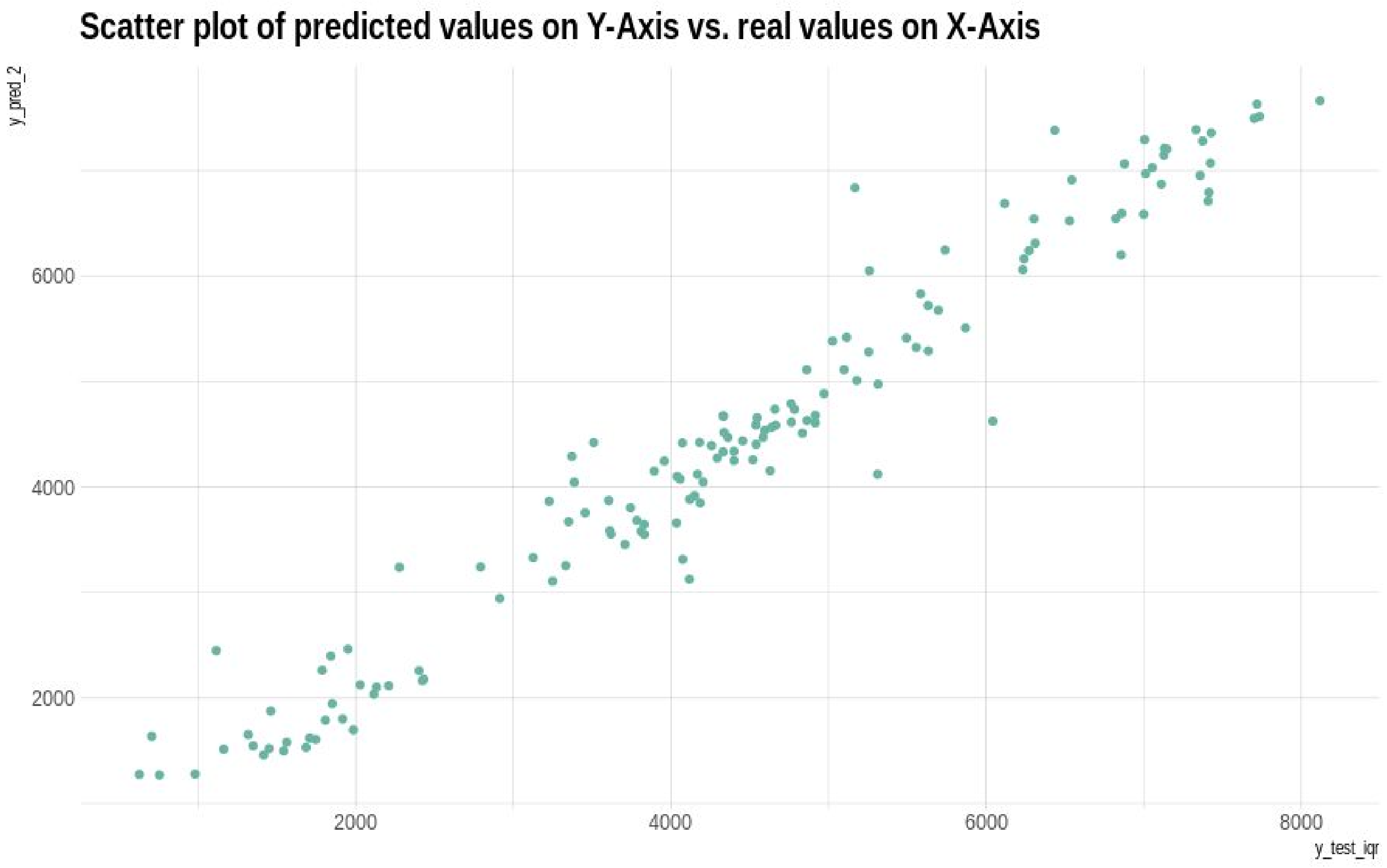


Fig 3.2 Scatterplot of predicted output.

3.3 Sample Input & Output

Sample input is provided in the form of a .CSV file having the same features in the same format as required by the model. It can be tweaked to handle other types of data as well.

References

https://topepo.github.io/caret/

https://www.r-graph-gallery.com/