Prediction of Bike Rental Count in Python

Sagar B.

2nd September 2019

Contents

Chapter 1

Introduction

1.1 Instructions to run the code file

- Extract the 'Bike_Rental_Prediction.zip' file.
- There will be two folders "Python" and "R".
- Open Jupyter notebook in the browser and upload 'Bike_Rental_Count_Prediction.ipynb' in the Python folder using the upload button provided on the home screen of Jupyter Notebook.
- After uploading is done, the notebooks can be accessed from the list on the home screen.

1.2 Problem Statement

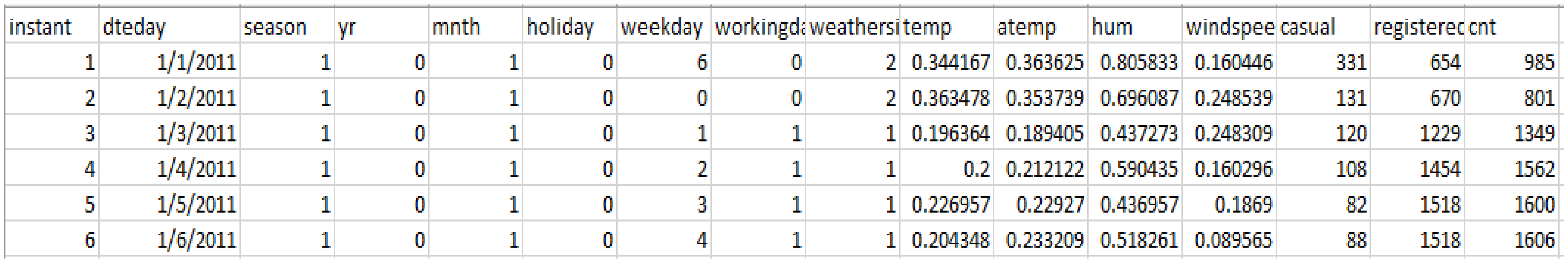
The objective of this Case is to Predication of bike rental count on daily basis based on the environmental and seasonal settings.

1.3 Dataset

Note that only one dataset with 731 observations was provided and there was no separate test dataset. The dimensions of the dataset were 731 x 16. The features in the dataset were as follows:

- Instant – It is the index number of the observations.
- Dteday – Date when the observation was recorded.
- Season – 1 : spring , 2 : summer, 3 : fall, 4 : winter
- Yr – 0 : 2011 , 1 : 2012
- Mnth – Months in a year from 1-12
- Holiday – 1 : Holiday , 0 : Not a Holiday
- Weekday – Day of the week from 0-6
- Workingday – 1 : Neither a holiday nor weekend , 0 : Holiday
- Weathersit – 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 – Temperature on that day (Normalized)
- Atemp – Feeling temperature (Normalized)
- Hum – Humidity (Normalized)
- Windspeed – windspeed (Normalized)

- Casual – Number of casual users who rented bike .
- Registered – Number of registered users who rented bike.
- Cnt – Total count Casual + Registered.



instant	dteday	season	yr	mnth	holiday	weekday	workingda	weathersi	temp	atemp	hum	windspee	casual	registerec	cnt
1	1/1/2011	1	0	1	0	6	0	2	0.344167	0.363625	0.805833	0.160446	331	654	985
2	1/2/2011	1	0	1	0	0	0	2	0.363478	0.353739	0.696087	0.248539	131	670	801
3	1/3/2011	1	0	1	0	1	1	1	0.196364	0.189405	0.437273	0.248309	120	1229	1349
4	1/4/2011	1	0	1	0	2	1	1	0.2	0.212122	0.590435	0.160296	108	1454	1562
5	1/5/2011	1	0	1	0	3	1	1	0.226957	0.22927	0.436957	0.1869	82	1518	1600
6	1/6/2011	1	0	1	0	4	1	1	0.204348	0.233209	0.518261	0.089565	88	1518	1606

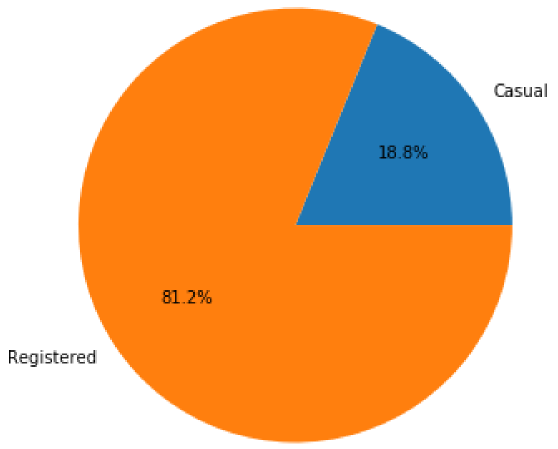
Above is a sample of the dataset on which we will be working on. The table shows first 6 rows with all its features.

Chapter 2

Data Cleaning and Exploratory Data Analysis

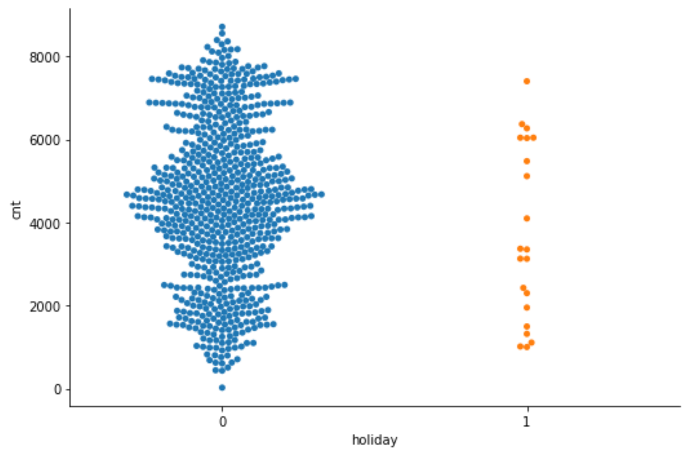
- First, we started off by importing all the necessary libraries in the python environment.
- Then, the dataset was imported and some preliminary observations of the dataset were made.
- There were no missing values present in the dataset.

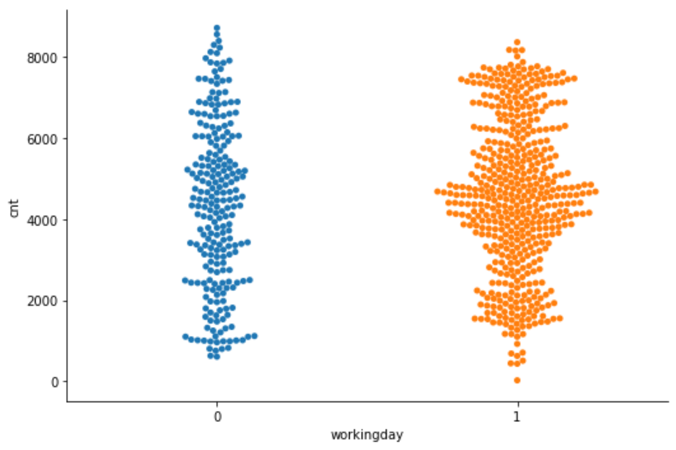
2.1 Pie Chart for comparing Casual and Registered users



- Then, some exploratory data analysis was done. Here, a pie chart was plotted to know how much percentage of casual and registered users make up for the total count. It was observed that registered users are more then casual users.

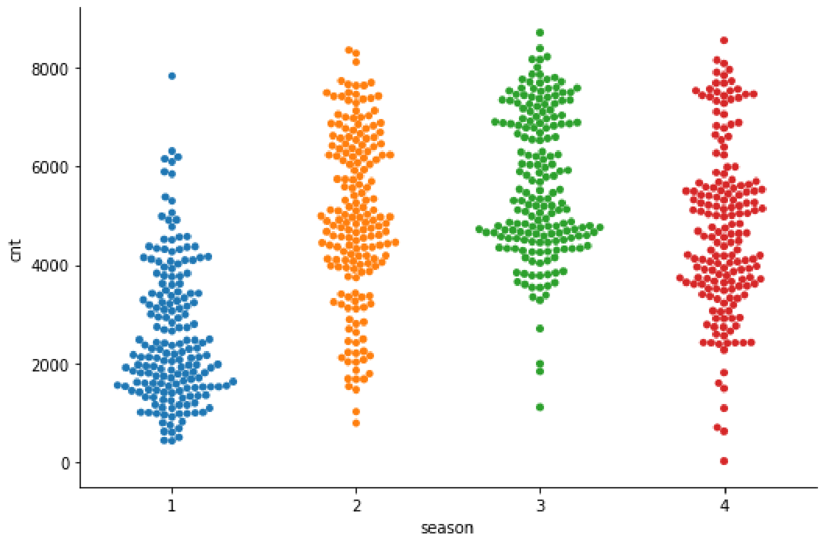
2.2 Effect of holiday and workingday on rental count





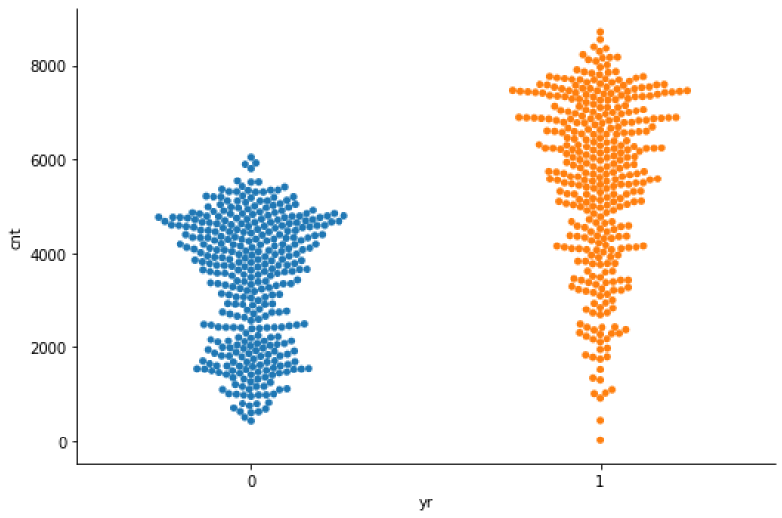
- From these graphs of holiday vs count and workingday vs count, we can infer that most people rent bikes on working days rather than on holidays. From workingday vs. count graph we can also infer that the count is bit significant for holidays, it may be that people mostly rent bikes on weekends rather than when they have holiday on other days.

2.3 Effect of season on rental count

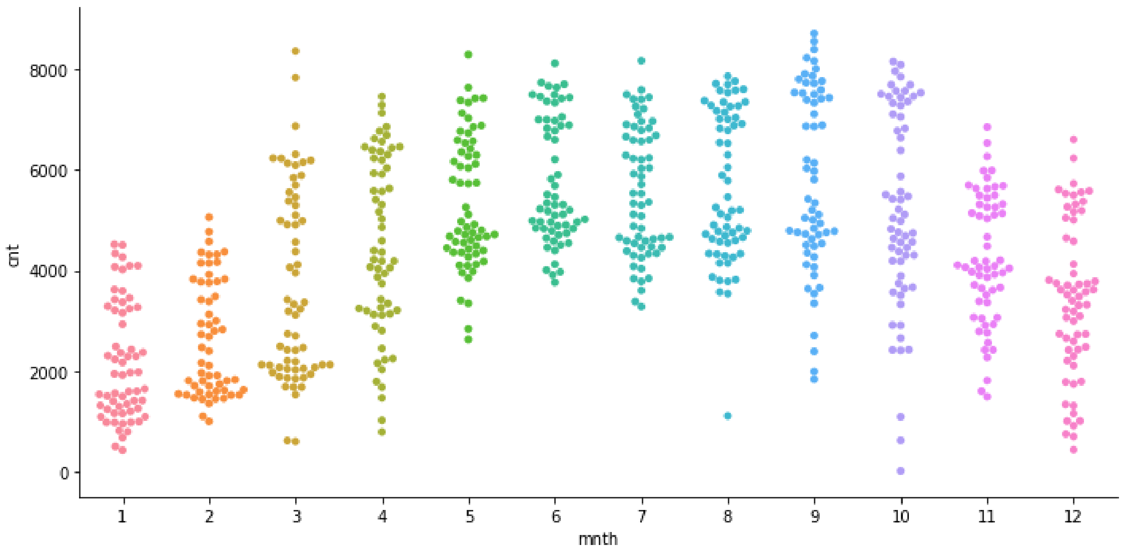


- From the graph of season vs count, we observe that count is least for spring season.

2.4 Effect of year and month on rental count

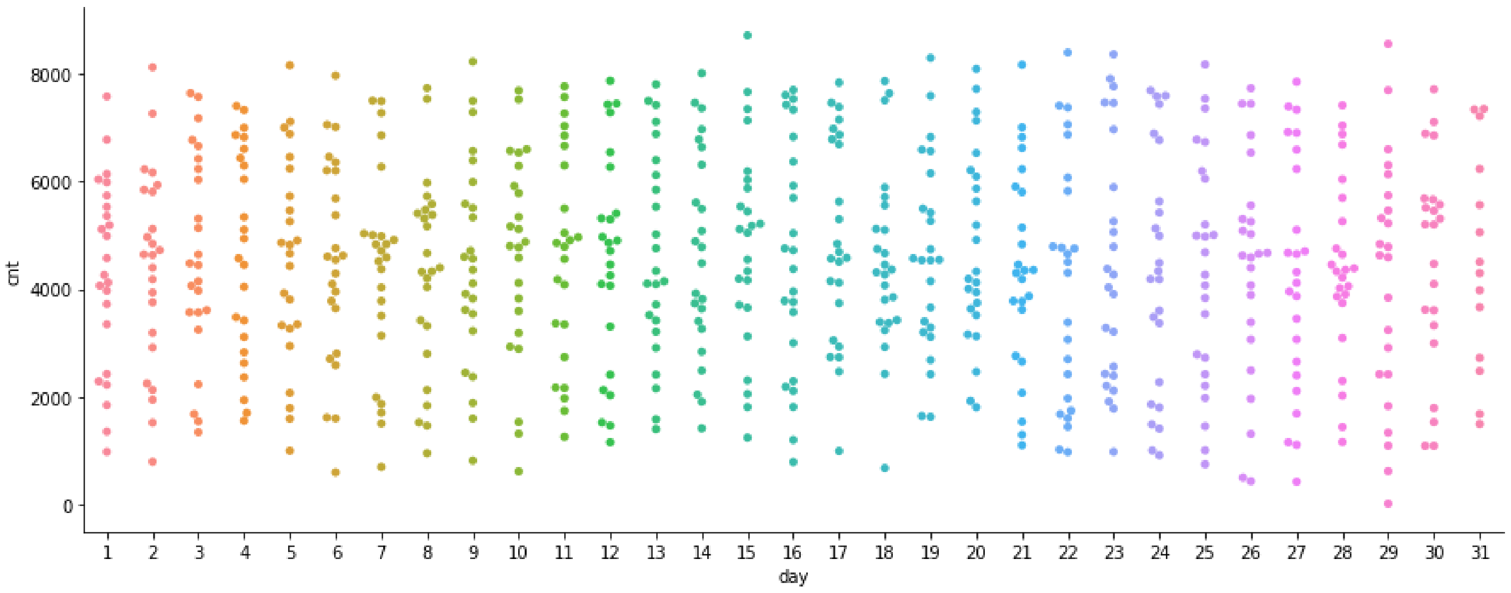


- From the graph of year vs count, we observe that rental count is higher in the year 2012.

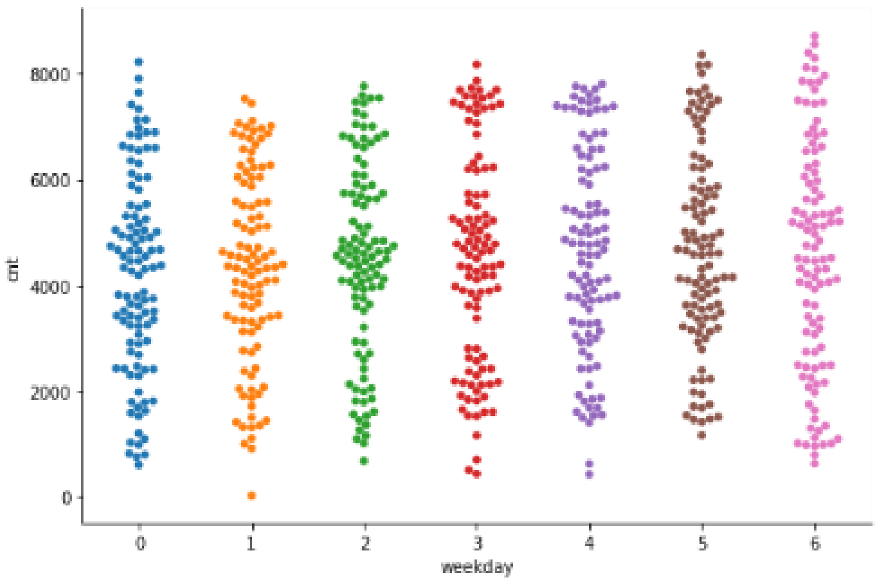


- From the above graph, we can infer that rental count is generally high from May to September.

2.5 Effect of date and weekday on rental count

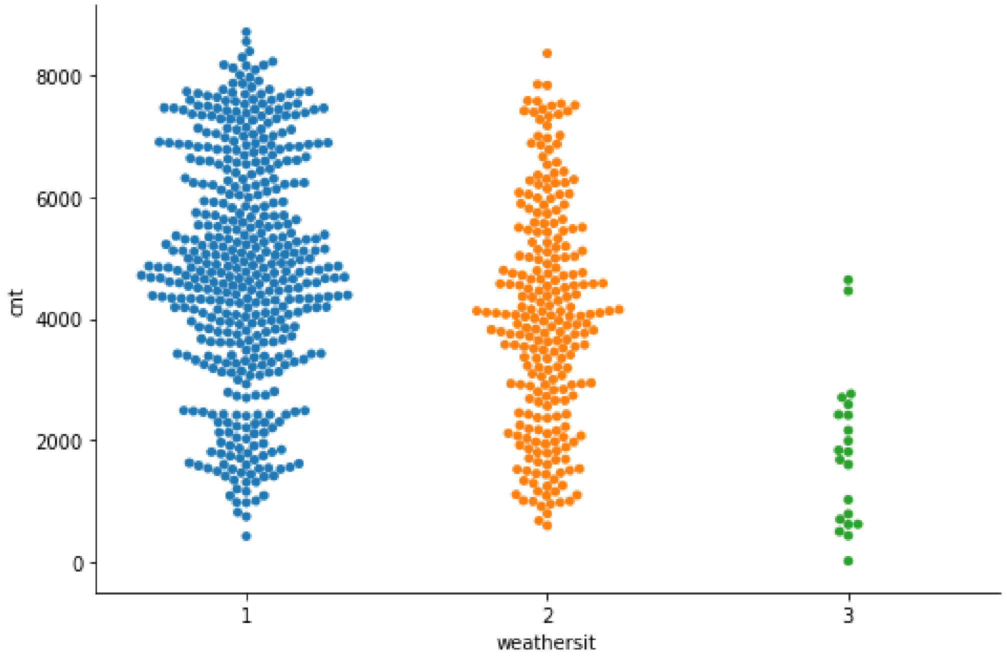


- The rental counts seem to be consistent throughout the month with few variations.



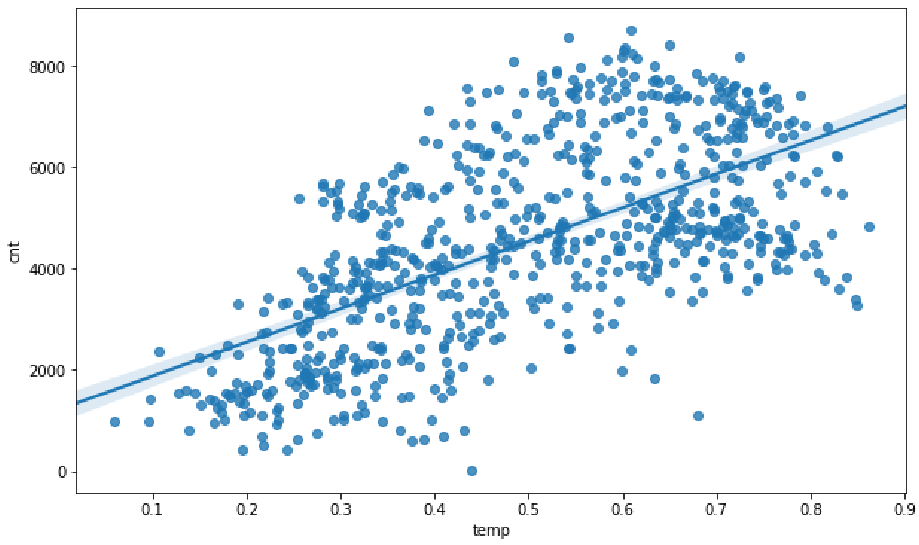
- The rental counts seem to be consistent throughout the week. Although, highest seem to be on Sunday (6).

2.6 Effect of weather on rental count

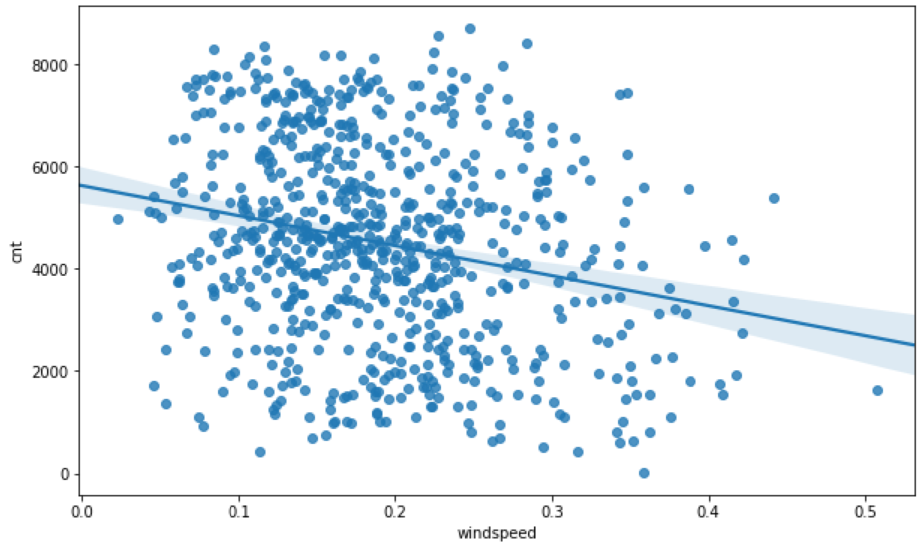


- From above graph of weather vs count, we can infer that most people rent bike when the weather is clear. The rental count decreases as the weather turns bad.

2.7 Effect of temperature and windspeed on rental count

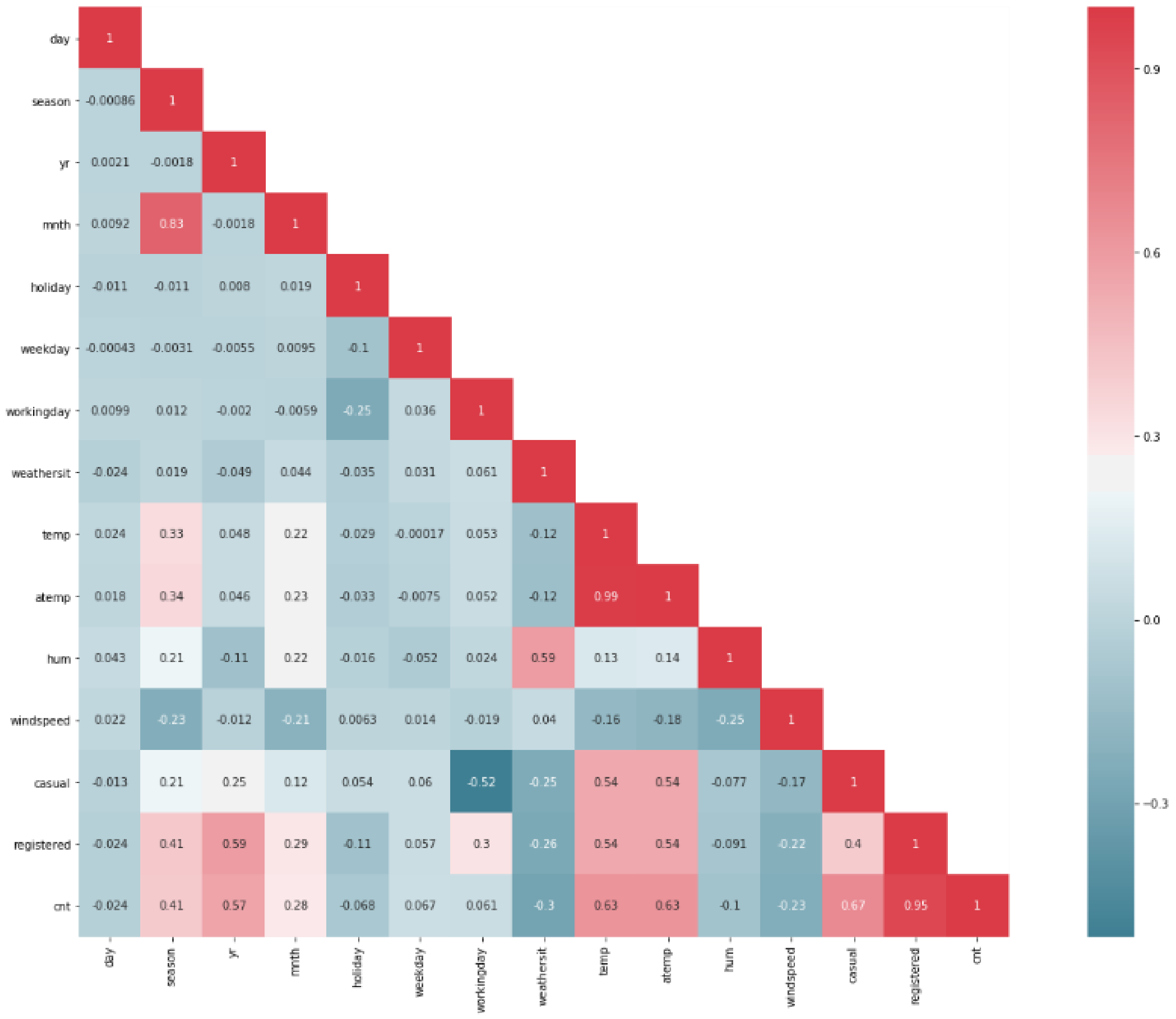


- From scatter plot of temperature vs count, we observe that very few people rent bike when it is cold. The count increases as temperature increases. Here, there is a linear relationship.



- Here, in scatter plot of windspeed vs count, we see that as the windspeed increases, the rental count decreases. This indicates an inverse relationship.

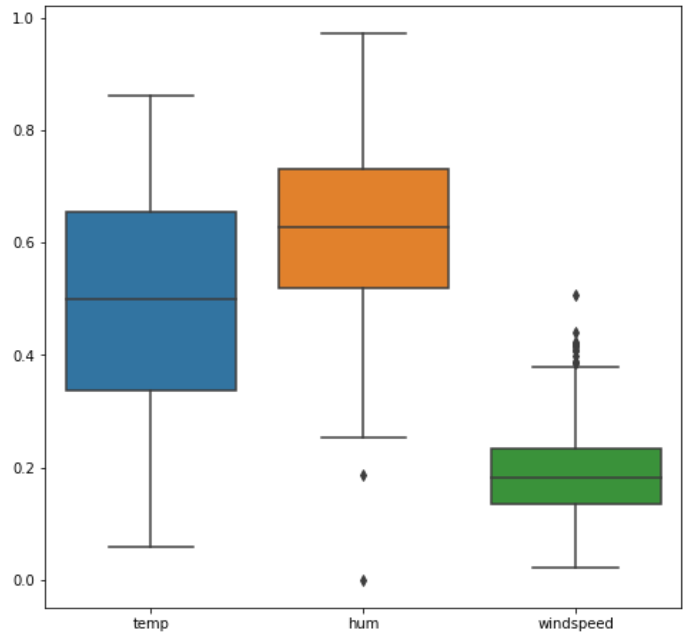
2.8 Correlation plot

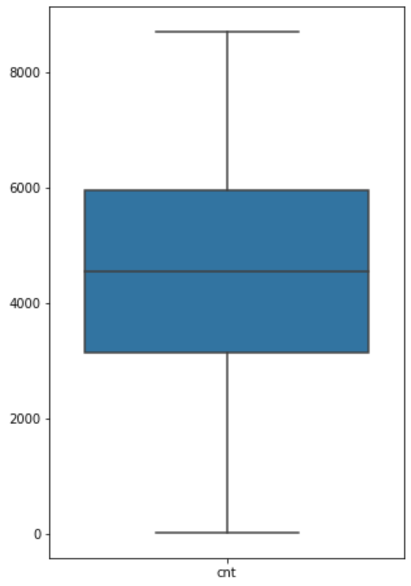


- Finally, we observe correlation plot. Here, we can see that temp and atemp are highly positively correlated and convey the same information. Thus, it is better to drop either one. In this case, atemp was dropped. Also, the count variable is basically sum of casual and registered variables and since we are only concerned with the total rental count prediction based on external factors, these variables were removed.

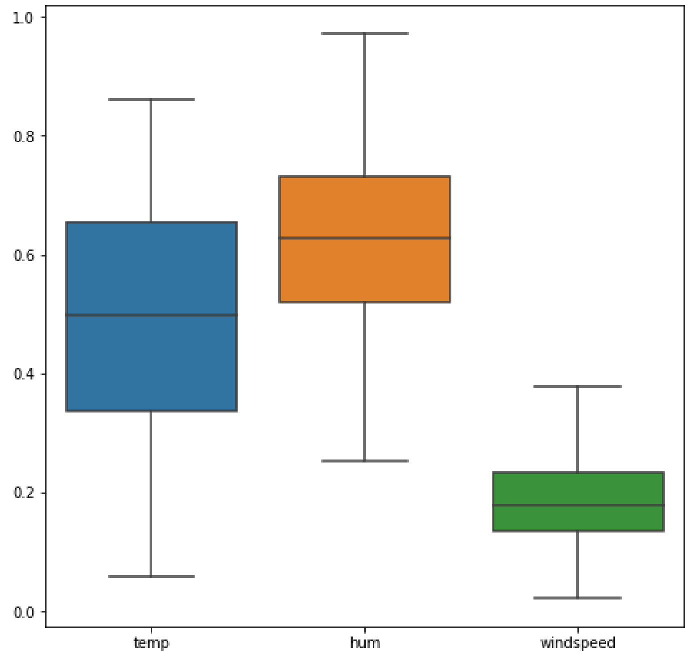
2.9 Outlier Analysis

- Now, since this is a regression problem and we have categorical variables like season and weathersit with multiple categories, there is a chance that our algorithm may take these categories as numeric values. To avoid this, we create dummy variables for each category.





- Then, we checked for outliers in continuous variables. Only variables – humidity and windspeed contained outliers as shown above.



- These outlier values were replaced with NaN and imputed using MissForest library which is a Random Forest based imputer.

Chapter 3

Modelling and Rental Count Predictions

- Here, we start of by importing important libraries for our modelling.
- We then split our data into train (80%) and validation (20%).
- Then a function was created to calculate error metrics.
- Now, we move on to modelling phase. Note that basic multiple linear regression was implemented but it was predicting negative count values. So, the model was not considered for further analysis.

  Note: Even though many error metrics were calculated, the focus was to minimize the RMSLE value. We have chosen RMSLE for this project because the count values are in the range of thousands and thus other error metrics will be either in the range of hundreds or thousands. If we use log-based error metric like RMSLE, we can get a better scale to interpret the performance of our model.

3.1 Implementation of models

i. **Random Forest Regression: -**
This is ensemble-based regression model in which several decision trees are generated where each of them has equal vote to decide what the prediction value of the target variable should be.
Here, Random Forest gave RMSLE of 0.2046 for train data and 0.2768 for validation data.

ii. **AdaBoost Regression: -**
Similar to random forest but generates several stumps rather than trees and feeds the error from previous stump to the next stump. Thus, as consequent stumps are influenced by the errors of previous stumps, all stumps will not get equal vote on the final value of the prediction.
Here, AdaBoost gave RMSLE of 0.2751 on train data and 0.3319 on validation data.

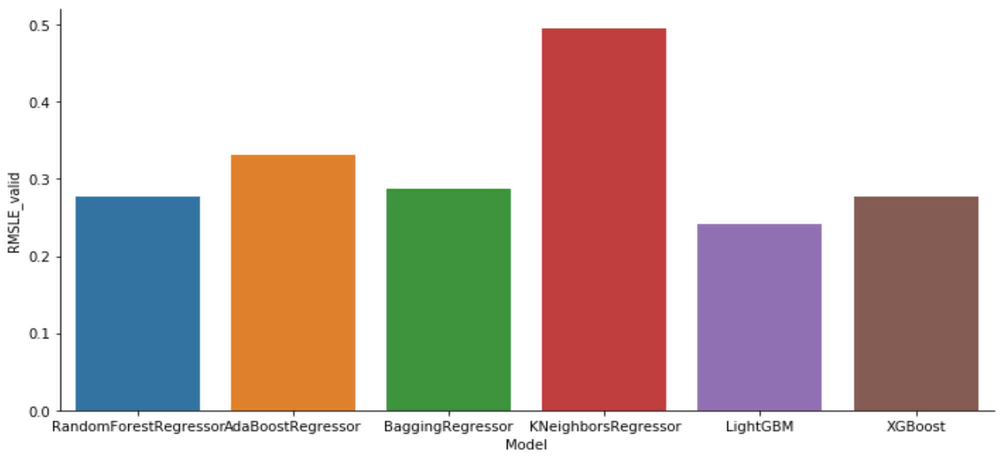
iii. **Bagging Regression: -**
This is also an ensemble-based learning algorithm. Here, there are several learners (or trees) that are trained on different data. The training data is randomly sampled and bagged with replacement.
Here, Bagging gave RMSLE of 0.1696 on train data and of 0.2867 on validation data.

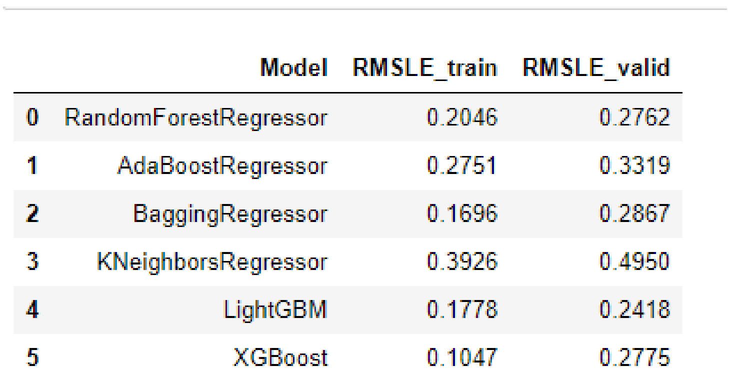
iv. **KNN Regression: -**
This algorithm predicts the target variable based on the patterns in the training data and decides the prediction value based on the K – nearest neighbors.
KNN gave RMSLE of 0.3926 on train data and of 0.495 on validation data.

v. **Light Gradient Boost Regression: -**
LGBM has trees of fixed length and are shorter than the trees in decision tree and random forest. It is also based on ensemble method like random forest, but the difference is that all the trees in LGBM do not have equal say on the output. The say (or vote) of each tree depends upon the error of previous tree and how that current tree reduces the error.
In the model, most of the default parameters were set and we implemented our model.
Here, LGBM gave RMSLE of 0.1778 on training data and of 0.2418 on validation data.

vi. **Xtreme Gradient Boost Regression: -**
This model is similar to LGBM. In fact, LGBM improves upon XGBoost.
Here, XGBoost gave RMSLE of 0.1047 on training data and 0.2775 on validation data.

3.2 Performance Summary of models





	Model	RMSLE_train	RMSLE_valid
0	RandomForestRegressor	0.2046	0.2762
1	AdaBoostRegressor	0.2751	0.3319
2	BaggingRegressor	0.1696	0.2867
3	KNeighborsRegressor	0.3926	0.4950
4	LightGBM	0.1778	0.2418
5	XGBoost	0.1047	0.2775

3.3 Hyperparameter Tuning

- As, we can see, Light Gradient Boosting algorithm gave us the least RMSLE for validation data. To further improve its performance, some hyperparameter tuning was done on LightGBM using Grid Search.
- The best parameters found were:

```
params = {
        'boosting_type':'gbdt',
        'objective': 'regression',
        'nthread': -1,
        'verbose': 0,
        'num_leaves': 25,
        'learning_rate': 0.075,
        'max_depth': 5,
        'subsample': 0.9,
        'subsample_freq': 1,
        'colsample_bytree': 0.6,
        'reg_aplha': 1,
        'reg_lambda': 0.001,
        'metric': 'rmse',
        'min_split_gain': 0.5,
        'min_child_weight': 1,
        'min_child_samples': 10,
        'scale_pos_weight':1
    }
```

- By fine tuning our LGBM model we reduced our RMSLE from 0.2418 to 0.2398 on validation data.

3.4 Prediction on Sample Dataset

- As separate test dataset was not provided to make predictions, a sample dataset was made and values were predicted using our trained LGBM model.
- The file was saved as ‘Final_Predictions_Python.csv’ and is included in the folder.

Chapter 4

Conclusion

- The aim of this project was to predict the bike rental count value based on various factors such as season, weather, temperature, windspeed, holiday/ working day etc.
- For this, we first gained insights from the data about the features and how they were related to our target variable 'count'.
- From above EDA, we gained following key insights: -
    - More registered users rent bikes as compared to casual users.
    - The rental count is more on working days rather than on holidays.
    - The rental count was high in the year 2012.
    - The count was high from May to September.
    - As weather gets bad, the rental count decreases.
- Then we trained several models and out of them, LightGBM proved to be the best for our case as it gave least RMSLE of 0.2398 on validation after hyperparameter tuning.
- A sample dataset with random observations was created and the count value was predicted for each case using our trained LightGBM model.