Predicting Bike Rental Count

By

Kiran Kumar

24 May 2019

Contents

Chapter 1

Introduction

1.1 Problem Statement

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

1.2 Data

The details of data attributes in the dataset are as follows -

instant: Record index

dteday: Date

season: Season (1:spring, 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 from Holiday Schedule)

weekday: Day of the week

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

weathersit: (extracted from Freemeteo)

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

Given below is a sample of the data set that we are using to predict the bike rental count:

	instant	dteday	season	yr	mnth	holiday	weekday	workingday	weathersit	temp	atemp	hum	windspeed	casual	registered	cnt
0	1	2011-01-01	1	0	1	0	6	0	2	0.344167	0.363625	0.805833	0.160446	331	654	985
1	2	2011-01-02	1	0	1	0	0	0	2	0.363478	0.353739	0.696087	0.248539	131	670	801
2	3	2011-01-03	1	0	1	0	1	1	1	0.196364	0.189405	0.437273	0.248309	120	1229	1349
3	4	2011-01-04	1	0	1	0	2	1	1	0.200000	0.212122	0.590435	0.160296	108	1454	1562
4	5	2011-01-05	1	0	1	0	3	1	1	0.226957	0.229270	0.436957	0.186900	82	1518	1600

As you can see in the table below we have the following 12 variables, using which we have to correctly predict the bike rental count:

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

We have dropped the following variables from our dataset:

- **Instant**- it represents the index of a record
- **Casual** and **Registered**- they are leakage variables in nature (dependent) and need to dropped during model building to avoid bias. (casual + registered = count)

Chapter 2

Methodology

2.1 Pre Processing

We begin by exploring the data, cleaning the data as well as visualizing the data through graphs and plots, which is often called as **Exploratory Data Analysis (EDA).**

To start this process we will first get a quick glance on the distributions of the variables. Most analysis like regression, require the data to be normally distributed. We can visualize that in a glance by using **pandas_profiling** library to generate profile reports.

For each column the following statistics are presented in an interactive HTML report:

- Essentials: type, unique values, missing values
- Quantile statistics like minimum value, Q1, median, Q3, maximum, range, interquartile range
- Descriptive statistics like mean, mode, standard deviation, sum, median absolute deviation, coefficient of variation, kurtosis, skewness
- Most frequent values
- Histogram
- Correlations highlighting of highly correlated variables, Spearman and Pearson matrixes

We can access the statistics of each variable generated in the HTML report via our saved repository.

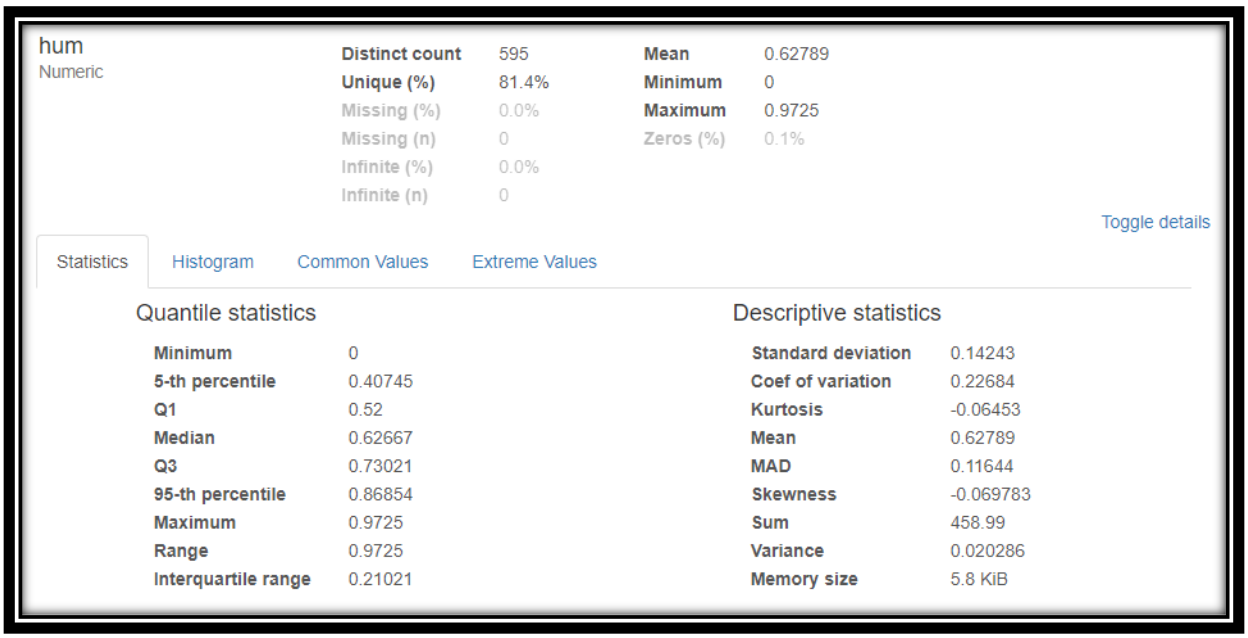


Fig 2.1 : Statistics of Humidity variable

For example, we have taken the humidity variable here, in Fig 2.1 it shows a **minimum of zero** in Quantile Statistics.

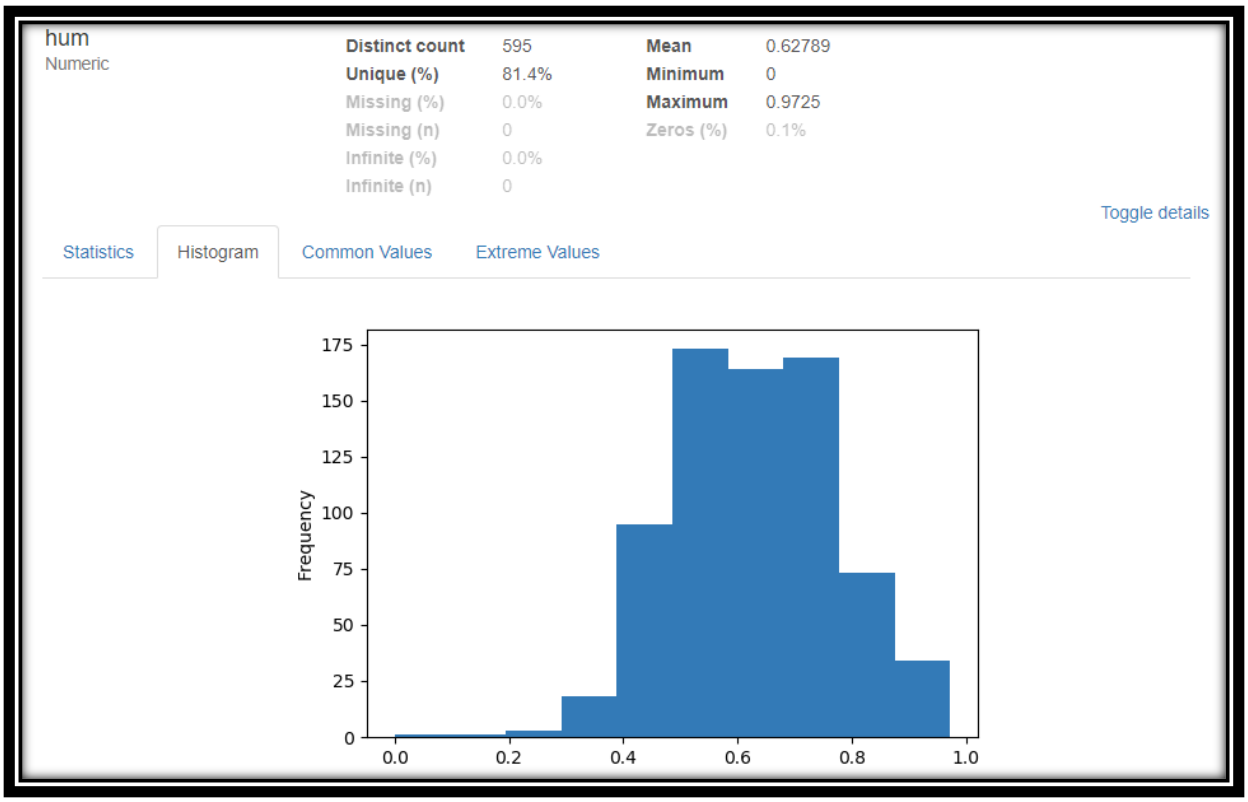


Fig 2.2 : Histogram of Humidity variable

The distribution of humidity variable also seems to be normally distributed. Other features like common values and Extreme values can also be accessed via toggle details option in HTML report.

2.1.1 Missing Value Analysis

- Visualisation of missing values done using the **missingno** library.
- No missing value were found.

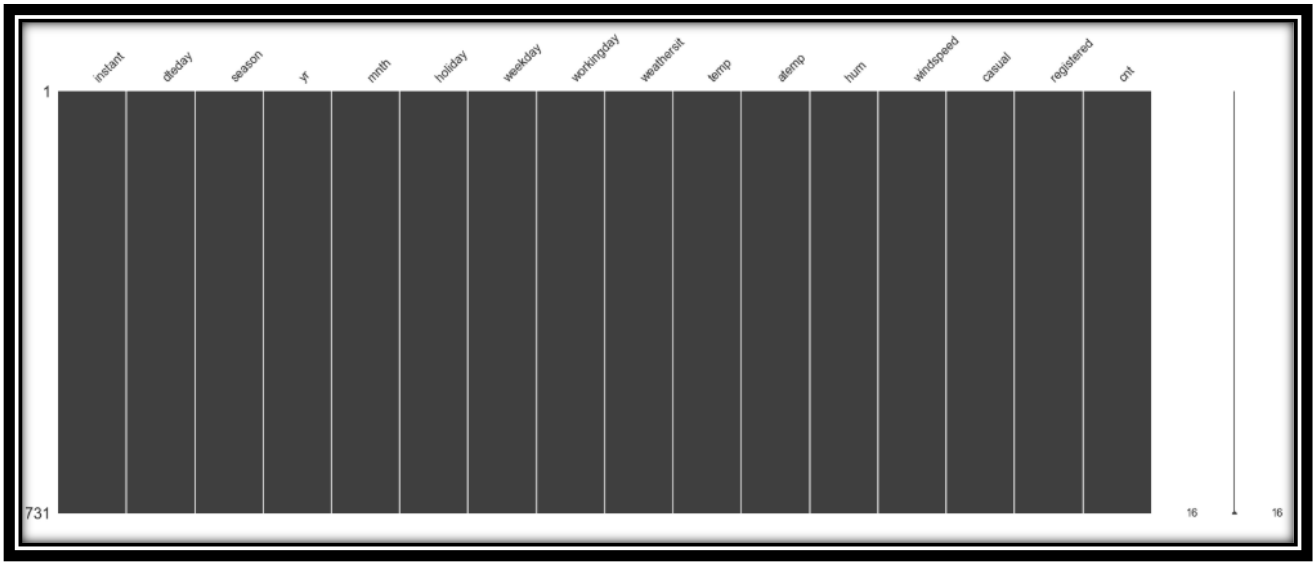


Fig 2.3 : Visualisation of missing values

2.1.2 Univariate and Bivariate analysis of all Categorical features

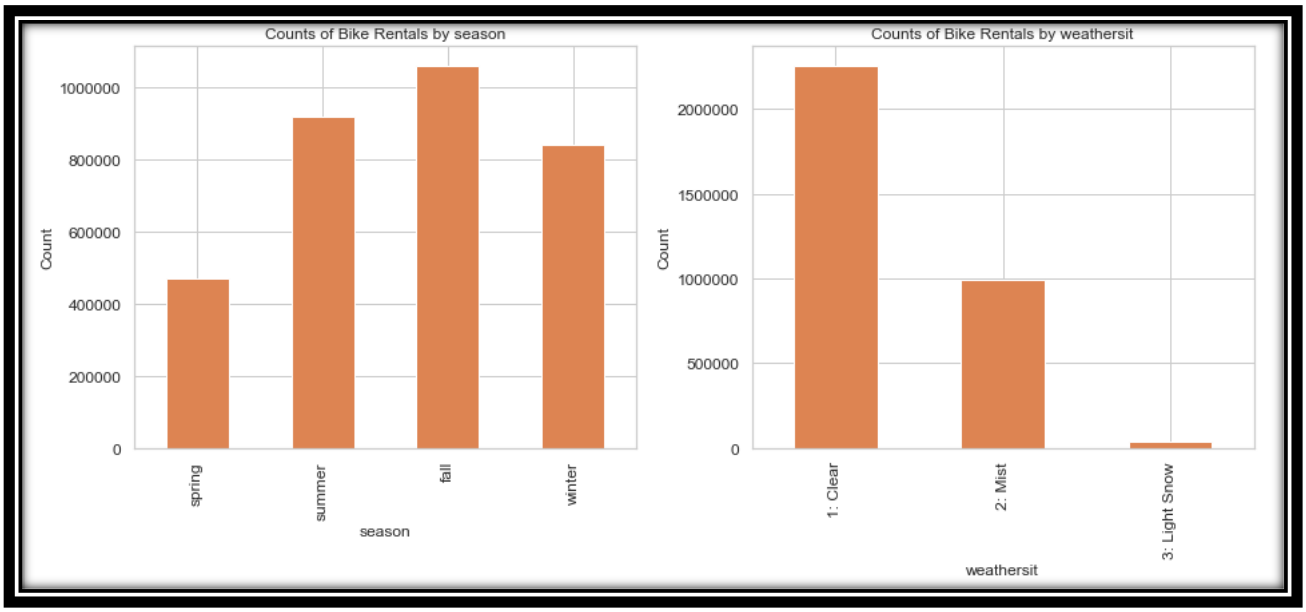


Fig 2.4 : Usage of bikes by type of season and weather

- People like to rent bikes more when the sky is clear.
- The count of number of rented bikes is maximum in fall (Autumn) season and least in Spring season.

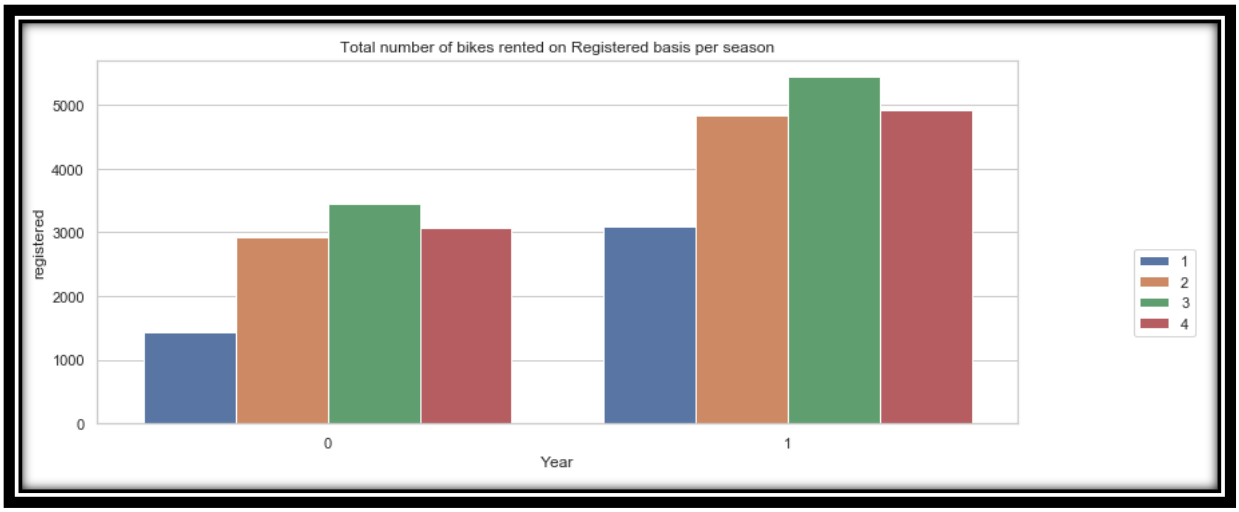


Fig 2.5 : number of bikes rented on registered basis per season

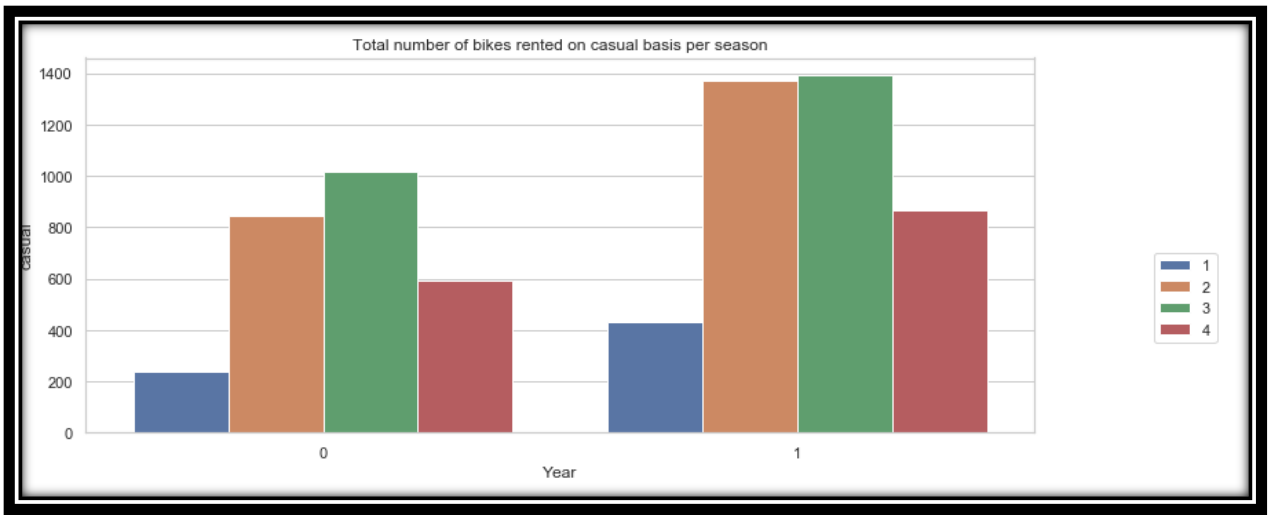


Fig 2.6 : number of bikes rented on casual basis per season

- On comparing Fig 2.5 and Fig 2.6 we can see that the number of bikes rented per season over the years has increased for both casual and registered users.
- Also, registered users have rented more bikes than casual users overall.

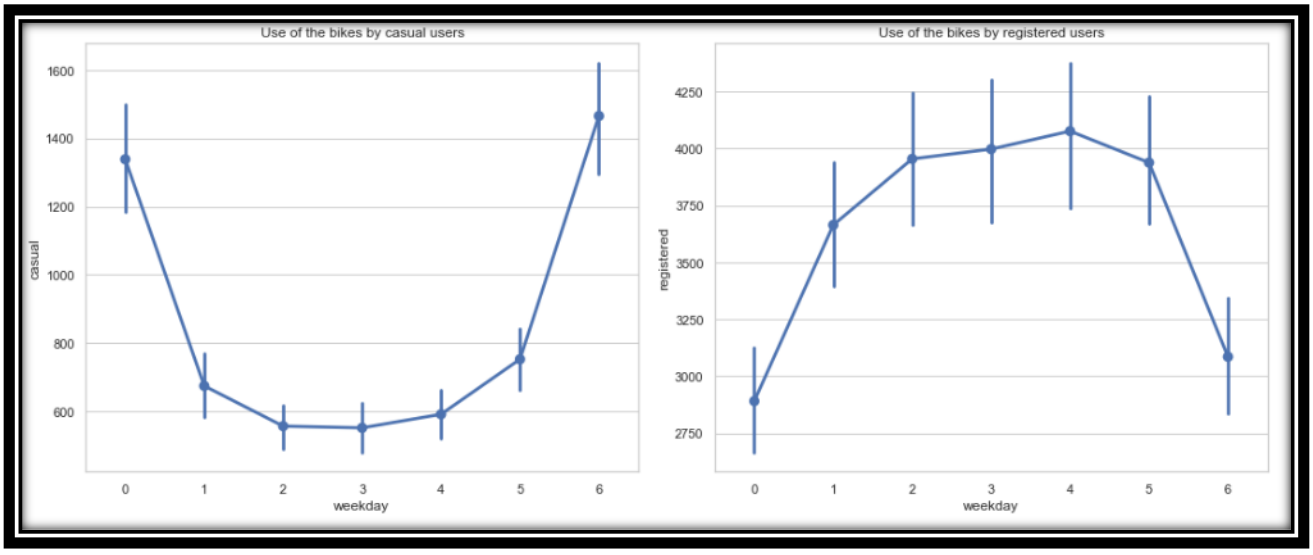


Fig 2.7 : usage of bikes by casual V/S registered users on daily basis

- In our dataset the weekends are encoded with **Saturday-6 & Sunday-0**
- Casual users like to travel more in weekends as compared to registered users.
- Registered users rent more bikes during working days as expected for commute to work / office.

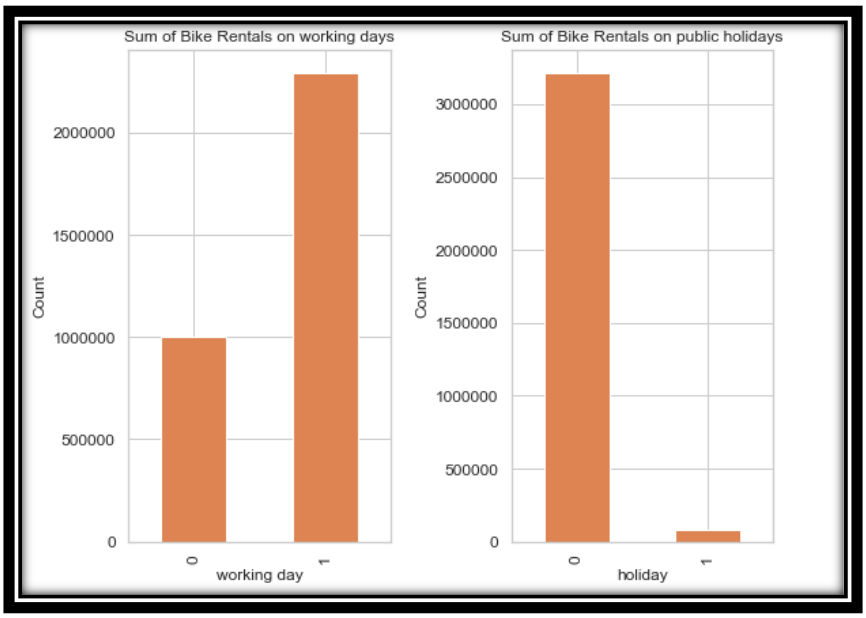


Fig 2.8 : usage of bikes on working days and holidays

- demand for bikes are more on working days as compared to holidays and weekends.

2.1.3 Univariate and Bivariate analysis of all continuous features

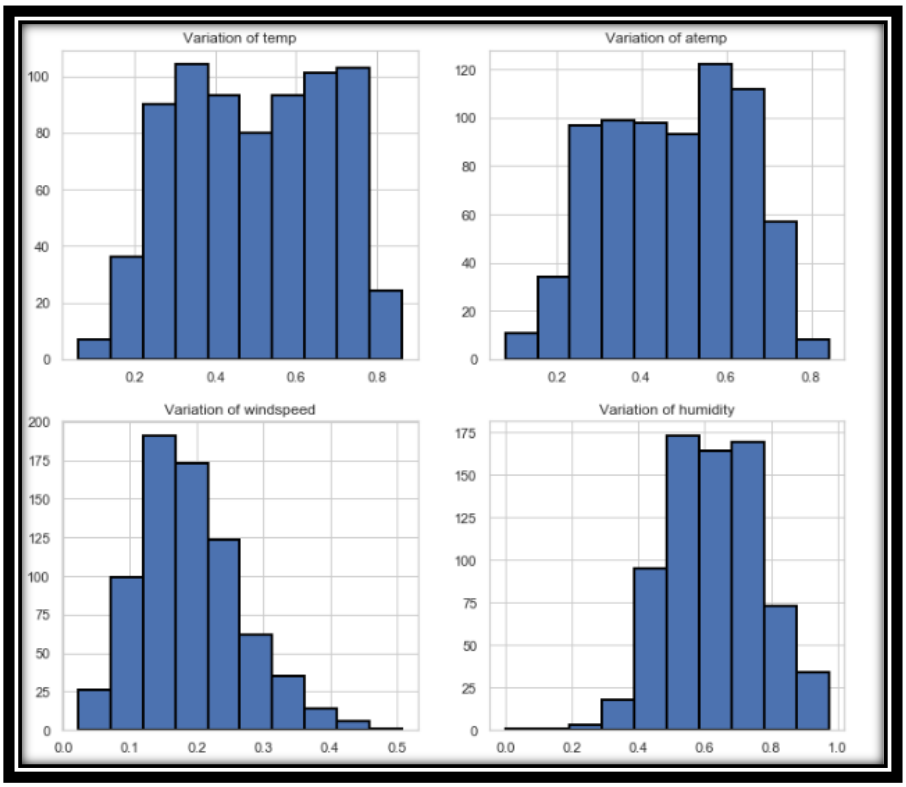


Fig 2.9 : Univariate Analysis of continuous features

- Temp, Atemp and hum variable looks normally distributed.
- Windspeed variable seems to be slightly positively skewed (long tail towards right).

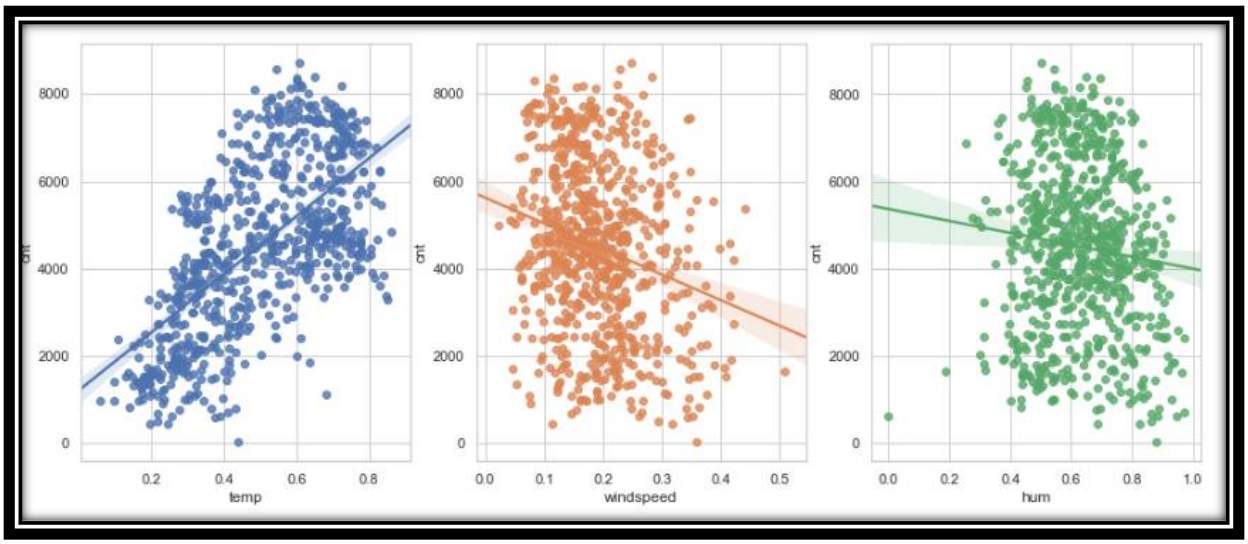


Fig 2.10 : Regression plot between continuous features and target variable (cnt)

- A + ve correlation with temperature was observed (sky becomes clear with increase in temperature, hence people prefer to ride more bikes)
- A - ve correlation with humidity and windspeed was observed as people avoid travelling when weather is very windy or humid.

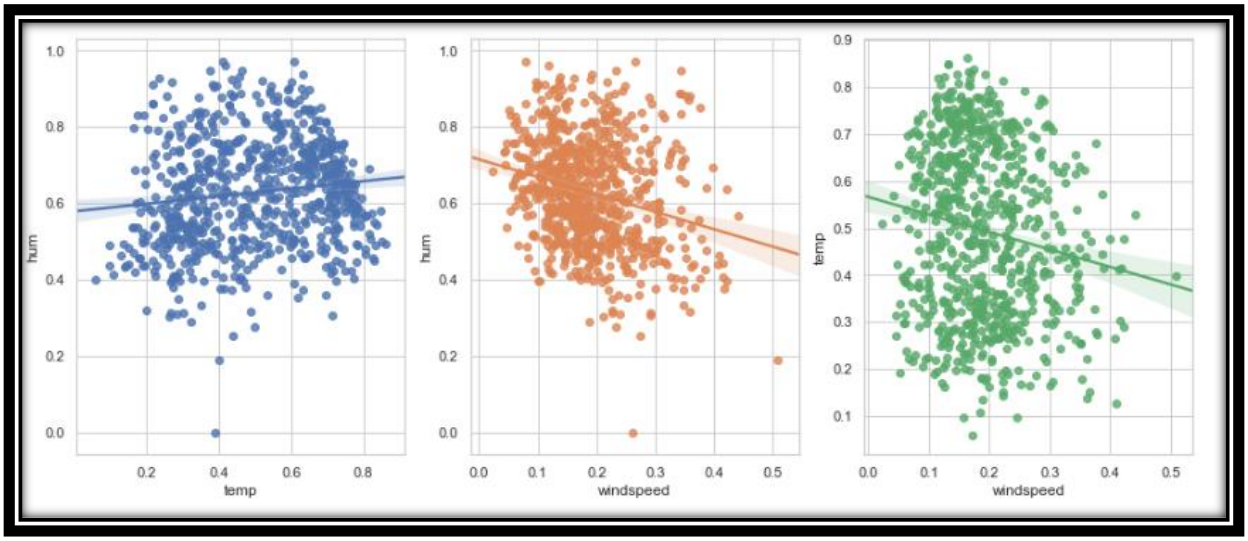


Fig 2.11 : Regression plot among continuous features

- A + ve correlation between humidity and temperature was observed (as temp increases the amount of water vapour present in the air also increases)

- A - ve correlation between windspeed with humidity and temperature was observed (as wind increases, it draws heat from the body, thereby temperature and humidity decreases)

2.1.4 Outlier Analysis

Few bivariate outliers were observed from the box plots and pair plots below.

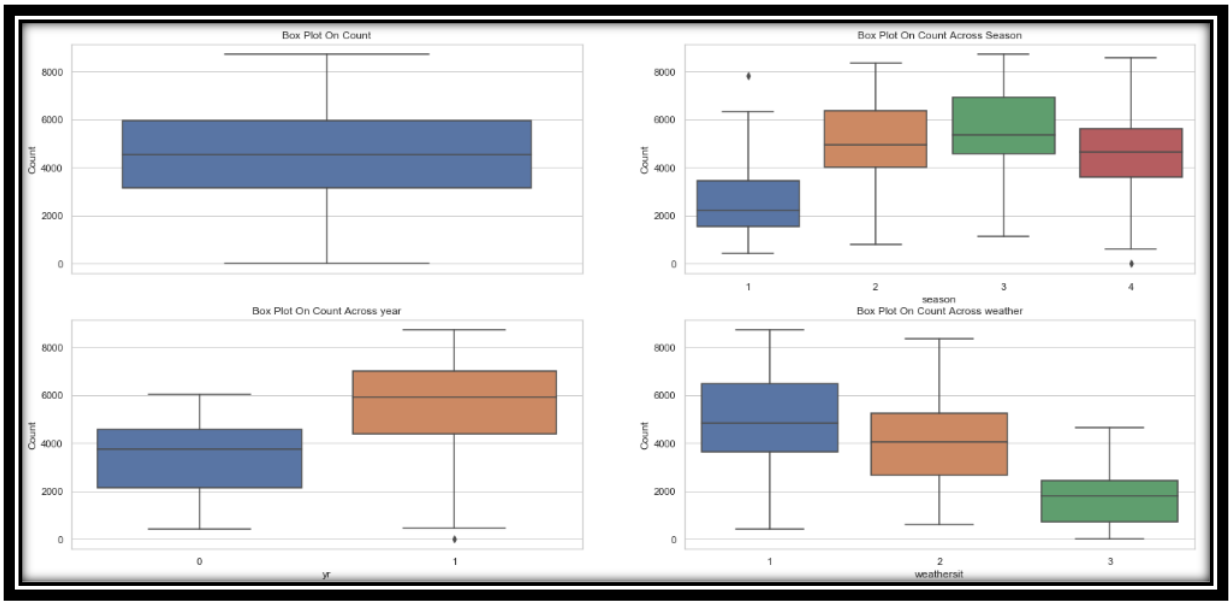


Fig 2.12 : box plot of categorical features with target variable (cnt)

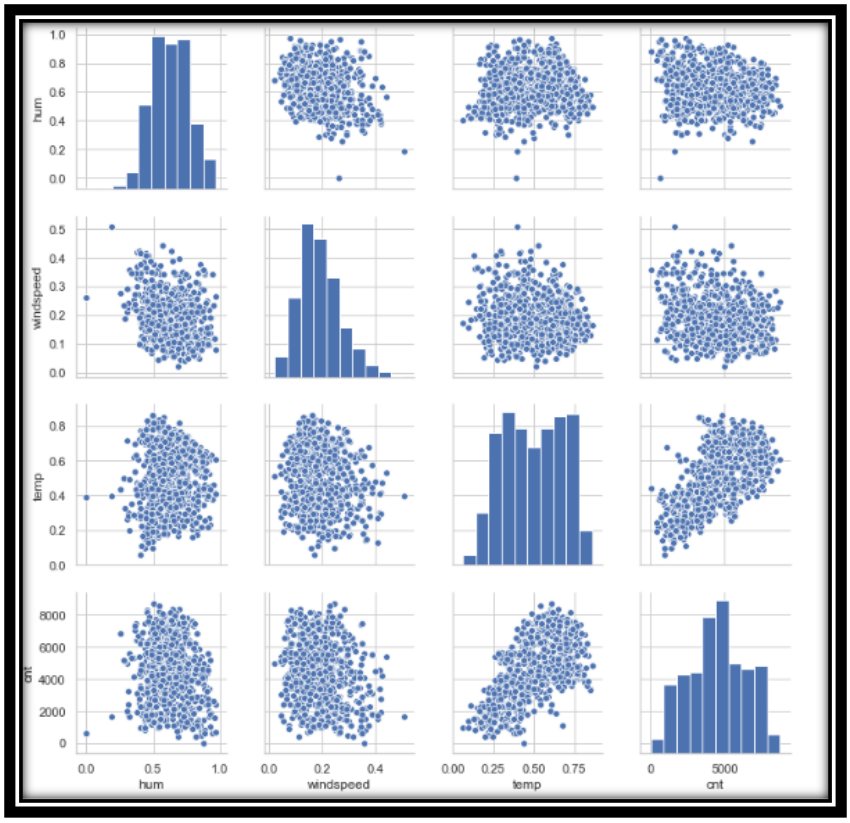


Fig 2.13 : pair plot of continuous features with target variable (cnt)

We will use **PyOD** library, for detecting outlying objects in multivariate data.

- **k-Nearest Neighbors** will be used here as a Outlier Detection Algorithm
- For each data point, the distance from its k nearest neighbors are looked at for outlier detection.

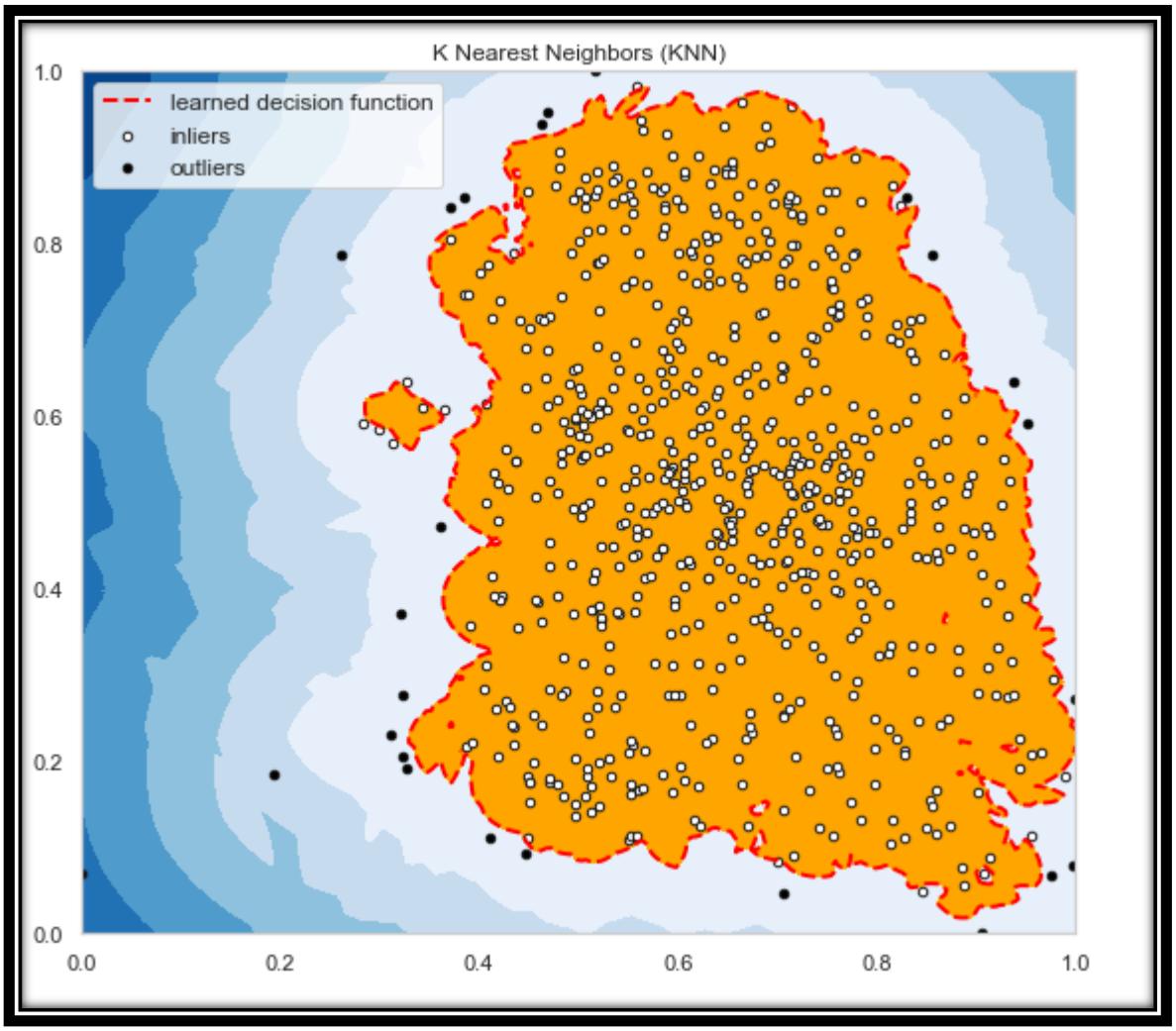


Fig 2.14 : Outlier Detection Algorithm : KNN (OUTLIERS : 25 , INLIERS : 706)

- In the above plot, the white points are inliers surrounded by red lines, and the black points are outliers in the blue zone.
- All the 25 outliers we dropped from the dataset consisting of 731 observations.

2.1.5 Feature Engineering

- A new feature **"isweekend"** was created, which denotes Sat or Sun as 1, and 0 otherwise.

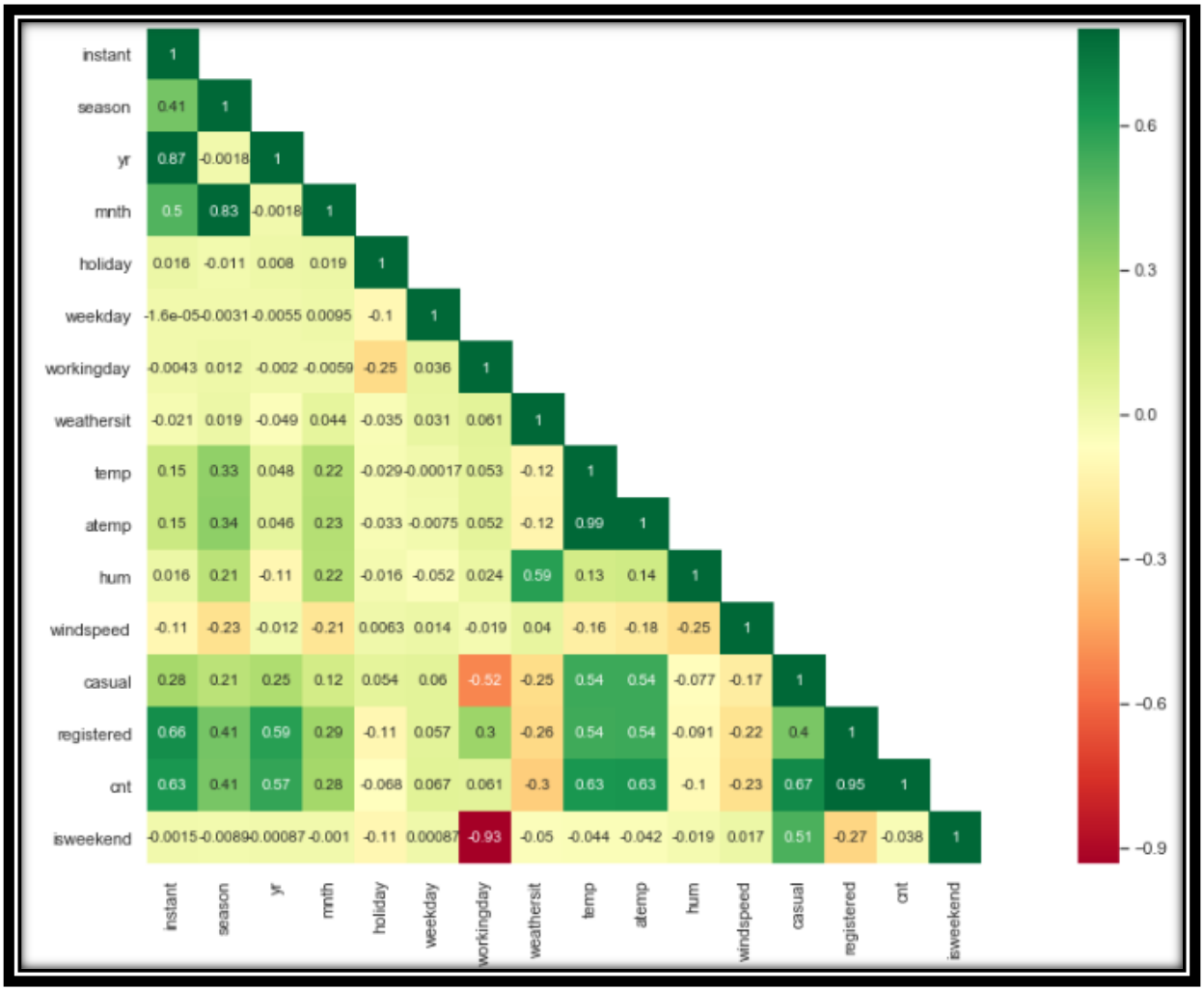


Fig 2.15 : Correlation Analysis of all features using HeatMap

- "atemp" and "temp" variable has got strong correlation with each other. During model building any one of the variable has to be dropped since they will exhibit multicollinearity in the data.
- "weathersit" and count are inversely related. This is because for our data as weather increases from (1 to 4) implies that weather is getting more worse and so lesser people will rent bikes.
- We will use **pd.get_dummies()** function for **One-Hot Encoding** the categorical features below

Categorical Features	Numerical Features	Features to Drop
season	temp	casual
holiday	hum	dteday
weathersit	windspeed	instant
weekday		registered
mnth		atemp
yr		
isweekend		
workingday		

Chapter 3

Modeling

We will start our model building from the most simplest to more complex. Therefore we use Simple Linear Regression first as our base model.

Since our problem is regression type , both R2 and RMSE can indicate the goodness of the fit.

R-squared is scaled between 0 and 1, whereas RMSE is not scaled to any particular values.

Even though R-squared can be more easily interpreted, but with RMSE we explicitly know how much our predictions deviate, on average, from the actual values in the dataset. Hence, we choose **RMSE as our error metric**.

3.1 Linear Regression

Pipeline concept is used to perform sequence of different transformations on our dataset before applying the final estimator.

Here following steps are used:

- StandardScaler() – data is normally distributed with mean of 0 and standard deviation of 1.
- Regressor- a list of linear models containing [LinearRegression(), Ridge(), Lasso()]

Model	RMSE	R^2
Linear Regression	721.318	0.838
Ridge	721.173	0.838
Lasso	720.313	0.838

- Here, R2 tells us 83.8% of the variability in the bike rental count is explained by our model
- Even after tuning the hyperparameters with GridSearchCV for Ridge and Lasso, very small improvement was observed.

3.2 Random Forest

- Base model gave RMSE Value : 711.725 and R^2: 0.842
- After tuning the hyperparameters with GridSearchCV (5 fold cross validation), and predicting on the best estimator gave an **RMSE Value of: 675.178** and R^2: 0.858

3.3 Gradient Boosting

- Base model gave RMSE value: 636.47
- After tuning the hyperparameters with GridSearchCV (5 fold cross validation), and predicting on the best estimator gave an **RMSE Value of: 600.132** and R^2: 0.888

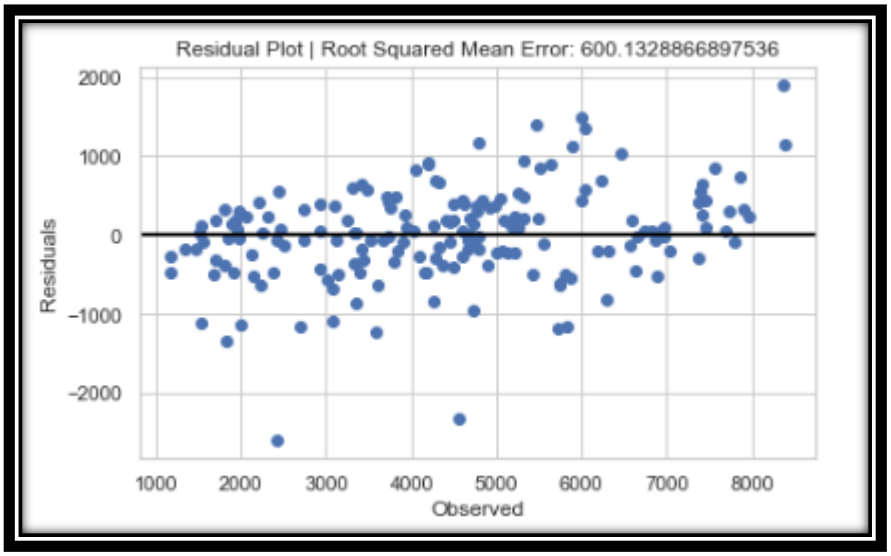


Fig 2.16 : Residual plot

3.4 Need for Bayesian approach of hyperparameter optimization (Hyperopt)

- An ML algorithm generally has a loss/cost function which needs to be minimized subject to the values of hyperparameters and parameters of the algorithm.

- Weights and Biases are decided on the basis of Optimization function of the ML algorithm. Selecting good hyperparameters further minimizes the loss function, and makes model more robust and accurate, increasing the overall efficiency of the model.

- For instance in Gradient Descent, it is very important to select a good learning rate to reach the convergence point in shortest time, because if it is large the cost function will overshoot and won't be able to find minima or if too small it will take forever to reach the minima.

- Now even though Scikit-Learn provides us Grid Search and Random Search, these algorithms are brute force and the computation time grows as grid becomes more dense. Even the best possible combination might be far from the optimal.

- For example, the **common implementation of random search completely ignores information on the trials already computed**, and each new sample is drawn from the same initial distribution.

- Fortunately, there is a way to account for them. Say, you were tuning C - regularization parameter for logistic regression. If one particular value gives really bad results - the points in vicinity will also perform poorly, so there is really very little need to sample from this region.

- We would like to incorporate this information into our strategy - in other words, we want to get more points from the regions with high probability of yielding good result and get less points from elsewhere. That's exactly what hyperopt module can do.

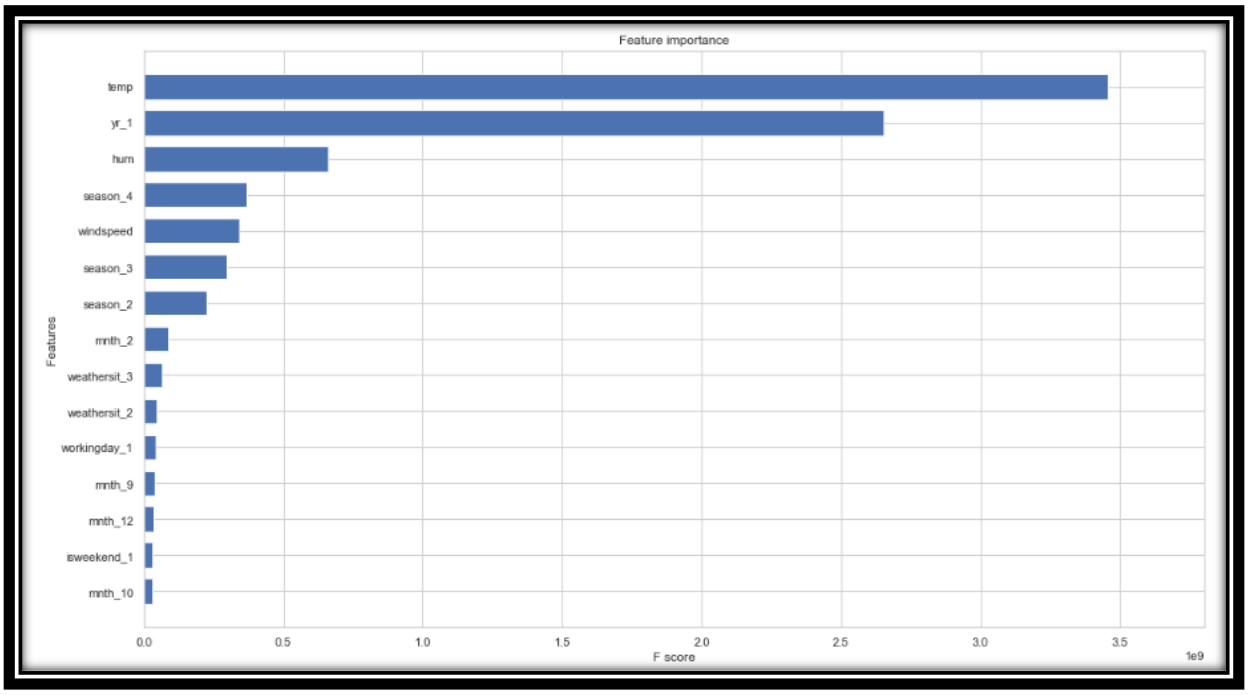
- hyperopt uses Bayesian approach for intelligent points sampling from search space. It adjusts prior distribution using history of target function evaluations, concentrating probability mass in the region where function performs better.

3.5 XGBoost

- Base model gave RMSE value: 625.023

- Bayesian hyperparameter optimization with hyperopt gave us the following best parameters {'alpha': 3.0, 'colsample_bytree': 0.700, 'eta': 0.025, 'gamma': 0.55, 'max_depth': 6, 'min_child_weight': 6.0, 'n_estimators': 400.0, 'subsample': 0.950}

- After training the model with the best parameters we got **RMSE value of 615.854**



- Observing the feature importance plot, it looks like **temperature and year** are two most important features affecting bike rental count.

3.6 CatBoost

- CatBoost is an algorithm for gradient boosting on decision trees. Developed by **Yandex** researchers and engineers.

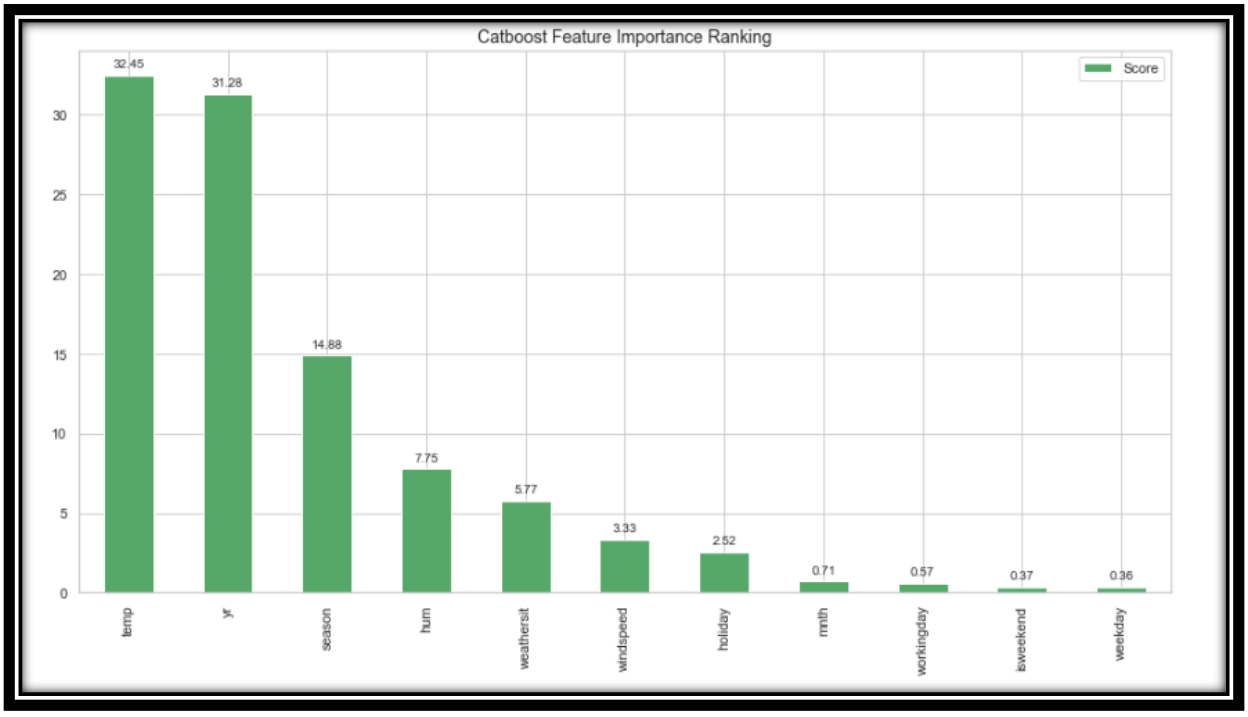
- Catboost's **power lies in its categorical features pre-processing**, since in our dataset almost 70% of features are categorical in nature.

- CatBoost has a very important parameter **cat_features** — This parameter is a must in order to leverage Catboost pre-processing of categorical features, you don't need to encode the categorical features manually, just need to pass the columns indices which are categorical.

- Base model gave an RMSE value: 594.490

- Bayesian hyperparameter optimization with hyperopt gave us the following best parameters {'bagging_temperature': 0.640, 'depth': 6.0, 'l2_leaf_reg': 16.652, 'learning_rate': 0.0511, 'n_estimators': 1900.0, 'rsm': 0.618}

- After training the model with the best parameters we got **RMSE value of 592.37**



- Observing the feature importance plot, **temperature, year, season, humidity and weather** are the top 5 features affecting bike rental count.

Chapter 4

Conclusion

- Among all the models, Catboost performed best with an RMSE of **592.37.**
- **temperature, year, season, humidity and weather** appeared to be the top 5 features affecting bike rental count.
- the dataset contained very less samples (around 731), due to which a large RMSE value was observed while training different models.
- By increasing the no of samples, the model can learn better, and overall error will be reduced.
- **AutoML** function of **H2O.ai** was also tried, which automates the process of building large number of models, with the goal of finding the “best” model without any prior knowledge.
- Stacked-ensemble model topped the leaderboard with RMSE of **616.646, R^2=0.897** on test data.

References

- https://catboost.ai/docs/concepts/python-reference_catboostregressor.html
- https://medium.com/analytics-vidhya/gentle-introduction-to-automl-from-h2o-ai-a42b393b4ba2
- https://pyod.readthedocs.io/en/latest/pyod.models.html#module-pyod.models.knn
- https://fizzylogic.nl/2018/08/21/5-must-have-tools-if-youre-serious-about-machine-learning/
- https://towardsdatascience.com/an-introductory-example-of-bayesian-optimization-in-python-with-hyperopt-aae40fff4ff0