**Seoul Bike Sharing Demand Prediction**

**Akash Salmuthe**

**Data Science Enthusiast**

**AlmaBetter, Bangalore**

**Abstract:**

The main goal of the project is to forecast the bike rental demand on the hourly based throughout the year. To perform that we are provided with the one year full data of rental bike count. With the help of the Exploratory Data Analysis on the given data set we finding the correlation of the various feature with the target variable. The objective of the model building from the given data we perform the various regression technique to get the best result for that we perform the train test split with the important feature. This gives us the good predictive module to reach out our main goal of the supply of rental bike count on the hourly base through the year.

***Keywords: machine learning, EDA, Bike sharing demand prediction***

**1. Introduction**

Seoul is a city spread over more than 600 km2, six times the size of Paris. It is necessary to take precautions when it comes to traveling, especially at peak hours, which generate endless traffic jams in the city's streets. Their customers can download their app on smartphones and book a bike from anywhere in the cities they operate in.

They, in turn, search for cabs from various service providers and provide the best option to their clients across available options.

According to recent studies, it is expected that more than 60% of the population in the world tends to dwell in cities, which is higher than 50% of the present scenario. Some countries around the world are practising righteous scenarios, renderings mobility at a fair cost and reduced carbon discharge. On the contrary other cities are far behind in the track. Urban mobility usually fills 64% of the entire kms travelled in the world. It ought to be modelled and taken over by inter-modality and networked self-driving vehicles which also provides a sustainable means of mobility. Systems called Mobility on Demand has a vital part in raising the vehicles’ supply, increasing its idle time and numbers.

 Bike-sharing MOD systems are already firmly holding the effective part in short commuting and as ‘last mile’ mobility resources on inter-modal trips in several cities. Certain issues prevail in the maintenance, design, and management of bike-sharing systems: layout of the station design; fleet size and capacity of the station; detecting broken, lost, or theft bikes; pricing; monitoring of traffic and customer activities to promote behaviour virtuously; and marketing using campaigns etc. System balancing is the hardest endeavour: In the daytime, some stations are likely to be crowded with bike flow, while leaving other stations empty, which hampers pick-up and drop-off, respectively. So, to restore the balance, several manual techniques, like shifting bikes through trucks, cars and even by volunteers are employed. Data analysis techniques and studies focus on dynamic systems and optimization methods are utilised for complementing the knowledge base of employing optimum rebalancing policies.

Today, bike-sharing systems are blooming across more cities around the world. To complete a short trip renting a bike is a faster way when compared to walking. Moreover, it is eco-friendly and comfortable too compared to driving.

**2. Problem Statement**

Currently Rental bikes are introduced in many urban cities for the enhancement of mobility comfort. It is important to make the rental bike available and accessible to the public at the right time as it lessens the waiting time

### Eventually, providing the city with a stable supply of rental bikes becomes a major concern. The crucial part is the prediction of bike count required at each hour for the stable supply of rental bikes.

**3. Dataset Description**

The data description phase starts with an initial data collection and proceeds with activities in order to get familiar with the data. Identifying data quality problems, discovering first insights into the data and detecting interesting subsets to form hypotheses from hidden information are activities of this step. Data which is collected from a rented bike provider company form Seoul to get analysed, involves usage details of customers from. The data was taken from rented bike Provider Company. It has 8760 rows and 14 columns. Most columns related to hourly bike count for rent. Other column was indicative of weather condition affecting bike count per hour.

|  |  |
| --- | --- |
| **Feature Name**  Date : year-month-day  Rented Bike Count  Hour  Temperature(**°C)**  Humidity (%)  Wind speed (m/s)  Visibility (10m)  Dew Point temperature (**°C)**  Solar Radiation (MJ/m2)  Rainfall (mm)  Snowfall(cm)  Seasons  Holiday  Functioning day | **Type**  Date  Int64  Int64  Float64  Int64  Float64  Int64  Float64  Float64  Float64  Float64  Object  Object  Object |

**Feature Description:**

* **Date**: The date of the day, during 365 days from 01/12/2017 to 30/11/2018, formatting in DD/MM/YYYY, *we need to convert into date-time format.*
* **Rented Bike Count**: Number of rented bikes per hour which our dependent variable and we need to predict that
* **Hour:** The hour of the day, starting from 0-23 it's in a digital time format
* **Temperature (°C):**  Temperature of the weather in Celsius and it varies from -17***°****C to 39.4****°****C*.
* **Humidity (%)**:*Availability of Humidity* in the air during the booking and ranges from 0 to 98%.
* **Wind speed (m/s):** Speed of the wind while booking and ranges from 0 to 7.4m/s.
* **Visibility (10m):** Visibility to the eyes during driving in “m” and ranges from 27m to 2000m.
* **Dew point temperature (°C)**: Temperature
* At the beginning of the dayand it ranges from -30.6**°**C to 27.2**°**C.
* **Solar Radiation (MJ/m2):**  Sun contribution or solar radiation during ride booking which varies from 0 to 3.5 MJ/m2.
* **Rainfall (mm):** The amount of rainfall during bike booking which ranges from 0 to 35mm.
* **Snowfall (cm):** Amount of snowing in cm during the booking in cm and ranges from 0 to 8.8 cm.
* **Seasons:** Seasons of the year and total there are 4 distinct seasons i.e. summer, autumn, spring and winter.
* **Holiday:** If the day is holiday period or not and there are 2 types of data that is holiday and no holiday.

## **4. Reasons for less Renting bikes**

The reasons for less counting of bikes:

* More Humidity
* More Rainfall
* Heavy Snowfall

# **5. How Renting bikes Increase** There are times when so many people are requesting renting bikes that they don’t have bikes and to go quickly anywhere on the road to help take them all. Good whether rush hour, and special events, for instance, may cause unusually large numbers of people to take renting bikes

**6. Steps involved:**

* **Exploratory Data Analysis**

After loading the dataset we performed this method by comparing our target variable that is ‘Rented Bike Count’ with other independent variables. This process helped us figuring out various aspects and relationships among the target and the independent variables. It gave us a better idea of which feature behaves in which manner compared to the target variable.

* **Null values Treatment**

Our dataset contains no null values or missing values which may tend to our accuracy at the beginning of our project in order to get a better result.

* **Encoding of categorical columns**

We used Label Encoding to produce binary integers of 0 and 1and 2 and 3 to encode our categorical features (Holiday, Seasons, Functioning day) because categorical features that are in string format cannot be understood by the machine and needs to be converted to numerical format.

* **Feature Selection**

In these steps we using information gaining method finding correlation coefficients between variables mostly effects that are removed from our data set i.e temperature and dew-point temperature are most effects for this we taken weighted average and removed this items removed from the dataset.

* **Standardization of features**

Our main motive through this step was to scale our data into a uniform format that would allow us to utilize the data in a better way while performing fitting and applying *Standard scaler after that Normalized* algorithms to it.

The basic goal was to enforce a level of consistency or uniformity to certain practices or operations within the selected environment.

* **Fitting different models**

For modelling we tried various classification algorithms like:

1. Linear Regression
2. Lasso Regression
3. Ridge Regression
4. Random Forest
5. Gradient Boosting
6. Xgboost

* **Tuning the hyperparameters for better accuracy**

Tuning the hyperparameters of respective algorithms is necessary for getting better accuracy and to avoid overfitting in case of tree based models like Random Forest Regressor and XGBoost Regressor.

**7.1 Algorithms:**

1. **Linear Regression:**

Linear regression is a quiet and the simplest statistical regression method used for predictive analysis in machine learning. Linear regression shows the linear relationship between the independent (predictor) variable i.e. X-axis and the dependent (output) variable i.e. Y-axis, called linear regression*.*If there is a single input variable X (dependent variable), such linear regression is called *simple linear regression*.

The function used in Linear Regression is given by:

***y = mx + b***

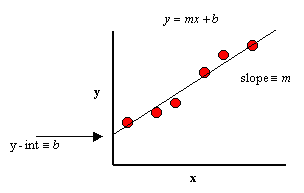


Fig.1 Linear Regression

.

1. **Lasso Regression:**

Lasso is used for Regularization when the coefficients very high or overfitting we use LASSO REGULARIZATION technique and this is used for variable selection and one axis is zero and shrink coefficients all the way to zero

Using hyper parameter tuning we get good results:

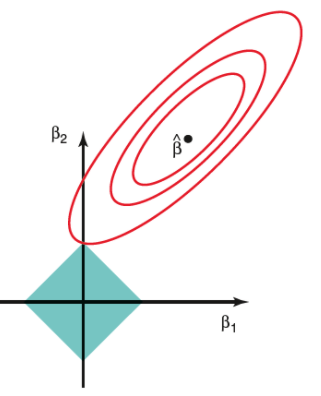


Fig.2 Lasso Regression

1. **Ridge Regression:**

Ridge is used for Regularization when the coefficients very high or overfitting we use RIDGE REGULARIZATION technique and this is used for better accuracy and there is no variable selection and both axis must have some value shrinking coefficients towards zero but rarely reach zero.

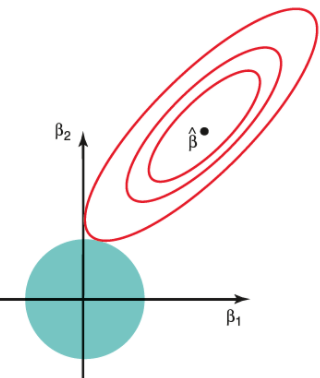


Fig.3 Ridge Regression

1. **Random Forest:**

Random Forest is a bagging type of Decision Tree Algorithm that creates a number of decision trees from a randomly selected subset of the training set, collects the labels from these subsets and then averages the final prediction depending on the most number of times a label has been predicted out of all.

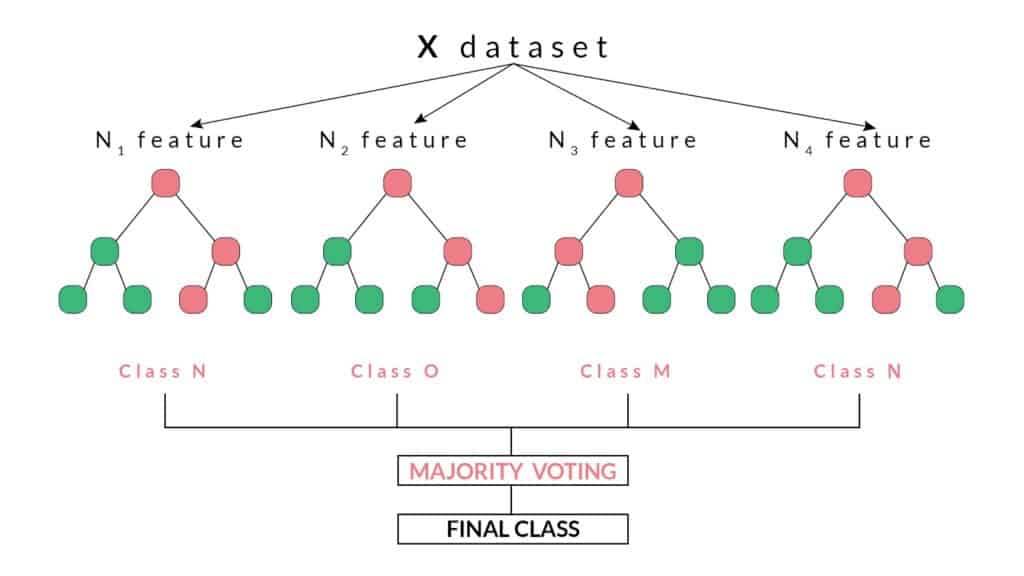


Fig.4 Random Forest

1. **Gradient Boosting:**

Gradient boost sequentially combines many weak learners to form a strong learner. Typically Gradient boost uses decision trees as weak learners.

Gradient boost is one of the most powerful techniques for building predictive models for both classification and Regression problems.

To understand the Gradient boost below are the steps involved. In Gradient boosting weak learners are decision trees.

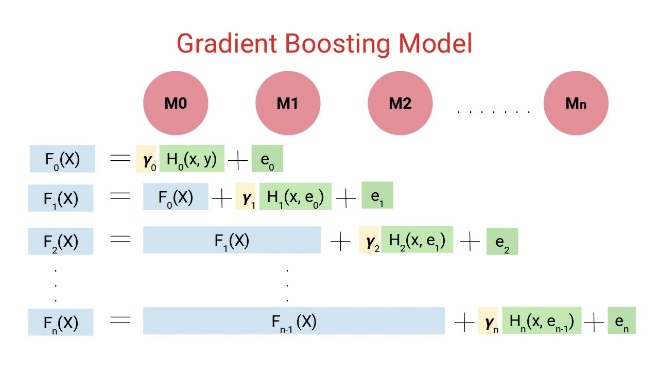


Fig.5 Gradient Boosting Regressor

**Step1**: Construct a base tree with a single root node. It is the initial guess for all the samples.

**Step2**: Build a tree from errors of the previous tree.

**Step3**: Scale the tree by learning rate (value between 0 and 1). This learning rate determines the contribution of the tree in the prediction

**Step4**: Combine the new tree with all the previous trees to predict the result and repeat step 2 until maximum number of trees is achieved or until the new trees don't improve the fit.

The final prediction model is the combination of all the trees.

1. **XGBoost:**

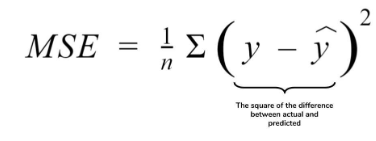
XG Boost is one of the fastest implementations of gradient boosting trees. It does this by tackling one of the major inefficiencies of gradient boosted trees: considering the potential loss for all possible splits to create a new branch (especially if you consider the case where there are thousands of features, and therefore thousands of possible splits). XG Boost tackles this inefficiency by looking at the distribution of features across all data points in a leaf and using this information to reduce the search space of possible feature splits.

**7.2. Model performance:**

Model can be evaluated by various metrics such as:

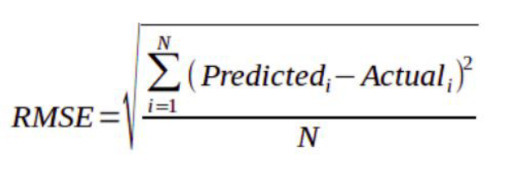
**1. Mean Squared Error (MSE):**

MSE or Mean Squared Error is one of the most preferred metrics for regression tasks. It is simply the average of the squared difference between the target value and the value predicted by the regression model.

****

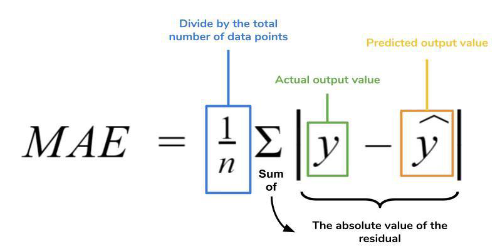
**2. Root Mean Squared Error (RMSE):**

RMSE is the most widely used metric for regression tasks and is the square root of the averaged squared difference between the target value and the value predicted by the model. It is preferred more in some cases because the errors are first squared before averaging which poses a high penalty on large errors.



**3. Mean Absolute Error (MAE):**

MAE is the absolute difference between the target value and the value predicted by the model. The MAE is more robust to outliers and does not penalize the errors as extremely as MSE. MAE is a linear score which means all the individual differences are weighted equally. It is not suitable for applications where you want to pay more attention to the outliers.

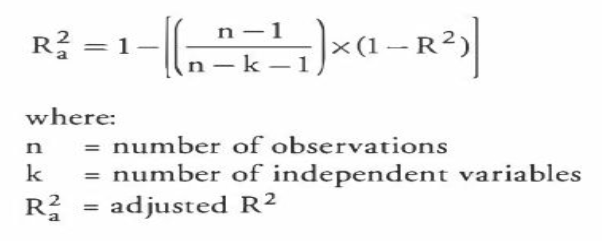
****

**4. R2 (R – Squared)**:

Coefficient of Determination or R² is another metric used for evaluating the performance of a regression model. The metric helps us to compare our current model with a constant baseline and tells us how much our model is better. The constant baseline is chosen by taking the mean of the data and drawing a line at the mean. R² is a scale-free score that implies it doesn't matter whether the values are too large or too small, the R² will always be less than or equal to 1.

**5. Adjusted R2:**

Adjusted R² depicts the same meaning as R² but is an improvement of it. R² suffers from the problem that the scores improve on increasing terms even though the model is not improving which may misguide the researcher. Adjusted R² is always lower than R² as it adjusts for the increasing predictors and only shows improvement if there is a real improvement.

****

**7.3. Hyper parameter tuning:**

Hyperparameters are sets of information that are used to control the way of learning an algorithm. Their definitions impact parameters of the models, seen as a way of learning, change from the new hyperparameters. This set of values affects performance, stability and interpretation of a model. Each algorithm requires a specific hyperparameters grid that can be adjusted according to the business problem.

Hyperparameters alter the way a model learns to trigger this training algorithm after parameters to generate outputs.

We used Grid Search CV for hyperparameter tuning. This also results in cross validation and in our case we divided the dataset into different folds. The best performance improvement after using this.

**Grid Search CV-**

Grid Search combines a selection of hyperparameters established by the scientist and runs through all of them to evaluate the model’s performance. Its advantage is that it is a simple technique that will go through all the programmed combinations. The biggest disadvantage is that it traverses a specific region of the parameter space and cannot understand which movement or which region of the space is important to optimize the model.

**8. Conclusion:**

That's it! We reached the end of our exercise.

* Starting with loading the data so far we have done EDA, null values treatment, encoding of categorical columns, feature selection and then model building.
* In all of these models our accuracy revolves in the range of 55% to 87%.
* There is a huge improvement in accuracy score after hyperparameter tuning.
* So the accuracy of our best model R2 SCORE approx. is 87% which can be said to be good for this large dataset.

**References-**

1. [*https://towardsdatascience.com/*](https://towardsdatascience.com/)
2. [*https://stackoverflow.com/*](https://stackoverflow.com/)
3. *https://medium.com/*