**Seoul Bike Demand Prediction**

**Abhishek Singh Rawat, Jatin, Abhilasha.M**

**Data science trainees,**

**AlmaBetter, Bangalore**

**Abstract:**

Dataset used in this project is the Seoul Bike Share program data. This dataset contains information about the total count of rented bikes at each hour, as well as the date of observation and meteorological information (Humidity, Snowfall, Rainfall, Temperature Season, and so on) for that hour.

The observations in the dataset were recorded during a span of 365 days, from December 2017 to November 2018.

Our experiment can help understand what could be the reason for the Regression of such labels by feature selection, data analysis and prediction with machine learning algorithms taking into account previous trends to determine the correct Regression.

**1.Problem Statement**

With the growing demand and user base for bike-sharing system, providing the city with a stable supply of rental bikes could eventually become a challenging task. The success of bike-sharing system relies in ensuring that the quality of facilities provided, meets the needs and expectations of the users. Therefore, it is important to ensure that rental bikes are available and accessible to the users at right time, as it reduces the waiting time. Forecasting the number of bikes required and identifying the key factors that influence the demand for rental bikes can greatly help in managing the bike-sharing system.

**2. Introduction**

## Bike sharing system is an innovative transportation strategy that provides individuals with bikes for their common use on a short-term basis for a price or for free. Over the last few decades, there has been a significant increase in the popularity of bike-sharing systems all over the world. This is because it is an environmentally sustainable, convenient and economical way of improving urban mobility. In addition to this, this system also helps to promote healthier habits among its users and reduce fuel consumptions.

## **3. Understanding the Variables**

### Solar radiation - MJ/m2

### Rainfall - mm

### Snowfall - cm

### Seasons - Winter, Spring, Summer, Autumn

### Holiday - Holiday/No holiday

### Functional Day - NoFunc(Non Functional Hours), Fun(Functional hours)

### Date: year-month-day

### Rented Bike count - Count of bikes rented at each hour

### Hour - Hour of the day

### Temperature-Temperature in Celsius

### Humidity - %

### Windspeed - m/s

### Visibility - 10m

### Dew point temperature - Celsius

## **4. Libraries we used**

* NumPy
* Pandas
* Seaborn
* Matplotlib
* Date Time
* Sklearn

# **5. Graphs used in this project**

* Point Plot
* Bar Plot
* Dis Plot
* Line Plot
* Reg Plot
* Box Plot
* Heat Map
* Scatter Plot

**6. Steps involved:**

* **Exploratory Data Analysis**

Exploratory Data Analysis stands for **EDA. ‘EDA’** is applied to **investigate** data and summarize the key **insights**. Here we are understanding the data distributions, some of the values by manipulating data and much more by using **Python functions and Machine Learning Algorithms.**

* **Null values Treatment**

Our dataset contains a large number of null values which might tend to disturb our accuracy hence we dropped them at the beginning of our project in order to get a better result.

* **Encoding of categorical columns**

We used One Hot Encoding to produce binary integers of 0 and 1 to encode our categorical features 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 used algorithms like Extra Tree classifier to check the results of each feature i.e which feature is more important compared to our model and which is of less importance.

* **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 different 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. **Elastic Net Regression**
5. **Decision Tree**
6. **Random Forest Classifier**
7. **Gradient Boosting**
8. **XGBoost classifier**

* **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 Classifier and XGBoost classifier.

**7.1. Algorithms:**

1. **Linear Regression**

Linear Regression is **a machine learning algorithm based on supervised learning**. It performs a regression task. Regression models a target prediction value based on independent variables. It is mostly used for finding out the relationship between variables and forecasting.

1. **Lasso Regression:**

Lasso regression is **a regularization technique**. It is used over regression methods for a more accurate prediction. This model uses shrinkage. Shrinkage is where data values are shrunk towards a central point as the mean. The lasso procedure encourages simple, sparse models (i.e. models with fewer parameters).

1. **Ridge Regression**

**Ridge regression can be used for the analysis of prostate-specific antigen and clinical measures among people who were about to have their prostates removed**. The performance of ridge regression is good when there is a subset of true coefficients which are small or even zero.

1. **Elastic Net Regression**

Elastic net linear regression **uses the penalties from both the lasso and ridge techniques to regularize regression models**. The technique combines both the lasso and ridge regression methods by learning from their shortcomings to improve the regularization of statistical models.

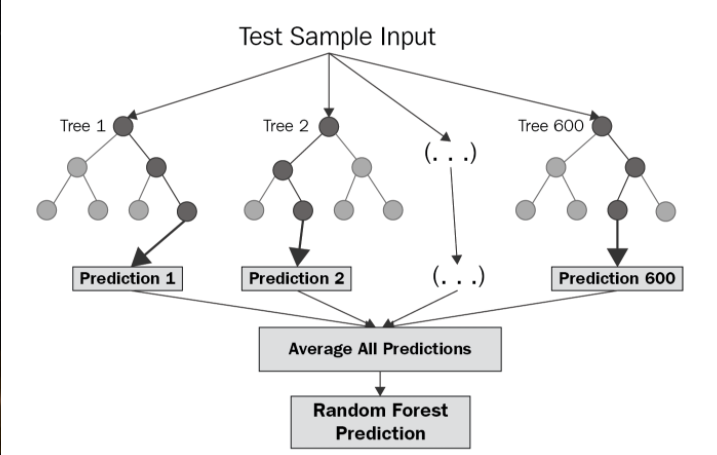
1. **Decision Tree Regression**

Decision tree regression **observes features of an object and trains a model in the structure of a tree to predict data in the future to produce meaningful continuous output**. Continuous output means that the output/result is not discrete, i.e., it is not represented just by a discrete, known set of numbers or values.

1. **Random Forest Regression:**

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.

Random Forest is an ensemble technique capable of performing both regression and classification tasks with the use of multiple decision trees and a technique called Bootstrap and Aggregation, commonly known as bagging.



1. **XGBoost-**To understand XGBoost we have to know gradient boosting beforehand.

* **Gradient Boosting-**

Gradient boosted trees consider the special case where the simple model is a decision tree

****

In this case, there are going to be 2 kinds of parameters P: the weights at each leaf, w, and the number of leaves T in each tree (so that in the above example, T=3 and w=[2, 0.1, -1]).

When building a decision tree, a challenge is to decide how to split a current leaf. For instance, in the above image, how could I add another layer to the (age > 15) leaf? A ‘greedy’ way to do this is to consider every possible split on the remaining features (so, gender and occupation), and calculate the new loss for each split; you could then pick the tree which most reduces your loss.

**XGBoost** 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). XGBoost 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. 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, Randomized Search CV and Bayesian Optimization for hyperparameter tuning. This also results in cross validation and in our case we divided the dataset into different folds. The best performance improvement among the three was by Bayesian Optimization.

1. **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:**

During the time of our analysis, we initially did EDA on all the features of our datset. We first analysed our dependent variable, 'Rented Bike Count' and also transformed it. Next we analysed categorical variable and dropped the variable who had majority of one class, we also analysed numerical variable, found out the correlation, distribution and their relationship with the dependent variable. We also removed some numerical features who had mostly 0 values and hot encoded the categorical variables.

Next we implemented 7 machine learning algorithms Linear Regression, Lasso Regression, Ridge Regression, Elastic Net Regression, Decision Tree, Random Forest and XGBoost. We did hyperparameter tuning to improve our model performance.