Abstract:

Bike sharing systems are a means of renting bicycles where the process of obtaining membership, rental, and bike return is automated via a network and locations throughout a city. The goal of this project is to combine the historical bike usage patterns with the weather data to forecast bike rental demand.

Introduction:

Bike sharing systems are a means of renting bicycles where the process of obtaining membership, rental, and bike return is automated via a network of kiosk locations throughout a city. Using these systems, people are able rent a bike from a one location and return it to a different place on an as-needed basis.

One of the most important problems from a business point of view is to predict the bike demand on any particular day. While having excess bikes results in wastage of resource (both with respect to bike maintenance and the land/bike stand required for parking and security), having fewer bikes leads to revenue loss (ranging from a short-term loss due to missing out on immediate customers to potential longer-term loss due to loss in future customer base), Thus, having an estimate on the demands would enable efficient functioning of these companies.

The goal of this project is to combine the historical bike usage patterns with the weather data to forecast bike rental demand.

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.

The dataset contains weather information (Temperature, Humidity, Windspeed, Visibility, Dewpoint, Solar radiation, Snowfall, Rainfall), the number of bikes rented per hour and date information.

Attribute Information:

1. Date: year-month-day
2. Rented Bike count - Count of bikes rented at each hour
3. Hour - Hour of he day
4. Temperature-Temperature in Celsius
5. Humidity - %
6. Windspeed - m/s
7. Visibility - 10m
8. Dew point temperature - Celsius
9. Solar radiation - MJ/m2
10. Rainfall - mm
11. Snowfall - cm
12. Seasons - Winter, Spring, Summer, Autumn
13. Holiday - Holiday/No holiday
14. Functional Day - NoFunc(Non Functional Hours), Fun(Functional hours)

Objective of our Project:

- Observation of factors which effect the Bike demand in our dataset.
- Selection the best model for prediction.
- Count of Bike demand w.r.t 'Hour', 'Seasons', 'Month'.

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 doesn’t contain null values.
- ## Encoding of categorical columns
  I have 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

  I have used correlation matrix to and also checked the distribution of all the features. According to importance of features in correlation matrix selected the features for 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 different algorithms to it.
  The basic goal was to enforce a level of consistency or uniformity to certain practicesor operations within the selected environment.

- ## Fitting different models

  For modelling we tried various classification algorithms like

1. Linear Regression
2. Lasso Regression
3. Decision Tree
4. Random Forest
5. XGBoost Algorithm

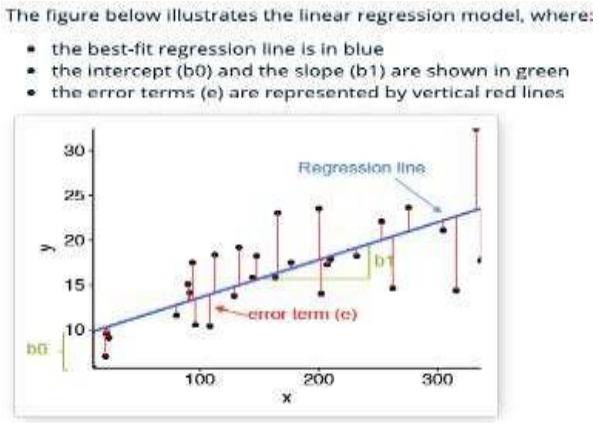
- ## Tuning the hyperparameters for better accuracy

  Tuning the hyperparameters of respective algorithms is necessary for getting better accuracy and to avoid overfitting in case of tree-based models like Decision Tree, Random Forest Classifier and XGBoost classifier.

- # Algorithms:

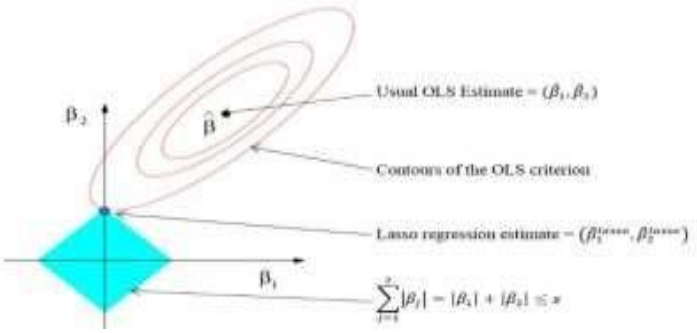
➢ Linear Regression-

linear regression is a regression model that estimates the relationship between one independent variable and one dependent variable using a straight line. Both variables should be quantitative. Linear regression most often uses mean-square error (MSE) to calculate the error of the model.



➢ Lasso Regression-

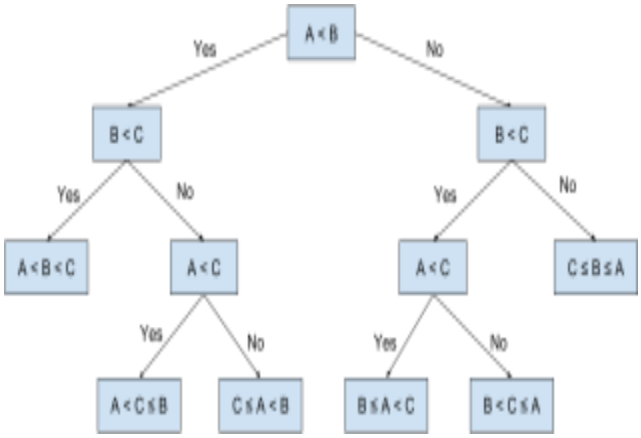
Lasso regression is a type of linear regression that uses shrinkage. Shrinkage is where data values are shrunk towards a central point, like the mean. The lasso procedure encourages simple, sparse models (i.e. models with fewer parameters). This particular type of regression is well-suited for models showing high levels of multicollinearity or when you want to automate certain parts of model selection, like variable selection/parameter elimination. The acronym “LASSO” stands for Least Absolute Shrinkage and Selection Operator



➢ Decision Tree-

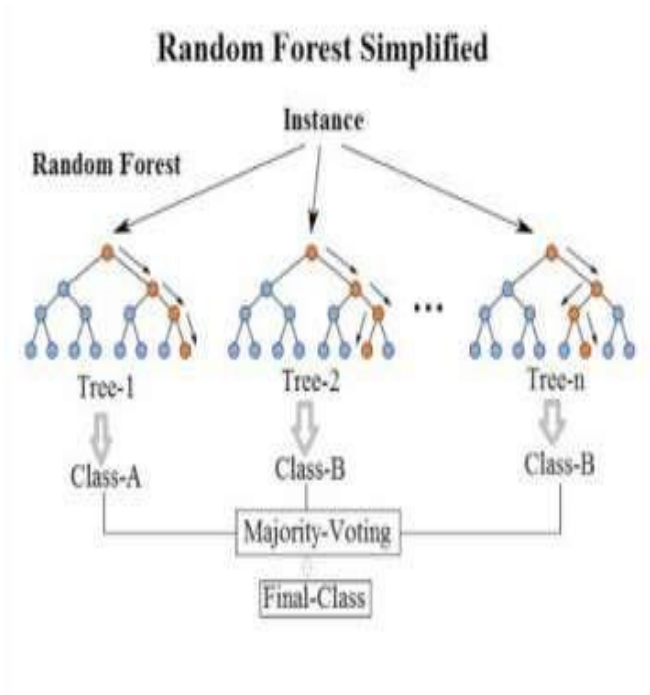
A decision tree is a graphical representation of all the possible solutions to a decision based on certain conditions. Tree models where the target variable can take a finite set of values are called classification trees and target

variable can take continuous values (numbers) are called regression trees.



➢ Random Forest Classifier-

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.



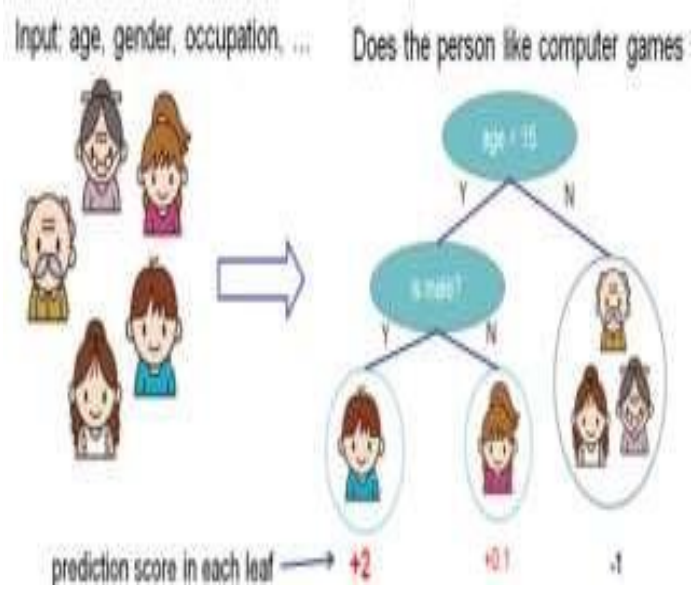
➢ XGBoost-

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.



- ## Model performance:

Model can be evaluated by various metrics such as:

1. R2 Score - The most common interpretation of r-squared is how well the regression model fits the observed data. For example, an r-squared of 60% reveals that 60% of the data fit the regression model. Generally, a higher r-squared indicates a better fit for the model.

2. Adjusted R2 Score –

Adjusted R-squared is a modified version of R-squared that has been adjusted for the number of predictors in the model. The adjusted R-squared increases when the new term improves the model more than would be expected by chance. It decreases when a predictor improves the model by less than expected. In other fields, the standards for a good R-Squared reading can be much higher, such as 0.9 or above.

3. Mean Square Error-

MSE represents the residual error which is nothing but sum of squared difference between actual values and the predicted / estimated values.

4. Mean Absolute error –

In the context of machine learning, absolute error refers to the magnitude of difference between the prediction of an observation and the true value of that observation. MAE takes the average of absolute errors for a group of predictions and observations as a measurement of the magnitude of errors for the entire group.

5. Root Mean Absolute error-

The RMSE is a quadratic scoring rule which measures the average magnitude of the error. The equation for the RMSE is given in both of the references. Expressing the formula in words, the difference between forecast and corresponding observed values are each squared and then averaged over the sample.

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. I 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 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.

Exploration conclusion-

Summary of exploratory data analysis-

- Working or Non- working Day We see 2 rental patterns across the day in bike rentals count - first. for a Working Day where the rental count high at peak office hours (8am and 5pm) and the second for a non-working day where rental count is more or less uniform across the day with a peak at around noon.
- Temperature- People generally prefer to bike at moderate to high temperatures.

We see highest rental counts between 32 to 35 degrees Celsius

- **Season-** Demand of rental bikes is high between February to October.
- Weather- As one would expect, we see highest number of bike rentals on a clear day and the lowest on a snowy or rainy day.
- Humidity- With increasing humidity, we see decrease in the number of bike rental count.

Conclusion –

So from the above operations we are coming to the conclusion that for XGBoost model performing very well than the other models. So in future if we want to do some prediction with this data then the XGBoost model will fit perfectly to do the prediction with good accuracy.

References-

1. MachineLearningMastery
2. GeeksforGeeks
3. Analytics Vidhya
4. TowardsDataScience
5. Stack Overflow