**NYC Taxi Time Prediction Analysis**

**Ajith R**

**Abstract:**

New York City taxi rides form the core of the traffic in the city of New York. Predicting the duration of a taxi trip is very important since a user would always like to know precisely how much time it would require for him to travel from one place to another. Prediction of duration can help users to plan their trips properly, thus keeping potential margins for traffic congestions. It can also help drivers to determine the correct route which in-turn will take lesser time as accordingly. Moreover, the transparency about trip duration will help to attract users.

We used six months data, of the year 2016 which contains data of customers who would provide at the start of a ride, or while booking a ride.

**Introduction**

New York City is one of the highly advanced cities of the world with extensive use of taxi services. Along with a vast population, the requirement of commonly available transportation serves the common purpose as it provides a very large transportation system. New York facilitates one of the largest subway systems in the world and comprises various green and yellow cabs with approximately counts of around 13,000 taxis. Most of the population of New York depends upon public transport, and it has been estimated that 54 percent of the people do not own a car or a personal vehicle. As a matter of fact, it accounts for almost 200 million taxi trips per year. Our goal here is to build a predictive model, which could help in predicting the time duration of each trip.

**Problem Statement:**

Task is to build a model that predicts the total ride duration of taxi trips in New York City. Your primary dataset is one released by the NYC Taxi and Limousine Commission, which includes pickup time, geo-coordinates, number of passengers, and several other variables.

* **id** - a unique identifier for each trip
* **vendor\_id** - a code indicating the provider associated with the trip record
* **pickup\_datetime** - date and time when the meter was engaged
* **dropoff\_datetime** - date and time when the meter was disengaged
* **passenger\_count** - the number of passengers in the vehicle (driver entered value)
* **pickup\_longitude** - the longitude where the meter was engaged
* **pickup\_latitude** - the latitude where the meter was engaged
* **dropoff\_longitude** - the longitude where the meter was disengaged
* **dropoff\_latitude** - the latitude where the meter was disengaged
* **store\_and\_fwd\_flag** - This flag indicates whether the trip record was held in vehicle memory before sending to the vendor because the vehicle did not have a connection to the server - Y=store and forward; N=not a store and forward trip
* **trip\_duration** - duration of the trip in seconds. (DV)

**Steps involved:**

**Data Pre-processing:**

Data preprocessing is a process of preparing the raw data and making it suitable for a machine learning model. It is the first and crucial step while creating a machine learning model. It involves the following:

* **Check Missing Data:** No null, NaN or missing values found in the dataset.
* **Create new variables:** New variables were derived using datetime variable and also distance and speed using coordinates and trip duration.
* **Encoding Categorical Data:** The categorical variables such as store and forward flag, months and days are encoded using one hot encoding.
* **Reducing data correlation:** This involves a correlation heatmap and based on that some features are removed in order to reduce collinearity to minimum.
* **Splitting dataset into training and test set:** The data is split into 80-20 format i.e. 80% for training and 20 % for testing purpose.

**Exploratory Data Analysis:**

Exploratory Data Analysis refers to the critical process of performing initial investigations on data so as to discover patterns, to spot anomalies, to test hypothesis and to check assumptions with the help of summary statistics and graphical representations. This includes univariate and bivariate analysis.

* **Univariate Analysis:** Univariate analysis is the simplest form of analysingdata. Each variable is analyzed separately.
* **Bivariate Analysis:** Bivariate analysis is stated to be an analysis of anyconcurrent relation between the dependent variable trip\_duration and other independent variables.

**Model Implementation:**

The models implemented for this dataset are:

* Linear Regression
* Lasso Regression
* Ridge Regression
* Decision Tree Regressor
* XGB Regressor

**Cross Validation and Hyperparameter Tuning:**

The cross validation and hyperparameter tuning are done using Grid Search CV. This is to reduce overfitting of the models.

**Algorithms:**

**Linear Regression:**

Linear regression analysis is used to predict the value of a variable based on the valueof another variable.

The variable you want to predict is called the dependent variable. The variable you are using to predict the other variable's value is called the independent variable.



**Data Preparation for Linear Regression:**

**Linear Assumption**: Linear regression assumes that the relationship between your independent and dependent is linear.

**Remove Outlier**: Linear regression assumes that your independent and dependent variables are not noisy.

**Remove Collinearity**: Linear regression will over-fit your data when you have highly correlated input variables.

**Gaussian Distributions:** Linear regression will make more reliable predictions if your independent and dependent variables have a Gaussian distribution.

**Rescale Inputs**: Linear regression will often make more reliable predictions if you rescale input variables using standardization or normalization.

**Regularisation**

### Regularized linear regression models are very similar to least squares, except that the coefficients are estimated by minimizing a slightly different objective function. we **minimize the sum of RSS and a "penalty term"** that penalizes coefficient size.

### **Lasso regression** shrinks coefficients all the way to zero, thus removing them from the model.

### 

### **Ridge regression** shrinks coefficients toward zero, but they rarely reach zero.

### 

### Visualizing Regularization

### 

### For **ridge**, this region is a **circle** because it constrains the square of the coefficients. Thus, the intersection will not generally occur on an axis, and so the coefficient estimates will typically be non-zero.

### For **lasso**, this region is a **diamond** because it constrains the absolute value of the coefficients. Because the constraint has corners at each of the axes, and so the ellipse will often intersect the constraint region at an axis. When this occurs, one of the coefficients will equal zero. In higher dimensions, many of the coefficient estimates may equal zero simultaneously. In the figure above, the intersection occurs at β1=0, and so the resulting model will only include β2.

### Random Forest

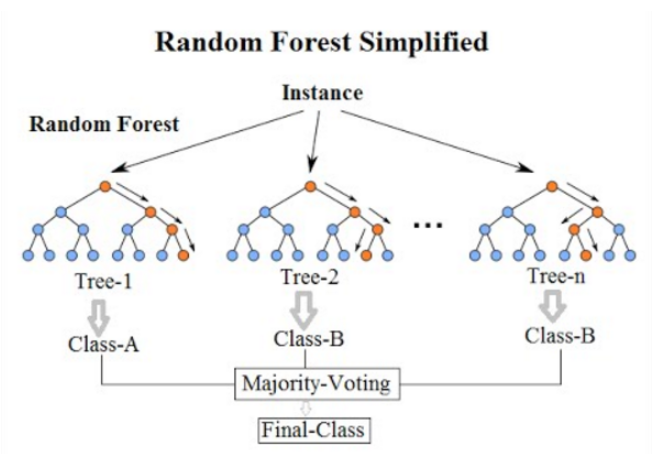
### In Random Forest, we grow multiple trees as opposed to a single tree in CART model.

### We construct trees from the subsets of the original dataset. These subsets can have a fraction of the columns as well as rows.

### To classify a new object based on attributes, each tree gives a classification and we say that the tree “votes” for that class.

### The forest chooses the classification having the most votes (over all the trees in the forest) and in case of regression, it takes the average of outputs by different trees.

### How does it work ?

****

**It works in the following manner:**

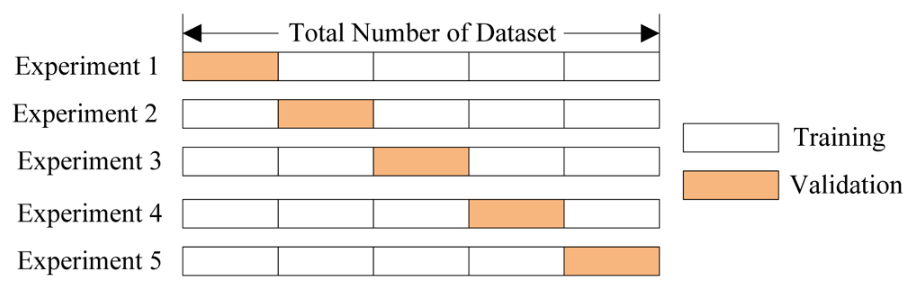
* Assume if the number of rows in the training set is N. Then, a sample of n < N rows is taken at random but *\*with replacement.\** This sample will be the training set for growing the tree.
* If there are M input variables, a number m < M is specified such that at each node, m variables are selected at random out of the M. The best split on these m is used to split the node. The value of m is held constant while we grow the forest.
* Each tree is grown to the largest extent possible and there is no pruning.
* Predict new data by aggregating the predictions of the n tree trees (i.e., majority votes for classification, average for regression).

**XGB Regressor**

XGBoost is an implementation of Gradient Boosted decision trees. In this algorithm, decision trees are created in sequential form. Weights play an important role in XGBoost. Weights are assigned to all the independent variables which are then fed into the decision tree which predicts results. The weight of variables predicted wrong by the tree is increased and these variables are then fed to the second decision tree. These individual classifiers/predictors then ensemble to give a strong and more precise model. It can work on regression, classification, ranking, and user-defined prediction problems.

**Cross Validation and Hyperparameter tuning**

### In cross-validation, we run our modelling process on different subsets of the data to get multiple measures of model quality. For example, we could have 5 folds or experiments. We divide the data into 5 pieces, each being 20% of the full dataset.



Hyperparameters are sets of information that are used to control the way of learning an algorithm. Each algorithm requires a specific hyperparameters grid that can be adjusted according to the business problem.

We have used GridSearchCV which is available in sci-kit learn. Grid Search combines a selection of hyperparameters 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.

**Model Performance**

A regression model performance can be evaluated using the following metrics.

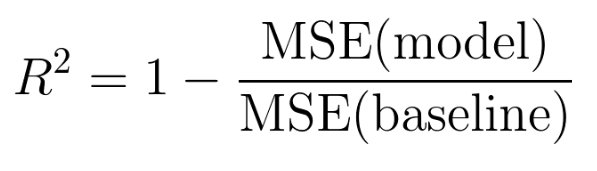
* R^2
* Adjusted R^2
* Mean Squared Error
* Root Mean Squared Error

**R^2:**

This 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.

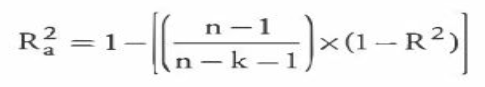


**Adjusted R^2:**

Adjusted R² depicts the same meaning as R² but is an improvement of it.

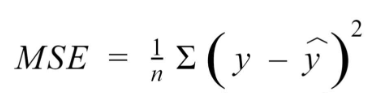
R² suffers from the problem that the scores improve on increasing terms even though the model is not improving which may misguide the researcher.

Adjusted R² is always lower than R² as it adjusts for the increasing predictors and only shows improvement if there is a real improvement.



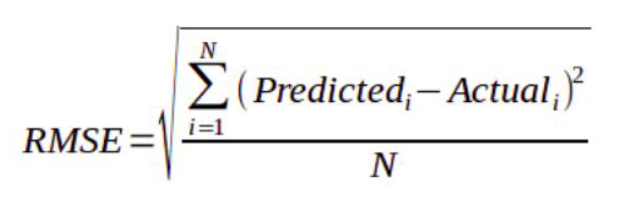
**Mean Squared Error:**

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.



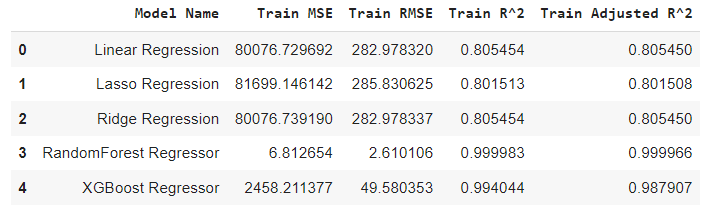
**Root Mean Squared Error:**

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.

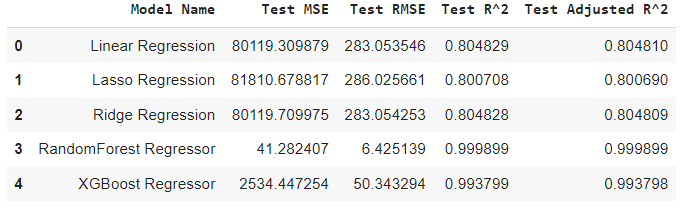


**Model Analysis of Train and Test data using above metrics**

**Train Data:**



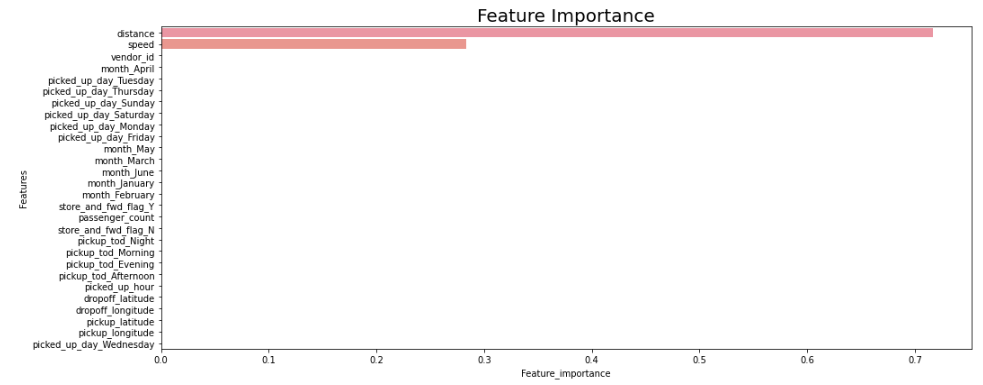
**Test Data:**



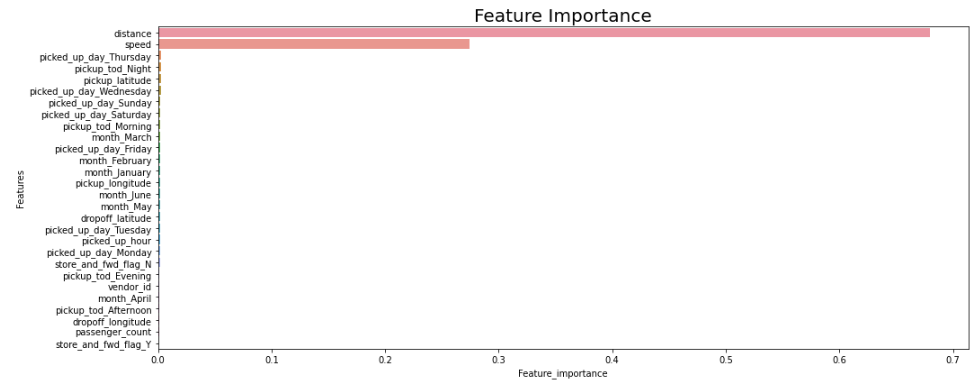
Random Forest Regressor and XGBoost Regressor are the best models.

**Feature Importance**

**Random Forest Regressor**



**XGB Regressor**



**Conclusion**

* Out of all the models Randomforest Regressor performed well in prediction of trip duration time with R2 score of 99.99% and also XGBoost Regressor did well too
* Linear Regression model too performed well with R2 score of 80.48% which isn't a bad score but we had to drop correlated variables for linear assumption
* Whereas both ensemble models did well when we included correlated variables since they're Non parametric models
* RandomForest Regressor has the best score and less error, also if we compare with Residual sum squares this model has the least value
* RandomForest Regressor is the best model for trip duration prediction

**References**

* Almabetter
* GeeksforGeeks
* AnalyticsVidhya