**NYC Taxi Trip Time Prediction**

**Krushna Chaure**

**Data Science Trainee,**

**AlmaBetter, Bangalore**

**Abstract**

In New York City many of people commute to different regions of city via taxi. A lot of streets and roads in New York city are quite busy due to traffic jams, construction, or road blockage etc. Therefore, it is very important to predict the trip duration of taxi so that the user will know how much time it will take to commute from one place to other. Also, due to the increasing popularity of app-based taxi such as ola or uber and there competitive pricing levels. Decisions has to be taken by the user for opting which one to choose based on trip pricing and duration. This prediction also helps drivers to choose route having lesser trip time. We were provided with dataset which is released by NYC Taxi and Limousine Commission. This dataset contains pickup time, drop-off time, geo- coordinates, number of passengers, trip duration and several other variables.

**Introduction**

More than 7 billion people exist on earth. With necessities of food, water and shelter there also a key requirement of commutating from one place to other. Rapid advancement in technology in the last two decades leads to adaption of a more efficient way of transportation via internet and app-based transport system. New York city is one of such advanced city with extensive use of transportation via subways, buses and taxi services. New York has more then 10,000 plus taxi and nearly 50% of population doesn’t have a personal vehicle. Due to this facts most people used taxi has a there primary mode of transport and it accounts for more than 100 millions taxi trips per year.

Our primary motives are to analyse the dataset, perform feature engineering to comes up with suitable independent features and building a good model that will help us in predicting the trip duration of NYC taxi.

Here, for prediction the taxi trip duration we have applied a linear regression, Decision Tree Regressor and then we have applied Random Forest regressor and XGBoost To find out which will give better accuracy and with lesser amount of prediction time. At last, a comparison of the two mentioned algorithms facilitates us to decide that XGBoost is more fitter and efficient than Multi-Layer Perceptron for taxi trip duration-based predictions.

**Problem Statement**

The main objective is to build a predictive model, which could help them in predicting the trip duration of taxi. This would in turn help them in matching the right cabs with the right customers quickly and efficiently.

**Dataset Description**

The dataset is based on the 2016 NYC Yellow Cab trip record data made available in Big Query on Google Cloud Platform. The data was originally published by the NYC Taxi and Limousine commission (TLC). This dataset contains around 1458644 records and 11 features.

**The data set contains the following columns:**

* 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.

**Feature Engineering**

**1. Data Loading and general check ups**

We have loaded the data from the given csv files using a function from pandas library. Then we checked the general information about data. We observed that the data contains 1458644 records and 11 features. We see that our data contains three different data types i.e. floats, strings and datetime objects.

**2. Null values Treatment**

We inspected the dataset and found out that our dataset has no null value present in it. So, no need to do null value treatment.

**3. Exploratory Data Analysis**

We begin our EDA by first checking the distribution of our dependent variable i.e. trip duration. We observed that the data is highly positively skewed. We also plotted the box plot and observed that there are many outliers present in the variable. To cross check this trip duration we have calculated the difference in pick and drop off timing and matched with trip duration we observed no difference. Thus, there is no miscalculation or falsified entries. To eliminate the outliers, we have segregated the data variable into different segment and observed that majority of trip duration is within an hour some observations are within two days but a very few observation are having more than two days. We eliminate such values from out dataset.

We removed id variable as it doesn’t give much interpretation. We then calculated the distance based on haversine formula from pickup and drop-off latitude and longitude. Then we plotted the box plot for the variable and observed there are many outlier so we segregate this variable and see that most of the trip are within 10km, some trip are within 50km while a very few trip crosses 50km. so we eliminate trip with 0 and above 50km distance.

We then checked for categorical variable store\_and\_fwd\_flag and passenger\_count. We observed the store and fwd. flag contain majority of one category. So we drop this feature. Passenger count variable has entries from 0 to 9. Since there is no trips with 0 passenger either this a miss entry or the driver forgot to enter passenger count of that trip. Also in a taxi maximum six person are allowed to sit including minor. So we eliminate 0 and 7-9 records from our dataset.

We also created some more feature i.e. pickup month, pickup weekday and pickup hour. To get a good insight of trip duration and drop pickup date and drop-off time column. Then we checked for correlation between variables and observed that geographic coordinates are very less correlated and VIF is also high between this variables so we drop off this variable from our data set.

**4. Encoding of categorical columns**

Since some of our categorical variable are in string format. So we cannot passed this variable to our model directly so we have to use one hot encoding(like dummify dataset) to convert it into numerical variable having binary integers 0 and 1.

**5. Standardization of features**

This is one of the important step for getting good accuracy as you can see there are some columns having different ranges of values then other column. Therefore. It is important to do scaling the data so that our data set will have uniformity and we get good accuracy. So, here we use PCA(Principal Component Analysis) function.

**Machine Learning Models Implementation**

Here we implemented 5 ML Model:-

1. Multiple Linear Regression
2. Regularized Lasso Regression
3. Regularized Ridge Regression
4. Random Forest Regressor
5. XGBoost Regressor

**1. Multiple Linear Regression**

Linear Regression is a regression of dependent variable on independent variable. It is a linear model that assumes a linear relationship between dependent (y) and independent variables (x). The dependent variable

(y) is calculated by linear combination of independent variable (x).

Y=B0+B1x1+B2x2

The cost function for linear regression is given by:

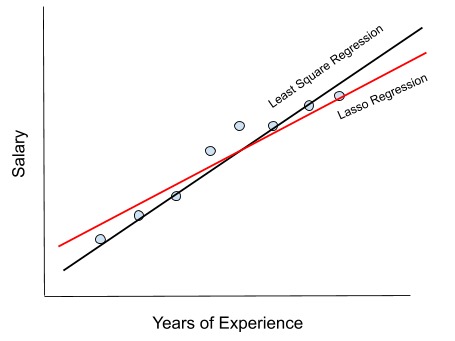
Minimum sum of square error MSSE= 𝑛(𝑌 𝑎𝑐𝑡 − 𝑌 𝑝𝑟𝑒𝑑)2



**2. Regularized 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).

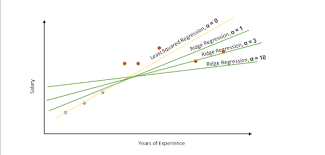
The main advantage of a LASSO regression model is that it has the ability to set the coefficients for features it does not consider interesting to zero. This means that the model does some automatic feature selection to decide which features should and should not be included on its own.



**3. Regularized Ridge Regression**

Ridge regression is a model tuning method that is used to analyse any data that suffers from multicollinearity. This method performs L2 regularization. When the issue of multicollinearity occurs, least-squares are unbiased, and variances are large, this results in predicted values being far away from the actual values.

It protects the model from overfitting. It does not need unbiased estimators. There is only enough bias to make the estimates reasonably reliable approximations to the true population values.

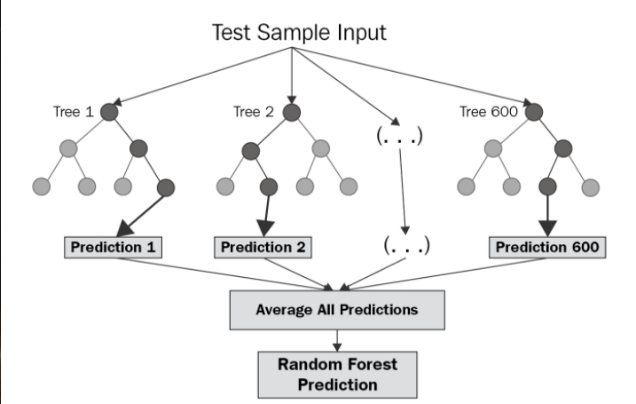


**4. Random Forest Regressor**

A random forest regressor. A random forest is a meta estimator that fits a number of classifying decision trees on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting.

Random forest regression is used to solve a variety of business problems where the company needs to predict a continuous value: Predict future prices/costs.

It can perform both regression and classification tasks. A random forest produces good predictions that can be understood easily. It can handle large datasets efficiently. The random forest algorithm provides a higher level of accuracy in predicting outcomes over the decision tree algorithm

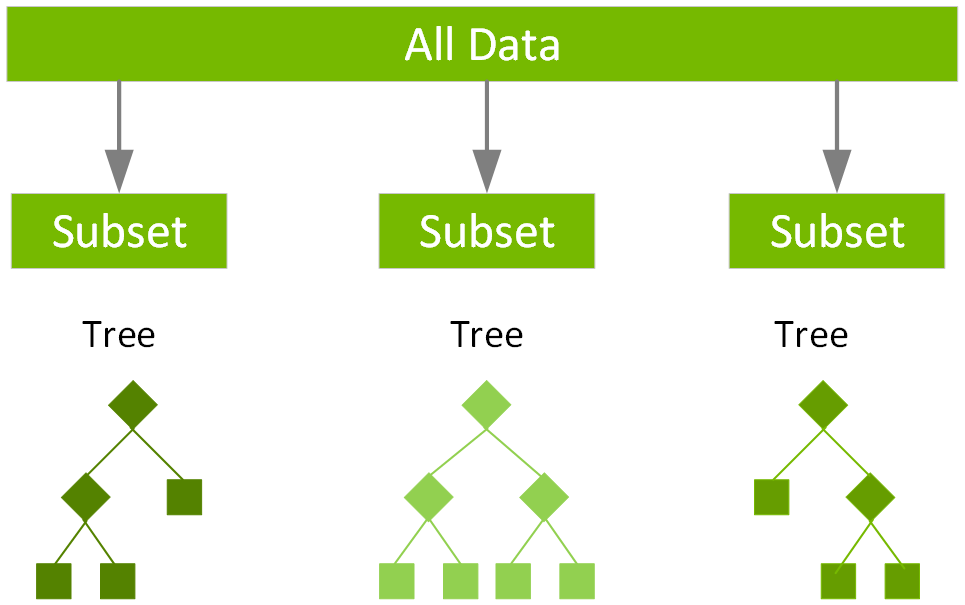


**5. XGBoost Regressor**

XGBoost stands for Extreme Gradient Boosting is a popular and efficient open-source implementation of the gradient boosted trees algorithm. Gradient boosting is a supervised learning algorithm, which attempts to accurately predict a target variable by combining the estimates of a set of simpler, weaker models.

XGBoost is a boosting algorithm that uses bagging, which trains multiple decision trees and then combines the results. It allows XGBoost to learn more quickly than other algorithms but also gives it an advantage in situations with many features to consider.

it's the efficiency, accuracy, and feasibility of this algorithm. It has both linear model solver and tree learning algorithms. So, what makes it fast is its capacity to do parallel computation on a single machine.

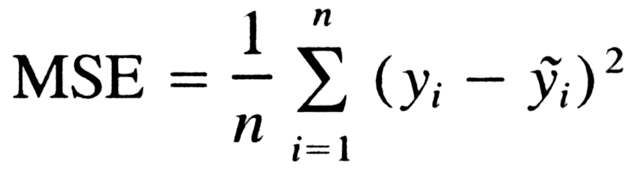


**Model Performance**

The model performance can be evaluated by various regression metrics such as:

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

Mean squared error is the most widely used evaluation metric for regression task. It is the average of squared difference between actual and predicted value of dependent variable



**2. Root Mean Square Error**

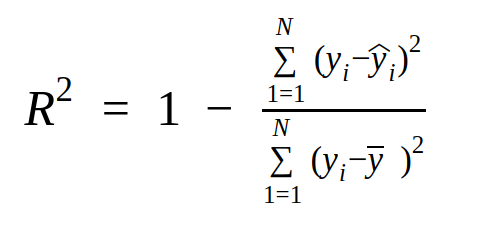
RSME (Root mean square error) calculates the transformation between values predicted by a model and actual values. In other words, it is one such error in the technique of measuring the precision and error rate of any machine learning algorithm of a regression problem.

Root mean square error or root mean square deviation is one of the most commonly used measures for evaluating the quality of predictions. It shows how far predictions fall from measured true values using Euclidean distance.



**3. R2 Score**

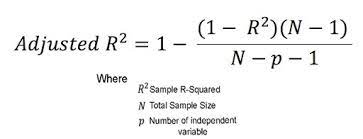
Coefficient of determination also called as R2 score is used to evaluate the performance of a linear regression model. It is the amount of the variation in the output dependent attribute which is predictable from the input independent variable(s). It is used to check how well-observed results are reproduced by the model, depending on the ratio of total deviation of results described by the model.



**4. Adjusted R2 Score**

Adjusted R2 is a corrected goodness-of-fit (model accuracy) measure for linear models. It identifies the percentage of variance in the target field that is explained by the input or inputs. R2 tends to optimistically estimate the fit of the linear regression. It always increases as the number of effects are included in the model. Adjusted R2 attempts to correct for this overestimation. Adjusted R2 might decrease if a specific effect does not improve the model. Adjusted R squared is calculated by dividing the residual mean square error by the total mean square error (which is the sample variance of the target field). The result is then subtracted from 1.

Adjusted R2 is always less than or equal to R2. A value of 1 indicates a model that perfectly predicts values in the target field. A value that is less than or equal to 0 indicates a model that has no predictive value. In the real world, adjusted R2 lies between these values.



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.

**2. Randomized Search CV**

In Random Search, the hyperparameters are chosen at random within a range of values that it can assume. The advantage of this method is that there is a greater chance of finding regions of the cost minimization space with more suitable hyperparameters, since the choice for each iteration is random. The disadvantage of this method is that the combination of hyperparameters is beyond the scientist’s control.

**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 these models our accuracy revolves in the range of 70 to 74%. And there is no such improvement in accuracy score even after hyperparameter tuning. So, the accuracy of our best model is 73% which can be said to be good for this large dataset. This performance could be due to various reasons like no proper pattern of data, too much data, not enough relevant features.

**Acknowledgement**

This project was completed by Krushna Chaure. I am extremely grateful to the celebrated authors whose precious works have been consulted and referred to in this project work. I also wish to convey my appreciation to our peers who provided encouragement and timely support in the hour of need.

**References**

1. Stackoverflow
2. Geeksforeeks
3. Jovian
4. Towardsdatascience
5. Medium