NYC Taxi Trip Time Prediction

Manas Ranjan Behera,

Data Science Trainee, Bangalore

Abstract :

Predicting a trip duration isn't something that has not been though upon. With the use of Google maps API one can find the estimated time it would take to move between two points in the city. However, a detailed analysis of the factors affecting a trip between two points in a city can be very useful for accurate and robust prediction. Trip duration is not as simple as it seems.

It is data dependent and is governed by a lot many factors apart from distance and speed. This research primarily focuses on the possible important factors that are used as attributes for the trip duration prediction in the New York City. This data can be used by taxi vendors for better services to the users. The research work not only uses a prediction model but also gives an in-depth analysis of the factors associated with the New York City taxi trips. A city like New York is expected to have various factors and variations with respect to the trip durations.

This research work involves application of relevant machine learning algorithms such as linear regression, lasso regression, ridge regression, decision tree and XGBoost algorithms for completion of the task. The final algorithm used in this research work is XGBoost algorithm as it yields the best result when compared with other methods employed for the same task. A root mean square error of 0.1177 was achieved when the test data that consisted of about3,64,000 data points were given as an input to the training model.

1 . Introduction

For a good taxi service and its integration with the existing transportation system the project serves as a right means to comprehend the traffic system in the New York City. For prediction purposes factors such as pick up latitude, pick up longitude, drop off latitude and drop off longitude etc.is considered. The primary focus of this project is in depth analysis of the factors associated with a taxi trip in NYC.

Primarily I used different types of machine learning models like Linear Regression, Lasso Regression, Ridge Regression, XGBoost, Decision Tree, from which we got the accurate trip duration prediction between two points in the New York City by the help of XGBoost.

The machine learning model also gives out analysis graphs where the average speed of travel is plotted against factors such as the month of the year, time of the day, the day of the week. These plots help in analysing the trends of city traffic throughout the year. Finally, the machine learning model gives an insight about how the data obtained from a trip can be used not only for prediction but also for finding trends and patterning the taxi rides that form such an important part of urban cities like New York

2 . Methodology

The trip duration and analysis can be broadly divided into three working modules. These modules have been explained below

- **Data Pre-processing**

In order to prepare the data for the experiment, the data is cleaned to remove outliers. It involves removing the null values, replacing missing values with dummy values, and changing the format of the date and time values so as to utilize them in the algorithm. It also includes removal of outliers.

- **Feature Creation**

In order to extract the features, the distance between the pickup and drop-off locations was found using grat_circle function from geopy.distance library. We remove columns which are not important for further analysis such as id, pickup_datetime, dropoff_datetime,

store_and_fwd_flag, pickup_weekday, dropoff_weekday, pickup_weekday_num, pickup_timeofday, trip_duration,speed.

We draw heat map to find correlation between different independent features and dependent feature. If correlation between independent features are high and has very less relation with dependent feature, remove them.

● Training and Testing

We used 5 different models.

A - Linear Regression : The linear regression model finds the set of θ coefficients that minimize the sum of squared errors.

B - Lasso Regression : The lasso method was used to shrink coefficients. For duration prediction models, lasso was run using a range of values for the penalizing parameter, λ . Grid Search was used to find the lasso model with the lowest error and select the value of λ to use.

C - Ridge Regression : To further confirm the best set of covariates to use, the regression method was used. It performs L2 regularization, i.e. adds penalty equivalent to square of the magnitude of coefficients.

D - Decision Tree : The decision trees was also built on the training data in order to improve prediction accuracy .We used GridSearch to tune the hyper parameters of Decision Tree to get the best possible test score.

E - XGBoost : It was used for final prediction of the trip duration in the test dataset. The dataset was very large, as a result for this type of problem XGBoost was applied in which all the attributes were taken and parallel processing of boosting trees executed. Another aspect of XGBoost is that it keeps a nice check between bias and variance which helps in better prediction. The results were interpreted by using GridSearch, the XGBoost hyper parameters.

3 . Dataset- New York City Taxi Duration Dataset

Dataset is based on the 2016 NYC Yellow Cab trip record datamade available in Big Query on Google Cloud Platform. Thetraining set (contains 1458644 trip records) and the testing set(contains 625134 trip records). The attribute features of the datasetinclude:

● 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.

4 . Steps Involved

● Exploratory Data Analysis

After loading the dataset we performed the method of comparing our target variable that are passenger_count, pickup_longitude, pickup_latitude, dropoff_longitude, dropoff_latitude, pickup_hour, month, distance with other independesnt 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 a large number of null values which might tend to disturb our accuracy hence we dropped them at the beginning of our project inorder to get a better result.

● **Feature Engineering**

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.

● **Correlation Analysis**

I applied correlation analisys in every new features to get the idea between highly correlated features to apply the model on it. I droped the features which have high correlation ie. 1.

● **Standardization of fratures**

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.

● **Recursive Feature Elimination (RFE)**

I add a constant variable the model for extra precaution in model creation

● **Fitting Different Model**

From modelling er tried various classification algorithms like :

1 . Linear Regression

2 . Lasso Regression

3 . Ridge Regression

4 . Decision Tree

5 . XGBoost

● **Tuning the hyper parameters for better Accuracy**

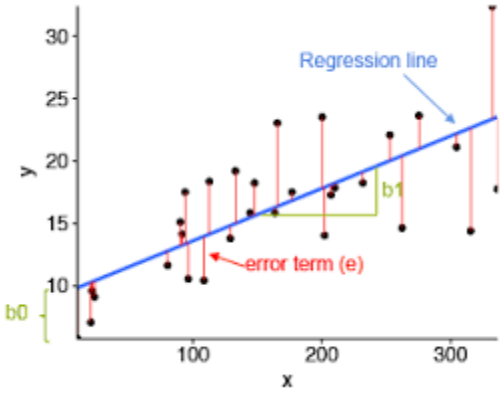
Tuning the hyper parameters of respective algorithms is necessary for getting better accuracy and to avoid over fitting in case of tree based models

like Decision Trees and XGBoost classifier.

5 . Algorithms

A . Linear Regression

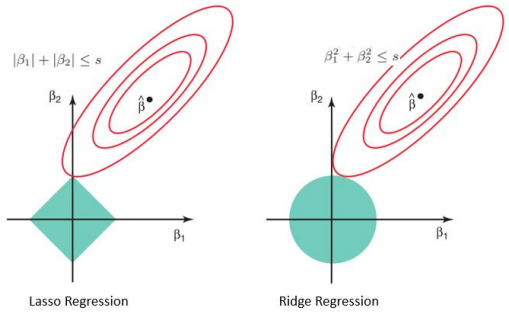
Linear regression analysis is used to predict the value of a variable based on the value of another variable. The variable you want to predict is called the dependent variable. The variable you are using to predict the other variable's value is called the independent variable.



This form of analysis estimates the coefficients of the linear equation, involving one or more independent variables that best predict the value of the dependent variable. Linear regression fits a straight line or surface that minimizes the discrepancies between predicted and actual output values. There are simple linear regression calculators that use a “least squares” method to discover the best-fit line for a set of paired data. You then estimate the value of X (dependent variable) from Y (independent variable).

B . Lasso Regression

Lasso regression performs L1 regularization, which adds a penalty equal to the absolute value of the magnitude of coefficients. This type of regularization can result in sparse models with few coefficients; Some coefficients can become zero and eliminated from the model. Larger penalties result in coefficient values closer to zero, which is the ideal for producing simpler models. On the other hand, L2 regularization (e.g. Ridge regression) doesn't result in elimination of coefficients or sparse models. This makes the Lasso far easier to interpret than the Ridge.



Lasso Regression Ridge Regression

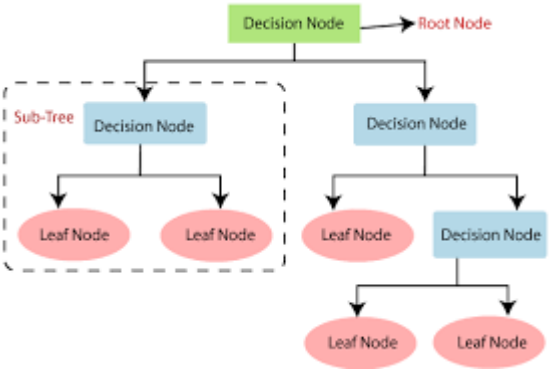
C . Ridge Regression

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.

The cost function for ridge regression:

Min(||Y – X(theta)||^2 + λ||theta||^2)

D . Decision Tree Classifier



Decision tree is a type of supervised learning algorithm that can be used for both regression and classification problems. The algorithm uses training data to create rules that can be represented by a tree structure.

Like any other tree representation, it has a root node, internal nodes, and leaf nodes. The internal node represents condition on attributes, the branches represent the results of the condition and the leaf node represents the class label.

To arrive at the classification, you start at the root node at the top and work your way down to the leaf node by following the if-else style rules. The leaf node where you land up is your class label for your classification problem.

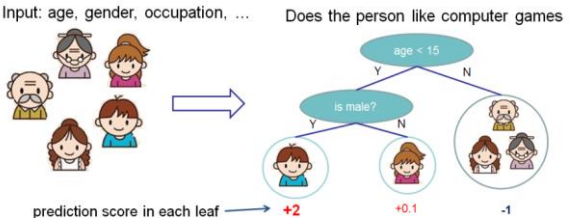
Decision tree can work with both categorical and numerical data. This is in contrast with other machine learning algorithms that cannot work with categorical data and requires encoding to numeric values.

E . 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 ossible feature splits.

6. Feature Importance

Feature Importance refers to techniques that calculate a score for all the input features for a given model — the scores simply represent the "importance" of each feature. A higher score means that the specific feature will have a larger effect on the model that is being used to predict a certain variable.

Feature Importance is extremely useful for the following reasons:

1) Data Understanding.

Building a model is one thing, but understanding the data that goes into the model is another. Like a correlation matrix, feature importance allows you to understand the relationship between the features and the target variable. It also helps you understand what features are irrelevant for the model.

2) Model Improvement.

When training your model, you can use the scores calculated from feature importance to reduce the dimensionality of the model. The higher scores are usually kept and the lower scores are deleted as they are not important for the model. This not only makes the model simpler but also speeds up the model's working, ultimately improving the performance of the model.

3) Model Interpretability.

Feature Importance is also useful for interpreting and communicating your model to other stakeholders. By calculating scores for each feature, you can determine which features attribute the most to the predictive power of your model.

I have taken the features like month, distance, and price per weekdays like Monday Saturday Sunday Thursday to calculate the weight of importance for all the features.

7 . Model Evaluation :

Model evaluation is the process of using different evaluation metrics to understand a machine learning model's performance, as well as its strengths and weaknesses. Model evaluation is important to assess the efficacy of a model during

initial research phases, and it also plays a role in model monitoring.

We have calculated the MSE, RMSE, R2 and adjusted R2 for each model. WEhave found that the the model with having least value of RMSE and MSE is XGBoost model also it has the lowest value of R2.

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. Also find out the feature importance of the variables.

MSE and RMSE of XGBoost model are very similar and their R2 is 82.

From above table, we can conclude XGBoost is best model for our dataset.

So the accuracy of our best model(XGBoost) is 82% 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.