Technical Documentation

NYC Taxi Trip Time Prediction

Completed by:

Ankush Kumar

Data Science Trainees @ AlmaBetter, Bangalore

Abstract:

Our model's goal is to accurately forecast the time it will take a taxi to travel between one pickup location and another drop-off location. In the hurried, time-constrained world of today, everyone wants to know how long it will take to get where they're going so they can move their plans along. Therefore, we already have billion-dollar startups like Uber and Ola where we can track the duration of our trips for their peace of mind. As a result, we suggested a method by which each taxi service provider could provide precise trip duration to their customers while taking into account variables like traffic, time, and day of pickup.

We suggest a technique for predicting trip duration in which we combine several algorithms, fine-tune the corresponding algorithmic parameters by comparing each parameter to the RMSE, and then predict the trip duration. We used a variety of models, including Linear Regression, Random Forest, and Decision Trees, to make our prediction. In order to handle missing data, eliminate duplication, and resolve data conflicts, we also examined a number of data mining techniques.

Problem Statement and Objective:

The task is to create a model that forecasts how long taxi rides will last overall in New York City. Your main dataset, which includes pickup time, geo-coordinates, the number of passengers, and several other variables, was made public by the NYC Taxi and Limousine Commission.

Introduction

There are numerous ways to get from one place in a city to another, but the taxi trip has more uses than any other mode of transportation in urban areas. When given the necessary set of parameters that influence trip duration, it becomes extremely important to analyse and forecast trip duration between two points in the city. The project serves as an appropriate way to understand the traffic system in New York City in order to provide a good taxi service and integrate it with the current transportation system. Predictions are made taking into account variables like pick-up latitude, pick-up longitude, drop-off latitude, drop-off longitude, etc.

Technical Documentation

The total trip duration is predicted using these geographic locations along with other crucial variables like the pick-up date and time. This project's main objective is an in-depth examination of the elements involved in a taxi ride in New York City.

Data information:

Based on data from the 2016 NYC Yellow Cab trip records made available in Big Query on the Google Cloud Platform, the dataset was created. The NYC Taxi and Limousine Commission initially released the information (TLC). For the purposes of this project, the data was cleaned and sampled. Predict the length of each trip in the test set based on its unique trip attributes.

the training set: NYC Taxi Data (contains 1458644 trip records)

- **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.

METHODOLOGY FOR WHOLE ANALYSIS:

- Exploratory Data Analysis
- Correlation
- Transformation
- Baseline Model
- Performance Measures
- Optimization
- Feature Importance
- Improved Models
- Conclusions

Technical Documentation

Exploratory Data Analysis (EDA):

Data Cleaning and Removal of Duplicity:

- There was not any null value
- And also, no duplicate data was present in our data.

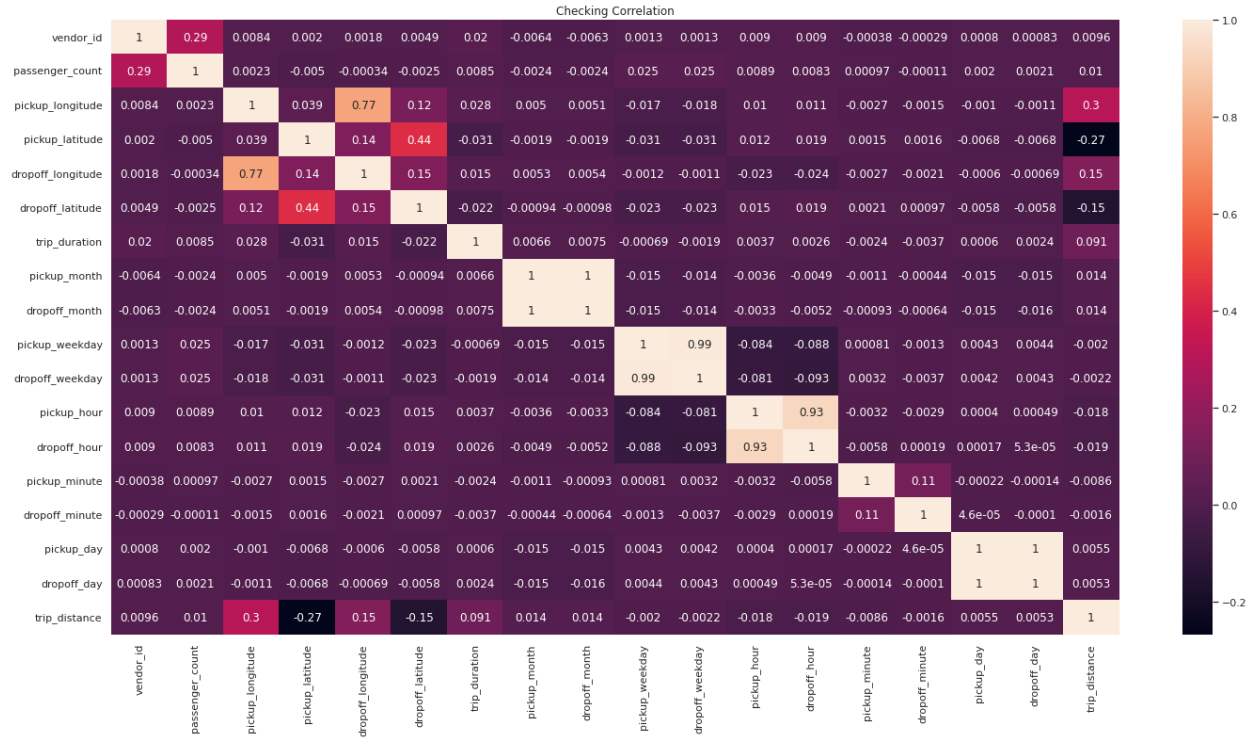
Data Visualization

- We have performed different types of visualization on the basis of Univariate, Bivariate and multivariate analysis.
- Firstly, we have conducted Univariate analysis as we also need to understand various features/columns individually that what kind of importance and insights they bring for our analysis.
- Secondly, we have performed Bivariate analysis so that we can analyse the impact of one column/feature to another feature and where these insights lead us.
- At last, we have performed Multivariate analysis in which we came to know the impact of multiple features.

We have concluded some important and leading insights from our EDA analysis:

- Vendor 2 there are more number of bookings which is of 54 %.
- When a taxi ride is booked by only a single person there are more number of bookings is high as compare to multiple people booking the taxi ride.
- Weekends 4 - Friday | 5 - Saturday there are high booking rate for taxi as compare to other days.
- This indicates that people use to go out for their celebrations | parties | or may be for other personnel works on weekends.
- Morning after 10 O'Clock people use to book taxi because they want to go out to their work places.
- And at in the evening after 6 O'Clock the taxi demand tends to in peak.
- both pickup and dropoff day time count plot booking count is maximum in the EVENING day time.
- Least trip duration occured in january and highest started occuring after the month of march.
- Trip duration start increases from 3rd month before that it is quite constant.

Correlation: As we know that correlation is a very important observation task to get the proper idea that how our features are independent from each other and if they are not independent and are related to each other then there will be lots of ambiguity for our model creations that misleads our analysis.



'dropoff_day', 'dropoff_hour', 'dropoff_month', 'dropoff_weekday' are highly corelated we can drop these features.

Feature Transformation:

- We have many features that have some categorical values in them like **'store_and_fwd_flag', 'pickup_day_time', 'dropoff_day_time'**

We have applied LabelEncoder from Scikit Learn and transformed these features to numeric format.

LabelEncoder:

Sklearn is a very efficient tool for encoding categorical feature levels into numeric values. LabelEncoder encodes labels with values ranging from 0 to n classes-1, where n denotes the number of distinct labels.

Technical Documentation

Different machine learning models are used to predict the label class and also used PCA decomposition:

Random Forests

Definition:

- The random forest classification technique is made up of several decision trees. When building each individual tree, it employs bagging and feature randomness in an attempt to produce an uncorrelated forest of trees whose forecast by committee is more accurate than that of any individual tree.

Algorithm

- Each tree in the random forest generates a class prediction, and the class with the most votes becomes the prediction of our model.
- A large number of relatively uncorrelated models (trees) acting as a committee will outperform any of the individual constituent models.
- The reason for this wonderful effect is that the trees protect each other from their individual errors (as long as they don't all err in the same direction all of the time).

Decision Trees

Definition:

Decision Tree is a predictive modelling tool that finds ways to divide a data set based on various conditions. It is built using an algorithmic approach. It is among the most popular and useful techniques for supervised learning. A non-parametric supervised learning technique called decision trees is used for both classification and regression tasks. The objective is to learn straightforward decision rules inferred from the data features in order to build a model that predicts the value of a target variable.

Working

- In order to generate the best decision trees possible, the variables that act as nodes in the decision trees are determined by calculating the Information Gain of each independent variable in the dataset.
- To define Information Gain precisely, we must first define entropy, a measure commonly used in information theory that measures the level of impurity in a set of examples.

Linear Regression:

Definition: A variable's value can be predicted using linear regression analysis based on the value of another variable. The dependent variable is the one you want to be able to predict. The

independent variable is the one you're using to make a prediction about the value of the other variable.

Working

- Regression is a statistical technique used in the fields of finance, investing, and other disciplines that aims to establish the nature and strength of the relationship between a single dependent variable (typically denoted by Y) and a number of independent variables (known as independent variables).

XGBOOST Regressor:

Definition:

- Extreme Gradient Boosting (XGBoost) is a distributed, scalable gradient-boosted decision tree (GBDT) machine learning library. The top machine learning library for regression, classification, and ranking issues, it offers parallel tree boosting.

Working

- It is predicated on the hunch that when previous models are combined with the best possible next model, the overall prediction error is minimized. Setting the desired results for this subsequent model in order to reduce error is the key concept.

Principal Component Analysis (PCA):

Definition:

- In order to reduce the dimensionality of large data sets, a technique known as principal component analysis, or PCA, is frequently used. PCA works by condensing a large set of variables into a smaller set that still retains the majority of the data in the larger set.

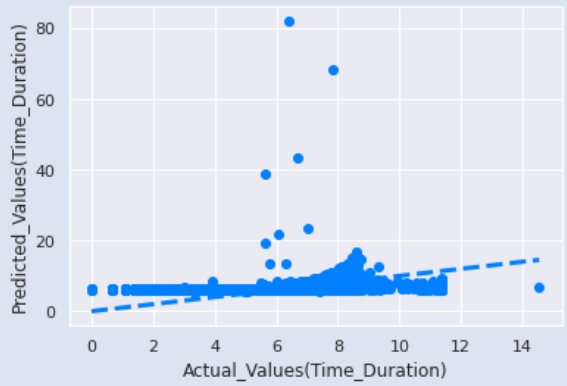
Working

- Set the range of continuous initial variables to a standard
- Make a covariance matrix calculation to find correlations.
- To find the principal components, compute the eigenvectors and eigenvalues of the covariance matrix.
- To decide which principal components to keep, create a feature vector.
- Transform the data along the axes of the principal components

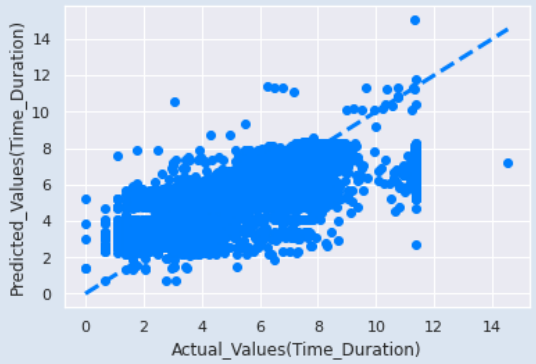
Results:

Observation by applying baseline model:

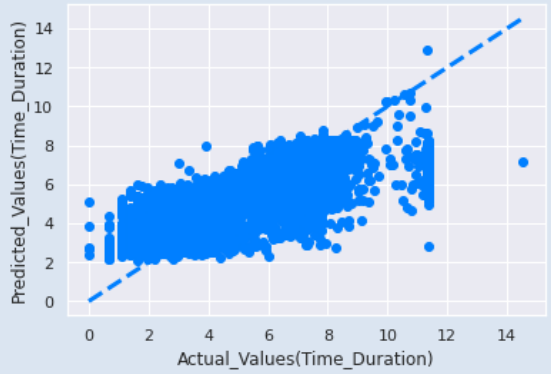
Linear Regression



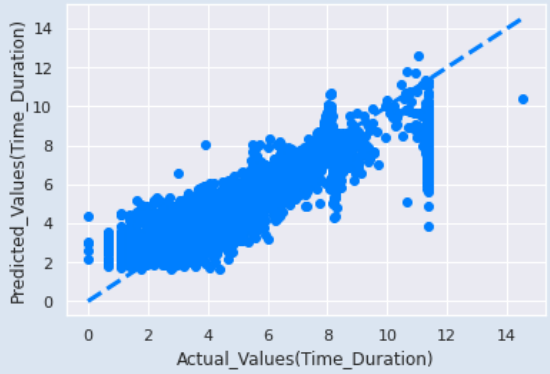
Decision Tree



Random Forest



XGBOOST



	Training_Score	Testing_Score	R2_Score	ADJ_R2	MSE	RMSE
Linear Regression	0.298175	0.274146	0.274146	0.274117	0.449004	0.670078
Decision Tree Regression	0.610109	0.601903	0.601903	0.601887	0.246258	0.496244
Random Forest Regression	0.627799	0.621592	0.621592	0.621577	0.234078	0.483816
XGBOOST	0.924322	0.907907	0.907907	0.907904	0.056967	0.238678

In terms of accuracy measure, which is not up to par, as well as in terms of R2, which is very low and RMSE, which is quite high, the linear regression model is not performing well at all. In terms of accuracy, R2, and RMSE evaluation for all matrices, Decision Tree Regressor and Random Forest Regressor perform marginally better than Linear Regression.

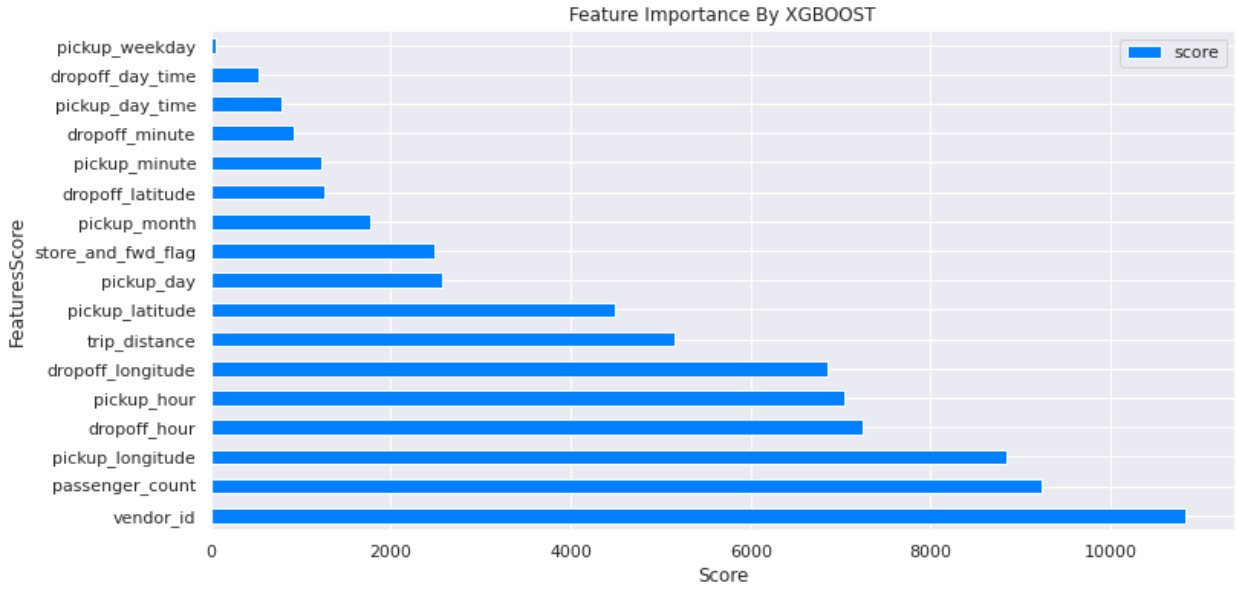
But in terms of accuracy, R2 Score, and RMSE evaluation, Random Forest Regressor is marginally superior to Decision Tree Regressor. As we can see, XGBOOST is doing a fantastic job and offering us excellent training and testing accuracy. Additionally, the R2 score is much better than other models, and the RMSE value is also very low.

Optimized Scaled Feature Result:

	Training_Score	Testing_Score	R2_Score	ADJ_R2	MSE	RMSE
Linear Regression	0.290377	0.292821	0.292821	0.292794	0.435413	0.659858
Decision Tree Regression	0.612110	0.600369	0.600369	0.600354	0.246054	0.496039
Random Forest Regression	0.631115	0.621728	0.621728	0.621713	0.232904	0.482601
XGBOOST	0.921048	0.903219	0.903219	0.903215	0.059589	0.244108

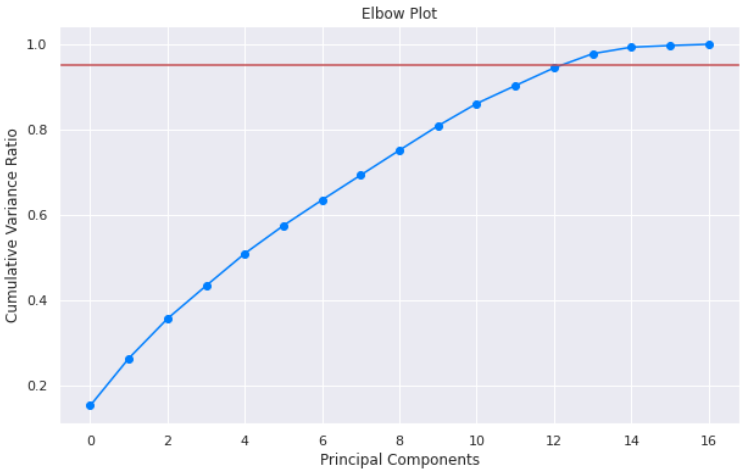
- The results indicate that there hasn't been much of a change in the resultant metrics, and we're essentially getting the same outcomes after standardising our features.

Feature Selection using Backward Elimination and XGBOOST Feature Importance:



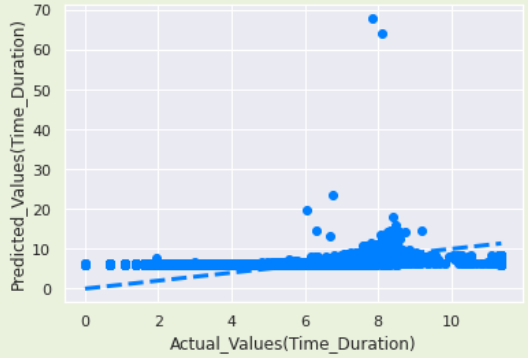
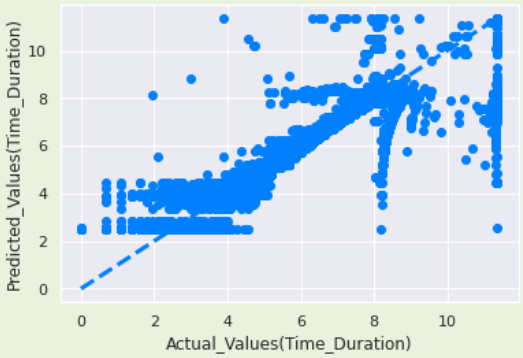
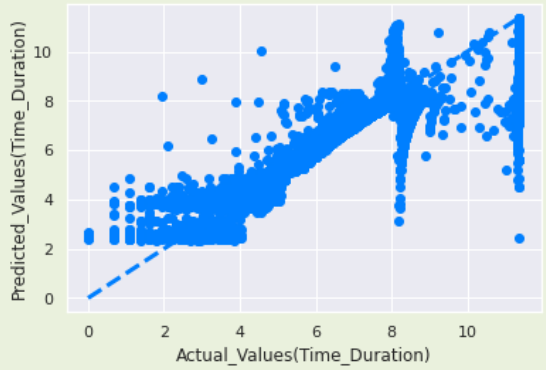
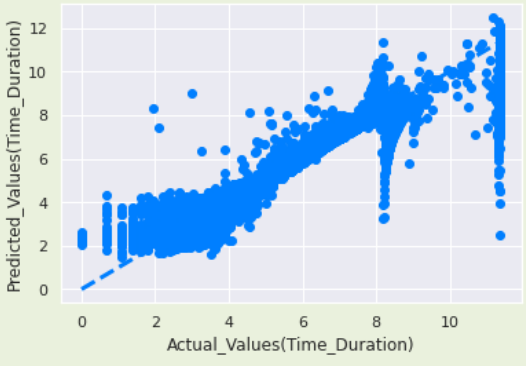
- Although Backward Feature Selection did not identify any features to remove, we can remove pickup_weakday from our data set so that our model can be trained.
- After selecting the features using the backward elimination method and feature importance, the impact on the results is minimal.

Using PCA Dimension reduction and transformation:



- 12 PCA components were found to account for more than 90% of the variance.

Final Results after PCA Transformation:

Linear Regression	Decision Tree
	
Random Forest	**XGBOOST**
	

	Training_Score	Testing_Score	R2_Score	ADJ_R2	MSE	RMSE
Linear Regression	0.283090	0.283905	0.283905	0.283886	0.444148	0.666444
Decision Tree Regression	0.965253	0.957100	0.957100	0.957098	0.026608	0.163121
Random Forest Regression	0.967994	0.962333	0.962333	0.962332	0.023363	0.152849
XGBOOST	0.979301	0.971078	0.971078	0.971077	0.017938	0.133934

- The linear regression model is completely underperforming.
- After reducing the Dimension of our features, Decision Tree and Random Forest are now performing very well here, and other evaluation matrices are also producing respectable results.
- But in terms of accuracy, R2 Score, and RMSE evaluation, Random Forest Regressor is marginally superior to Decision Tree Regressor.
- As we can see, XGBOOST is doing a fantastic job and offering us excellent training and testing accuracy. Additionally, the R2 score is much better than other models, and the RMSE value is also very low.

Conclusion and Summary:

- We can see that we applied several models using various methodologies, and that, with the exception of Decision Tree and Random Forest Regression, XGBOOST performed flawlessly in each case.
- But after applying PCA to our features and successfully reducing the feature dimension, we noticed that Decision Tree and Random Forest were performing just as well as they had before transformation, if not even better.
- Additionally, XGBOOST is also performing better after PCA transformation, and it also displays marginally better results in terms of RMSE | MSE, which are lower than those of other models, as well as R2 | Adj R2 scores, which are also higher than those of other models.
- As a result, we can use the XGBOOST Regressor to get good prediction rates and less error prone results. It can also be further optimized by using more tuned hyperparameters.

Future work recommendation:

- Since this data set only spans almost 6 months, I believe there should be more data for over a year as well as additional features, allowing us to train our models with more relevant data and improve their learning efficiency in order to achieve higher performance from machine learning models.

- Additionally, by adding more features, we can get more information about this data and further our understanding of this type of data.

References:

- Towardsdatascience
- Analyticsvidya
- Becominghumanai
- Siteminder.com
- Tmstudies.net