Predicting New York City Taxi Trip Duration using Multiple Machine Learning Regression Techniques

Jaron Baldazo
Schulich School of Engineering
University of Calgary
Calgary, Canada
jaron.baldazo@ucalgary.ca

Manjot Singh
Schulich School of Engineering
University of Calgary
Calgary, Canada
manjot.singh1@ucalgary.ca

***Abstract*— New York City taxis have been a major contributor to its transportation infrastructure. Majority of the population rely on public transportations such as taxis to get around the city. The ability to predict trip duration for customers using taxis can be a powerful tool since this allows them to plan accordingly to where they need to go. This is also helpful to taxi drivers as it can help them optimize routes during peak and rush hours. In this paper, different machine learning algorithms are built to predict trip duration times in New York City using real time data provided by the New York City Taxi and Limousine Commission. Models such as linear regression with stochastic gradient descent (SGD), ridge regression, decision tree regression, random forest regression, and gradient boosted trees were compared with one another. After the models were trained, it was found that gradient boosted trees outperforms the rest.**

Keywords—SGD, ridge regression, decision tree regression, random forest regression, gradient boosted trees

I. PREAMBLE

The project was completed by a group of two students, Jaron Baldazo and Manjot Singh. The group worked together on the project collectively as partners, where both members would work on the project at the same time. This ensured that the time put in the project and that the effort achieved by the members were the same.

TABLE I. CONTRIBUTION BETWEEN MEMBERS

Member	***Contribution***	***Contribution Percentage***
Jaron Baldazo	- EDA - Linear Models - writing and formatting paper	50%
Manjot Singh	- Preprocessing of data and RDD - Tree Models - writing and formatting paper	50%

Table I shows the contribution summary between members. The EDA of the project was handled by Jaron, whereas preprocessing of the data and transformations of the RDD's were handled by Manjot. The different ML models were split equally between members with Jaron creating the Linear Models such as Linear and Ridge Regression Models, and Manjot doing the different Tree models, being the Random Forest, Decision Tree, and Gradient Boosting Models. The written report was also completed by members equally with constant feedback being provided back and forth.

Declaration of all information above being accurate and true can be found below.

Jaron Baldazo Manjot Singh

The public repository for our notebook can be found at https://github.com/ManjotSin/SENG550-Project.

II. INTRODUCTION

A core symbol for New York City (NYC), taxis have been used as a means of public transportation by most of the city's population. With the growth of public transportation (taxis, Uber, Lyft) and population, challenges may arise within the transportation infrastructure. Longer trip times due to congested roads, traffic jams, or being unable to find parking spots are just some challenges that both taxi drivers and customers have to face every day. A solution to these challenges can stem from the predictions of trip duration for New York City taxis. Knowing the trip duration will allow customers to plan for their trips accordingly by selecting the best form of transportation. Taxi drivers can also benefit from trip duration predictions as it will allow them to optimize the routes that they must take.

Many papers and projects attempt to tackle this problem using different machine learning techniques using python and Scikit-learn (Sklearn). One paper [1] by Poongodi, M., Malviya, M., Kumar, C. et al uses the machine learning library Sklearn to predict NYC taxi trip times by modeling the dataset using XGBoost and MLP. These two machine learning techniques were then compared with one another. It was concluded in the

paper that XGBoost provided better results in terms of accuracy and efficiency. Other work done [2] by Kaggle user headsortails within this field includes performing exploratory data analysis and adding external data such as weather reports to further improve accuracy of trip duration. However, papers and work done on this field perform these predictions using Sklearn. In this paper, large scale data analytics is considered to provide scalability for larger datasets. Apache Spark will be used as a large-scale data processing framework while still performing similar machine learning techniques.

In this paper, exploratory data analysis is performed to detect any anomalies in the dataset. Machine learning models will then be built and trained upon using Apache Spark and Pyspark (interface for python in Apache Spark) to predict trip durations. Models such as linear regression with stochastic gradient descent (SGD), ridge regression, decision tree regression, random forest regression, and gradient boosted trees are trained and compared with one another. A baseline model is also created to be compared with the models mentioned. Data cleaning such as removing outliers and null values are also done on the dataset to further improve accuracy. Evaluation metrics that will be used is the root mean squared error (RMSE) which shows how close the predicted values are to the trained model.

III. Background

The machine learning models that will be used are linear regression with SGD, ridge regression, decision tree regression, and random forest regression, and gradient boosted trees. Linear regression with SGD is a simple model that uses multiple input variables to determine the coefficients in order to predict the output variable. SGD is used to optimize the linear regression model for further improving the best fit between the train and test sets. In ridge regression, the simple linear regression model is tuned to take multicollinearity into account. Multicollinearity refers to independent variables (features) that are correlated to one another. Decision tree regression is a model that uses a decision tree to break down the subsets of the dataset into different classes. Random forest regression builds upon the idea of decision tree regression. In the random forest model, multiple decision trees are trained. One input into a random forest model will query every decision tree and average all the predictions to find the final prediction. Finally, the gradient boosted trees model is similar to the random forest model in which multiple decision trees are made. However, in gradient boosted trees, the decision trees are created based on the previous trees created.

IV. Related Work

The paper by Poongodi, M. et al uses the machine learning library Sklearn to build models that predict trip duration. The dataset is cleaned where outliers are removed. XGBoost and MLP models are then trained on the dataset where trip duration is set as the target while other columns are used as the features. After running XGBoost, it was found that the average RMSE is 0.39 and 0.44 for the training and testing dataset respectively. This outperforms MLP with training and testing scores of 0.2740 and 0.41 respectively.

In the Kaggle report by headsortails similar preprocessing techniques were done such as removing outliers and null values. Another dataset is introduced that shows the weather data within the same time frame of the taxi trip dataset. Features such as rain or snowfall were added to the original dataset since it was correlated to the trip duration. Finally XGBoost was done on the model with different features added to the original dataset. Evaluation metrics includes RMSE and root mean squared logarithmic error (RMSLE).

V. Dataset

The dataset used [3] New York City Taxi Trip Duration was pulled from a previous Kaggle competition in 2016. The dataset was originally based on the 2016 NYC Yellow Cab Record Data [4] released by the NYC Taxi and Limousine Commission. Two datasets are provided in the kaggle that separates the test and train datasets. The train.csv file consists of over a million data points with information such as pickup_datetime, dropoff_datetime, trip_duration, etc. For the models being created, the column name of trip_duration was used as the target while the remaining columns are used as the features.

VI. Methodology

The proposed methodology used in this paper consists of five components. Setup, exploratory data analysis (EDA), preprocessing, model creation, and model evaluation. In each of these subsections, a general overview of the steps taken to perform each task is provided.

A. Setup

Using Google Colab, a Pyspark environment was set up using the source code provided by teaching assistant and the website [5]**.** This environment was chosen as it made collaborating while using Pyspark easier and for our situation of having only two members on the team and allowed for seamless cooperation. We also chose to use Apache Spark as our Big Data framework due to it being more familiar to us, and due to its improved efficiency in dealing with Machine Learning Algorithms, then say other Big Data frameworks such as Apache Hadoop. For our machine learning algorithms, we will be implementing different types of regression models, which will be created using the PySpark MLlib library, more specifically the RDD-based library [6]. Using this library, we will be implementing our regression and tree techniques in hopes of creating an accurate Taxi Trip Duration Prediction model.

B. Exploratory Data Analysis

Our experimental process began with exploratory data analysis, using Pyspark to explore our data and understand what

it contains. First the csv file (train.csv) is read and stored as a pyspark data frame. Using the show method, the first three data points are shown as example data. The different column headers such as id, vendor_id, pickup_datetime, dropoff_datetime, passenger_count, pickup_longitude, pickup_lattitude, dropoff_longitude, dropoff_latitude, store_and_fwd_flag, and trip_duration is shown. Two different ids are present in the dataset. The id corresponds to the unique identifier for each trip and the vendor_id corresponds to the trip provider. The column names pickup_datetime and dropoff_datetime corresponds to the date and time the taxi meter was started and ended. The passenger count is the number of passengers for each trip. The pickup_longitude, pickup_lattitude, dropoff_longitude, and dropoff_latitude all corresponds to the location where a trip has started and ended. The store_and_fwd_flag is a flag that indicates whether a trip record was held in the taxi memory. Finally, trip_duration is the amount in seconds for a trip to complete. Using the describe method, a summary of the statistical information is displayed for each column. This includes count, mean, standard deviation (stddev), minimum value (min) and maximum value (max) for each column. For this paper, the trip duration mean and standard deviation is used to detect and remove outliers. This is summarized in Table II. Finally, the data set is checked for null and empty value. In this case, the dataset did not contain any missing values.

TABLE II. TRIP DURATION MEAN AND STDDEV

Column Name	***Mean***	***Standard Deviation***
trip_duration	962	5853

C. Preprocessing

We began the preprocessing of our data by converting our train.csv file into an RDD that we will be able to transform and manipulate such that it can be used in our ML algorithms. Since the csv file included header information at the top of the file, it was necessary for us to transform our RDD so that this unwanted information is removed.

Next, when doing our EDA it was determined that some of our values were extremely high compared to the mean of our data, and thus would skew the results of our model by a substantial amount. In order to counter this, we removed outliers from our RDD by filtering out any record in which the trip duration of the record was greater than the mean plus the standard deviation. This would make sure that we don't have large outliers affecting our data when training and testing our ML models. After, it was decided that the id column of the data should be extracted, as each id of a record was unique, and not necessary for our model, thus we created a new RDD which filtered out the ids of each record.

After, we had to change the format of the datetime records as well as the store_and_fwd flags such that they are no longer considered strings, and can be used when creating our Labeled Point RDD. For both pickup_datetime and dropoff_datetime, the records were transformed and split into new features month, day and hour of pickup and dropoff respectively. Year was excluded and extracted as the year for all records in the dataset is from the same year 2016, thus it is unnecessary for our model. For the store and forward feature the records were in the format of either Y (yes) or N (no) which was converted to an acceptable format of 1 and 0 respectively.

Our final steps of preprocessing included converting our filtered RDD into a LabelPoint RDD such that it is in an acceptable format for our ML models found in the MLlib library. This was done using MLlib's LabelPoint function where we were able to create a tuple with the label, being trip_duration, and features, being the remaining features in the RDD. Once this RDD was created, it was then transformed to a normalized RDD using the function normalizeFeatures()**.** This function allowed us to normalize our features using the standard deviation and mean of the feature and prepare our final RDD before we begin the creation of our ML models. Functions such as normalizeFeatures() was derived from an example that the teaching assistant provided, Sarah Shah [7].

D. Model Evaluation

One evaluation metric is used for the models being created. The RMSE will be used to test the models and compare with one another. The equation for the RMSE in seconds is shown below.

$$RMSE = \sqrt{\sum_{i=1}^{N} \frac{(Predicted_i - Actual_i)^2}{N}}$$

Where N is the number of data points and i is the ith data point. Predicted represents the value predicted by the model whereas actual represents the real value.

E. Model Creation

The first step in building the models is to create a train test split. This was done by splitting the dataset into two RDDs, train and test. Train contains 80% of the datapoints while the test contains the remaining 20%. These RDDs were cached to prevent the models from re-reading every time it was used. The train and test consists of 838,012 and 208,903 datapoints respectively.

A baseline model is first created in which the other models are evaluated into. The mean and RMSE was calculated with values of 835.27 and 656.39 respectively.

The first model created was linear regression with SGD. The parameters used for this model was 500 iterations, step parameter of 1 for SGD, miniBatchFraction of 1, regression type of l2 and setting intercept to true. After training the model with these parameters, a predictions were then made using the test data set. Finally, this was evaluated by the RMSE of the train and test of 604.18 and 600.90 respectively.

In another variation of the linear regression model, a ridge regression model was created. Similar parameters were used with the first linear regression with SGD model. The parameters set were 500 iterations, step parameter of 1, miniBatchFraction of 1 and setting intercept to true. After fitting the model, predictions were made on the test data set. An RMSE of 604.25 and 601.04 was evaluated for the train and test respectively.

After creating the linear regression models, we moved on to decision trees. The parameters used were setting impurity to variance, max depth of 10, and max bins of 32. The impurity represents the criteria used to calculate information gain. The max depth represents the stopping point of creating new decision nodes. Bins are used to find the splits for each node. Fitting and running the model on the train and test yielded an RMSE of 431.69 and 435.23 respectively.

In an extension to decision trees, a random forest regressor model was created. Similar parameters were used with and additional parameter of number of trees. In this model, the number of trees was set to 13 which represents the number of decision trees to be created. An RMSE of 418.92 and 421.61 was obtained for train and test respectively.

Finally, a gradient boosted trees model were created to compare the different decision trees model. The parameters used for this model were setting max depth to 5 and learning rate to 0.5. The learning rate corresponds to the contribution for each tree created. The train and test models resulted in an RMSE of 307.63 and 312.35 respectively.

VII. Results

From our results it can be seen that Gradient Boosted Trees was our best model for predicting taxi trip duration. With the lowest RMSE for the testing data at 312.35 seconds, it does not have the best fit, however it was quite a ways better then our other models.

If we look at our baseline model to begin with, derived from the mean of trip durations in our dataset, which has a RMSE of 656.39 seconds. With this as our baseline, we started with the Linear Regression Model, which provided a test RMSE of 600.90 s, being very close to our baseline model. Similar results were achieved from the Ridge Regression Model, with a RMSE of 601.04 s. With both our linear regression models, providing such similar results to the baseline model, it shows that they will not be very helpful when predicting taxi trip durations.

From this we moved on to creating Tree Regression Models, starting with the Decision Tree Regressor Model. This model showed better results than our linear regression models having a RMSE of 435.23 seconds. It can be assumed that with better parameters, for example maxDepth and maxBins, this model will perform better, however the time it will take to train said model would be much longer. The next model we trained was a Random Forest Regressor Model, which showed slightly improved results from our previous Decision Tree Model, with a RMSE of 421.61 seconds. Having over 200 seconds improvement in the error shows us that using Tree regression models may prove more helpful then using the linear regression models before.

With this in mind, we finally ended up on the Gradient Boosted Trees Model, which as said before, had the greatest results, with an RMSE of 312.35 seconds and thus would be the model we go forward with if we were to predict Taxi trip durations, having an RMSE which is more than a 100 seconds lower than the next best model. Table III shows a summary of results obtained for each of the models and its corresponding train and test RMSE.

TABLE III. Sumamry of Results

Model	***Train RMSE***	***Test RMSE***
Baseline	656.39	656.39
Linear Regression with SGD	604.18	600.90
Ridge Regression	604.25	601.04
Decision Trees	431.69	435.23
Random Forest	418.92	421.61
Gradient Boosted Trees	307.63	312.25

VIII. Conclusion

In the dataset provided, multiple regression techniques were performed to predict taxi trip duration. The techniques used can be classified into two types, linear regression models and decision tree models. Linear regression models consist of linear regression with SGD and ridge regression. On the other hand, decision tree models consist of decision trees, random forest, and gradient boosted trees. It was found that amongst the linear regression models, both linear regression with SGD and ridge regression performed similarly based on the RMSE of the train and test data sets. Moving onto the decision tree models, gradient boosted trees model outperforms both decision tree and random forest. When comparing both linear regression and

decision tree models, it was found that the models used in decision trees performed better than that of linear regression.

Future work for this can include more parameter tuning for the models created. During the EDA, more preprocessing and feature engineering where feature prioritization and selection is performed can further boost the scores. In terms of the evaluation metrics, other metrics such as accuracy, recall, and f score can be calculated to show the differences in each model. Finally, adding more data sets such as weather conditions or traffic information can supplement the initial dataset by finding more variables that are correlated to trip duration time.

REFERENCES

[1] M. Poongodi, M. Malviya, C. Kumar, M. Hamdi, V. Vijayakumar, J. Nebhen, and H. Alyamani, "New York City taxi trip duration prediction using MLP and XGBoost - International Journal of System Assurance Engineering and Management," SpringerLink, 01-Jul-2021. [Online]. Available: https://link.springer.com/article/10.1007/s13198-021-01130-x.

[2] Headsortails, "NYC Taxi Eda - update: The Fast & The Curious," Kaggle, 22-Feb-2020. [Online]. Available: https://www.kaggle.com/code/headsortails/nyc-taxi-eda-update-the-fast-the-curious#a-simple-model-and-prediction.

[3] "New York City taxi trip duration," Kaggle. [Online]. Available: https://www.kaggle.com/competitions/nyc-taxi-trip-duration/data.

[4] "TLC Trip Record Data," TLC Trip Record Data - TLC. [Online]. Available: https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page.

[5] A. Bhandari, "PySpark Google Colab: Working with pyspark in Colab," *Analytics Vidhya*, 23-Nov-2020. [Online]. Available: https://www.analyticsvidhya.com/blog/2020/11/a-must-read-guide-on-how-to-work-with-pyspark-on-google-colab-for-data-scientists/. [Accessed: 11-Dec-2022].

[6] "MLlib (RDD-based)¶," *MLlib (RDD-based) - PySpark 3.3.1 documentation*. [Online]. Available: https://spark.apache.org/docs/latest/api/python/reference/pyspark.mllib.html.

[7] S. Shah, "ML_Linear_Regression," Central Authentication Service. [Online]. Available: https://d2l.ucalgary.ca/d2l/le/content/468644/viewContent/5609670/View.