Prediction of Taxi Fares in New York City

Pradyumna YM
PES1201700986
CSE dept., PES University

Anush V Kini
PES1201701646
CSE dept., PES University

Punit Pranesh Koujalgi
PES1201701502
CSE dept., PES University

***Abstract*—Every month, about 1.5 million taxi trips are completed in the city of New York. With the rise of taxi aggregation services like Uber and Lyft who have established a monopoly in this business in the recent times, taxi fares have become unpredictable due to various schemes like surge pricing. We predict the taxi fares given the time and pickup and drop-off coordinates. The yellow cabs of New York City have partnered with Google to solve this task of predicting taxi fares, which has published a large dataset on kaggle through a competition. We approach this problem by using techniques to improve the data and come up with a model that can predict taxi fares accurately and efficiently using this published data in order to avoid the hassles of data collection. We feel that this will help the yellow cabs regulate their prices for a given trip according to inputs from our model. This also allows us to optimise the travel costs for passengers by predicting the time of departure for the least cost from point A to point B. In this report, we describe our approaches to engineering more features for this dataset. We tried a number of models of increasing complexity, and were able to achieve an RMS error rate of about 2.9 on the test set, which is equivalent to 153^{rd} rank on the kaggle competition.**

***Index Terms*—Regression, Fare Optimisation, Machine Learning, Data Analytics**

I. Introduction

The taxi cab has established itself as one of the important modes of transportation in urban areas. In New York City, the yellow taxi cabs have been dominating the public transportation industry for years. The past few decades have seen a sharp rise in the usage of taxi cabs. According to the New York City Taxi and Limousine Commission in 2015, there were 143,674 taxi cabs operating in the city of New York. Recent years have seen the rise of app based taxi services like Uber and Lyft who have been providing stiff competition to the yellow cab services. This sudden growth in competition has left traditional cab services at a severe disadvantage. Further, most app based taxi services use the concept of 'surge pricing' to inflate the service rates according to supply and demand. This has led to unpredictable taxi fares putting the consumer at a severe disadvantage. Predicting the fare of a taxi ride can help passengers determine a favourable time to begin their journey while also assisting drivers to choose more profitable rides. Taking this idea a step further, a system that would predict the optimal time for a ride with least fare be a useful tool for passengers in suggesting the best time to take as cab ride.

'New York City Taxi Fare Prediction' is a competition on Kaggle, hosted in partnership with Google and Coursera. The data consists of about 55 million rows of pickup and dropoff details, with the target column being the fare amount for the ride. From the raw data, we have drawn inferences through visualization. We then have feature engineered various columns to the dataset. From here, we have explored different models and evaluated them on their predictions. We then have built a system which recommends the best time for a passenger to embark on their journey to reduce the taxi cab fare.

In this paper, we expand on our attempt to build a machine learning model which predicts the fare amount with the least error. In section 2, we briefly look at some of the related works. In section 3, we elaborate on our problem statement. In section 4, we explain the preprocessing methods that we have used to clean our dataset. We also take a look at the feature engineered columns we have added to our dataset. In section 5, we describe our methodology and explore the details of various models we have trained. Further, we give an overview of the final pipeline. In section 6, we expand on other experiments we have undertaken. In section 7, we summarise on the important aspects of our report.

II. Literature Survey

The taxi fare problem has mostly been approached as a regression problem. Various techniques have been modeled to predict the fare with minimal error.

Rishabh Upadhyay et al. [1] have classified the taxi fare amount into 5 different categories, (low, normal, med, high, extra high). Further, they have feature engineered columns holiday, time, weekend, distance from airport, city centre and popular tourist places. They have approached this problem with 2 different methods. Firstly, a 5 layer DNN, with ReLU units, and softmax output layer. Secondly, They have used a stacked classifier approach, where there are a set of base classifiers trained on the training data. Next, there is a classifier trained on these outputs of the base classifiers and the test data, which will be used to classify the taxi fares. Though the validation results of the neural network are promising, they fail to generalise to the test set. Further, no comparative study has been performed on similar approaches to the stacked classifier like AdaBoost classifier. Further, the classification approach used is of little use in the real world.

Xinwu Qian et al. [2] have used an approximate dynamic programming approach to model the underlying semi-Markov process. In their paper, they focus particularly on the time

of pickup as a feature, which is actually an important factor in deciding the taxi fares. Also, the method discussed in this paper is computationally economical, considering the fact that no large ML models are trained.

Hai Yang et al. [3], focus on the linear distance-fare model used in Hong Kong and it's downfalls. They have proposed to model the taxi fares in Hong Kong as a nonlinear profitability-based taxi service model, which factors in the initial flag-fall charge of taxis and distance of the taxi ride.

K. Tziridis et al. [4] have investigated into the important features of influence for the prediction of Airfares,where they concluded that the day of the week and holidays are very important.They have approached the problem as both a regression problem and as a classification problem. With a limited data of about 1800 flights, they were able to achieve good results using bagging regression trees.

Frank Ivis et al. [5] have compared various techniques that can be employed to compute geographical 2-dimensional and spherical distances. Further, they have discussed methods of handling latitude and longitude data., which is pivotal in solving our regression problem.

Nitin R Chopde et al. [6], have used the Haversine formula as a reliable approximation to the distance between two GPS coordinates in their work to find the shortest path between two Coordinates in Google Maps, using Djikstra's algorithm.

Christophoros Antoniades et. al [7] have used various models such as linear regression with forward selection, and higher order terms of attributes, Lasso Regression, and Random Forests. They propose to transform the GPS coordinates in such a way that the streets of the city of New York are parallel to the x and y axes, as this might lead to a better split of the data in case of Random Forest Regressors. They have obtained impressive results through these models.

Tianqi Chen et al. [8], in their paper, provide insights on cache access patterns, data compression and sharding to build a scalable tree boosting system. By combining these insights, they were able to build a highly scalable framework in order to perform regression using tree boosting algorithms.

III. Problem Statement - Our Approach

Most of the above approaches have focused mainly on predicting the optimal fares prices for the driver-passenger dynamic with the primary aim of aiding the taxi drivers. We propose to first perform a comparative study of the state of the art machine learning models as mentioned in the above literature. Further, we will then optimize these models to suit our problem statement of predicting the fare given information only available prior to the trip. We then plan to supplement these models by adding a system which determines the optimal time to embark on a trip. This system will aid passengers to minimise their fare amount. This solves the problems that are faced by the consumers due to the uncertain pricing schemes used by Taxi aggregators, while it can also be used by yellow cabs to fairly approximate their fares.

IV. Data

We chose a dataset on NYC taxi fare prices had 7 columns and 55 million rows, due to which we loaded the data in chunks of 5 million rows, and cleaned, pre-processed the data, performed feature engineering and then concatenated these chunks after changing the datatypes of some columns to reduce the memory required. The descriptions of the attributes can be found here:

1) key: A unique string identifying each row.
2) pickup_datetime: Timestamp of when the taxi ride began.
3) pickup_longitude: The longitude coordinate of where the taxi ride began.
4) pickup_latitude: The latitude coordinate of where the taxi ride began.
5) dropoff_longitude: The longitude coordinate of where the taxi ride ended.
6) dropoff_latitude: The latitude coordinate of where the taxi ride ended
7) passenger_count: The number of passengers in the taxi ride.

For data cleaning, imputation was not performed as there was a surplus of data. Hence, we dropped the rows that had:

- missing values
- pickup or drop-off coordinates outside the bounding box of the city of New York.
- passenger count greater than 8.
- fare amount greater than 250
- negative fare amount

After carefully examining the dataset, we decided to add more features to the dataset as the ones already present were too crude for models to make sense of. So, we decided to add the following features:

1) distance: a Haversine distance between the pickup and drop-off coordinates.
2) weekday: an integer representing the day of the week.
3) time: time in minutes since 12:00 AM that day.
4) holiday: boolean value indicating whether it is a public holiday or not.
5) year: the year during which the taxi ride was taken.
6) distance from Manhattan, John F Kennedy Airport, Laguardia Airport, Statue of Liberty, Central Park, Times Square, Brooklyn Bridge, Rockerfeller: These destinations are prominent tourist spots and have a high amount

of taxi trips. Hence, can have an effect on the effective taxi fare.

7) day: This corresponds to the particular day of the month.
8) month: the month of the year of the transaction.

V. Methodology

A. Model selection

In this section, we discuss the series of models we went through and their shortcomings that lead to us trying the next model.

The first model we tried was OLS linear regression. Closed form expression of the parameters was used to obtain the model, as performing gradient descent on a dataset this size was time consuming. This model was used as the baseline for future models to improve on.

Next, we tried Ridge Regression and lasso regression. The shrinkage factor for both of these was determined using a grid search method. As we can see in Figure 2 and Figure 1, we get best results for $\alpha = 10^{-5}$ and $\alpha = 10^{-3}$ for lasso and ridge regression respectively. As we can see from the results section, these models performed only marginally better than the linear regression model. From this, we could conclude that the features are not redundant/there was no need for regularization of the model.

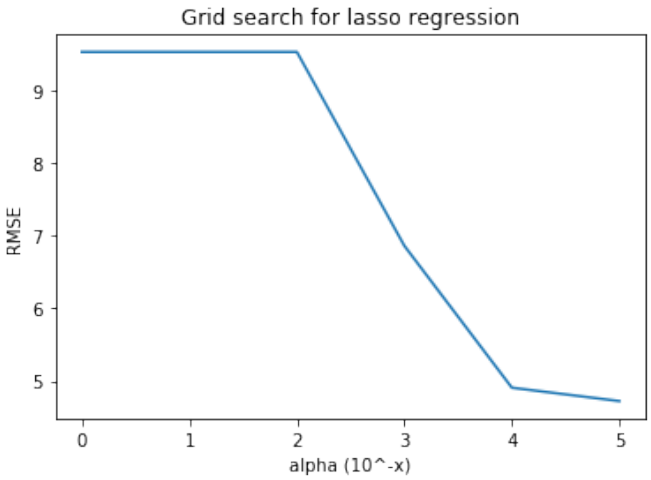

Fig. 2. Grid search for Lasso model

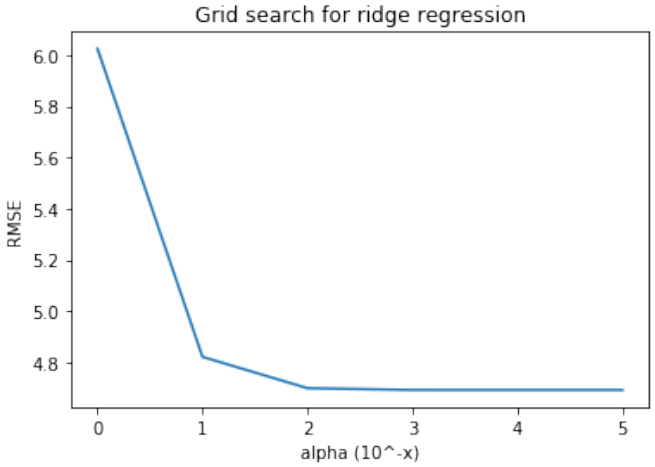

Fig. 1. Grid search for Ridge model

We then trained a random forest regressor on the dataset. Each tree in the ensemble is built from a sample drawn with replacement (bootstrapping) from the train set. This model chooses a random subset of the attributes in order to train multiple decision trees. The model then averages the predictions of all the decision trees in order to output a single prediction.

K Nearest Neighbours The KNN regressor algorithm assigns a target value to the query point based on the mean of target values of its K nearest neighbours. A plot of k vs error rate was performed. As we can see from Figure 3, this plot gave us the least RMS error rate at k=1. This can be explained by the sheer size of the dataset. The dataset has 5.5×10^7 data points. Just merely searching for a similar record gave us an RMS error rate of 3.5. Bagging was used to train 55 different models seperately in order to reduce the time taken to perform inference on the whole dataset at once.

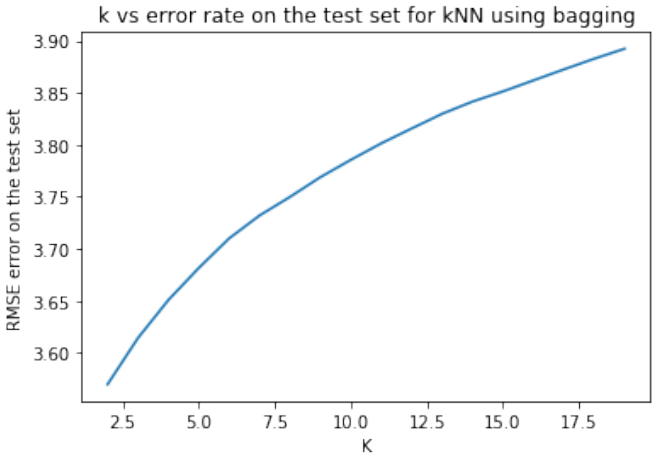

Fig. 3. K vs RMSE for kNN model

ANN: keras framework was used in order to train an ANN with 5 hidden layers, whith ReLU and tanh activation functions. The The data was normalized before passing it to the ANN for training as neural networks perform better when the data is not skewed. The loss curve for ANN model can be seen in Figure 4. We can note that the model has converged.

Xgboost and LightGBM: are two of the most popular implementations of gradient boosting decision trees (GBDT) in python. Gradient boosting is also another form of ensemble

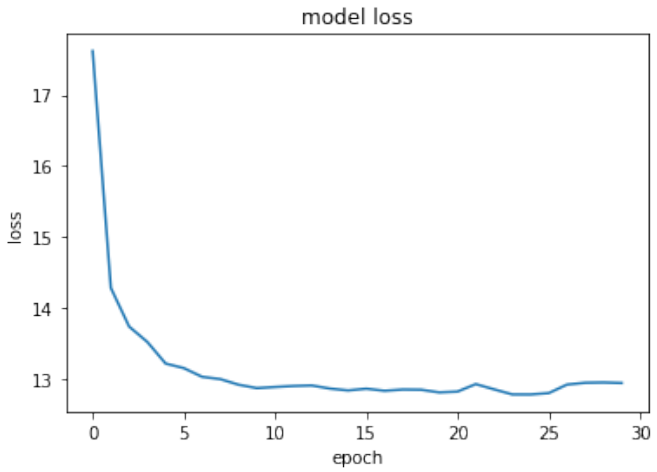

Fig. 4. Loss curve for ANN model

learning that tries to create a good predictor by combining multiple weak learners(decision trees in this case), where each learner tries to fit the residual errors of the previous learners. Guolin Ke et al., [9] in their publication, have presented a new, more lightweight and highly efficient framework called LightGBM, an implementation of Gradient Boosting Decision trees for regression and classification problems.

We were able to obtain an RMS error rate of just 2.93 using the features we engineered and LightGBM regressor, and an RMSE of 3.11 with XGBoost. This is because ensemble learning models are computationally heavy, and we were able to train the LGBM and XGBoost models on 15 and 2 million data points respectively. Provided enough computational resources and data, these models are capable of producing even better results. These tree models provide us with a feature importance graph. This graph can be seen in Figure 5.

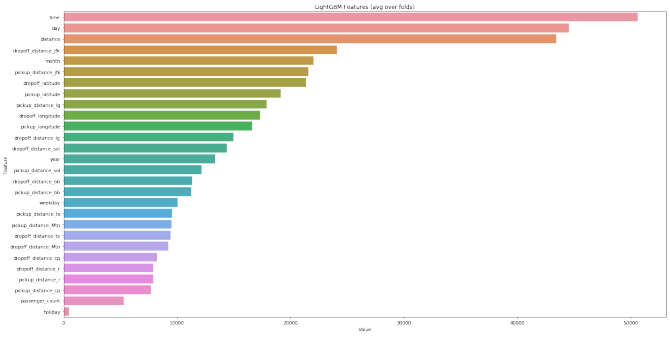

Fig. 5. Feature Importance graph of LGBM model

B. Solution Pipeline

We propose a pipeline in order to predict taxi fares and to suggest optimal departure time in order to minimise the travel cost, as described by Figure 6.

First, The training data was cleaned and engineered for more features. Data points having missing values were dropped, as we have a surplus of data. Next, the features described above were added to the dataset.

We then trained models on the training set, and Based on their performance on a test set, the best model was chosen. Once we have a model that can be used to predict taxi fares.

We take the user's details, such as source, destination, approximate time of departure and date. To get the coordinates of the pickup and drop-off points from the addresses, we use an online geolocation API, Opencage. Using requests library, we send an HTTP get request with the given address. The response obtained from the API is then parsed using the json module in order to obtain the coordinates.

Once we have the 7 required features, we use the same feature engineering pipeline in order to obtain the required features for the given trip. We then use the trained model in order to predict the fare at different time intervals. Based on the predictions, the best departure time is chosen.

C. Evaluation Metrics

The evaluation metric used is the root mean squared(RMS) error. The RMS error takes into account the difference between the predicted values of the model and the corresponding ground truth values. One of the important advantages of using the RMS error is that it the error is in the units being measured and hence, it makes the error easy to analyse. The RMS error is given by:

$$RMS\ Error = \sqrt{\frac{1}{n}\Sigma_{i=1}^{n}(\hat{y}_i - y_i)^2} \quad (1)$$

The models were evaluated based on a part of the data reserved for testing. As RMS error rate gives a high weight to examples with a large error, and a large error in fare prediction is undesirable, while small errors in most of the predictions are acceptable. We also submitted our predictions on a test set on the kaggle competition from which the data was taken, and were able to obtain impressive RMS error rates there as well.

VI. Experiments and Results

One of the major problems encountered while using large datasets is the lack of computational resources. The raw dataset consists of about 55 million rows with 7 columns of data. Loading this raw data onto the dataset used up a lot of memory and any further computing would cause the system to crash.

We overcame this problem by reading the data in chunks of 1 million data points. We then performed feature engineering on these individual chunks. We made sure that we downcasted the datatypes to the smallest possible type. python's garbage collector was used often in order to prevent memory errors

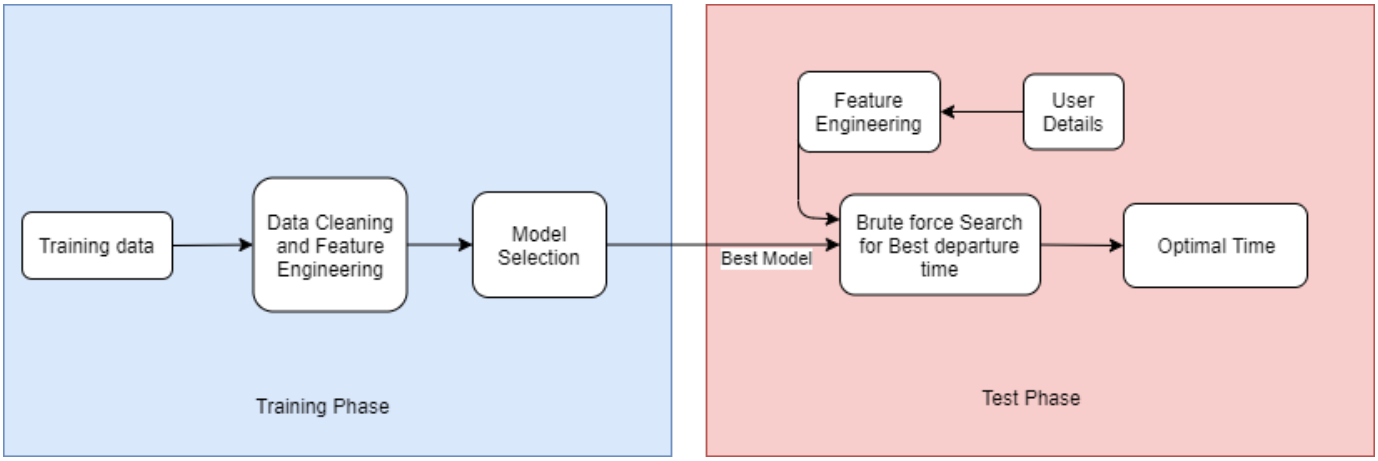

Fig. 6. The Proposed Pipeline

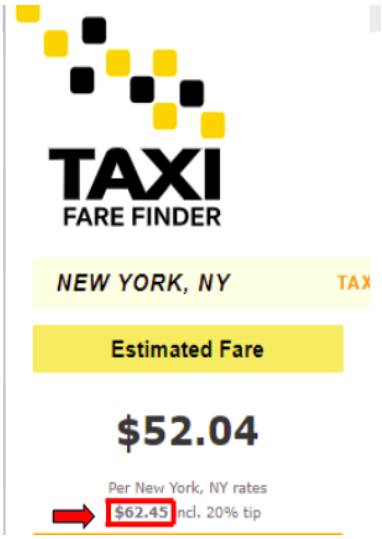

Fig. 7. Results from the taxi finder predictor for a taxi ride from Laguardia Airport to JFK Airport

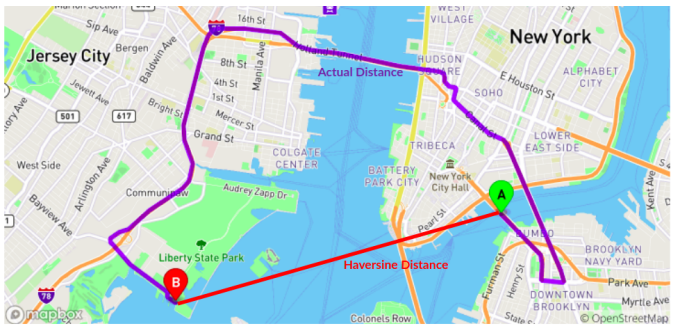

Fig. 8. Actual Distance vs Haversine Distance

due to the large size of the data. For instance, we had a large number of attributes that were just coordinates.These are float64 by default. Theses attributes were downcasted to float32, which saved us a lot of memory, while not hampering the accuracy of the models.

After performing many iterations of feature engineering and testing on hold out data, we trained the models on the training set and submitted our predictions to the kaggle competition and obtained the results as seen in Table 1.

Taxi Fare Finder is a website which gives accurate taxi fares given source and destination at the current time. We verified the predictions made by our model with a company's website online. AS seen in table II, the predictions of our models were pretty close to the actual fare in most cases. So, we can rest assured that the time suggested by the system can be relied on. However, our model does perform poorly in some cases. Our model predicts the fare amount for a ride from Brooklyn bridge to statue of liberty to be 21.84 USD. However, the actual cost is about 47 USD. This can be attributed to the fact that our model calculates the distance using a basic haversine formula. This formula dosen't work well in presence of water bodies as can be seen in Figure 8. We feel that having some more data about the roads of the city of new york in the form of a graph data will help the model in improving its performances in such cases greatly.

VII. Conclusion and Future Works

We explore the current state of the art models in fare prediction in this work. After careful analysis of the dataset through visualization, we have feature engineered columns and modified the dataset. We were successful in predicting taxi fares fairly accurately and were able to create a system in order to suggest users an optimal departure time.

TABLE I
A TABULATION OF THE RESULTS OBTAINED USING VARIOUS MODELS

Model	RMS error
Linear Regression (OLS)	5.18
Ridge Regression	5.18
Lasso Regression	5.05
Random Forest Regressor	4.43
1 Nearest Neighbour	3.54
Artificial Neural Network	3.39
Gradient Boost Decision Tree (XGBoost)	3.11
Gradient Boost Decision Tree (LGBM)	2.93

TABLE II
A COMPARISION BETWEEN TAXI FARE FINDER AND OUR MODEL PREDICTION

Source	Destination	TFF(Including tip)	Prediction
Laguardia Airport (OLS)	JFK Airport	62.45	65.54
Laguardia Airport	Statue of Liberty	72	69
Brooklyn Bridge	Statue of Liberty	47	21.884
Brooklyn Bridge	Laguardia Airport	45.3	44.35

As discussed above, the models can be greatly enhanced by making use of the data on the roads/streets in the city of New York by treating the entire city as a graph with nodes and edges, with the weights of the edges having information such as traffic intensity, distance, etc. encoded within them.

VIII. CONTRIBUTIONS

This project was done by a cohesive duo of Pradyumna YM and Anush V Kini, as our teammate had to withdraw from the course due to an unavoidable reason. Both of us have been involved equally in the process of data cleaning, feature engineering and model selection.

IX. ACKNOWLEDGEMENT

We would like to thank Prof. Gowri Srinivasa for her timely inputs and advice which went a long way in guiding us on this project.

REFERENCES

[1] R. Upadhyay and S. Lui, "Taxi fare rate classification using deep networks," 09 2017.
[2] X. Qian and S. V. Ukkusuri, "Time-of-day pricing in taxi markets," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 6, pp. 1610–1622, June 2017.
[3] H. Yang, C. Fung, K. Wong, and S. Wang, "Nonlinear pricing of taxi services," *Transportation Research Part A: Policy and Practice*, vol. 44, pp. 337–348, 06 2010.
[4] K. Tziridis, T. Kalampokas, G. A. Papakostas, and K. I. Diamantaras, "Airfare prices prediction using machine learning techniques," in *2017 25th European Signal Processing Conference (EUSIPCO)*, Aug 2017, pp. 1036–1039.
[5] F. J. Ivis, "Calculating geographic distance: Concepts and methods," 2006.
[6] M. Nichat, "Landmark based shortest path detection by using a* algorithm and haversine formula," 04 2013.
[7] C. Antoniades, D. Fadavi, and A. F. Amon, "Fare and duration prediction : A study of new york city taxi rides."
[8] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," *CoRR*, vol. abs/1603.02754, 2016. [Online]. Available: http://arxiv.org/abs/1603.02754
[9] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "Lightgbm: A highly efficient gradient boosting decision tree," in *NIPS*, 2017.