Predicting Uber Ride Fares

Hilary Le
hale@scu.edu

Austin Nguyen
anguyen10@scu.edu

Abstract

This project explores the prediction of Uber ride fares using machine learning models trained and tested on the Uber Fares Datasets (Kaggle and H). The dataset provides a comprehensive view of related parameters, enabling a detailed analysis of the factors that influence the final price. This project evaluates the performances of different regression methods: LightGBM Regression, Random Forest Regression, Extra Trees Regression, Polynomial Regression, Ridge Regression, Linear Regression, and Decision Tree Regression.

Evaluation of each model's performance used metrics such as Mean Squared Error (MSE), Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and R-Squared (R^2) scores. The project goes over the pros and cons of each model through the scores of the training and testing of each. From this analysis, conclusions are drawn that provide practical implications in choosing appropriate predictive models for fare predictions on similar datasets.

The report concludes with the identification of the best-performing models, considering the evaluation metrics, and potential next steps for our findings.

1 Introduction

The rapid growth of ride-sharing services like Uber has revolutionized the means of urban transportation, offering more convenient alternatives to taxis. One of the most critical factors of customer satisfaction and efficiency of Uber's service is fare prediction. Accurate fare estimation both enhances the user experience by providing upfront prices and helps drivers optimize their routes and procedures. Despite Uber's existing advanced algorithms, understanding and replicating fare predictions using open datasets is a valuable exercise in machine learning and for similar predictive modeling.

This project utilizes the Uber Fares Dataset from Kaggle to train and test different regression models to predict these ride fares.

The dataset includes several features that influence pricing, such as:

- **fare_amount**: the cost of the trip in USD,
- **passenger_count**: the number of passengers entered by the driver,
- **pickup_datetime**: the date and time when the meter was engaged,
- **pickup_longitude** and **pickup_latitude**: the geographic coordinates of the pickup location,
- **dropoff_longitude** and **dropoff_latitude**: the geographic coordinates of the dropoff location.

The project aims to identify the most suitable regression model to predict fares with high accuracy and reliability. A systematic approach will be adopted, training and testing with a set of regression models: LightGBM Regression, Random Forest Regression, Extra Trees Regression, Polynomial Regression, Ridge Regression, Linear Regression, and Decision Tree Regression. Each model will be evaluated using key performance metrics, including Mean Squared Error (MSE), Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and R-squared (R^2) scores. These will determine which models perform better under the given conditions to draw insights into the factors behind calculating ride fares.

2 Background & Related Work

According to Uber, the app uses a dynamic pricing algorithm that adjusts based on many different variables, "such as time and distance of your route, traffic and the current rider-to-driver demand" (Phillips, 2017).

In a National Bureau of Economic Research report, the authors state that Uber's rider-to-driver

demand pricing algorithm is called "surge pricing," which monitors rider demand and Uber's supply of available drivers. It institutes a price multiplier on the total when the demand heavily outweighs the supply. This is meant to increase the supply at times of higher demand, however, it will be out of the scope of our project. (Cohen et al., 2016) The reason for this variable to be missing from our project is the lack of transparency Uber has shared about the exact multiplier of this factor on the base price. It was observed that lower surges of 1.2x to 1.5x the base price are fairly common, however while rare, it is seen that surges can go as high as 3-4x base price (Atienza et al., 2023).

From Chao's work, we understand that many different factors impact Uber's pricing (Chao, 2019). Uber collects lots of data from cities. With our dataset, we do not have as much data, meaning that our project is quite limited by the dataset we have.

While Uber has much more access to a wider range of variables, such as hotspots, number of drivers, and more, our Kaggle dataset has fewer features. For example, Chao highlights how even the weather is an important factor in Uber's dynamic pricing model (Chao, 2019).

A very similar case study to ours was conduced on a pool of Uber rides in Madrid, Spain, using regression models to try to most accurately predict ride fares. It uses similar variables such as the fare amount, passenger count, pickup and dropoff latitude and longitudes as well as the time of pickup, and it concluded that the Random Forest Regression algorithm performed the best (Silveira-Santos et al., 2023). It is important to know how Uber and forms of app-based transportation services act around the world. In China, the amount of drivers registered in these apps are commonly regulated, leading to datasets coming from cities like Beijing, Shanghai, and Hangzhou not having as many features to analyze (Zhong et al., 2021). New York City, the location our dataset uses, has adapted to Uber seamlessly as it has become another option to the public there in addition to taxis and the subway (Willis and Tranos, 2020).

Although our dataset limits our testing, we aim to predict ride fares based on what is available in our dataset using statistical and machine learning techniques.

3 Approach

3.1 Data Clean-Up & Validation

To begin, we imported our data and performed clean-up and validation steps:

- **Remove Extraneous Columns**
  - `key`: This was a date and time stamp; as this information was already provided, we removed the column.
  - `unnamed`: This was the index column, which was not important for our purposes and was likely provided as a key of some sort.
- **Validate Latitude and Longitude Coordinates**
  - Latitude: Must be between -90 and 90 (inclusive).
  - Longitude: Must be between -180 and 180 (inclusive).
  - Rows with invalid coordinates were removed.
- **Validate Passenger Count**
  - Passenger count must be between 1 and 6 people (inclusive).
  - Rows with invalid passenger counts were removed.
- **Calculate Trip Distance (Displacement)**
  - We hypothesized that trip distance would influence fare prediction. In traditional cabs and taxis, trip distance is a significant factor in pricing, typically measured using a meter.
  - Using the Haversine formula, we calculated the displacement between the start and end coordinates.
- **Split Time Data**
  - Time was represented in a date-timestamp format. We extracted various features such as year, month, day, hour, minute, and second.

After clean-up and validation, the following features were retained:

- `fare_amount`
- `pickup_longitude`

- pickup_latitude
- dropoff_longitude
- dropoff_latitude
- passenger_count
- pickup_year
- pickup_month
- pickup_day
- pickup_hour
- pickup_minute
- pickup_second
- trip_distance

3.1.1 Plotting Data

3.2 Finding & Removing Outliers

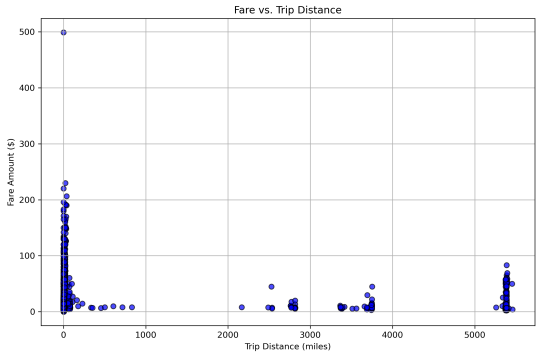

Figure 1: Fare vs. Trip Distance

When plotting Fare v. Trip Distance, it is clear that the distribution is irregular. Considering that the shortest distance between the east and west coasts of the US is slightly more than 2000 miles, we infer that the larger distances may be invalid data points. We also see an outlier around (0, 500), a short distance with an extremely expensive fare amount.

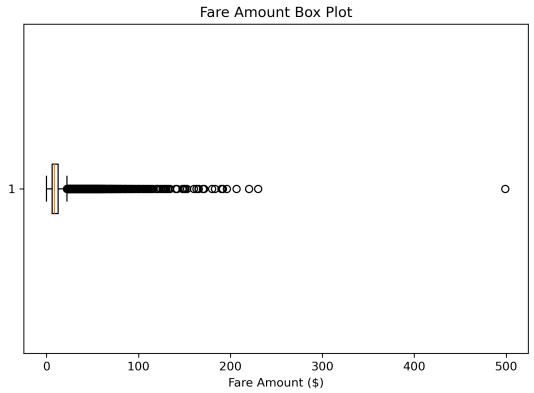

Figure 2: Fare Amount Box Plot

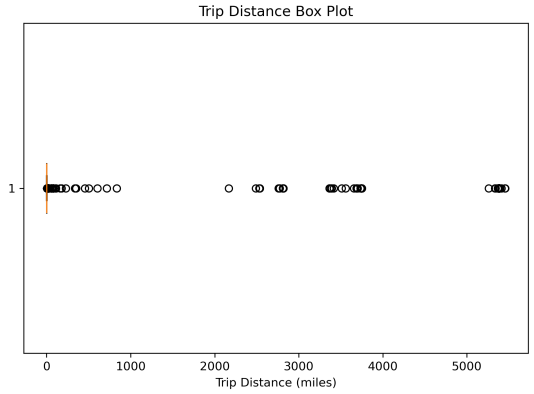

Figure 3: Trip Distance Box Plot

When viewing the box plots for both the trip distance and the fare amount, we observed that the distribution is irregular. Since there is some clustering, we wanted to see how changing the dataset would affect the results. With these two different datasets, we performed training and testing separately to see if the results would differ greatly.

We split the data in two different ways: removing obvious outliers and removing outliers based on the Inter-quartile Range (IQR).

3.2.1 Reduced Data

We removed obvious outliers that represented invalid data. Outliers were removed if they were extreme outliers. This data is more raw and retains more of the original information.

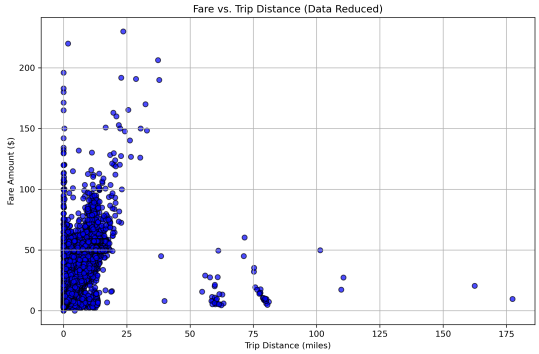

Figure 4: Fare vs. Trip Distance (Data Reduced)

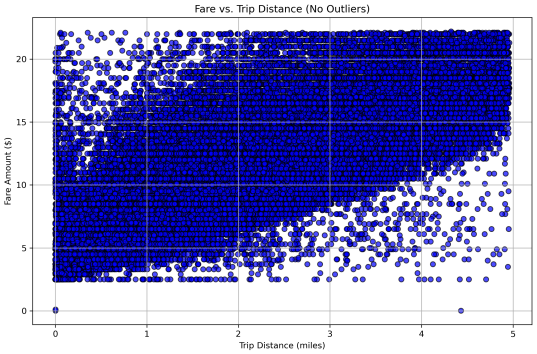

Figure 7: Fare vs. Trip Distance (No Outliers)

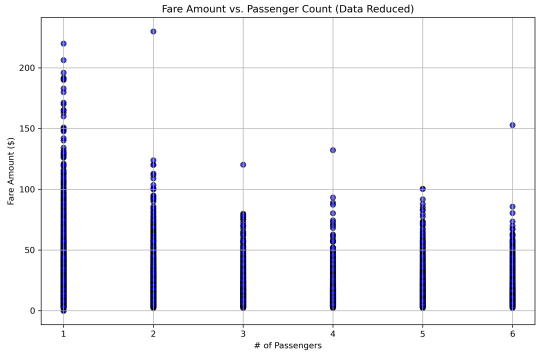

Figure 5: Fare vs. Passenger Count (Data Reduced)

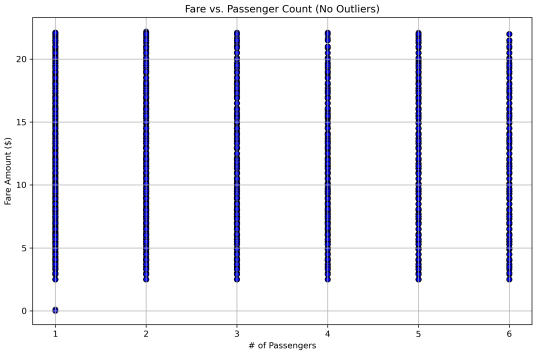

Figure 8: Fare vs. Passenger Count (No Outliers)

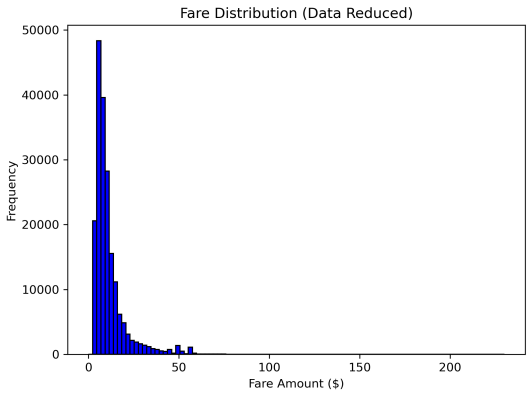

Figure 6: Fare Distribution (Data Reduced)

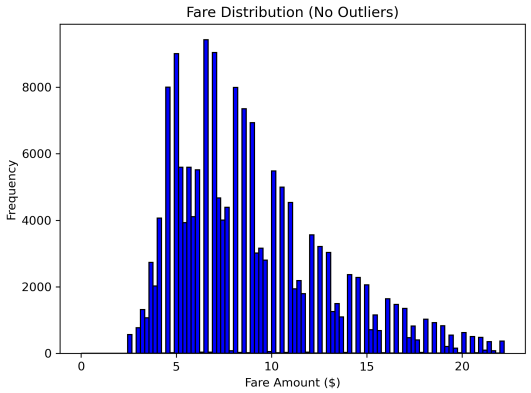

Figure 9: Fare Distribution (No Outliers)

With the reduced data, the data points are still very spread apart.

3.2.2 No Outliers

We removed outliers based on the IQR, using upper and lower bounds. The IQR is calculated as IQR $= Q_3 - Q_1$, where Q_1 and Q_3 are the first and third quartiles, respectively. The upper bound is given by $Q_3 + 1.5 \times$ IQR, and the lower bound is given by $Q_1 - 1.5 \times$ IQR.

By removing outliers, the distributions are much less random, as seen in the figures without outliers.

4 Experiment

4.1 Our Dataset

Using the Uber Fares Dataset from Kaggle, we cleaned up the data into two different datasets.

4.1.1 Reduced Data

We removed obvious outliers that represented invalid data. Outliers were removed if they were

extreme outliers. This data is more raw.

4.1.2 No Outliers

We removed outliers based on the Interquartile Range (IQR), using upper and lower bounds. Outliers were removed based on the IQR, utilizing the upper and lower bounds.

4.2 How We Ran Experiments

Using our chosen models, we did slightly different training for each. For each type, we created two models: one for the *reduced dataset* and one for the *no outliers dataset*.

4.2.1 Linear Regression

We used standard training without any regularization or cross-validation for both models.

4.2.2 Ridge Regression with Cross Validation

We performed Ridge Regression with k-fold cross-validation. With cross-validation, we can find the best regularization by splitting the data into multiple training and validation sets. We used $k = 5$.

4.2.3 Polynomial Regression

We transformed the input features into higher degrees. We tuned the degree of the polynomial to find the best model performance.

4.2.4 Decision Tree Regression

We trained two separate Decision Tree Regression models. Both models were trained independently with the same random state for reproducibility.

4.2.5 Random Forest Regression

Both models were configured with 100 estimators (trees) and no maximum depth to allow the trees to grow freely, with the random state set for reproducibility.

4.2.6 Extra Trees Regression

We iterated through different values of $n_{\text{estimators}}$ (25, 50, 100, and 200). For each iteration, we found the R^2 score for the corresponding $n_{\text{estimators}}$ to find the optimal number of estimators for the Extra Trees model. The model with the best score was selected as the final model.

4.2.7 LightGBM Regression

Both models were trained with the same hyperparameters:

- $n_{\text{estimators}} = 100$ (number of trees)
- learning_rate = 0.1 (step size)
- max_depth = 6 (tree depth)
- subsample = 0.8 (fraction of data used per tree)
- colsample_bytree = 0.8 (fraction of features used per tree)

4.3 Our Evaluation Metrics

4.3.1 Mean Squared Error (MSE)

Lower MSE is better. The formula for MSE is:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

4.3.2 Root Mean Squared Error (RMSE)

Lower RMSE is better. The formula for RMSE is:

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

4.3.3 Mean Absolute Error (MAE)

Lower MAE is better. The formula for MAE is:

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

4.3.4 Coefficient of Determination / R^2 Score

Measures how well a statistical regression model predicts an outcome. The R^2 score is between 0 and 1, with a higher R^2 score being better. The formula for R^2 is:

$$R^2 = 1 - \frac{\text{RSS}}{\text{TSS}} = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

Where:

- **RSS** = sum of squared residuals
- **TSS** = total sum of squares

5 Results

5.1 Reduced Dataset

Model	MSE	RMSE	MAE	R^2 Score
Linear	33.1398	5.75672	2.74994	0.642316
Ridge	32.9448	5.73976	2.73933	0.64348
Polynomial	20.1092	4.48432	2.14270	0.782958
Decision Tree	26.9403	5.19040	2.55280	0.709229
Random Forest	13.5814	3.68530	1.81709	0.853413
Extra Trees	13.8624	3.72323	1.87869	0.850381
LightGBM	13.7136	3.70319	1.79868	0.851986

Table 1: Model Performance on Reduced Dataset

5.1.1 MSE Comparison

Model	MSE
Random Forest	13.5814
LightGBM	13.7136
Extra Trees	13.8624
Polynomial	20.1092
Decision Tree	26.9403
Ridge	32.9448
Linear	33.1398

Table 2: MSE Comparison for Models on Reduced Dataset

Random Forest Regression performed the best under MSE.

5.1.2 RMSE Comparison

Model	RMSE
Random Forest	3.6853
LightGBM	3.70319
Extra Trees	3.72323
Polynomial	4.48432
Decision Tree	5.1904
Ridge	5.73976
Linear	5.75672

Table 3: RMSE Comparison for Models on Reduced Dataset

Random Forest Regression performed the best under RMSE.

5.1.3 MAE Comparison

Model	MAE
LightGBM	1.79868
Random Forest	1.81709
Extra Trees	1.87869
Polynomial	2.1427
Decision Tree	2.5528
Ridge	2.73933
Linear	2.74994

Table 4: MAE Comparison for Models on Reduced Dataset

LightGBM Regression performed the best under MAE.

5.1.4 R^2 Score Comparison

Model	R^2 Score
Random Forest	0.853413
LightGBM	0.851986
Extra Trees	0.850381
Polynomial	0.782958
Decision Tree	0.709229
Ridge	0.64348
Linear	0.642316

Table 5: R^2 Score Comparison for Models on Reduced Dataset

Random Forest Regression performed the best under the R^2 Score.

5.2 No Outliers

Model	MSE	RMSE	MAE	R^2 Score
Linear	4.02562	2.00639	1.46312	0.688978
Ridge	4.02562	2.00639	1.46312	0.688818
Polynomial	3.6847	1.91956	1.36297	0.715317
Decision Tree	6.55217	2.55972	1.79811	0.493775
Random Forest	3.18987	1.78602	1.27378	0.753549
Extra Trees	3.36745	1.83506	1.32416	0.739829
LightGBM	2.9133	1.70684	1.20636	0.774916

Table 6: Model Performance on No Outliers Dataset

5.2.1 MSE Comparison

Model	MSE
LightGBM	2.9133
Random Forest	3.18987
Extra Trees	3.36745
Polynomial	3.6847
Ridge	4.02562
Linear	4.02562
Decision Tree	6.55217

Table 7: MSE Comparison for Models on No Outliers Dataset

LightGBM Regression performed the best under MSE.

5.2.2 RMSE Comparison

Model	RMSE
LightGBM	1.70684
Random Forest	1.78602
Extra Trees	1.83506
Polynomial	1.91956
Ridge	2.00639
Linear	2.00639
Decision Tree	2.55972

Table 8: RMSE Comparison for Models on No Outliers Dataset

LightGBM Regression performed the best under RMSE.

5.2.3 MAE Comparison

Model	MAE
LightGBM	1.20636
Random Forest	1.27378
Extra Trees	1.32416
Polynomial	1.36297
Ridge	1.46312
Linear	1.46312
Decision Tree	1.79811

Table 9: MAE Comparison for Models on No Outliers Dataset

LightGBM Regression performed the best under MAE.

5.2.4 R^2 Score Comparison

Model	R^2 Score
LightGBM	0.774916
Random Forest	0.753549
Extra Trees	0.739829
Polynomial	0.715317
Linear	0.688978
Ridge	0.688818
Decision Tree	0.493775

Table 10: R^2 Score Comparison for Models on No Outliers Dataset

LightGBM Regression performed the best under the R^2 Score.

6 Analysis

Comparing tests for the reduced dataset, it appears that the Random Forest Regression produced the best results on average. For the no outliers dataset, it appears that the LightGBM Regression produced the best results on average. For both datasets, both the Random Forest and LightGBM Regression performed well. This trend, however, highlights the sensitivity of the dataset due to how depending on whether or not outliers were included, there were two different regression models performed best.

More generally, the more complex regression models performed better than those that were not as complex in implementation. For Random Forest Regression's case, its ability to train complex interactions between features without much preprocessing is very beneficial given these factors of high variability and influence on each other. In the case of LightGBM, being able to better handle datasets with higher dimensionality gave it an edge with this dataset and led to this model producing better results.

Between the reduced and no outliers datasets, coefficients of determination for the reduced dataset were generally better. This could be because models trained on the no outliers dataset were overfitted since there were no outliers, meaning that they were worse at predicting. These differences suggest that the outliers, while extreme, contributed information that enhanced the models, whereas their removal likely oversimplified the model or removed valuable data.

7 Conclusion

7.1 What We Learned

From this project, we were able to create models that explained the Uber fare amounts relatively well. Considering that we did not have all the data that Uber has, we got high coefficients of determination. We also learned about the importance of trying different models. If we had only tried linear or ridge regression models, the best model would have explained much less of the data and would have been worse overall.

We also learned about the importance of features and large datasets. There are so many things involved with algorithms, so it makes sense that many features are needed to predict. A large and high-quality data set makes a big difference when it comes to creating different models. Although a large dataset might seem good, it is not always the case. The data points might be invalid or the features are irrelevant. It is always important to analyze the data before training and testing models.

7.2 What's Next

Looking ahead, the top-performing regression models identified in this project can be applied to other datasets to evaluate their consistency and effectiveness in different regression analysis scenarios. Additionally, the scalability of these models can be tested beyond this project's scope with larger datasets to replicate real-world scenarios with millions of data points in a given dataset.

With larger datasets, access to more features would help create a more complex model that might get closer to mimicking Uber's dynamic pricing model. We could also try using Principle Component Analysis to see what are the most important components. We could also do more feature engineering, figuring out what features we can use or what features we can combine.

More features could also help us uncover potential biases within their pricing algorithm. For example, we can analyze the specific locations more and determine hotspots. Here, we can determine whether the algorithm is fair or perpetuating specific societal inequalities. This could reveal whether the algorithm disproportionately affects communities in specific locations or perpetuates certain societal or economic inequalities. This analysis could have a huge impact on both drivers and riders.

Uber is often under fire for unethical practices, so future work can help highlight flaws within its current practices and urge Uber to change its processes to be more equitable and uplift its customers.

8 References

References

Geanloren Atienza, Emmanuel Peňones, Diwanna Faye Lim, Xanrifae Lyn Tabliga, and Trisha Camille Valencia. 2023. Case study: Uber pricing strategy.

Pallab Banerjee, Biresh Kumar, Amarnath Singh, Priyeta Ranjan, and Kunal Soni. 2020. Predictive analysis of taxi fare using machine learning. *International Journal of Scientific Research in Computer Science, Engineering and Information Technology*, pages 373–378.

Junzhi Chao. 2019. Modeling and analysis of uber's rider pricing. In *Proceedings of the International Conference on Advances in Economics, Business and Management Research (AEBMR)*. Atlantis Press.

Florence Chee. 2018. An uber ethical dilemma: Examining the social issues at stake. *Journal of Information, Communication and Ethics in Society*, 16(3).

Peter Cohen, Robert Hahn, Jonathan Hall, Steven Levitt, and Robert Metcalfe. 2016. Using big data to estimate consumer surplus: The case of uber. Working Paper 22627, National Bureau of Economic Research, Cambridge, MA.

Elizabeth Rani. G, Sakthimohan. M, Revanth Raj. R, Sri Ganesh. M, Shyam Sunder. R, and Karthigadevi. K. 2022. An automated cost prediction in uber/call taxi using machine learning algorithm. In *2022 2nd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE)*, pages 764–767.

Kaggle and M Yasser H. Uber fare dataset. https://www.kaggle.com/datasets/yasserh/uber-fares-dataset.

Jessica Phillips. 2017. How Uber's Dynamic Pricing Model Works. https://www.uber.com/en-GB/blog/uber-dynamic-pricing/. Accessed: 2024-12-09.

Tulio Silveira-Santos, Anestis Papanikolaou, Thais Rangel, and Jose Manuel Vassallo. 2023. Understanding and predicting ride-hailing fares in madrid: A combination of supervised and unsupervised techniques.

George Willis and Emmanouil Tranos. 2020. Using 'big data' to understand the impacts of uber on taxis in new york city. *Computers, Environment and Urban Systems*, 84.

Kai Zhao, Denis Khryashchev, and Vo Huy. 2019. Predicting taxi and uber demand in cities: Approaching the limit of predictability. *IEEE Transactions on Knowledge and Data Engineering*, PP:1–1.

Yuanguang Zhong, Tong Yang, Bin Cao, and T.C.E. Cheng. 2021. On-demand ride-hailing platforms in competition with the taxi industry: Pricing strategies and government supervision.