PREDICTING NEW YORK CITY TAXI FARES WITH SUPERVISED MACHINE LEARNING

ZUBAID AMADXARIF
ID: 200320353

Word Count: 5951

SUPERVISOR
NINA OTTER

Submitted to the School of Mathematical Sciences of Queen Mary, University of London

BSc Mathematics

January 2023

CONTENTS

FIGURES

TABLES

1 Introduction

1.1 Introduction

I will be predicting taxi fares in NYC by using a few prediction machine learning methods on a large dataset. Despite the popularity of taxis in the city, their usage has decreased in recent years due to the availability of app-based ride-hailing services like Uber and Lyft, which offer fare estimates before the trip begins. Taxi services in New York City do not have this capability unless they are using an e-hail partner, which was introduced in 2013. These partners work similarly to Uber, but a portion of the fare is taken as a fee. There are multiple factors that contribute to taxi fares beyond just the meter, and the goal of this project is to develop a model that can accurately predict taxi fares to give customers a more informed decision about whether to use a taxi or not.

1.1.1 New York Transport

NYC is split up into zones and boroughs with a population of 8 million people [21]. There are many ways to travel in New York City [1] – the MTA (metropolitan transportation authority) includes the subways and buses using a metro card which costs $5.50 to buy. One subway/bus fare costs $2.75 and other options such as unlimited travel are also available. The subways and buses are available 24 hours a day and 7 days a week, with buses as frequent as every 15 minutes. There is also the Roosevelt Island Tram which is a tram service available seven days a week and you can use the MetroCard on it.

You can also hire a car (to drive like Zipcar or to ride like Uber), ride your own bike or hire the Citibikes available 24/7. Then comes New York's staple – the yellow taxicab.

Uber and Lyft are app-based ride-hailing services that compete with the yellow-taxicab. According to data from June 2017, Uber alone provided approximately 289,000 rides per day, while the taxi service provided 277,000 journeys on the same day [5]. These app-based services have become increasingly popular due to their accessibility and as a result, have disrupted the traditional taxi industry. However, the New York Taxi and Limousine Commission (TLC) forbid drivers to give an estimate of taxi fares because "it is impossible to pre-calculate a fare because the meter rate depends on traffic, construction, weather, and route to the destination." [6].

Too many factors depend on the calculation of the fare ride. It would be implied that a tariff-based prices would mean that the fare is purely calculated as costs per mile and per second travelled. When compared to fixed travel systems like trains, factors go into place like the different routes taken and the congestion on the road, so it makes it difficult to say how much a ride would cost beforehand. Uber have an advantage over the taxis as they give a predicted price and predicted arrival time before a customer chooses to book. This means that the user can choose if they want to accept the ride or if they want to establish another mode of transport.

1.1.2 Price Calculation

The data that is given to us about how the taxi fares are calculated are [7]:

- Minimum metered fare is \$2.50
- Increases \$0.50 for every 0.2 miles or every minute of travel

Then the following surcharges are added:

- MTA state surcharge of \$0.50 per ride
- \$0.30 improvement surcharge which goes to the Taxi Improvement Fund
- \$0.50 if it is overnight 8pm to 6am
- \$1 rush hour surcharge from 4pm to 8pm Monday to Friday
- NY State Congestion Surcharge of \$2.50 (Yellow Taxi) or \$2.75 Green Taxi or \$0.75 cents for all trips that begin, end or pass-through Manhattan south of 96th Street.
- Any Tips and Tolls for crosses bridges etc

As we can see, there are many internal factors that go into working out the taxi fare – and the time of the day is an important factor. Uber do not have a systemised fare structure like this. During busy times, Uber bring surge pricing [31] which boost the prices up to handle demand. They believe that as the prices are so high, it will encourage more drivers to work, and as more drivers come to work, the supply matches the demand, and the prices would decrease or reach an equilibrium.

1.1.3 Main Objectives and Deliverables

The main issue that taxi customers face is price uncertainty. Price certainty is important as it enables better cost planning, reducing corrupt pricing, and customers have better information to compare prices to competitors.

The project aims are as following:

1. Build machine learning models to predict NYC taxi price fares.
2. Compare and assess the performance of multiple machine learning models between tree ensemble models, tree boosting models and linear models.
3. Identify which factors influence the price the most from days, locations, time, and other factors.

1.2 Related Literature

During my research, I have studied various prediction-based techniques involving neural networks, deep learning, bagging, and boosting algorithms to gain a comprehensive understanding of their processes and potential benefits for my project. Examining existing papers on the subject provided insights into the advantages and disadvantages of different algorithms and methodologies for addressing my research question.

One study compared the effectiveness of gradient boosting with XGBoost, and a deep learning method called multi-layer perceptron (MLP) [18] for predicting trip duration and found that XGBoost with an RMSE log function outperformed the MLP model when all variables were considered. However, the authors noted that the MLP model could potentially be improved through autotuning, at the cost of additional time.

There has also been some exploratory data analysis comparing the data between app-based taxi services and the TLC taxis and seeing the effects of deploying a taxi price comparison to see what would happen if prices for both uber and taxis were known by the user [2]

A third study used linear regression and random forests to predict both time and price [19] for taxi journeys and found that the random forest model was more accurate than the linear regression model, even when higher order terms were added. This was because the data they had was nonlinear and the linear model did not fit it well.

The present study aims to enhance the current body of knowledge by utilizing the NYC (New York City) taxi fare dataset to predict fares through the implementation of multiple machine learning models such as those used in previous studies as well as newly released boosting models such as CatBoost. The performance of these models will be assessed using various metrics, and the results will be compared between models. Additionally, the study will delve deeper into the model results to identify the factors that have the greatest impact on price, which has not yet been explored in the extant literature.

2 Data Description

2.1 Dataset

The data for this paper was obtained from Kaggle [1] and consists of records collected by two vendors contracted by the NYC TLC over a couple months in 2021: Creative Mobile Technologies and Verifone Inc. The data was collected for the purpose of maintaining records for the NYC government and is a subset of a dataset that is regularly updated on the TLC's main website [8]. While the TLC claims to review all records to ensure their accuracy, they have stated that there may be some errors that they are not responsible for which should not pose a significant issue, as any outliers can be easily identified and removed if necessary.

2.2 Variable Description

The dataset consists of nineteen variables and has a size of 83961 x 20 variables. Below is the table that describes each variable, its type, and what it means:

FIELD NAME	VARIABLE TYPE	DESCRIPTION
VendorID	float64	A code indicating the TPEP provider that provided the record. 1= Creative Mobile Technologies, LLC; 2= VeriFone Inc.
tpep_pickup_datetime	object	The date and time when the meter was engaged.
tpep_dropoff_datetime	object	The date and time when the meter was disengaged.
Passenger_count	float64	The number of passengers in the vehicle. This is a driver-entered value.
Trip_distance	float64	The elapsed trip distance in miles reported by the taximeter.
PULocationID	int64	TLC Taxi Zone in which the taximeter was engaged.
RatecodeID	float64	The final rate code in effect at the end of the trip. 1= Standard rate 2=JFK 3=Newark 4=Nassau or Westchester 5=Negotiated fare.
DOLocationID	int64	TLC Taxi Zone in which the taximeter was disengaged
Store_and_fwd_flag	object	This flag indicates whether the trip record was held in vehicle memory

		before sending to the vendor, aka “store and forward,” because the vehicle did not have a connection to the server. Y= store and forward trip N= not a store and forward trip.
Payment_type	float64	A numeric code signifying how the passenger paid for the trip. 1= Credit card 2= Cash 3= No charge 4= Dispute 5= Unknown 6= Voided trip.
Fare_amount	float64	The time-and-distance fare calculated by the meter.
Extra	float64	Miscellaneous extras and surcharges. Currently, this only includes the $0.50 and $1 rush hour and overnight charges.
MTA_tax	float64	$0.50 MTA tax that is automatically triggered based on the metered rate in use
Trip type	float64	Type 1 is inner city journey; type 2 is outer city journey
Improvement_surcharge	float64	$0.30 improvement surcharge assessed trips at the flag drop. The improvement surcharge began being levied in 2015.
Tip_amount	float64	Tip amount – This field is automatically populated for credit card tips. Cash tips are not included.
Tolls_amount	float64	Total amount of all tolls paid in trip.
Total_amount	float64	The total amount charged to passengers. Does not include cash tips.
Congestion_Surcharge	float64	Total amount collected in trip for NYS congestion surcharge.

Table 1 - Dataset Description

2.3 Requirements

For the project solutioning, I will be using Python 3.10. Libraries used include “Pandas” will be using to visualize and create data frames to manipulate data. “NumPy” for all mathematical calculations such as working with vectors. A combination of “matplotlib” and “seaborn” will be used to generate graphs for visualization of the data. “XGBoost” and “CatBoost” libraries for gradient boosting. “Sklearn” includes functions for Linear Regression and Random Forests. Lastly the “datetime” library which will allow me to work with the datetime objects and manipulate the dates. Python allows for all these libraries to work together flawlessly with the functions given in base Python.

2.4 Cleaning Data

For prediction, we will only be referring to the “fare_amount” variable as our target and will not be using the columns that include any surcharges or extras. This allows us to predict the variable that is affected by many factors. Datetime variables of “lpep_pickup_datetime” and “lpep_dropoff_datetime” are imputed into a 3 columns: ‘time taken’, “day” and “hour”. All the extra payment columns are removed (taxes, tolls, and surcharges), and we are left with the following table:

	PULocationID	DOLocationID	passenger_count	trip_distance	day
count	83190	83190	83190	83190	83190
mean	108.340966	133.121229	0.797932	193.859863	2.995829
std	70.35481	77.140694	1.00053	4394.073593	1.850199
min	3	1	0	0	0
25%	56	69	0	1.35	1
50%	75	132	1	2.75	3
75%	166	205	1	6.19	4
max	265	265	32	260517.93	6

	hour	time_taken	trip_type	fare_amount
count	83190	83190	83190	83190
mean	13.228898	19.868885	0.630304	20.343863
std	4.931973	16.15171	0.523854	15.391564
min	0	0.016667	0	-150
25%	10	8.716667	0	9
50%	13	15	1	16
75%	17	26	1	26.81
max	23	119	2	480

Table 2 - Summary Statistics for dataset

The summary statistics in table 2 show potential outliers in the dataset. For example, where passenger count and trip types are 0, these cells must be removed as you cannot have these as 0, and any journeys whose times and distances are below zero must be removed too. This is completed manually, and the errors could be due to measurement error or invalid activity such as leaving the meter running when fares are not on. To ensure other outliers do not affect our model, we remove these by calculating a z-score for each column using:

$$z = \frac{(x - \bar{x})}{\sigma}$$

$where\ z\ is\ the\ z\ score, x\ is\ the\ data\ point, \bar{x}\ is\ the\ mean\ of\ the\ column, \sigma\ is\ the\ standard\ deviation\ of\ the\ column$

and compare the z-score to the mean-to-standard deviation ratio for each corresponding column. Each of these datapoints are added to a new dataset where it is has no outliers. We can see the updated values in table 3 with over 45,000 values:

	PULocationID	DOLocationID	passenger_count	trip_distance	day
count	45647.0	45647.0	45647.0	45647.0	45647.0
mean	96.1	135.0	1.3	2.4	3.0
std	62.9	77.6	1.0	2.0	1.9
min	3.0	3.0	1.0	0.0	0.0
25%	52.0	74.0	1.0	1.1	1.0
50%	75.0	135.0	1.0	1.8	3.0
75%	119.0	215.0	1.0	3.2	4.0
max	265.0	265.0	7.0	41.8	6.0

	hour	time_taken	trip_type	fare_amount
count	45647.0	45647.0	45647.0	45647.0
mean	13.7	12.7	1.0	11.2
std	5.1	8.5	0.2	6.0
min	0.0	0.0	1.0	0.0
25%	10.0	6.7	1.0	7.0
50%	14.0	11.0	1.0	9.5
75%	18.0	16.7	1.0	14.5
max	23.0	116.3	2.0	29.7

Table 3 - Summary Statistics cleaned dataset

3 Exploratory Data Analysis

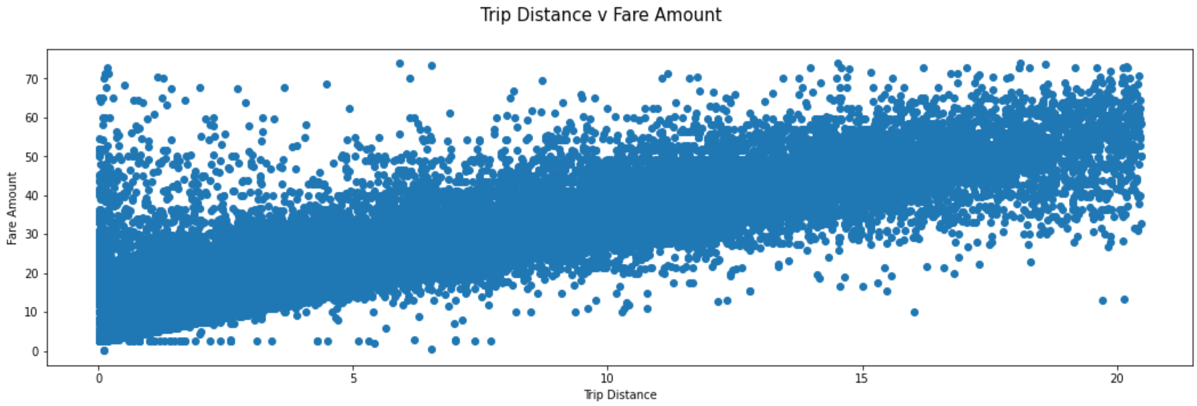


Figure 1 - Trip Distance v Fare Amount

Figure 1 shows a positive correlation between (0.882) between trip distance and fare amount with variation. The variation suggests other factors will also affect fare amount even if trip distance is short.

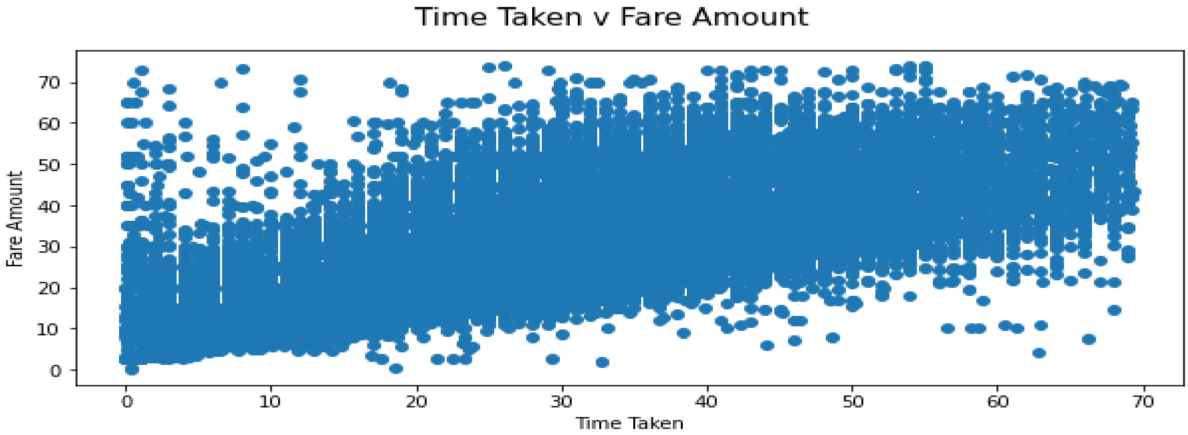


Figure 2 – Time Taken vs Fare Amount

Similarly, to the graph above, we can see that the graph does show a similar positive correlation (0.824) between the time taken and total fare, but we can see that the data is more spread between the fare ranges.

3.1 Trends in Rides

As fare amount seems to be affected by other factors, we assess these in further detail here. Figure 3 plots the number of rides across different days. We find that the most popular days are Friday, Thursday, and Saturday in that respective order with Sunday being the least popular. We look further into this and see the number of rides of the individual days.

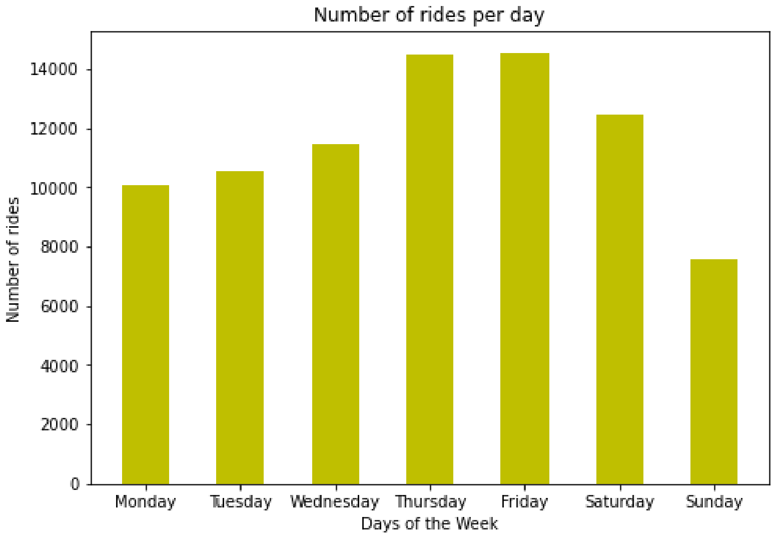
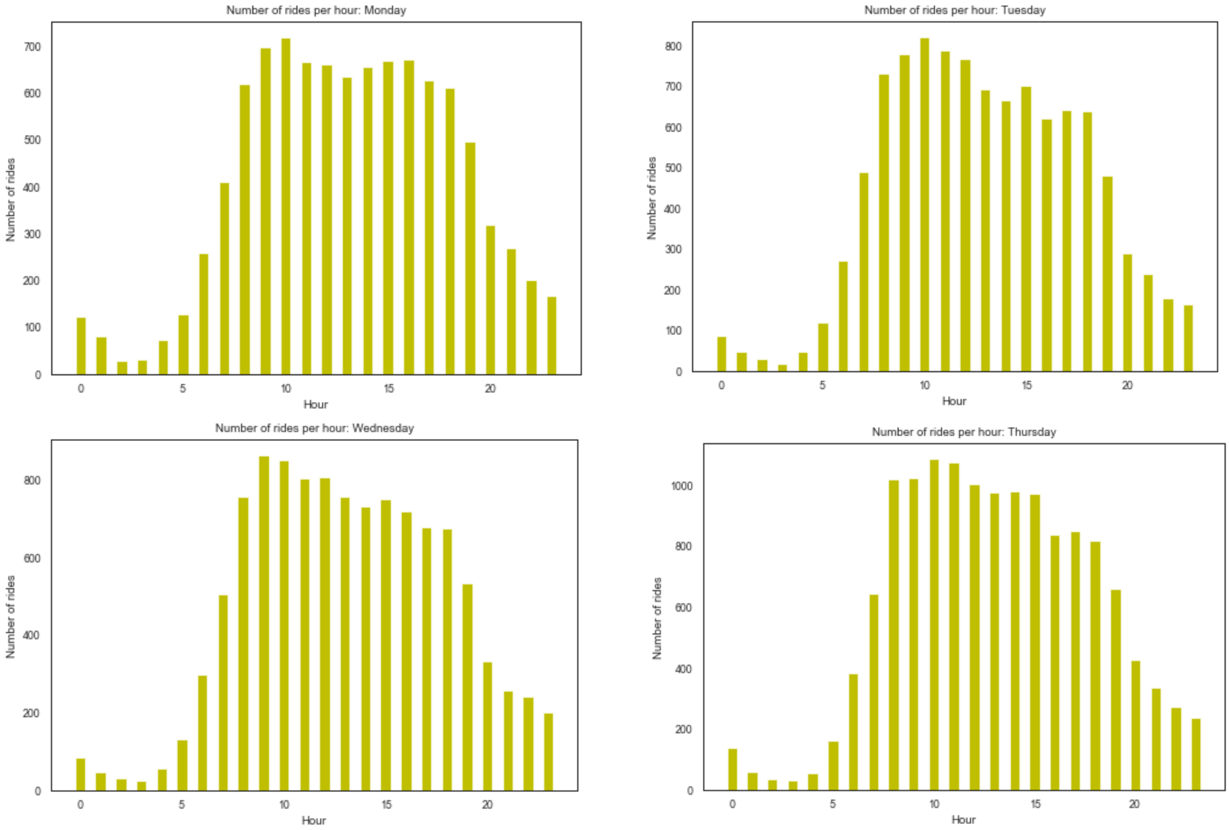


Figure 3 – Number of rides per day of the week

Figure 4 plots the number of rides per hour for each day. Hourly trends look very similar for each day where midnight to 5am are quiet and 8am to 8pm are the busiest windows. We would expect mornings and evenings to be busiest with work or school journeys, but busiest times are throughout the day.



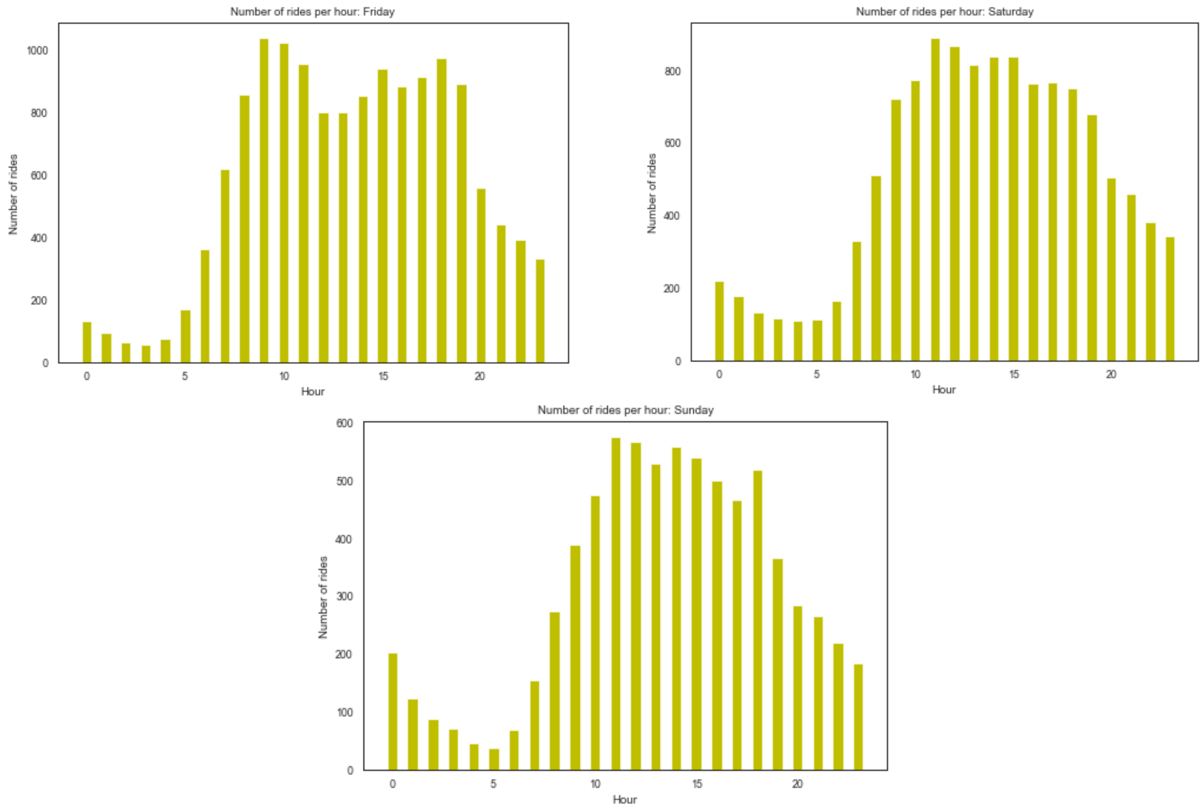


Figure 4 - Number of Rides per hour for each day of the week

3.2 Linear Regression model

In this section, we fit a regression line between our target ‘fare amount’ and all factors. We then assess metrics and the distribution of residuals to assess its fit.

Linear regression is a supervised learning algorithm that is used for predicting a continuous outcome (also known as a target or dependent variable) based on one or more predictor variables (also known as features or independent variables). In linear regression, the target variable is modelled as a linear function of the predictor variables, with the coefficients of the predictor variables representing the strength and direction of the relationship between the predictor variables and the target variable.

Here, we have modelled the target variable fare amount against the six variables – pick up location, drop off location, trip distance, day, time taken and trip type as per the specification (1).

$$fare\ amount = b_0 + b_1 * PU\ Location + b_2 * DO\ Location + b_3 * trip\ distance + b_4 * day \\ + b_5 * time\ taken + b_6 * trip\ type\ (1)$$

where b_0, $b_1, b_2, b_3, b_4, b_5\ and\ b_6$ are 7.57, 0.0989, 0.00243, 0.000007, 0.171, 0.7229 and -5.57 r espectively. All coefficients are highly significant at the 1% significance level with a p-value l ess than 0.01, except b_2 with a p-value of 0.29.

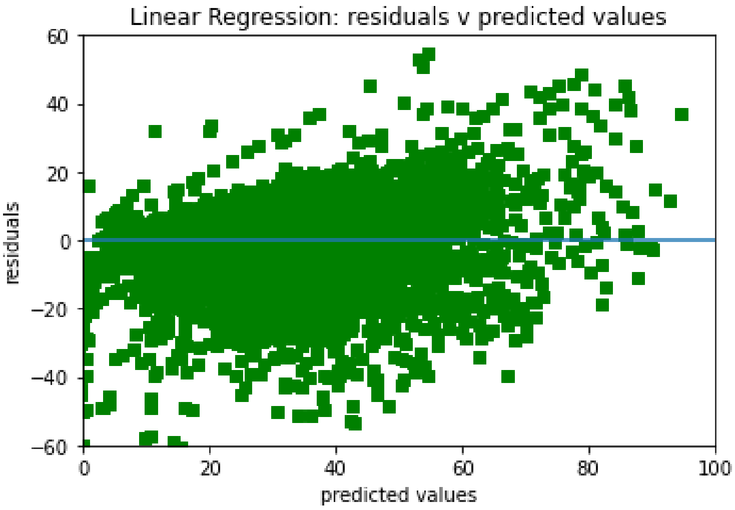


Figure 5 - Linear Regression - residuals v predicted values

To ensure coefficients are inferred correctly, residual terms must be independent and there should be no heteroskedasticity [24]. Figure 5 shows a trend between residuals and predictions which suggests the errors are not randomly distributed and possible heteroskedasticity. Therefore, whilst this does not mean estimators are unbiased, the resulting standard errors will be incorrect and any inference on coefficients will be misleading.

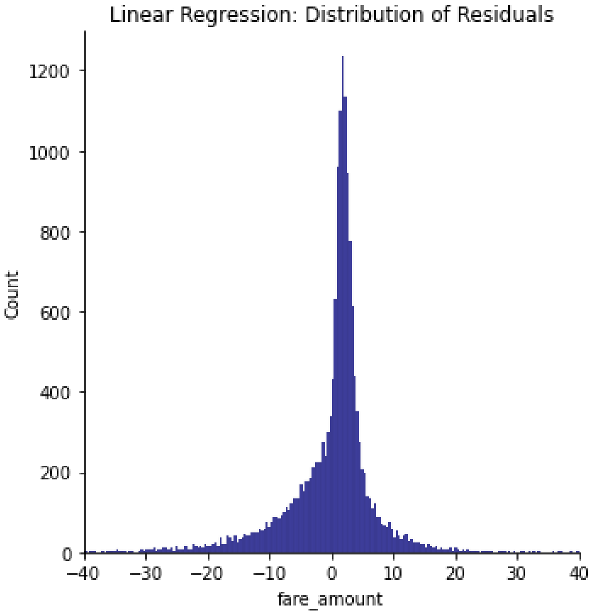


Figure 6 - Linear Regression: Distribution of Residuals

The normality assumption of linear regression is also violated here as this residuals display a mean of -0.1 rather than 0 and the distribution is skewed rather than normally distributed. This will also cause problems in inference as hypothesis tests such as t-tests will become unreliable [25].

4 Modelling

4.1 Machine Learning

Machine learning (ML) is a rapidly growing field with widespread applications in various aspects of life, including image processing for autonomous vehicles, targeted advertising, and more. ML refers to the ability of systems to learn from problem-specific training data and automate the process of constructing analytical models to solve related tasks [10]. It is a subset of artificial intelligence that involves discovering patterns and relationships in data through examples and building upon previous observations to find new insights. ML has the potential to revolutionize a wide range of industries and has already had a significant impact on many aspects of our lives.

There are three types of ML – Supervised Learning (SL), Unsupervised Learning (UL) and Reinforcement Learning (RL). [11]

SL relies on ML tasks to learn a function that maps an input to an output by learning the relationship between variables - which has been used in text categorisation and facial recognition [12]. Data are labelled with classes and outcomes and models then learn from this data (used for classification or regression). UL doesn't need the data to be labelled and puts data into classes itself and the model learns from unlabelled data using clustering, hierarchical clustering, and Principal Component Analysis (PCA). RL operates sequentially and at each iteration makes better decisions after failed attempts as models are based on reward or penalty. This type of Learning also uses classification. [11].

For this problem, we will use a supervised learning approaches as I have a structured dataset with features and a target, and the problem is a prediction problem.

4.1.1 Supervised Learning - Prediction

In the previous section, we used a linear regression model to predict fare amounts. This is also a supervised learning method, and we found some potential problems in inference if we used that approach. In the following sections, we compare the performance of two tree-based boosting algorithms, bagging tree-based algorithm and linear regression to identify the best performing model.

It is easy to assume that as the number of data values and variables increase, the prediction becomes more accurate but that doesn't always occur. Overfitting may occur when the model becomes too fit for our data, and it learns all the irregularities that our dataset may have [13]. One of the major issues is also a bias-variance trade-off, where a model could have a low bias indicating a good fit but high variability indicating a bad fit and vice-versa.

4.2 Models for Prediction

4.2.1 Decision Tree

A decision tree is a type of machine learning model that can be used for both classification and regression tasks [28]. The model learns a set of rules based on the features in the data (X variables) and applies these rules to make predictions about the target variable (Y variable). In this case, the target variable is the fare amount.

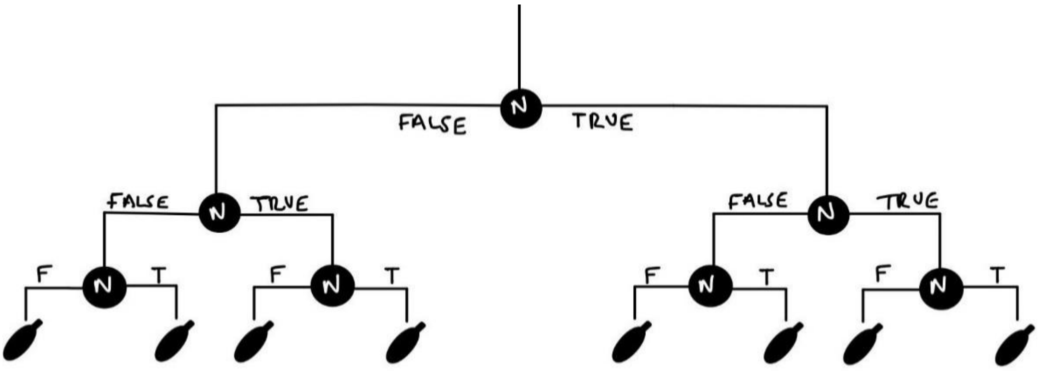


Figure 7 – Example of a Decision Tree

The decision tree model is structured like an inverted tree (figure 7), with the data being progressively divided into smaller and smaller subsets at each level [14]. Each node (labelled N) contains a true or false statement that determines how the data is split based on that condition. This process continues recursively until a leaf node is reached, which represents the final prediction made by the model.

For example, if we were using a decision tree to predict prices for a taxi journey from one location to another, the nodes at the first level might be labelled "Is the journey less than 5 miles?" and would split the data into two subsets based on this condition. At the second level, we might ask "Will the journey take longer than 10 minutes?" and continue to split the data in this way until we reach a leaf node with the final prediction. By breaking the data down in this way, the decision tree can make more accurate predictions by considering a variety of factors and their interactions.

In machine learning, weak learners are models that can be combined to create more complex models, but do not perform well on their own. Decision trees are a type of weak learner because they are prone to overfitting, which occurs when the model learns patterns in the training data that do not generalize to new, unseen data which can result in poor performance on test data. Additionally, decision trees may not be as effective at capturing complex relationships in the data as some other algorithms. Decision trees can still be useful to the problem as a component in an ensemble model like random forest or gradient boosting model. In these cases, the decision tree is combined with other models, which can help to improve the overall accuracy and reduce the risk of overfitting.

4.2.2 Random Forest – Bagging Algorithm

Random forests are a type of ensemble learning method for regression and classification tasks [27]. Ensemble methods are machine learning algorithms that combine the predictions of multiple individual models to make more accurate predictions than any of the individual models could achieve on their own.

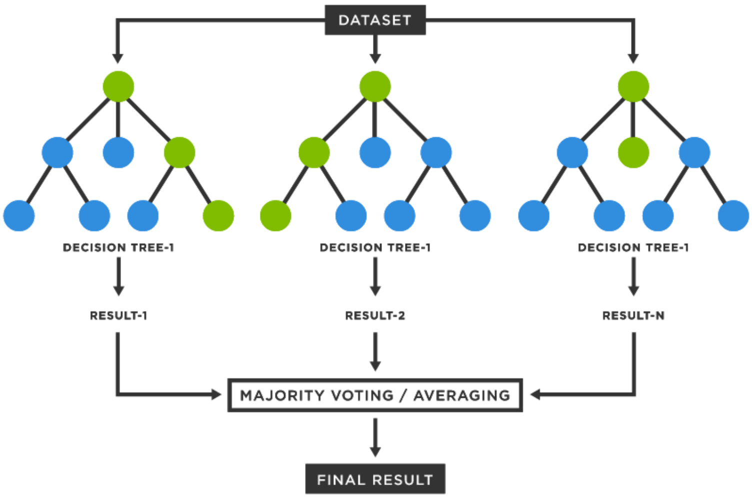


Figure 8 – visualised Random Forests - https://www.tibco.com/reference-center/what-is-a-random-forest

For regression, the individual models in the ensemble are decision trees. To train a random forest for regression, the algorithm generates many decision trees using a process called bootstrapping. This involves selecting random samples of the training data with replacement and using these samples to train each decision tree. The decision trees are trained on different samples of the data, and the features used to split the nodes in each tree are also randomly selected.

To make a prediction using the random forest, the algorithm processes the input data through all the decision trees in the ensemble. The final prediction is then calculated by taking the mean of the predictions made by the individual decision trees. Random forests have several advantages for regression tasks as they are capable of handling large amounts of data and many features, and they are relatively resistant to overfitting.

4.2.3 Gradient Boosting Method

Gradient boosting is a machine learning technique that creates an ensemble of weak learners (decision trees) and combines them to make a strong model that can make accurate predictions [29].

The gradient boosting algorithm works by fitting a weak learner to the data and using the gradient of the loss function to determine the direction in which the model should be improved. It then adds a new weak learner to the model in a manner that reduces the loss and repeats this process until the desired number of learners has been added or the loss has been minimized to an acceptable level.

The final prediction made by the gradient boosting model is the sum of the predictions made by the individual models in the ensemble. Gradient boosting is often used for regression and classification tasks, and it is known for being a very effective and powerful machine learning algorithm, especially when the individual models are decision trees.

Loss Function

A loss function is a measure of the difference between the predicted values of a model and the true values of the data. The goal of the model is to minimize the loss function, which indicates how well the model can predict the true values. There are various loss functions that can be used for regression problems [16], and the choice of loss function will depend on the specific requirements of the problem and the type of data being used.

In this study, the Root Mean Squared Error (RMSE) was chosen as the loss function to evaluate the performance of the models, as it is a commonly used metric that measures the average squared difference between the predicted and true values. It is often preferred because it is easy to interpret, and it places more weight on large errors.

RMSE is a measure of the square root of difference between the predicted values and the true values in a regression problem. It is defined by:

$$RMSE = (\frac{1}{n}\sum_{i=1}^{n}(y_i - \widehat{y_i})^2)^{0.5}$$

Minimising Loss Function

Decision Trees are used as the weak learner in gradient boosting and used in ensemble modelling. Each output is given, and the goal is to minimise loss by adding new models together. For example, if the value of the loss function is high for one decision tree, the value of the loss function should decrease when added to the first model. This can be denoted as:

$$F(m) = F(m-1) + \alpha * -\frac{\delta L}{\delta F(m-1)}$$

$$where\ F\ is\ the\ ensemble, m\ is\ the\ step\ of\ ensemble, alpha\ is\ the\ learning\ rate,$$

$$L\ is\ the\ loss\ function, the\ differential\ is\ the\ weak\ leaner\ at\ step\ m.$$

The learning rate is a hyperparameter that plays a crucial role in the training process of a machine learning model. It determines the step size at which the model updates its predictions based on the gradient of the loss function. A smaller learning rate results in smaller updates to the prediction error and may lead to more accurate predictions, but it may also increase the training time. Whereas a larger learning rate may result in faster training but may also lead to less accurate predictions. It is important to carefully tune the learning rate to ensure that the model can learn effectively and make accurate predictions.

4.3 Metrics and Measures for Model Performance

To measure the model performances for the four models created I will look at some common metrics.

4.3.1 R-Squared

A regression model's goodness of fit can be measured using R-Squared. It is a number that indicates how well the model fits the data and is between 0 and 1, with 1 representing the best fit and 0 representing no fit. It is calculated as:

$$R^2 = 1 - \frac{RSS}{TSS}$$

where:

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

$$TSS = \sum_{i=1}^{n} (y_i - \bar{y})^2$$

$$y_i \text{ is the actual value}, \hat{y}_i \text{ is the predicted value}, \bar{y} \text{ is the mean}, n \text{ is the number of obersations}$$

This will be a good indicator and a common metric to compare the models against each other. If the R-squared value is high, it would indicate a good model, but it could also be high if the model has been overfitted. The other metrics would confirm/negate this, and we could look at things like the residuals for normality assumptions or perform SHAP tests (Tree Models and Gradient Boosting) for feature importance.

4.3.2 Mean Squared Error (MSE)

MSE is the squared difference between the predicted and actual values, which is the average of the squared differences. It is calculated by:

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

$$\text{where } n \text{ is the number of observations}, y_i \text{ is the actual value and } \hat{y_i} \text{ is the predicted value}$$

The lower the MSE the more accurate the model, while a higher MSE indicates a less accurate model.

4.3.3 Mean Absolute Error (MAE)

MAE is the average absolute error between actual and predicted value. It is calculated as:

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

$$\text{where: } y_i \text{ is the actual value}, \hat{y_i} \text{ is the predicted value, and } n \text{ is the number of observations}$$

Without considering whether the predicted value is higher or lower than the actual value, MAE is a measure of the magnitude of the error. It is useful to us as it gives us a confidence interval on how far off our prediction is as it is measured in the same unit as our target variable.

4.3.4 Shapley Additive Explanations (SHAP)

SHAP is a ML algorithm that explains the output of a model by giving each feature importance to the final prediction. It uses concepts from game theory to calculate the contribution of each feature to the model [17].

There are a couple functions within the SHAP library which will help us visualise the importance of each feature in a model's prediction:

- SHAP values – represent the contribution of the feature to the model's output. A positive value indicates a positive impact on the prediction and the negative value indicates a negative impact on the prediction – this is especially helpful as it is in the unit of the target variable
- Summary Plots – will create a bar chart that displays the SHAP values for each feature, ordered by their importance in their model
- Dependence Plots – will help visualise the relationship between a single feature and the model's output by creating a scatter plot by putting the SHAP values on the x-axis and the model's output on the y-axis helping us identify the relationship between the feature and the model

Using these to compare the different ML models (CatBoost, XGBoost, Random Forest) will help to understand the individual features of each model and help us to decide which model is the best along with the other metrics stated above.

4.4 Prediction APIs

There are three different models that I have trained the data with that I have talked about above – Gradient Boosting, Linear Regression and Random Forest. There are four models I have run in total:

- CatBoost API
- XGBoost API
- Random Forest Regressor from sklearn library
- Linear Regression from sklearn library

CatBoost is an implementation of gradient boosting ML library made by a Russian company named Yandex. One of the key features of CatBoost is that it handles categorical data, missing values and allows us to plot variable importance to see how much each variable impacts the model for prediction. CatBoost has been used to solve real-world issues like recommendation systems, computer vision, and natural language processing making it very popular in industry.

XGBoost (eXtreme Gradient Boosting) is an implementation of the gradient boosting algorithm in Python, developed by Tianqi Chen [15] and has become one of the most popular machine learning libraries in recent years – I will be using this to compare the two different gradient boosting methods to see which one gives the best result.

Random Forest Regressor and Linear Regression algorithms will be used from the sklearn library. Sklearn is a library for machine learning in Python and is built on top of NumPy, SciPy, and matplotlib and provides a wide range of machine learning algorithms and tools for data analysis and predictive modelling which we have used.

4.4.1 One Hot Encoding

The cleaned dataset we have contains a mix of categorical and numerical values. We ensure categorical columns are used correctly as inputs to the models by carrying out one hot encoding. One-hot encoding is a method used to represent categorical variables as numerical data [30]. It involves creating a new binary column for each unique category in the categorical variable. The values in each column are then set to 0 or 1, depending on whether the row belongs to that category or not. The categories that will be one-hot encoded are the LocationIDs, time metrics (day, hour), passenger count and trip type.

4.5 Prediction Results

ML Method	R-Squared	Adjusted R-Squared	MSE	RMSE	MAE
CatBoost	0.945	0.942	1.95	1.40	0.541
XGBoost	0.938	0.934	2.20	1.48	0.568
Random Forest	0.937	0.933	2.24	1.50	0.524
Linear Regression	0.845	0.846	24.67	4.97	2.91

Table 4 - Metrics for Machine Learning Methods

Table 4 displays the metrics derived from training the four machine learning models and predicting fare amount on the testing data.

To note, the data was split into training (80%) and testing (20%) splits. For the XGBoost model, I have used 100 weak learners, and the learning rate was set at 0.3 as default whereas CatBoost used 1000 weak learners and the learning rate was 0.07.

The results show that the CatBoost algorithm reported the highest R-squared value, lowest MSE, lowest RMSE and second lowest MAE. The XGBoost algorithm performed like the CatBoost algorithm given its boosting configuration but performed slightly worse by a few decimal points in each metric. The Random Forest Regressor reported strong results too with a high R-squared, low RMSE, low MSE and the lowest MAE. The Linear regression model performed the worst and had results that were far from the other models.

We can see from figure 9 that the residuals are diamond shaped indicating that the model is making larger errors for certain combinations of input features and mostly for mid-range prediction values whereas low and high prediction values have smaller residuals. This suggests the models suffer from heteroskedasticity and the residuals are not randomly distributed. This could be due to many factors such as missing columns or lack of data in certain regions of the feature space. To improve the performance of the model in this case, it might be necessary to add more data to the training set, particularly in the regions where the errors are largest. Furthermore, this will mean inference at least for the linear regression model will be flawed although the coefficients will not be biased. For tree-based models, this property will not lead to flawed inference as this is not a requirement for tree-based models and for inference using SHAP. Figure 10 shows that CatBoost and XGBoost seem to the models whose residuals follow a distribution close to Normal.

Given this, we can conclude the CatBoost returns the best predictions compared to other models on unseen data (code can be found at [31]).

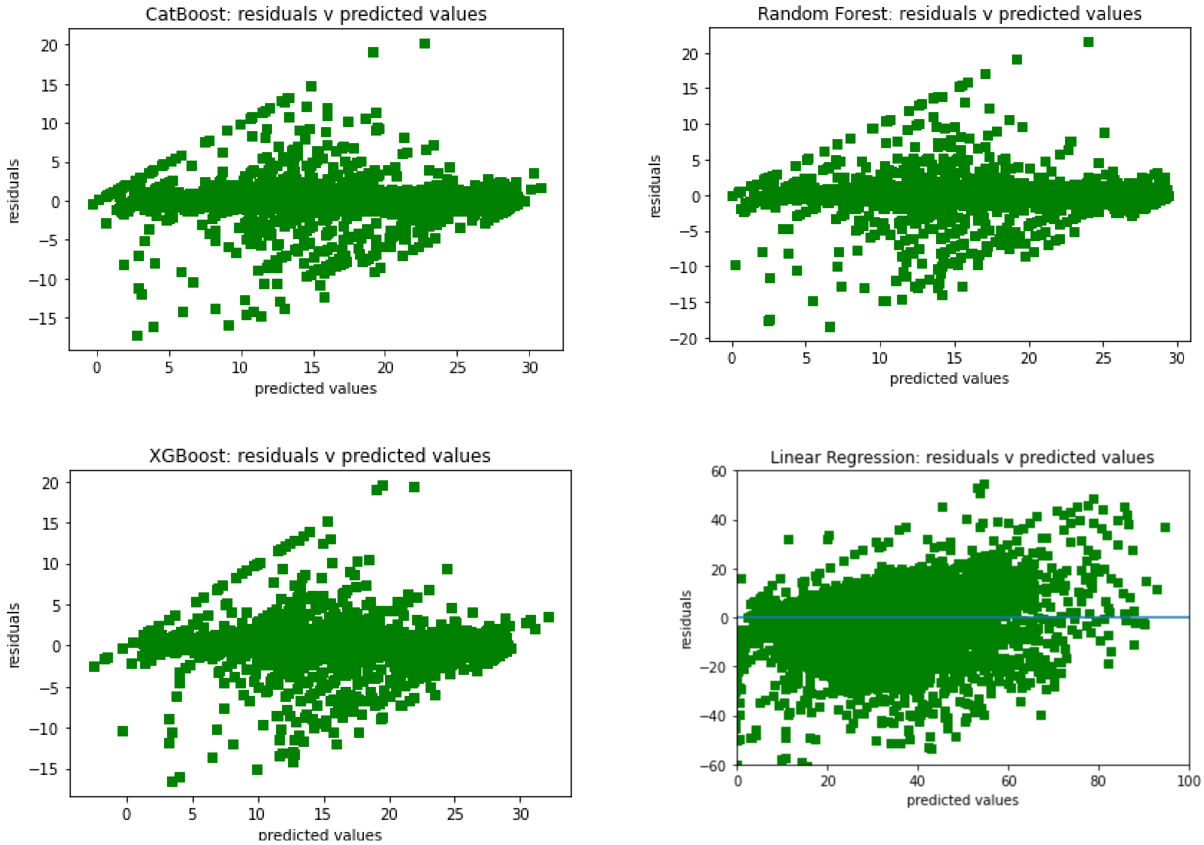


Figure 9 - Residuals for ML models

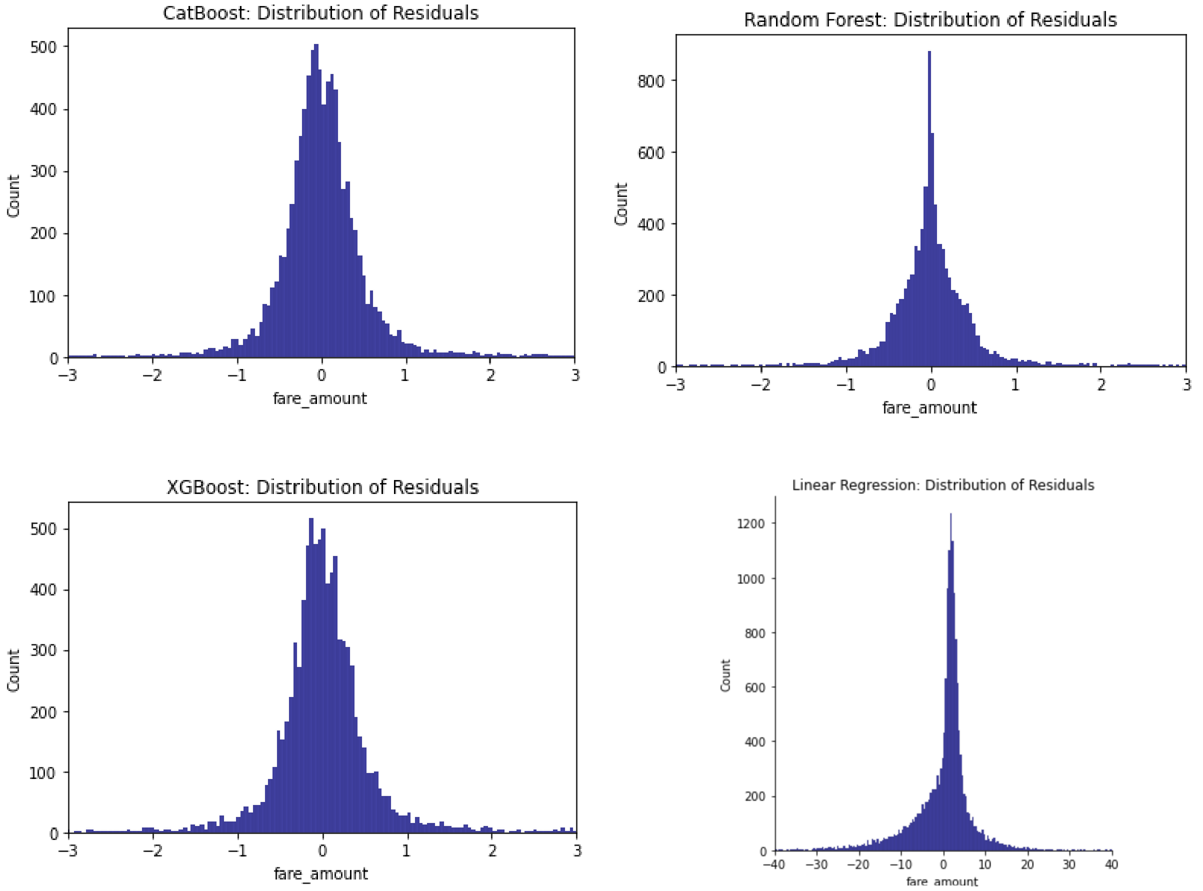


Figure 10 - Distribution of Residuals for all ML methods

4.5 Insights with SHAP

In this section, we dive further into the best performing trained CatBoost and XGBoost models using SHAP to identify which features were most important to the prediction and how different values of features affect the prediction. SHAP is the best approach for this as it does not require normally distributed residuals, homoskedasticity or other linear regression assumptions to perform inference.

4.5.1 Summary Plots

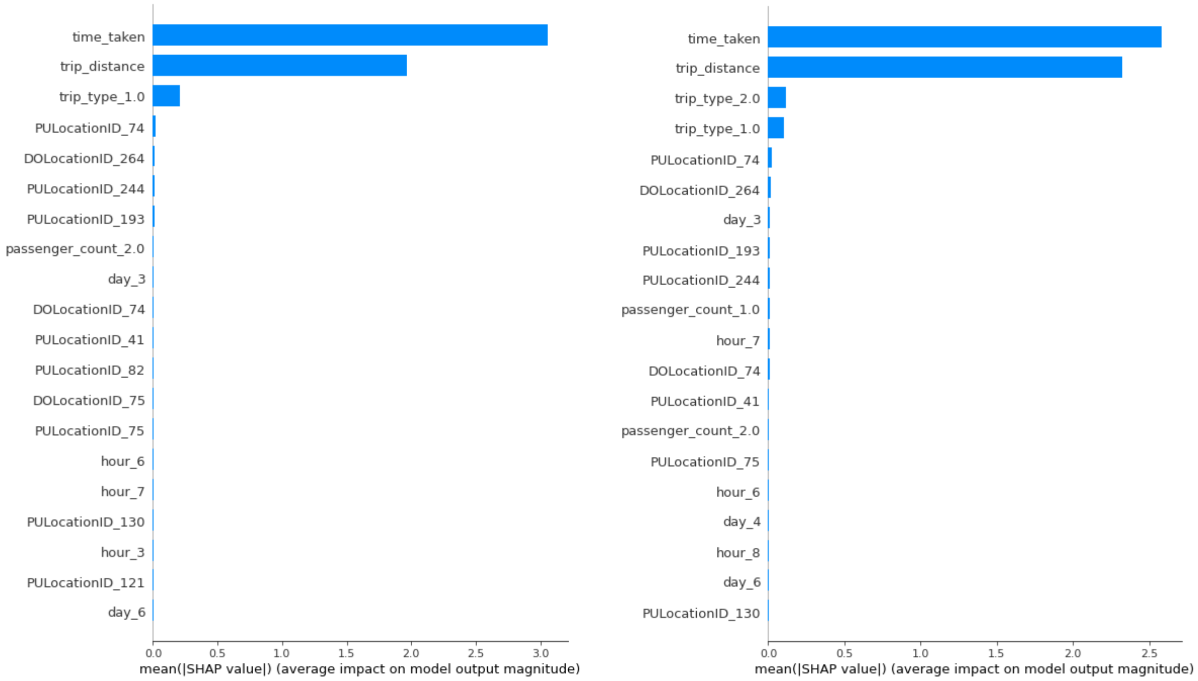


Figure 11 - SHAP Summary Bar Plots - a) XGBoost on left, b) CatBoost on right

Figure 11 displays the mean SHAP (impact on model) output for each feature. We find that 'time taken' has by far the highest impact in predicting fare amount followed by trip distance and both inner city and outer city trip types. XGBoost does not consider inner city trips to have much impact on the model whereas CatBoost considers outer city trips to have more impact than inner city trips. This is in line with what we assessed in the exploratory data analysis phase. It also seems as though pickup location 74 is the most influential with drop off location 264 but do not have as strong impact on the model.

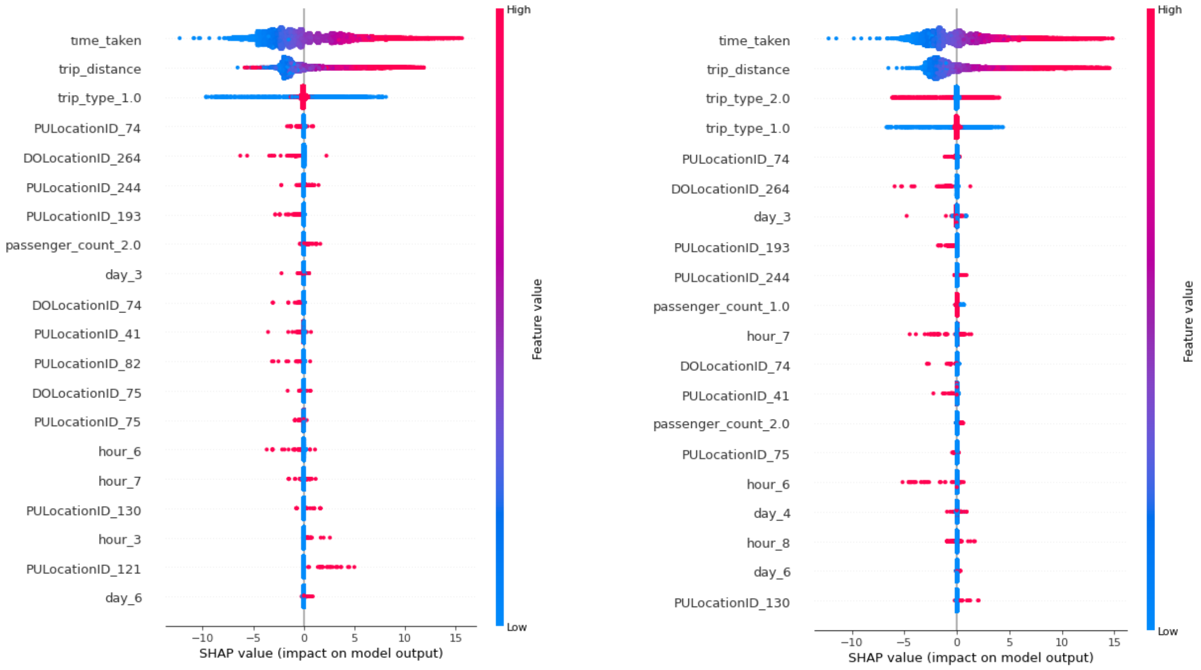


Figure 12 - SHAP summary plots for XGBOOST (left) and CATBOOST (right)

Figure 12 displays summary plots which allow us to analyse how different features affect predictions. We find a few interesting insights:

- The higher the distance of the journey, the higher the price of the fare
- The more time it takes for the journey, the higher the price of the fare
- The trip type decides if it is outer city and inner city – if it is outer city, it has a higher impact on price and if it is inner city then it is lower impact on price for both models
- The day & hour of the trip affects price a little
- Specific pickup and drop off locations affect the fare amount more than others (e.g., PULocationID 74 affects pricing most)

Each of these insights confirm our hypotheses as we learn that both time and distance are important in the prediction of the fare amount and the day, time it takes, and pickup drop off locations do affect the prices. This makes sense as the more time and distance it takes for a journey then it would cost more. Time includes things like congestion and different routes. If there was congestion it would take more time for the journey to complete. We hypothesised that the pickup locations and drop off locations should not affect the fare amount, but it seems as though specific locations do. This could be because some areas are more in demand or have a richer demographic, so prices could be a little inflated. It could also be because some areas are busier than others so there is more congestion which means more time taken.

4.5.2 Dependence Plots

We now look at a few dependence plots to gain some further insights in the models.

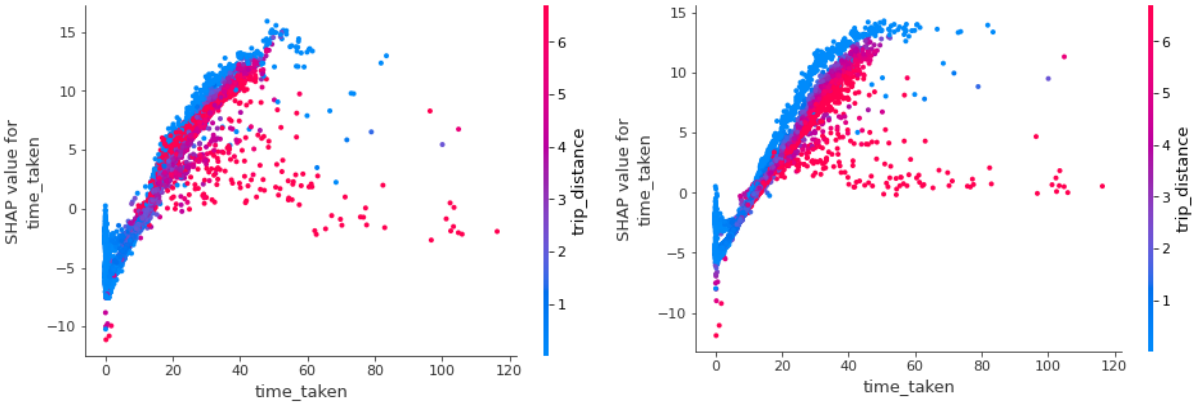


Figure 13 - SHAP dependence plot time taken and trip distance for XGBoost (left) and CatBoost (right)

We can see from figure 13 that both models show as time increases, the price increases, the left band shows that the trip distances when they are higher, they are the darker red indicating there is a strong relationship between the trip distance and the time taken.

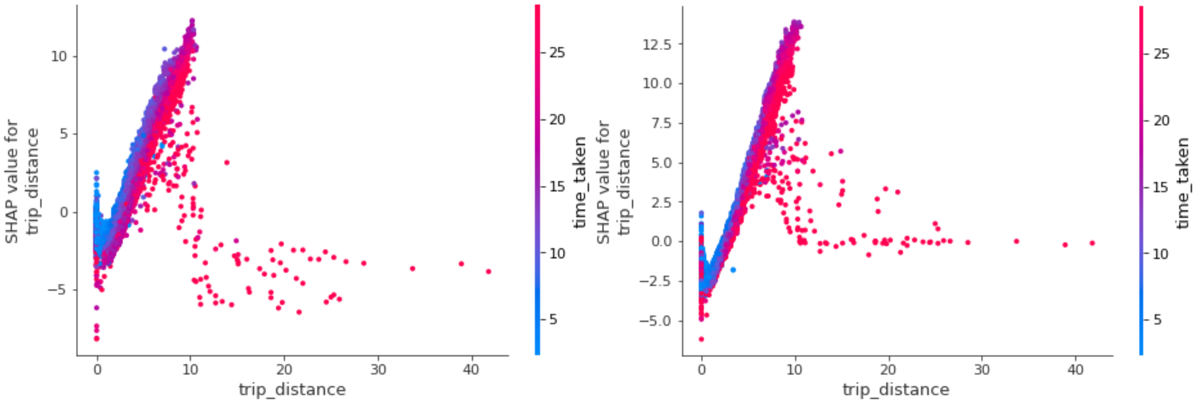


Figure 14 - SHAP dependence plots trip distance and time taken for XGBoost (left) and CatBoost (right)

Figure 14 shows us that XGBoost seems to have a greater negative impact on average compared to CatBoost which has little negative impact on the model.

5 Conclusions

In this paper, we put forward a first-step solution to solving the problem of price uncertainty for NYC cab customers by building a model to predict the fare amount. By gathering the dataset of historical NYC cab fares and their factors, we saw correlations between trip types, the time taken for a trip and the trip distance and its fare amount. We then built four machine learning models to predict fare amount using several factors. We find that the CatBoost model performs the best in predicting fare amount based on several metrics such as R-squared, RMSE, MSE and MAE. Other tree-based models such as XGBoost and Random Forest performed comparably with the linear regression model performing the worst. For this model, the most important factors in predicting fare amount were distance, time taken, location IDs and whether the trip type was inner city or outer city.

The second stage of solving the problem of price uncertainty for NYC cab customers would be to build a user-based tool which would provide users with outputs from this model for their desired trip. Given the strong performance of this model, it can provide customers with reliable predictions of fare amounts. Such a tool will target customers worried about cab drivers overcharging, customers who want to make informed cost-effective decisions between transport options as well as supporting the declining taxi industry whose customers are choosing competitors on app-based services who provide price prediction already. The TLC could create their own application in this way to help them keep up with the competition.

Any system which utilized this model must also provide a prediction for the time taken which feeds into the model as an input and is not known beforehand [19]. Furthermore, any machine learning architecture incorporating this model should follow MLOps best practices including updating the model by training on updated datasets on a frequent basis and ensure model performance is still adequate with time.

Whilst prediction performance was strong for tree-based models, we found that the residuals were not random and demonstrated a diamond trend with its predictions. This suggests that there are columns that are not in the dataset which could help further predict fare amount. Therefore, further research should aim to enhance the dataset incorporating additional datasets which provide additional information such as more accurate pick up and drop off location or other trip-related factors, environment-related factors, or competition fare factors [18][19]. Additional datasets could include satellite information, prices of competitors and public transport, weather information, demand for taxis and distance to nearest public transport spot. Alternatively, further research could attempt to utilize methodologies not used in this paper or the literature such as more complex deep learning models.

Bibliography

1. Ganapathi, A., NYC Gov, (2022). "Taxi Trip Data NYC", Version 1. Retrieved September 29, 2022, from https://www.kaggle.com/datasets/anandaramg/taxi-trip-data-nyc.

2. Noulas, A., Salnikov, V., Hristova, D., Mascolo, C. and Lambiotte, R., 2018, October. Developing and deploying a taxi price comparison mobile app in the wild: Insights and challenges. In *2018 IEEE 5th International Conference on Data Science and Advanced Analytics (DSAA)* (pp. 424-433). IEEE.

3. *TIME*, 2014, [online] Available: http://time.com/3633469/uber-surge-pricing/.

4. *Slate. The mirage of the marketplace*, 2015, [online] Available: https://slate.com/technology/2015/07/ubers-algorithm-and-the-mirage-of-the-marketplace.html

5. *Curbed*, 2017, [online], Uber surpasses yellow cabs in average daily ridership in NYC, https://ny.curbed.com/2017/10/13/16468716/uber-yellow-cab-nyc-surpass-ridership

6. Majasaki C., *Investopedia, 2021,* [online], *Uber vs. Yellow Cabs in New York City: What's the Difference?* https://www.investopedia.com/articles/personal-finance/021015/uber-versus-yellow-cabs-new-york-city

7. *NYC, Taxi and Limousine Commission, 2022, Taxi Fare,* [online], https://www.nyc.gov/site/tlc/passengers/taxi-fare.page [Accessed on 30/09/2022]

8. NYC, Taxi and Limousine Commission, 2022, TLC Trip Record Data, [online], https://www.nyc.gov/site/tlc/about/tlc-trip-record-data.page

9. *Dubois, J., Quanthub, 2021, [online], "What Programming Language is Most Useful for Data Science?",* https://quanthub.com/python-for-data-science/

10. Janiesch, C., Zschech, P. & Heinrich, K. Machine learning and deep learning. *Electron Markets* **31**, 685–695 (2021). https://doi.org/10.1007/s12525-021-00475-2

11. Raffaele Pugliese, Stefano Regondi, Riccardo Marini, Machine learning-based approach: global trends, research directions, and regulatory standpoints, Volume 4, 2021, Pages 19-29, ISSN 2666-7649, https://doi.org/10.1016/j.dsm.2021.12.002, (https://www.sciencedirect.com/science/article/pii/S2666764921000485)

12. *Sethi A. Analytics Vidhya, 2020, [online],* https://www.analyticsvidhya.com/blog/2020/04/supervised-learning-unsupervised-learning/

13. Tammy Jiang, Jaimie L. Gradus, Anthony J. Rosellini, Supervised Machine Learning: A Brief Primer, Behavior Therapy, Volume 51, Issue 5, 2020, Pages 675-687, https://doi.org/10.1016/j.beth.2020.05.002.

14. Chesta Dhingra, A Visual Guide to Decision Trees, 2020, [online] ,https://towardsdatascience.com/a-visual-guide-to-decision-trees-26606e456cbe

15. Tianqi Chen and Carlos Guestrin. 2016. XGBoost: A Scalable Tree Boosting System. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16). Association for Computing Machinery, New York, NY, USA, 785–794. https://doi.org/10.1145/2939672.2939785

16. *Prashanth Saravanan, Understanding Loss Functions in Machine Learning,* 2021, [online], https://www.section.io/engineering-education/understanding-loss-functions-in-machine-learning/#introduction

17. Scott Lundberg, 2018, SHAP documentation, [online], [Accessed on 24/12/2022] https://shap.readthedocs.io/en/latest/#:~:text=SHAP%20(SHapley%20Additive%20exPlanations)%20is,papers%20for%20details%20and%20citations).

18. Poongodi, M., Malviya, M., Kumar, C. *et al.* New York City taxi trip duration prediction using MLP and XGBoost. *Int J Syst Assur Eng Manag* **13** (Suppl 1), 16–27 (2022). https://doi.org/10.1007/s13198-021-01130-x [Accessed 27/12/2022]

19. Christophoros Antoniades, Delara Fadavi, Antoine Foba Amon Jr., Fare and Duration Prediction: A Study of New York City Taxi Rides December 16, 2016, http://cs229.stanford.edu/proj2016/report/AntoniadesFadaviFobaAmonJuniorNewYorkCityCabPricing-report.pdf (Accessed on 10/10/22)

20. Transportation Nation, NYC Ready to Launch Taxi E-Hail Technology, https://www.wnyc.org/story/284791-nyc-ready-to-launch-taxi-e-hail-technology/ [Accessed on 27/12/2022]

21. https://www.census.gov/glossary/#term_Populationestimates USA CENSUS GOV [Accessed on 27/12/2022]

22. Vishal Mendekar, Machine Learning – it's all about assumptions, 2022, https://www.kdnuggets.com/2021/02/machine-learning-assumptions.html [Accessed on 24/12/2022]

23. Abighyan, Understanding Decision Trees, 2020, https://medium.com/analytics-vidhya/understanding-decision-tree-3591922690a6 [Accessed on 24/12/2022]

24. Julien Royer, Conditional asymmetry in Power ARCH() models , Journal of Econometrics, 10.1016/j.jeconom.2021.10.013, (2022).

25. Jonathan Bartlett, The T-Test and Robustness to Non-Normality, (2013), [online] https://thestatsgeek.com/2013/09/28/the-t-test-and-robustness-to-non-normality/ [Accessed on 27/12/22]

26. Jason Brownlee, Strong Learners vs Weak Learners in Ensemble Learning, [online], (2021) https://machinelearningmastery.com/strong-learners-vs-weak-learners-for-ensemble-learning [Accessed on 12/11/22]

27. Breiman, L. Random Forests. *Machine Learning* **45**, 5–32 (2001). https://doi.org/10.1023/A:1010933404324 [Accessed on 14/11/22]

28. Kingsford, C., Salzberg, S. What are decision trees? *Nat Biotechnol* **26**, 1011–1013 (2008). https://doi.org/10.1038/nbt0908-1011 [Accessed on 15/11/2022]

29. Jerome H. Friedman, Stochastic gradient boosting, Computational Statistics & Data Analysis, Volume 38, Issue 4, 2002, https://doi.org/10.1016/S0167-9473(01)00065-2. [Accessed on 27/11/22]

30. Seger, C., 2018. An investigation of categorical variable encoding techniques in machine learning: binary versus one-hot and feature hashing. [Accessed on 29/12/22]

31. Chen, M. Keith, and Michael Sheldon. "Dynamic pricing in a labor market: Surge pricing and flexible work on the Uber platform." *Ec* 16 (2016): 455. [Accessed on 12/11/22]

32.https://github.com/zvbaid/NYC-TAXI-FARE-EDA PREDICTION/blob/main/training%20all%20models%20for%20taxi%20trip%20data-v3.ipynb