Machine Learning for Predicting Taxi Fare

Abdiakhmet Kozhamkulov, Assylbek Khalyk
Nazarbayev University

***Abstract*—This study presents a predictive modelling of New York City taxi fares using the publicly available data from NYC Taxi Limousine Commission [3] and the Kaggle NYC Yellow Taxi dataset [4]. Existing studies on non-linearity of prices [1] and taxi modeling [2] were used to built upon a new predictive model. More than 1 million trip records were used to train a predictive model. A structured machine learning pipeline was implemented. Classic ML models - XGBoost, Ridge Regression, HistGBM, and Random Forest were used for prediction. The performance of the model was assessed using RMSE and MAE in a holdout test set to prevent data leakage, achieving an RMSE of approximately 4.3 and an MAE of 1.9 in fare prediction. Feature importance visualizations reveal the highest demand during morning and evening rush hours and higher fare around airports and business districts, consistent with findings on non-linear pricing and equilibrium behavior in taxi markets [1].**

***Index Terms*—Machine Learning, Taxi Fare Prediction, Non-linear Pricing, Regression, Random Forest, Boosting.**

I. Introduction

A. Background & Motivation

Urban taxi services play a critical role in meeting on-demand transportation needs. Unlike public transportation systems, they provide point to point transportation. Early economic models have captured the demand-supply relationship of taxi markets and the spatial / geographical heterogeneity [1]. Recent work demonstrates that non-linear pricing schemes and dynamic dispatch can improve service efficiency and reduce idle times, especially in places with high demand such as airports and business districts [1].

B. Objectives

The main goal of this project is to develop and asses predictive models for individual taxi trip fares, and analyze demand patterns, enabling data-driven decisions for dynamic pricing.

C. Scope

The scope of data is from March of 2016.

II. Methodology

A. Data Collection & Description

Data was collected from official New York Taxi & Limousine Commission "Yellow Taxi Dataset" in Kaggle[3]. It has 19 features: VendorID, tpep_pickup_datetime, tpep_dropoff_datetime, passenger_count, trip_distance, pickup_longitude, pickup_latitude, RatecodeID, store_and_fwd_flag, dropoff_longitude, dropoff_latitude, payment_type, fare_amount, extra, mta_tax, tip_amount, tolls_amount, improvement_surcharge, and total_amount. Dataset contains total of 12 million rows.

B. Feature Engineering

To increase the accuracy of prediction, several new features were added. Firstly, distance related features:

1) Haversine Distance. It is the angular distance between two points on the surface of a sphere. It adds information on how far actually the trip was, not how long the route was. It was calculated by the following formula: $a = \sin^2(\frac{\Delta\phi}{2}) + \cos(\phi_1)\cos(\phi_2)\sin^2(\frac{\Delta\lambda}{2}),\ d = 2R, \text{atan2}(\sqrt{a}, \sqrt{1-a})$
2) Distance ratio. It adds information about the route directness. Higher ratio shows the trip was inside city, having many loops and turns, whereas lower ration shows the route was direct, possibly on a highway.

To avoid using latitude and longitude we have converted the coordinates to zones using the TLC zones data. Essentially, each location maps to a specific broader zone. These zone could be seen on a figure 1.

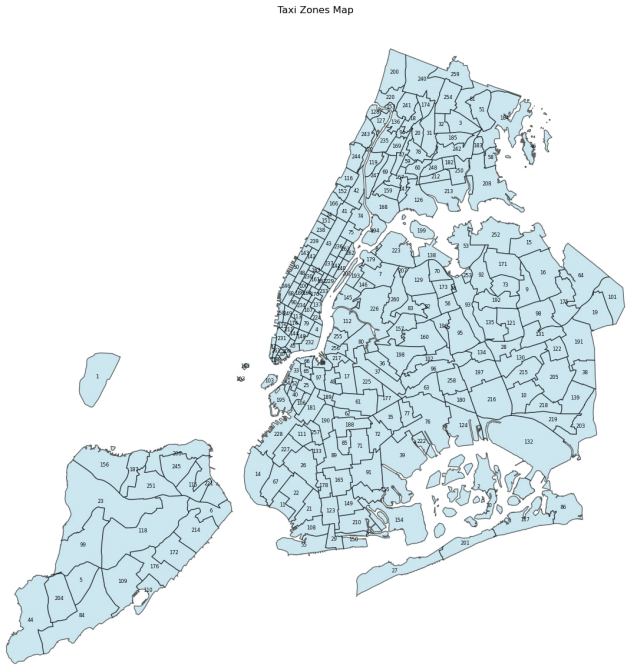

Fig. 1. TLC Taxi Zones

C. Modeling Pipeline

The new geo spatial features added are :

1) Pickup Zone (PU)
2) Dropoff Zone (DO)

3) PU_DO pair - pairs the pickup and dropoff zones
4) Is_airport. It flags if trip was to or from the airport.

Furthermore, we have also implemented new time features derived from original time column:

1) Day_of_week. It categorizes the datetime in original column to days of week (Mon, Tue, etc.)
2) Is_weekend. Boolean if the day is weekend.
3) Is_rush_hour. Boolean if the hour is considered a rush hour.
4) Hour_bin. Categorize the hour into: morning, midday, evening, night.

Most importantly, weather features were added via Open-Meteo public API [5]. To decrease computation time and cost, we have fetched data for each hour of the dataset's scope, and then simply merged the weather data with the original dataset. Following features were added:

1) Temperature. Temperature in celcius 2m above the ground.
2) Wind_speed. Wind speed in kmh 2m above the ground.
3) Precipitation.
4) Humidity.
5) Is_rainy.

D. Data Preprocessing

First, data was split into holdout test set and train-validation set to avoid any data leakage in hyperparmater and model selecting process. Once the splits are made, features and target are separated. The target was selected to be 'total_amount' which is the total amount charged for the taxi trip, including fare and tax.

Next, a correlation map of numeric features and the target was derived. The heatmap visualizes the pairwise Pearson correlation coefficients between all the numeric features and the target. It is important since it detects feature redundancy, and helps to ensure model stability by removing unnecessary information from model training.

Subsequently, we have dropped unnecessary columns that do not correlate with the target or add any new information. All other columns served as candidate predictors. Next, we have removed logically inconsistent rows, where the amount of passengers was equal to 0. Also, rows where the total trip fare was less than 2$ were removed since they are extreme outliers. Raw data was very clean, and only null values were contained in zones columns (PU, DO zones) since not all longitudes and latitudes values were inside the TLC zones. These null rows were dropped since they are out of scope of the zones and will be a noise in model training.

After all feature engineering, the new Pearson correlation heatmap is the following:

E. Model Training Pipeline

To balance model interpretability, nonlinearity capture, and computational efficiency, four supervised regressors were evaluated: Ridge regression, Random Forest, XGBoost and HistGradientBoosting.

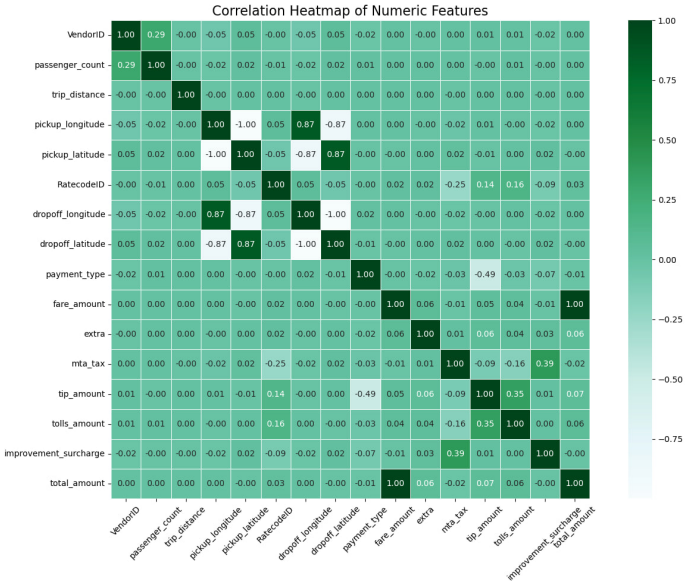

Fig. 2. Correlation Heatmap of Numeric Features

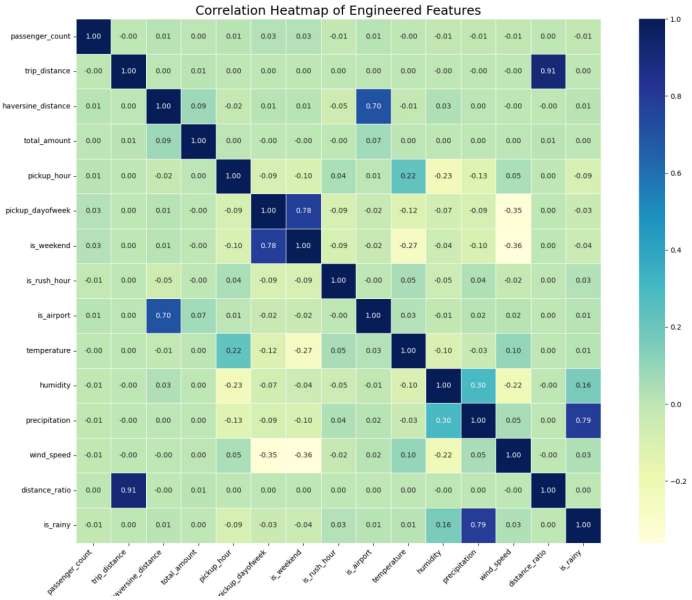

Fig. 3. Correlation Map After Feature Engineering

1) Split: A strict train-validation-test split was enforced to prevent data leakage. First, data was split into train+validation and hold-out set (20%). Then all hyperparameter tuning occurred via GridSearchCV on the training+validation folds with five-fold cross-validation, and final performance was reported against the hold-out test set.

2) Encoding data: Categorical features—pickup–dropoff zone pairs and hourly time bins were encoded to integers via an OrdinalEncoder. Ordinal Encoder was used because PU_DO pair feature is very high cardinality and OneHotEncoder would decrease performance significantly. Furthermore, tree based algorithms work well with OrdinalEncoder because

decision trees and their ensembles (Random Forest, XGBoost, HistGradientBoosting) simply test the splits. They treat each integer label as a distinct group without assuming that, for example, code "2" is twice as large as code "1.". For tree based algorithms, no scaling is needed. However, for the ridge linear model OneHotEncoder should be used since it treats codes of categories as continous values, and compares them directly as numbers. Also, linear models require scaling of numerical values. Therefore a StandartScaler was used.

- **Ridge Regression**. This linear regression algorithm uses the ridge regularization which shrinks coefficients of noisy features, stabilizing estimates when many correlated predictors exist. This model has achieved an $R^2 = 0.88$ and $RMSE = 4.27$
- **XGBoost**. This ensemble learning tree-based model further improves the model prediction accuracy and achieves an $R^2 = 0.91$ and $RMSE = 3.61$. Also, this model trains much faster than other models
- **Random Forest**. This model achieves a similar result of $R^2 = 0.91$ and $RMSE = 3.66$
- **HistGBM**. This is a modified version of Gradient Boosting and it achieves a somewhat similar performance of $R^2 = 0.91$ and $RMSE = 3.64$

III. Discussion

Metric	Random Forest	XGBoost	Ridge	HistGBM
MAE	1.93	1.85	2.33	1.91
R^2	0.91	0.91	0.88	0.91
RMSE	3.66	3.61	4.27	3.64

1) Performance: The performance comparison of models derives several insights. The large drop in MAE and rise in R^2 from Ridge to Random Forest shows the nonlinearity in fare determinants. The performance metrics R^2, $RMSE$, and MSE show that the taxi fare models are capturing the vast majority of the variation in taxi trip fare. An R^2 of 0.90 means roughly 90% of the fare fluctuations such as distance, zones, surcharges, time of day, etc. are being explained by the model. An RMSE of about $3.6 and MAE under $2 tells us that the typical prediction error is small in absolute terms.

In practical terms, this means that the system can reliably forecast fare price within a few dollars.

In real world application, this allows for dynamic price recommendations and adaptive taxi fleet deployment.

2) Feature Importances: Feature-importance analyses from Random Forest and Ridge coefficients converge on the primacy of distance metrics. Unsurprisingly, both trip_distance and haversine_distance are the first predictors of fare price. The next most significant predictor was is_airport flag. This could mean that usually airport trips are longer since they are far from the city center. However, this could also mean that there is a price discrimination for airport rides. Also, temporal features - time, is_rush hour somewhat contribute to the fare price as well. Ride prices appeared to increase in rush hours. Furthermore, even if not significantly, weather features also contributed to the ride prices. Adverse conditions modestly inflate fares perhaps due to increased idle time and longer ride times. However, it should be noted that the data was from March, which means that weather conditions were relatively modest, without extreme temperatures, storms, snowfalls, wind gusts, and other extreme conditions. Therefore weather could affect the fare price more if data was sampled in other months.

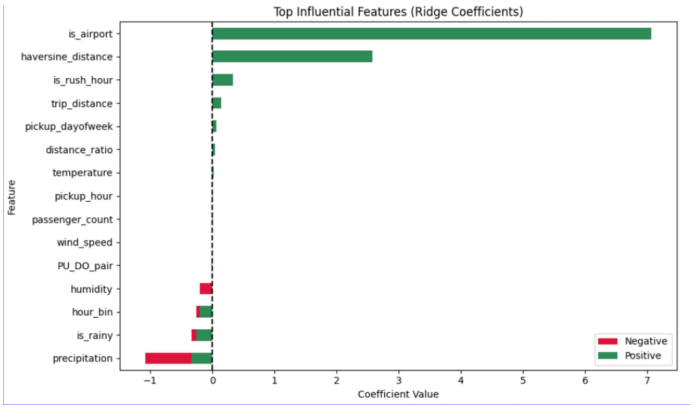

Fig. 4. Ridge Feature Importance

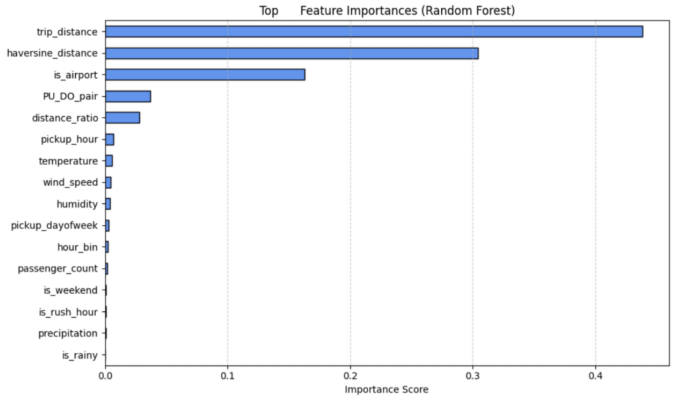

Fig. 5. Random Forest Feature Importance

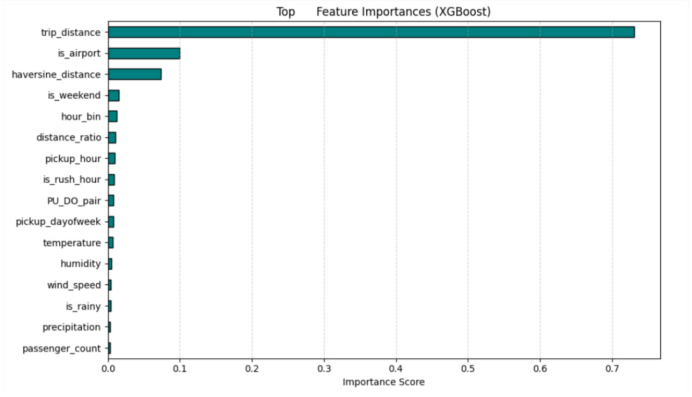

Fig. 6. XGBoost Feature Importance

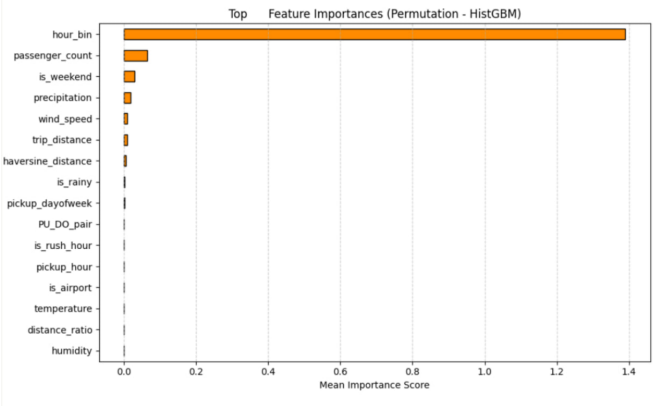

Fig. 7. HistGBM Feature Importance

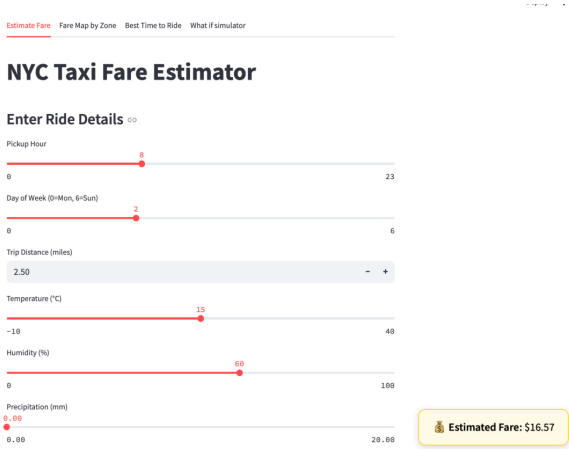

Fig. 8. Sample screenshot of Streamlit Dashboard

IV. Conclusion

This study shows that NYC Taxi fares can be predicted with a high accuracy using a structured machine learning pipeline. By predicting the fares within a few dollars on average, these models can be used for more effective dynamic-pricing strategies, resource allocation, and operational planning for taxi fleets.

V. Interactive Visualization

The Streamlit application provides an interactive interface for exploring and simulating fare estimates:

- Estimate Fare tab. Users input ride details and the model immediately outputs a fare prediction enabling real-time feedback. This is useful for seeing how variable affect ride price in real time.
- Fare Map by Zone tab. This tab visualizes predicted fares over a full 24-hour cycle for any chosen origin–destination pair. By fixing weather and distance parameters, users can see how fare fluctuates by time.
- Best Time to Ride tab Given a fixed route and optional weekend/rush-hour assumptions, the app identifies and highlights the cheapest and most expensive hours of the day.
- What-If Fare Simulator tab Users select one variable (e.g. pickup hour, trip distance, precipitation) to vary while holding all others constant. It ouptus a visualized graph of the selected variable versus the fare price.

VI. Limitations & Future work

One of the main limitations was computational power. Training models on full data (12 million rows) would require much more computational power, therefore only a subset of data (1.2 million rows) was used. Also, the data is from 2016, and inflation rates should be accounted for for using the model today. Furthermore, data was from March, which means extreme weather conditions were not captured and possibly in other months the feature importance of weather conditions could be more significant.

Future work should explore nested cross-validation for hyperparameter robustness, incorporate richer external data (traffic feeds, event calendars), and experiment with graph-based or deep-learning methods to further capture intricate urban mobility patterns.

References

[1] H. Yang, C. S. Fung, K. I. Wong, and S. C. Wong, “Nonlinear pricing of taxi services,” Transportation Research Part a Policy and Practice, vol. 44, no. 5, pp. 337–348, Apr. 2010, doi: 10.1016/j.tra.2010.03.004.
[2] J. M. Salanova, M. Estrada, G. Aifadopoulou, and E. Mitsakis, “A review of the modeling of taxi services,” Procedia - Social and Behavioral Sciences, vol. 20, pp. 150–161, Jan. 2011, doi: 10.1016/j.sbspro.2011.08.020.
[3] NYC Taxi & Limousine Commission, “TLC Trip Record Data,”https://www.nyc.gov/site/tlc/about/tlc-trip-record-data.page.
[4] Kaggle, “NYC Yellow Taxi Trip Data”.https://www.kaggle.com/datasets/elemento/nyc-yellow-taxi-trip-data.
[5] “Free Open-Source Weather API — Open-Meteo.com.” https://open-meteo.com/