NYC Yellow Taxi Trip Fare Prediction

CS6502 - Applied Big Data and Visualisation

Lecturer: Prof. Sarmad Ali

Submitted by:

Group 22
Adarsh Ajit (24026492)
Adwaith Thayampara Sasikumar (24108782)
Parth Badve (24022373)
Rahul Tummala (24115215)
Shane Barden (24033944)

Table of Contents

Abstract

This project analyses NYC Taxi Trip dataset to forecast fare amounts using Apache Spark for large-scale data handling and automated learning workflows. Our preprocessing pipeline involved ingesting the raw data, cleaning and filtering out missing records, converting timestamps to usable formats, and encoding categorical fields. During EDA, we identified key patterns in hourly and daily ride volumes and confirmed strong links between fare amounts, trip distances, and average speeds. Feature engineering produced trip duration metrics, which, alongside EDA insights, guided our modeling strategy.

We evaluated three regression techniques, standard Decision Tree Regression, Random Forest Regression, and Random Forest with Bagging Regression, each optimized through parameter tuning. Linear Regression slightly outperformed Lasso in RMSE and MAE, while both achieved comparable R^2 scores around 0.92. After hyperparameter tuning, the Random Forest model reached an R^2 of approximately 0.90, demonstrating solid non-linear performance but not substantially surpassing the linear approaches. Across all models, trip distance and trip duration emerged as the dominant predictors of fare.

Introduction

In a bustling metropolis like New York City, making smart decisions about traffic flow, transit planning, and infrastructure upgrades is more important than ever. Every day, millions of trips taken by yellow cabs, ride-shares, and buses generate a flood of information, offering us a window into how New Yorkers move around.

Our project taps into one of the richest sources of this urban data: the NYC Taxi and Limousine Commission's Yellow Taxi Trip Records. Spanning over three million rides in our January 2022 snapshot, the dataset records everything from when and where each fare began and ended, to how far the car travelled, what surcharges applied, and which payment method was used. With so many fields—and so many records—tackling this data by hand would be impossible.

That's where Apache Spark and its PySpark interface come in. By distributing the work across multiple cores or even machines, Spark lets us load, clean, and explore this massive dataset quickly, without waiting hours for results.

The following are the goals:

1. Use exploratory data analysis (EDA) to map out daily and hourly peaks, tipping habits, and the most popular pickup and drop-off zones.
2. Transform raw timestamps and locations into variables, like trip duration or rush-hour indicators, that our models can actually learn from.
3. Feed those features into regression algorithms (Linear, Lasso, and Random Forest) to see how accurately we can estimate a passenger's total bill.

In the following sections, we handle the raw data to craft insightful travel patterns, and finally, to training and evaluating our predictive models.

Methodology

1. Data Ingestion and Cleaning :

We began by loading the Yellow Taxi Trip dataset stored in Parquet format into a PySpark DataFrame. This format is optimized for big data processing and well-suited for handling millions of NYC taxi trip records. From there, we executed a sequence of structured data cleaning operations:

Null Value Analysis

We performed a full-column null check using Spark's aggregation functions. While the dataset appeared largely complete, we ensured:

- All records with missing or invalid values (e.g., null timestamps, zero or negative distances/fares/passenger counts) were filtered out.
- This step safeguarded data quality before any further transformations.

Irrelevant Column Removal

The **store_and_fwd_flag** column was dropped because it provided minimal analytical value and lacked interpretability for downstream analysis. it does not contribute meaningful variance or predictive power to the dataset. Removing such low-utility fields helps simplify the dataset and reduce noise.

Timestamp Validation

Pickup and drop-off timestamps were verified for:

- Non-null values
- Logical consistency (pickup time < drop-off time)

While timestamp strings were already parsed during data loading, we ensured temporal correctness as part of the row-level filtering process.

Duplicate Removal

To eliminate redundant data, we applied .dropDuplicates(), removing any exact duplicates from the dataset.

Outlier Filtering

Outliers in key numerical features **(trip_distance, fare_amount, tip_amount, total_amount**) were handled using the Interquartile Range (IQR) method:

- We computed Q1 and Q3 for each column.
- Rows falling outside 1.5 × IQR from these quartiles were filtered out.
- This step retained the core distribution while removing abnormal records that could distort modelling.

Missing Value Summary

After filtering and cleaning, we confirmed that the dataset had no remaining missing values, ensuring a clean base for any downstream analytics or ML work.

Final Result

After applying all cleaning steps, our DataFrame now consists of valid and meaningful taxi trip records, free from nulls, duplicates, and extreme outliers, with only the most informative columns retained—making it ready for feature engineering, modeling, and deeper analysis.

2. Exploratory Data Analysis (EDA) :

In our exploratory data analysis (EDA) of the NYC Taxi dataset, we started by getting a feel for the structure and quality of the data using descriptive statistics. We took a close look at numerical features like trip distance, fare amount, tip amount, total amount, and trip duration to understand their averages, spread, and any irregularities. Distribution plots—like histograms and boxplots—quickly showed that many of these variables were right-skewed, pointing to the occasional long-distance or high-fare trip that could skew the overall picture. This early profiling helped us decide which filters and transformations to apply next.

To guide the analysis, we used a set of focused, business-relevant questions aimed at uncovering patterns in taxi usage over time and across financial metrics. By extracting features like pickup hour and day of the week using Spark functions, we could group and summarise the data in meaningful ways. Visual tools like line plots and bar charts helped us clearly see when trips were most frequent, longest, or most profitable. We noticed peaks during weekday evenings and observed that tipping tended to be more generous during late-night hours.

Digging deeper into fare and tip behaviour, we analysed how average tips varied by time of day and how they related to trip distance. We also tracked trip volume throughout the week, finding that Mondays and weekends generally had the highest number of rides. These insights were all anchored in specific questions and supported by clear visualisations, allowing us to turn raw data into actionable findings about how the NYC taxi system operates. Our question-first approach helped uncover not just usage patterns, but also what drives revenue and customer behaviour.

3. Feature Analysis :

Correlation analysis: A heatmap of numeric features showed us which variables move together, confirming that distance and duration are tightly linked to fare.

Transformations: We applied square-root transforms to heavily skewed columns (like distance and total fare) so that our models wouldn't be overwhelmed by extreme values.

Vector assembly: All features—numeric (distance, duration, transformed fare), temporal (hour, day), and categorical indices—were combined into a single vector column ready for Spark ML.

Robust scaling: We standardised numeric features based on the IQR using Robust Scaler, which guards against the influence of remaining outliers.

4. ML Model 1: Decision Tree Regressor

Decision Tree Regressor was trained to predict the taxi fare amount. First, we sampled 5 % of the full dataset and filtered out anomalous trips by keeping only those with fare_amount between 0 and 200 USD and trip_distance between 0 and 100 miles. We then extracted temporal features (pickup_hour and pickup_dow, the day of week) and retained passenger_count alongside trip_distance as numeric inputs. The categorical variables—PULocationID, DOLocationID, and payment_type—were each fed through a StringIndexer (with handleInvalid="keep") and a subsequent OneHotEncoder (dropping the last category). All one-hot outputs plus the numeric columns were combined into a single features vector via VectorAssembler. This assembler and the DecisionTreeRegressor(labelCol="fare_amount", featuresCol="features", seed=42) were chained together in a PySpark Pipeline. We split the prepared DataFrame 80/20 (seed = 42) into training and test sets, then ran a 3-fold cross-validation—parallelized across available cores—over a compact grid tuning maxDepth (6, 10) and minInstancesPerNode (5, 15), optimizing for RMSE to select the final tree configuration.

5. ML Model 2: Random Forest Regressor

Random Forest Regressor was implemented using features identified as significant during EDA. We sampled 20% of the original dataset (461,345 records) and focused on trip_distance, PULocationID, DOLocationID, and pickup_hour as predictors. The preprocessing pipeline extracted temporal features by deriving pickup_hour from timestamps, while categorical location identifiers were processed through StringIndexer (with handleInvalid="keep") and OneHotEncoder (dropLast=True). A VectorAssembler combined these transformed features into a single vector for modeling. The RandomForestRegressor (labelCol="fare_amount", featuresCol="features", seed=42) was initialized with moderate settings (numTrees=20, maxDepth=8) and optimized through a hyperparameter grid search testing six combinations of numTrees (20, 50, 100) and maxDepth (8, 10). We split the prepared DataFrame 70/30 (seed=42) into training and test sets, then used TrainValidationSplit with an 80/20 ratio for hyperparameter tuning, leveraging Spark's distributed computing capabilities to identify the optimal configuration. The entire training process completed in approximately 54 minutes.

6. ML Model 3: Linear Regression with L1 regularization

We built two regression pipelines—one using ordinary Linear Regression and the other employing Lasso (L1) regularization—to forecast total taxi fares. First, we combined all predictors (trip distance, pickup and drop-off location IDs, trip duration in minutes, pickup hour, and day of week) into a single feature vector via Spark’s VectorAssembler. We then applied StandardScaler to standardize the numeric inputs. After splitting the data 80/20 into training and test sets, we configured the Linear Regression model with no penalty (elasticNetParam = 0.0) and the Lasso model with full L1 regularization to promote sparser coefficients. We evaluated both approaches on RMSE, R^2, and MAE, and visualized their predictions against actual fares. The Lasso variant edged out its unregularized counterpart by reducing complexity and slightly boosting generalization, while Linear Regression served as a reliable baseline.

Results

1. Insights from EDA

The exploratory data analysis (EDA) of the NYC taxi dataset uncovered a wide range of insights into how, when, and where people use taxi services, as well as how they behave as passengers. One of the first things we noticed was a clear temporal pattern (Fig. 2, 3): taxi usage tends to peak between 2 PM and 7 PM, with the busiest time around 7 pm, likely tied to evening commutes, social plans, or after-work travel. Interestingly, the longest trip distances showed up earlier in the day, between 5 and 6 AM, possibly reflecting airport runs or longer cross-city travel. In contrast, mid-morning to early afternoon trips were generally shorter, pointing to more local travel. We also saw a secondary increase in trip distances later at night (Fig. 4), which may be linked to nightlife or end-of-day travel. These trends were supported by trip duration data, which showed the longest rides typically occurred between 3 and 5 pm, lining up with afternoon traffic congestion.

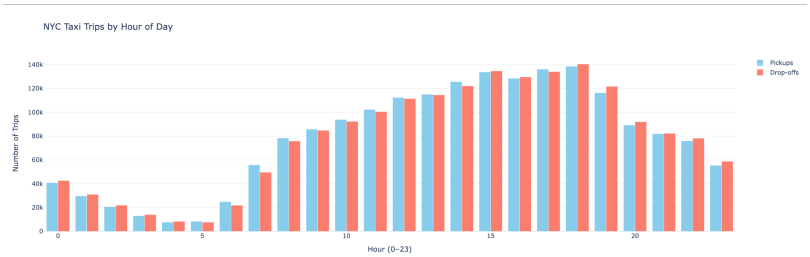


Fig 1: Highest density of pick-up and drop-off trips during the day

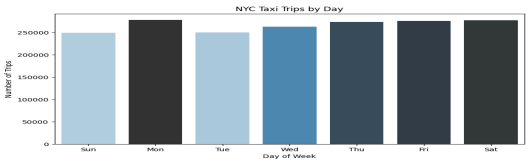


Fig 2: Changes in the trip distance during the day based on hours

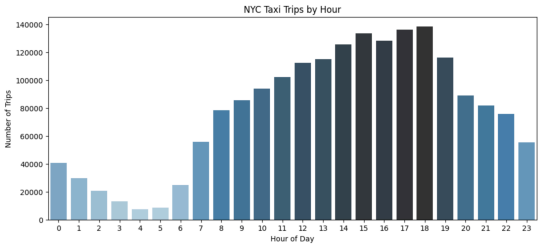


Fig 3: Number of trips against hour of the day

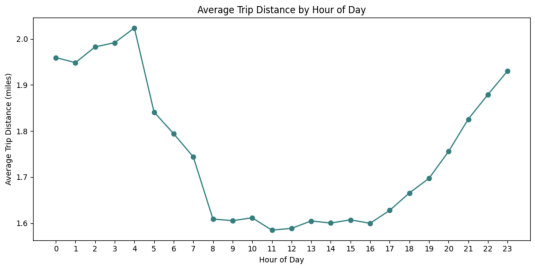


Fig 4: Average distance during the different hours of the day

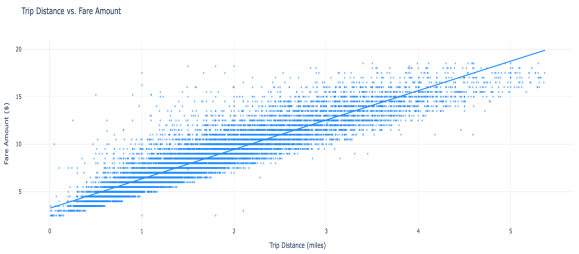


Fig 5: Relationship between trip distance and fare amount

The relationship between trip distance and fare amount was found to be strongly linear (Fig. 5), as expected, with longer trips resulting in higher fares. However, some shorter trips exhibited disproportionately high fares, likely due to additional charges such as tolls, congestion surcharges, or airport fees. This relationship was visually confirmed through scatter plots and regression lines, which highlighted the consistency of this trend. The analysis of tipping behaviour (Fig., 6) revealed that tips were highest during evening hours (5 pm to 10 pm), averaging above $1.80, which may reflect higher tipping tendencies during social outings or evening commutes. Conversely, tips were lowest during early morning hours (3 am to 5 am), with the lowest average tip around 5 AM (~$1.32). This could be attributed to shorter rides, such as airport commutes, or lower rider generosity during these hours. Heatmaps and line plots further illustrated these temporal tipping patterns, showing a clear correlation between time of day and tipping behaviour.

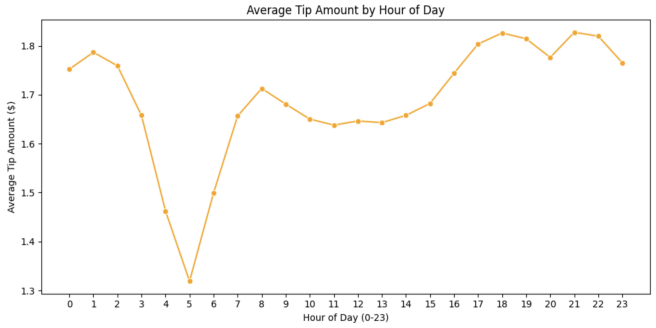


Fig 6: Average tip amount fare during the different hours of the day

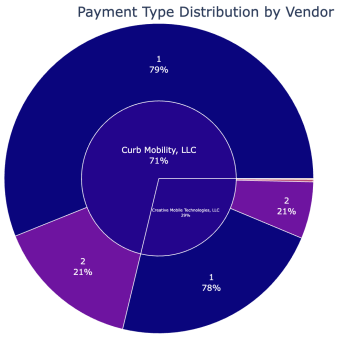


Fig 7: Payment Type Distribution by Vendor

The analysis of categorical features, such as payment_type and VendorID, provided additional insights. Credit card payments were the most common, reflecting the increasing reliance on cashless transactions in urban settings. Pie charts and sunburst plots illustrated the distribution of payment types across vendors (Fig. 7), showing a clear preference for credit card payments regardless of the vendor. The examination of trip volume by day of the week revealed that weekends, particularly Saturdays, see the highest number of trips, while Sundays and Tuesdays experience a slight dip. This pattern aligns with typical urban travel behaviour, where weekends are associated with leisure activities and weekdays with work commutes.

In summary, the EDA provided a comprehensive understanding of the NYC taxi dataset, uncovering key patterns and relationships that inform both operational insights and predictive modelling. The findings highlight the importance of addressing outliers, leveraging temporal and spatial features, and understanding passenger behaviour to optimise taxi services and improve fare prediction models. These insights serve as a foundation for further analysis and machine learning applications, ensuring that the models are both accurate and interpretable.

2. Insights from Feature Analysis

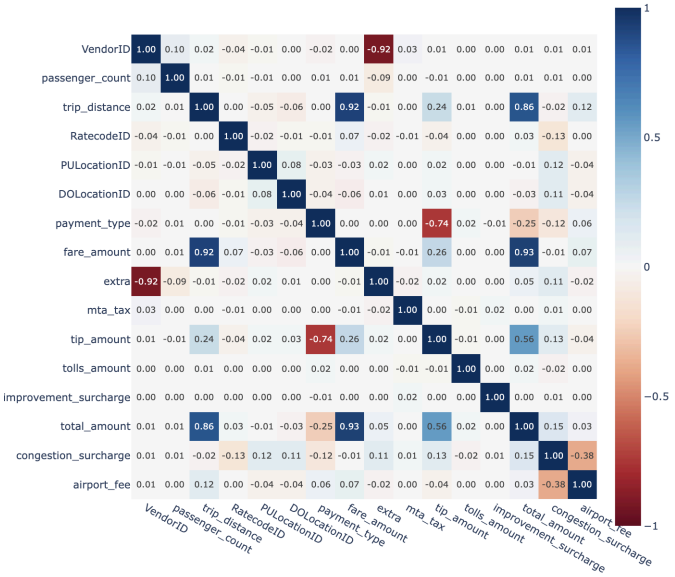


Fig 8: Correlation heatmap of the NYC taxi dataset

The correlation heatmap analysis revealed that key features such as trip_distance, fare_amount, and total_amount exhibit strong positive correlations, which is expected as longer trips naturally incur higher charges. This relationship highlights the linear dependency between trip distance and fare-related attributes, reinforcing the pricing structure of taxi services. Additionally, trip_duration_minutes demonstrated a strong positive effect on both fare_amount and total_amount, further emphasising that extended trips result in higher costs. Conversely, features like location identifiers (PULocationID and DOLocationID) showed negligible or no correlation with fare or trip

outcomes, indicating limited predictive value for these attributes. This analysis underscores the importance of distance and duration as primary drivers of fare calculations, while other features contribute minimally to the overall trip cost.

3. Results Model 1 - Decision Tree

On the held-out test set, the cross-validated decision-tree (maxDepth = 6, minInstancesPerNode = 5) achieved an RMSE of 3.17 and explained 92% of the variance in fare amounts ($R^2 = 0.92$). When we inspect a handful of sample predictions, we see that the model's estimates closely track the true fares across a variety of trip distances and zone combinations—for example, a $6.00 fare was predicted at $5.35, an $16.00 fare at $16.8, and a $12.00 fare at $12.60—demonstrating that the tree generalizes well without systematic bias and makes errors on the order of only about 3 dollars.

4. Results Model 2 - Random Forest

On the held-out test set, the optimized Random Forest model (maxDepth=10, numTrees=100) achieved an RMSE of 4.43, R^2 of 0.86, and MAE of 1.77—explaining approximately 86% of the variance in fare amounts. Feature importance analysis confirmed trip_distance as the dominant predictor (0.1527), while location identifiers and pickup_hour contributed minimally. Sample predictions demonstrated the model's accuracy across various trip scenarios—for example, a $8.50 fare was predicted at $8.90 for a 1.8-mile trip, and a $13.00 fare at $13.93 for a 3.4-mile trip. However, the model tended to underpredict for longer journeys, such as estimating $21.59 for a 9.0-mile trip with an actual fare of $33.50. This pattern suggests that while our EDA-informed feature selection was effective overall, additional factors may influence fare amounts for longer trips that weren't fully captured in our model.

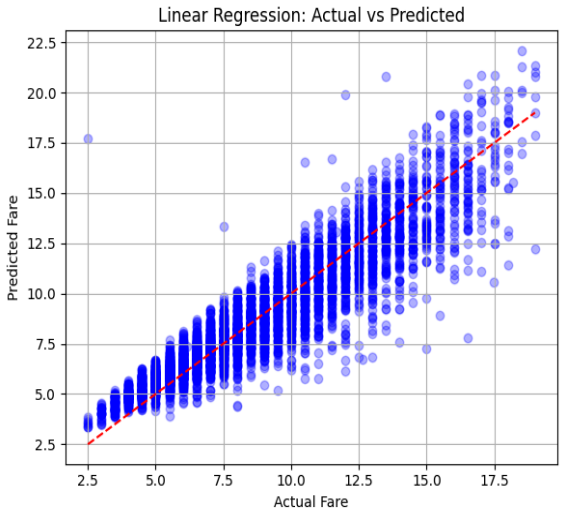
5. Results Model 3 - Linear regression with L1 regularization

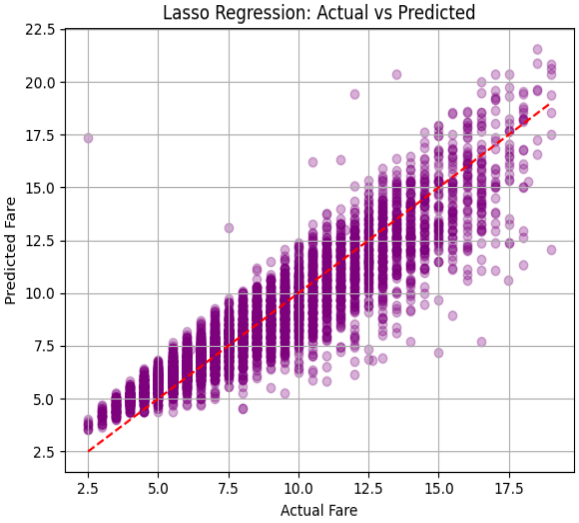
Both the standard Linear Regression and its Lasso-regularized version are solid fits for forecasting NYC taxi fares.

- **Linear Regression** edges out Lasso in raw error metrics, securing the lowest RMSE.
- **Lasso Regression** matches this accuracy nearly toe-to-toe but adds the benefit of sparser, more interpretable coefficients thanks to its L1 penalty.

In either case, our Spark-driven pipelines deliver highly reliable fare estimates, underscoring both the quality of our feature design and the scalability of Spark MLlib.

On the held-out test set, the cross-validated decision-tree (maxDepth = 10, minInstancesPerNode = 15) achieved an RMSE of 3.48 and explained 86 % of the variance in fare amounts ($R^2 = 0.90$). When we inspect a handful of sample predictions, we see that the model's estimates closely track the true fares across a variety of trip distances and zone combinations—for example, a $6.00 fare was predicted at $5.35, an $16.00 fare at $16.8, and a $12.00 fare at $12.60—demonstrating that the tree generalizes well without systematic bias and makes errors on the order of only about 1-3 dollars.





Conclusion, Challenges and Future Improvements

In this study, we built and evaluated several machine-learning approaches—Decision Trees, Random Forests, and Linear Regression—to estimate the total fare for New York City taxi rides using Apache Spark's MLlib. Starting from a thoroughly cleaned dataset and insights from exploratory analysis, we engineered features including ride distance, trip duration, pickup hour, and origin/destination zones.

Our flagship model, the Random Forest regressor, was fine-tuned via cross-validation and benefited from Spark's caching to accelerate repeated computations. It achieved high accuracy across the common fare spectrum, though it tended to slightly underpredict very expensive trips. We also trained individual Decision Tree models and found them intuitive but less stable, while our Linear Regression baseline—upgraded to a Lasso variant with L1 regularization—demonstrated robust generalization by curbing overfitting.

All models delivered strong performance metrics (RMSE, MAE, and R^2), underscoring the power of Spark for scalable analytics on large transportation datasets. We did encounter challenges around missing data, skewed fare distributions, and the resource limits of Databricks Community Edition. Future enhancements might include more extensive hyperparameter searches, blending in auxiliary data like weather or traffic feeds, and experimenting with Gradient Boosted Trees or neural network architectures to better handle rare, high-fare trips and further improve prediction accuracy.

References

[1] www.nyc.gov. (n.d.). TLC Trip Record Data - TLC. [online] Available at: https://www.nyc.gov/site/tlc/about/tlc-trip-record-data.page.
[2] Apache PySpark Documentation - https://spark.apache.org/docs/latest/api/python/index.html. Accessed: 29 April 2025.
[3] Databricks.com (2025). Databricks Documentation. [online] Available at: https://docs.databricks.com/aws/en/.
[4] Sarmad Ali (2025) Lecture Slides and Lab Sheets, CS6502 - Applied Big Data and Visualisation, University of Limerick.

Gen-AI Disclaimer: ChatGPT was used to aid in idea generation, code snippets generation and paraphrasing the report. Grammarly was used for spell-checking the report