New York City Yellow Taxi Data Traffic Analysis Using Spark

Adrian Cican
Faculty of Engineering
University of Porto, Portugal
Master in Data Science and Engineering
Email: up202400287@up.pt

Francisco Pinto
Faculty of Engineering
University of Porto, Portugal
Master in Data Science and Engineering
Email: up199504009@up.pt

João Matos
Faculty of Engineering
University of Porto, Portugal
Master in Data Science and Engineering
Email: up202400124@up.pt

João Soares
Faculty of Engineering
University of Porto, Portugal
Master in Data Science and Engineering
Email: up202105364@up.pt

João Vieira
Faculty of Engineering
University of Porto, Portugal
Master in Data Science and Engineering
Email: up202107689@up.pt

Manuel Silva
Faculty of Engineering
University of Porto, Portugal
Master in Data Science and Engineering
Email: up202108874@up.pt

Abstract—**This paper presents a data engineering and analytics project that investigates traffic and mobility patterns in New York City using the Yellow Taxi dataset. Using Apache Spark's distributed computing capabilities, we construct a complete data pipeline to perform large-scale data profiling, analytical queries, and machine learning tasks. The exploratory queries include temporal trends, seasonal patterns, geographic distribution, analysis of payment methods and correlation analysis. On the predictive side, we develop models to estimate fare and tip amounts, recommend drivers, and cluster trips based on profitability. Furthermore, we assess the runtime performance and scalability of these tasks by experimenting with various Spark cluster configurations and measuring key performance indicators such as training duration, memory usage, and regression metrics. The paper is organized into five main sections: introduction, data profiling, query and learning tasks on the dataset, runtime and scalability analysis, and conclusion. The overarching goal is to evaluate Spark's effectiveness in processing large-scale data and to provide insight into how system configurations impact analytical and predictive workloads.**

I. Introduction

With the exponential growth of urban mobility data, the analysis of transportation systems has become increasingly data-driven. This project focuses on exploring the New York City Yellow Taxi dataset through Apache Spark, a distributed computing framework designed for scalable data processing and analytics. The study addresses two key objectives: (1) to uncover patterns and relevant insights from the data using advanced query and machine learning techniques, and (2) to evaluate the runtime performance and scalability of these tasks across various Spark configurations.

On the data analysis front, we perform a range of query-based exploratory tasks to examine temporal and behavioral trends in taxi usage. These include temporal trends, such as time series analysis, monthly seasonality, day-of-week patterns, and hourly trip distributions; geographic distribution, including identification of the most frequent pickup and drop-off zones as well as the most common taxi routes; payment methods, focusing on the distribution and evolving trends in payment types; and correlation analysis, specifically examining the relationship between trip distance and trip duration. Collectively, these queries aim to characterize usage behavior across time and geography, revealing patterns such as rush hour activity, customer preferences, and high-demand zones.

In terms of predictive analytics, we implement multiple ML tasks, including fare amount prediction, tip value prediction, driver recommendations, and trip profitability clustering. These models serve to demonstrate Spark's ability to support end-to-end machine learning pipelines at scale.

To assess performance and scalability, we measure key indicators such as execution time, memory usage, and model evaluation metrics (RMSE and R^2 for regression models). Experiments are conducted on a Google Cloud Dataproc cluster under varying configurations: different numbers of executor instances, cores, and memory allocations, as well as varying the number of shuffle partitions. We analyze how these parameters affect both training duration and model performance for algorithms such as Linear Regression and Random Forest.

In general, this project not only extracts valuable information from a large-scale urban mobility dataset, but also provides an empirical evaluation of Spark's capabilities in real-world data engineering and machine learning workloads.

II. Related work

Since the NYC Taxi and Limousine Commission released detailed trip records, researchers have mined them to infer spatio-temporal demand, travel times, and pricing dynamics. Early single-node studies (e.g., Ferreira et al., 2013) profiled 2013 flows, while deep-learning approaches like those by Zhang et al. (2017) later fused weather and event feeds for demand prediction. More recent work explored fare estimation (e.g., through Kaggle competitions), ride-share matching, and driver repositioning (Camerer et al., 1997; Buchholz, 2022), yet most pipelines ran on limited snapshots and non-distributed stacks. Spark's scalability has been demonstrated for mobility data and for fare regression benchmarks (Shabib et al., 2024), but the effect of executor memory, cores, and shuffle partitions remains workload-specific, with optimization challenges highlighted by research on Spark tuning (Lyu et al., 2024). Unlike prior work, we analyze the post-pandemic 2023-2025 yellow-cab corpus, combine descriptive SQL analytics with multiple Spark MLlib models, and empirically tune cluster parameters on Google Cloud Dataproc, providing a unified, end-to-end assessment of Spark for contemporary urban-mobility applications.

III. Dataset Profiling

The dataset used in this project consists of NYC Yellow Taxi trip records, spanning from January 2023 to February 2025. It comprises **86,532,715 rows** and **22 attributes**, where each row represents a single taxi trip. The features include numerical and categorical variables related to timestamps, trip metadata, geographic locations, and fare components. These attributes consist of 2 timestamp fields, 4 long, 13 double, 2 integer, and 1 string type.

The data was initially ingested and explored using Apache Spark's RDD API for fine-grained transformation and error handling.

Each Parquet file was loaded using `spark.read.parquet()` and transformed with DataFrame operations such as `withColumn()`, `select()`, and `cast()` to split composite fields, convert data types, and detect and resolve schema inconsistencies.

Invalid or malformed rows—such as records with missing essential fields, negative fare amounts, zero-distance trips with non-zero charges, or illogical timestamps—were detected and removed with a `filter()` predicate.

Profiling revealed a high occurrence of missing values: 10 of the 22 attributes had missing values, and 96.8% of records had at least one missing field. To address this, missing values were filled following the dataset dictionary guidelines and industry best practices, ensuring consistency and analytical integrity.

Additionally, `distinct()` was applied to ensure no duplicate records. No duplicates were found.

The dataset includes the following attributes:

Field Name	**Description**
VendorID	Taxi provider ID
tpep_pickup_datetime	Trip start timestamp
tpep_dropoff_datetime	Trip end timestamp
passenger_count	Number of passengers
trip_distance	Distance in miles
RatecodeID	Rate classification code
store_and_fwd_flag	Whether the trip was temporarily stored (Y/N)
PULocationID	Pickup location (TLC zone ID)
DOLocationID	Dropoff location (TLC zone ID)
payment_type	Payment method
fare_amount	The time-and-distance fare by the meter.
extra	Additional charges
mta_tax	MTA state tax
tip_amount	Tip amount (card only)
tolls_amount	Total tolls paid
improvement_surcharge	Fixed startup fee
total_amount	Total charged amount, without cash tips
congestion_surcharge	Total NYS congestion fee
airport_fee	Airport pickup surcharge
cbd_congestion_fee	MTA congestion zone fee

TABLE I: Attributes in the NYC Yellow Taxi dataset

To support later querying and modeling tasks, several new features were engineered from existing fields:

- `trip_duration`: Duration of the trip in minutes.
- `pickup_hour`, `pickup_day_of_week`, `pickup_month`: Extracted from the pickup timestamp.
- `is_rush_hour`: Boolean indicator of whether the trip occurred during NYC peak traffic times.
- `fare_per_mile`: Fare amount divided by the trip distance in miles, representing the cost efficiency of the trip.
- `is_weekend`: Boolean indicator of whether the trip took place on a Saturday or Sunday.

These features were computed using `map()` transformations and Spark's built-in datetime functions.

After cleaning and feature engineering, the final dataset contains **85,086,541** records and **27 attributes**. This preprocessed dataset was then cached and reused for the querying, modeling and performance analysis stages of the project.

IV. Query and Learning Tasks on the Dataset

In this section, we describe the two main categories of work carried out on the dataset: (1) **Query Tasks**, which helped us explore and understand patterns in the NYC Yellow Taxi data, and (2) **Learning Tasks**, where we applied machine learning techniques to derive insights and build predictive models.

A. Query Tasks

To investigate temporal, spatial, and behavioral trends in NYC Yellow Taxi operations from January 2023 to February 2025, we conducted a comprehensive series of queries using PySpark, utilizing both the Spark SQL and DataFrame APIs. Most queries executed efficiently within 10–20 seconds. Additionally, we experimented with RDDs, but performance and ease of use were inferior compared to Spark SQL and the Data Frame API.

For each analytical question, we created visualizations to reveal key patterns and insights. Below, we summarize each query category, the corresponding visualizations used, and the insights derived.

1. Temporal Trends:

- **Time Series Analysis:** Monthly ride volumes across the 26-month window were visualized using line charts (fig 1), revealing temporal patterns.
  Clear seasonal variations were observed, with distinct peaks during holiday periods and major events.
  Additional time-series plots were used to track average fare per trip, total monthly revenue, average trip distance, and average tip amount (fig 2).

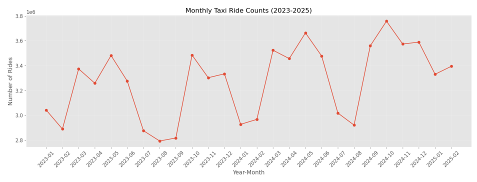


Fig. 1: Monthly ride count from January 2023 to February 2025.

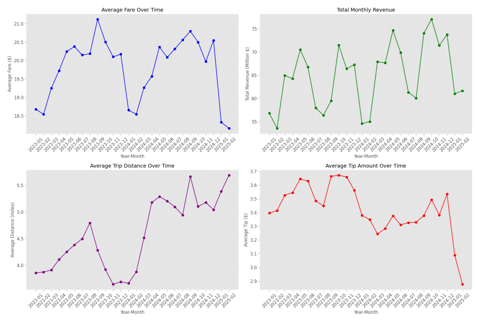


Fig. 2: Monthly average fare, distance, revenue, and tip amount over time.

2. Seasonal Patterns:

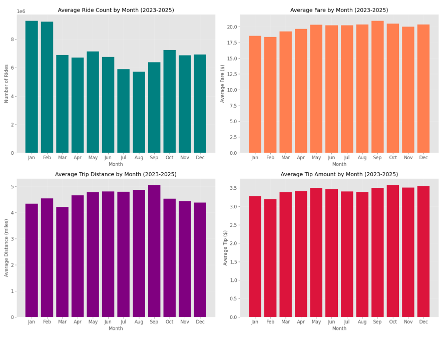
- **Monthly Patterns:** Aggregated trip data by month (fig 3a) revealed seasonal peaks, particularly at the beginning of the year.
- **Day-of-Week Trends:** Average daily ride volumes by weekday (fig 3b) highlighted consistent weekday patterns.
- **Hourly Distribution:** Trips grouped by pickup hour (fig 3c) showed pronounced usage spikes during morning (6AM) and evening rush hours (around 6PM), with significantly lower activity overnight.
  Additionally, average trip durations varied significantly by day of the week and time of day fig 12).

3. Geographic Distribution:

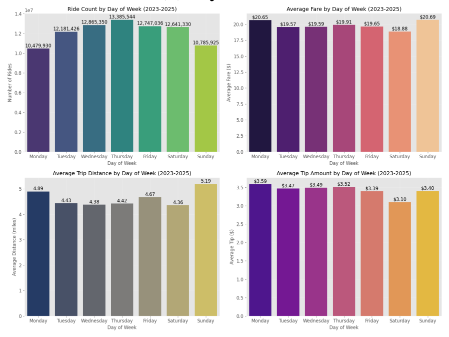
- **Top Pickup and Drop-off Zones:** By joining trip data with the Taxi Zone Lookup Table, we identified the top 20 pickup (fig 4a) and drop-off (fig 4b) zones by trip frequency. Additional visualizations revealed the zones generating the most total revenue, as well as those with the highest average revenue per trip (fig 8).
  Additionally, certain pickup zones consistently showed longer ride durations (fig 9), and the weekday patterns for the top 5 longest-duration pickup zones were further analyzed (fig 13).
- **Popular Routes:** We ranked the top 20 origin-destination pairs both by number of trips (fig 5), total revenue generated (fig 10) and average revenue per trip (fig 8b), allowing us to distinguish between volume-based and economically significant routes.

The most profitable taxi trips originate from JFK and LaGuardia airports, Times Square, and Midtown Manhattan, with frequent routes in the Upper East Side North and South.

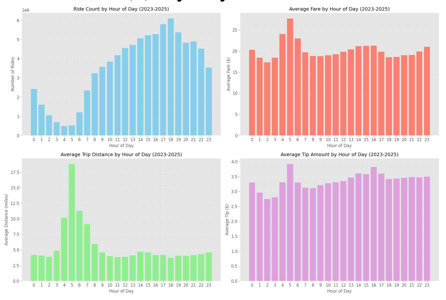
4. Payment Methods:



(a) By month



(b) By day of week



(c) By hour

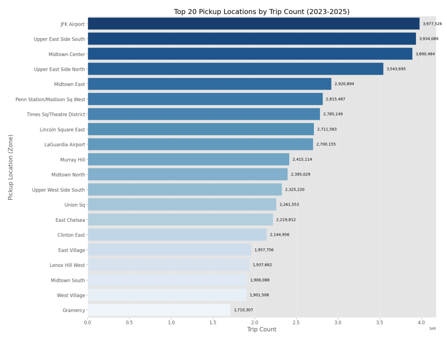
Fig. 3: Seasonal and daily usage patterns.

- **Payment Distribution:**
  The bar chart (fig 6) shows that credit cards were the dominant payment method (76.5%), followed by cash (14.6%). Time-series trends (fig 14) further highlighted changes in payment preferences over the study period.

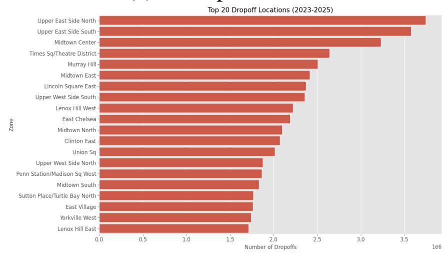
5. Correlation Analysis:

- **Trip Distance vs. Duration:** To examine the relationship between trip distance and duration, we used both scatter and time-series plots (fig 7).
  The scatter plot with a fitted trendline (fig 7a) revealed a strong positive linear correlation, longer trips tend to take more time. Temporal trends in average distance and duration were visualized monthly to observe long-term shifts (fig 7b).

By integrating PySpark SQL with dynamic visualizations, we were able to uncover actionable insights from a large-scale NYC taxi dataset. These graphical analyses highlighted key usage patterns, temporal shifts, geographic hotspots, and



(a) Pickup locations



(b) Drop-off locations

Fig. 4: Top 20 taxi locations by number of trips

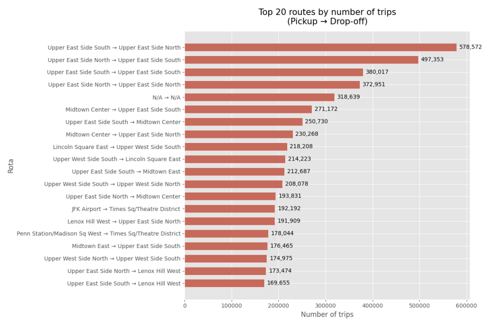


Fig. 5: Top 20 routes by number of trips

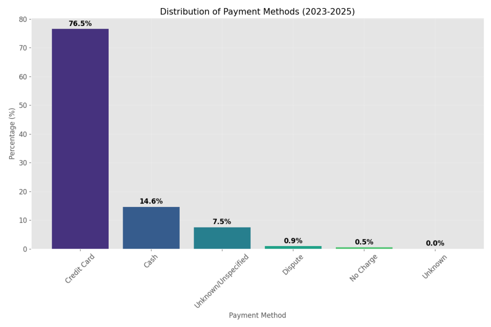
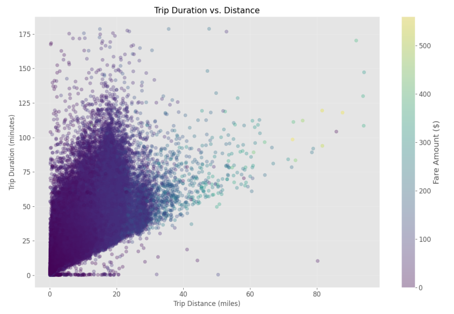
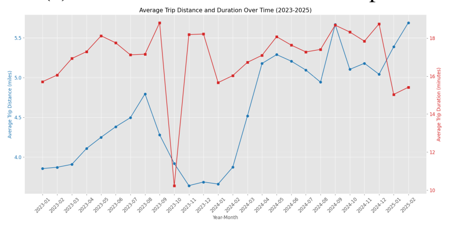


Fig. 6: Payment distribution



(a) Distance vs. Duration scatter plot



(b) Monthly averages over time

Fig. 7: Relationship between trip distance and duration.

behavioral correlations. Such findings can inform stakeholders in urban planning, transportation policy, fleet management, and customer service optimization.

B. Learning Tasks

We implemented several learning tasks using Spark MLlib to build predictive models and decision tools. These include regression and clustering techniques. The key tasks executed are as follows:

1. Predict Fare Amount: To estimate taxi fare amounts, we considered the standard pricing model used by New York City taxis, which includes:

- \$3.00 — Initial base fare when the trip begins.
- \$0.70 per 1/5 mile when traveling over 12 mph.
- \$0.70 per 60 seconds when traveling at 12 mph or less, or when stopped in traffic.

To enhance fare transparency and reduce the risk of overcharging, we developed a predictive model using a Random Forest Regressor. The model was trained on features including trip distance, duration, pickup and drop-off locations, rate code ID, pickup hour, passenger count, and day of the week.

- **Results:** RMSE: 2.69, MAE: 0.95, R^2: 0.97
- **Most important features:** *trip_distance* (0.45), *trip_duration* (0.28), *pickup_location* (0.104), *o*ther features had marginal impact (less than 0.1).

2. Predict Tip Amount: Usually, customers give tips to the drivers, and that information is recorded. In this task, we constructed a regression model to estimate tip amounts based on various trip and contextual features.

Another Random Forest model was built using a broader set of features: fare Amount, trip distance, duration, pickup/dropoff location, rate code id, pickup hour, vendor id, passenger count and day of the week. In addition, we engineered features such as `is_weekend` and `is_rush_hour`.

- **Results:** RMSE: 2.12, MAE: 1.02, R^2: 0.68
- **Most important features:** *fare_amount* (0.31), *trip_distance* (0.14), *trip_duration* (0.12), and locations.

3. Driver Recommendations: To support drivers in maximizing earnings, we analyzed historical trip data to identify the most profitable zones and time periods. Since drivers typically do not know the drop-off location, trip duration, or traffic conditions in advance, these variables were excluded from the training data.

- **Results:** RMSE: $13.52, R^2: 0.64

4. Trip Profitability Clustering: To categorize trips based on profitability, we applied K-means clustering using two key metrics: revenue per minute and revenue per mile. Trips were grouped into three distinct groups (low, medium, high), allowing operators and drivers to better understand which types of rides are economically efficient.

These tasks enabled us to extract meaningful knowledge from the dataset and illustrate how predictive models and data-driven recommendations can enhance the operation of urban taxi services.

V. Runtime and Scalability Analysis

This section evaluates the scalability and runtime performance of our data processing and model training pipelines. We performed controlled experiments by varying key Spark configuration parameters and analyzing their impact on job execution and model training efficiency.

A. Methodology

To assess the impact of system configurations on performance, we measured the following metrics:

- Job execution time
- Model training duration
- Resource utilization (CPU and memory consumption)
- Model performance (RMSE, R^2)

The default Spark session configuration was defined as follows:

- **Executor Instances:** 2 (number of distributed workers)
- **Executor Cores:** 1 (parallel tasks per executor)
- **Executor Memory:** 3 GB (RAM per executor)
- **Shuffle Partitions:** 50 (number of partitions during operations like joins or aggregations)

We conducted experiments by varying the following Spark parameters:

- **Executor Instances:** 1, 2, 3, 4
- **Executor Cores:** 1, 2
- **Executor Memory:** 2 GB, 3 GB, 4 GB
- **Shuffle Partitions:** 50, 100, 200

Two regression models were trained using the same set of predictors: `PULocationID`, `DOLocationID`, `pickup_hour`, `pickup_day_of_week`, and `pickup_month`, with the target variable being `total_amount`.

- **Linear Regression:** Trained on the full dataset (approximately 1.45 GB).
- **Random Forest Regression:** Trained on a representative subset (187 MB, 10 million rows) with 20 trees and a maximum depth of 10.

The experiments were executed on a Google Dataproc cluster with the following configuration:

- **Region:** `europe-west2`
- **Master Node:** 1 node (`n4-standard-2`, 100 GB RAM)
- **Worker Nodes:** 2 nodes (`n4-standard-2`, 200 GB RAM each)

B. Results

The results presented in this subsection stem from systematic variations of Spark configuration parameters during the training of the linear regression model. The primary findings are summarized as follows:

a) Executor Instances: Increasing the number of executor instances did *not* result in a measurable reduction in training time.

b) Executor Cores: Increasing the number of cores per executor led to a decline in model performance.

c) Executor Memory: Executor memory was found to have the most pronounced impact on both stability and performance. Configurations with 2 GB of memory per executor frequently resulted in job failures or inefficient execution. Assigning 3–4 GB ensured reliable completion of training tasks.

d) Shuffle Partitions: A configuration of 100 shuffle partitions provided the most effective trade-off between parallel execution and shuffle overhead. Increasing the number to 200 resulted in excessive task fragmentation, introducing latency without corresponding performance benefits.

When extending the experiments to the Random Forest regression model, we observed that the impact of Spark configuration parameters, although notable, was not significant Random Forest training proved more resilient to suboptimal configurations, exhibiting stable behavior across a wider range of settings. Nevertheless, performance gains were still observed under optimized configurations, particularly in terms of execution time and resource utilization.

VI. Conclusion

This project demonstrated the effectiveness of big data frameworks such as Apache Spark in handling and analyzing large-scale urban mobility datasets. By profiling and exploring the NYC Yellow Taxi dataset, we extracted meaningful insights into trip duration, spatial pickup/drop-off distributions, and temporal travel patterns. Clustering techniques further revealed spatial groupings that align with known city structures and usage behaviors.

We trained machine learning models, including linear regression and random forests, to predict trip costs, using location and time-based features. Our experiments showed that model performance depends not only on algorithm selection, but also on data volume, pipeline design and Spark configuration.

In our runtime and scalability analysis, we found that:

- **Executor memory** had the greatest impact on job success and model training stability.
- **Adding more resources** did not consistently improve performance and sometimes led to diminishing returns or even degradation.
- **Spark configurations** such as shuffle partitions and memory allocation must be carefully tuned depending on the model and data size.

Through this work, we confirmed that Spark is a powerful and flexible platform for big data processing. It supports efficient querying, scalable machine learning, and advanced parallelization strategies through easy-to-configure clusters. However, our experiments also revealed that tuning Spark is non-trivial, and performance gains are not always guaranteed by simply scaling up the cluster.

Future work may explore:

- Testing other functionalities of Spark (e.g. Spark Streaming)
- Testing the same workflows on other cloud providers such as AWS or Azure
- Continuing testing different Spark configurations

Overall, this project not only provided analytical insights into urban mobility but also deepened our understanding of distributed data processing and performance engineering using Apache Spark.

References

[1] H. Luu, *Beginning Apache Spark 3*. Apress, 2021. ISBN: 978-1-4842-7382-1.

[2] J. S. Damji, B. Wenig, T. Das, and D. Lee, *Learning Spark – Lightning-Fast Big Data Analysis*, 2nd ed. O'Reilly, 2020. ISBN: 978-1492050049.

[3] T. White, *Hadoop: The Definitive Guide: Storage and Analysis at Internet Scale*, 4th ed. O'Reilly, 2015. ISBN: 978-1491901632.

[4] T. John and P. Misra, *Data Lake for Enterprises: Lambda Architecture for Building Enterprise Data Systems*. Packt Publishing, 2017. ISBN: 978-1787281349.

[5] J. M. G. Barbosa, Moodle slides on Big Data Engineering topics, [Online]. Available through Moodle.

[6] J. J. Buchholz, *Spatial Equilibrium, Search Frictions and Efficient Regulation in the Taxi Industry*. Working Paper, Yale Department of Economics, 2022.

[7] C. Camerer, L. Babcock, G. Loewenstein, and R. Thaler, "Labor supply of New York City cabdrivers: One day at a time," *The Quarterly Journal of Economics*, vol. 112, no. 2, pp. 407–441, 1997.

[8] N. Ferreira, J. Poco, H. T. Vo, J. Freire, and C. T. Silva, "Visual exploration of big spatio-temporal urban data: A study of New York City taxi trips," *IEEE Transactions on Visualization and Computer Graphics*, vol. 19, no. 12, pp. 2149–2158, 2013.

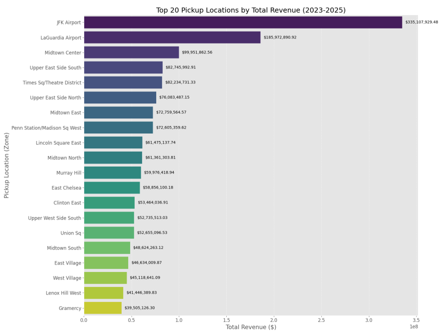
[9] C. Lyu, R. Zhang, Z. Li, S. Wu, Z. Zheng, Z. Wang, R. Yu, J. Xiao, J. Yuan, J. Zhou, Y. Diao, and A. K. H. Tung, "A Spark Optimizer for Adaptive, Fine-Grained Parameter Tuning," *Proceedings of the VLDB Endowment*, vol. 17, no. 7, pp. 1615–1629, 2024.

[10] D. Shabib, T. Cooper, and M. Farag, *Analysis of big data from New York taxi trip 2023: revenue prediction using ordinary least squares solution and limited-memory Broyden-Fletcher-Goldfarb-Shanno algorithms*. Preprint/Working Paper, Available on ResearchGate, 2024.

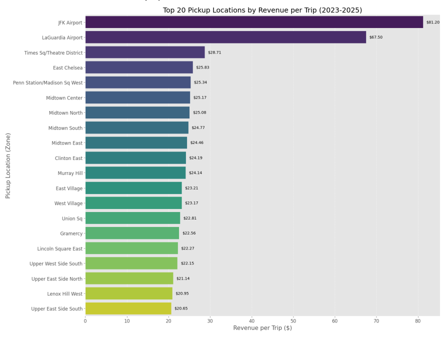
[11] J. Zhang, Y. Zheng, and D. Qi, "Deep Spatio-Temporal Residual Networks for Citywide Crowd Flows Prediction," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 31, no. 1, 2017.

[12] Kaggle, *New York City Taxi Fare Prediction*. [Online]. Available: https://www.kaggle.com/c/new-york-city-taxi-fare-prediction. Accessed on May 23, 2025.

A. Appendix



(a) Total revenue



(b) Average revenue per trip

Fig. 8: Top 20 pickup locations by economic metrics.

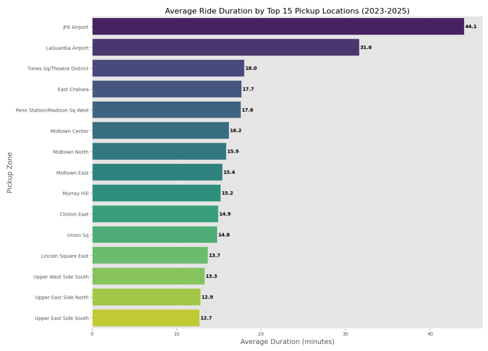


Fig. 9: Top 15 pickup locations by average ride duration

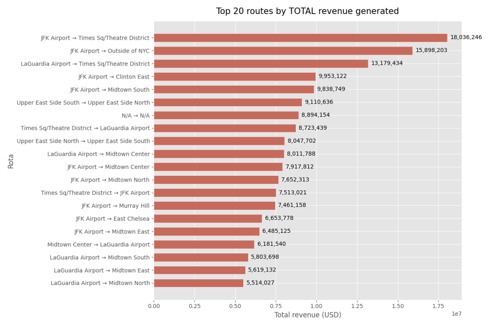


Fig. 10: Top 20 routes by total revenue

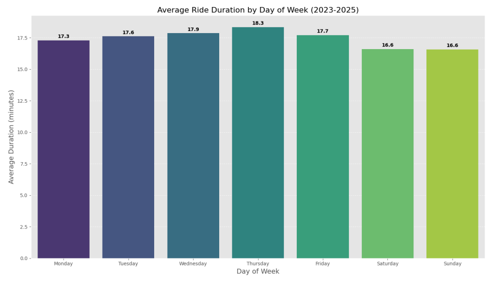


Fig. 11: Average ride duration by day of week

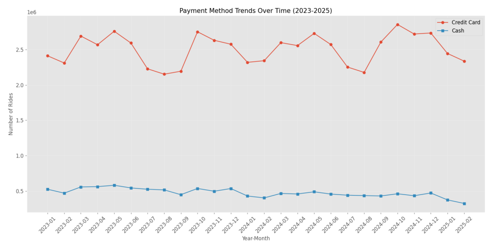


Fig. 14: Payment trends over time

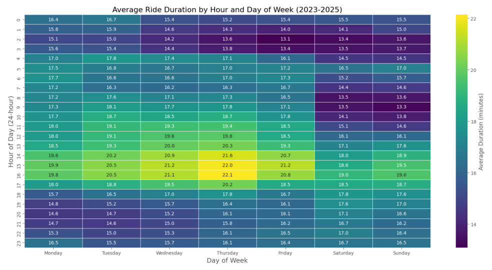


Fig. 12: Average ride duration by hour and day of the week

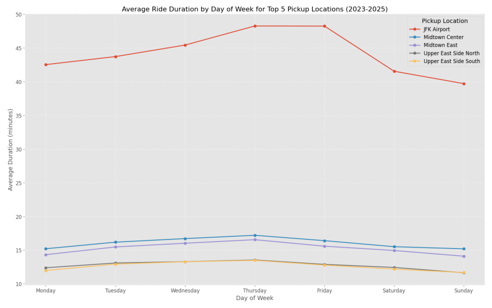


Fig. 13: Average ride duration by weekday for top 5 pickup locations (by duration)