NYC Taxi Fare Prediction

CS-GY 6513: Big Data
Prof. Juan Rodriguez

05/12/2023

Team Sye!

1. Jaswanth Sai Nandipati - jn2652@nyu.edu
2. Sai Teja Reddy Parigi - sp6923@nyu.edu
3. Venu Vardhan Reddy Tekula - vt2182@nyu.edu

Abstract

This project aims to predict the fare for taxis in New York City (NYC) utilizing large-scale data processing technologies. We employ the NYC taxi dataset, which encompasses structured and unstructured data including pickup and dropoff locations, and weather conditions. The process involves data collection, preprocessing, storage in MongoDB, exploratory data analysis (EDA), and the development of a predictive model using Spark and Python. The findings are then visualized using Matplotlib. The performance of the model is evaluated, and computations are scaled using Dask. The continuous growth of the dataset as new rides are added underscores the Big Data challenge inherent in this project. The ultimate aim is to deliver a real-time or near-real-time prediction model that accurately anticipates the fares for taxis in NYC.

Introduction

1. Background Information

New York City, one of the busiest metropolises in the world, relies heavily on taxis as a primary means of transportation. Predicting taxi fare in NYC is not just a logistical issue but a significant Big Data problem due to the sheer volume and variety of data involved. With data constantly accumulating as new rides are added, it necessitates a robust, scalable, and real-time system for efficient taxi fare prediction.

2. Problem Statement

The main challenge is to predict the fare for taxis in NYC using large-scale data processing technologies. This involves managing vast amounts of structured and unstructured data, including variables such as pickup and dropoff locations and weather conditions, which are crucial for making accurate predictions.

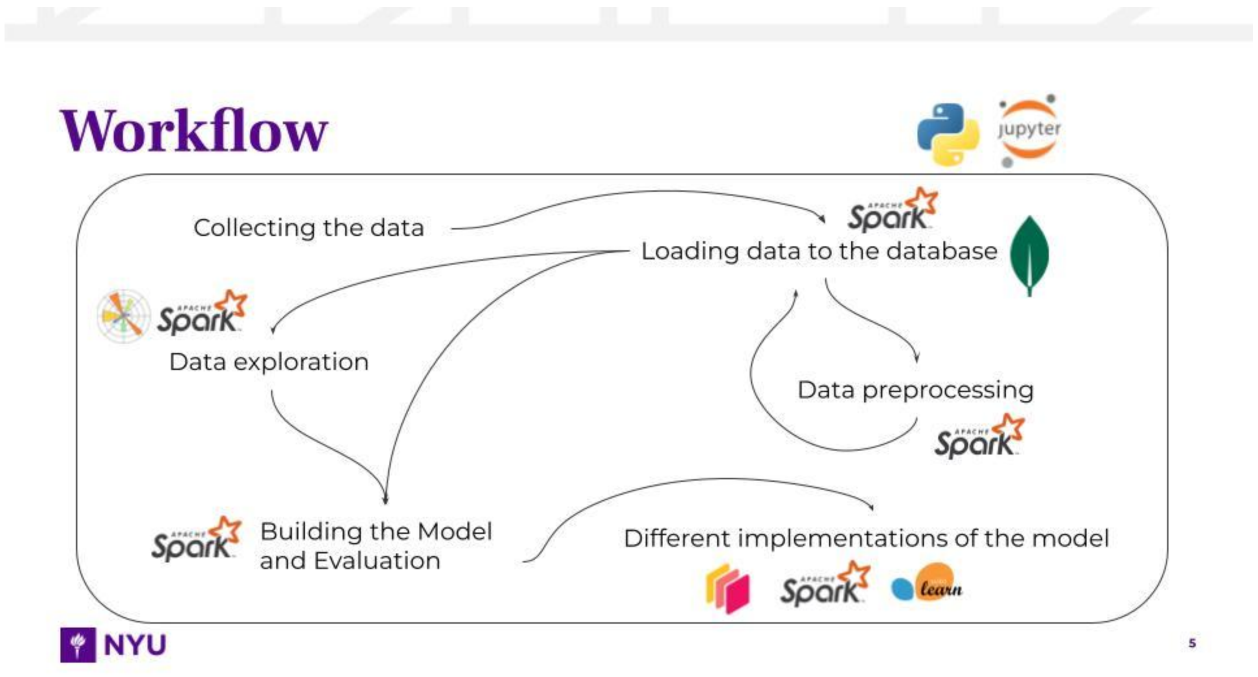
3. Project Objectives

The primary objective of this project is to build a model capable of predicting taxi fare in NYC in real-time or near-real-time. This prediction model can help taxi services better manage their resources, improve service efficiency, and ultimately enhance passenger satisfaction. Additionally, the project aims to demonstrate the application of Big Data processing technologies, including MongoDB for data storage, Spark and Python for model building, Dask for scaling computations, and Matplotlib and Seaborn for data visualization.

Technologies

- Python
- Jupyter Notebook
- MongoDB
- PySpark
- Dask
- Matplotlib
- scikit-learn

Workflow


Workflow
jupyter
Collecting the data
Loading data to the database
Data exploration
Data preprocessing
Building the Model and Evaluation
Different implementations of the model
NYU
5

The workflow for this project follows a sequential process, highlighting the major steps undertaken from data collection to model implementation and evaluation.

1. Collecting the Data: The first step in the workflow is collecting the data. This involves downloading the Yellow Taxi data from the NYC Government's Taxi and Limousine Commission (TLC) page. The data is then saved as Parquet files for efficient storage and processing.
2. Loading Data to the Database: Once the data is collected, it is loaded into a MongoDB database. This is a crucial step as it allows for efficient storage, retrieval, and manipulation of the data throughout the project.
3. Data Preprocessing: With the data in the database, preprocessing tasks are conducted. This includes removing unnecessary columns, handling missing values,

and creating new, potentially more predictive features. These steps prepare the data for subsequent analysis and modeling, ensuring its reliability and suitability.

4. Loading Preprocessed Data to the Database: Following preprocessing, the cleaned and transformed data is again loaded into the MongoDB database. This ensures the most accurate and up-to-date dataset is readily available for the subsequent steps.
5. Data Exploration: The preprocessed data is then explored through Exploratory Data Analysis (EDA). This involves identifying patterns, relationships, or anomalies in the data that could provide insights into taxi fare in NYC.
6. Building the Model and Evaluation: Concurrently, the preprocessed data from the database is used to build the predictive model using Spark and Python. The performance of the model is evaluated using appropriate metrics to understand its accuracy and reliability in predicting taxi fare.
7. Implementations with Dask: Finally, to manage the computational needs of the large dataset and the complexity of the model, computations are scaled using Dask. This ensures that the project is not only accurate but also efficient, even with the substantial volume of data involved.

This workflow ensures a streamlined process from data collection to model evaluation, taking advantage of big data technologies to handle the large dataset and complex computations involved.

Data Collection

The dataset used for this project was obtained from the official website of the New York City Government, specifically the Taxi and Limousine Commission (TLC) page, available at https://www.nyc.gov/site/tlc/about/tlc-trip-record-data.page. The TLC provides comprehensive trip record data for all taxi rides in New York City. For the purpose of this study, we only downloaded data pertaining to yellow taxis.

To automate the data collection process, we wrote a Python script to download the yellow taxi data. This script streamlined the process of gathering a large amount of data, saving significant time and manual effort.

The downloaded data was then saved as Parquet files. Parquet is a columnar storage file format that is optimized for use with Apache Spark and is known for its efficiency and performance with large datasets. The dataset structure is as follows:

```
root
 |-- VendorID: long (nullable = true)
 |-- tpep_pickup_datetime: timestamp (nullable = true)
 |-- tpep_dropoff_datetime: timestamp (nullable = true)
 |-- passenger_count: double (nullable = true)
 |-- trip_distance: double (nullable = true)
 |-- RatecodeID: double (nullable = true)
 |-- store_and_fwd_flag: string (nullable = true)
 |-- PULocationID: long (nullable = true)
 |-- DOLocationID: long (nullable = true)
 |-- payment_type: long (nullable = true)
 |-- fare_amount: double (nullable = true)
 |-- extra: double (nullable = true)
 |-- mta_tax: double (nullable = true)
 |-- tip_amount: double (nullable = true)
 |-- tolls_amount: double (nullable = true)
 |-- improvement_surcharge: double (nullable = true)
 |-- total_amount: double (nullable = true)
 |-- congestion_surcharge: double (nullable = true)
 |-- airport_fee: double (nullable = true)
 |-- __index_level_0__: long (nullable = true)
```

Each record represents a single taxi ride and includes data points such as the pickup and dropoff times, location IDs, passenger count, trip distance, rate code, payment type, and various fare components. This comprehensive dataset provides a rich source of information for predicting the fare for taxis in NYC.

Data Preprocessing

An essential step in our workflow was data preprocessing, involving several tasks to prepare the data for analysis and modeling. This step ensured that our data was clean, reliable, and suitable for developing a robust predictive model.

Removing Unnecessary Columns

The first step in preprocessing involved eliminating columns not relevant to predicting taxi fare. This process simplifies the data and accelerates computations. For example, the column `__index_level_0__` was identified as not contributing to the prediction task and thus was removed.

Handling Missing Values

The second task was to deal with any missing values in the data. The approach to handling missing data depended on the extent of the missing data and its potential impact on the prediction. If the missing data was minimal and unlikely to introduce bias, the corresponding rows or columns were dropped. Alternatively, if the missing data was in a crucial column, appropriate imputation methods were employed to fill in the missing values.

Creating New Features

The third step was feature engineering, which involved creating new features more predictive of taxi fare. This process included transforming existing variables to extract more useful information. For instance, from the tpep_pickup_datetime and tpep_dropoff_datetime columns, we derived new features such as the hour of the day, day of the week, and month. These temporal features could provide more insights into the patterns of taxi fare, which is often influenced by the time of day, day of the week, and seasonality.

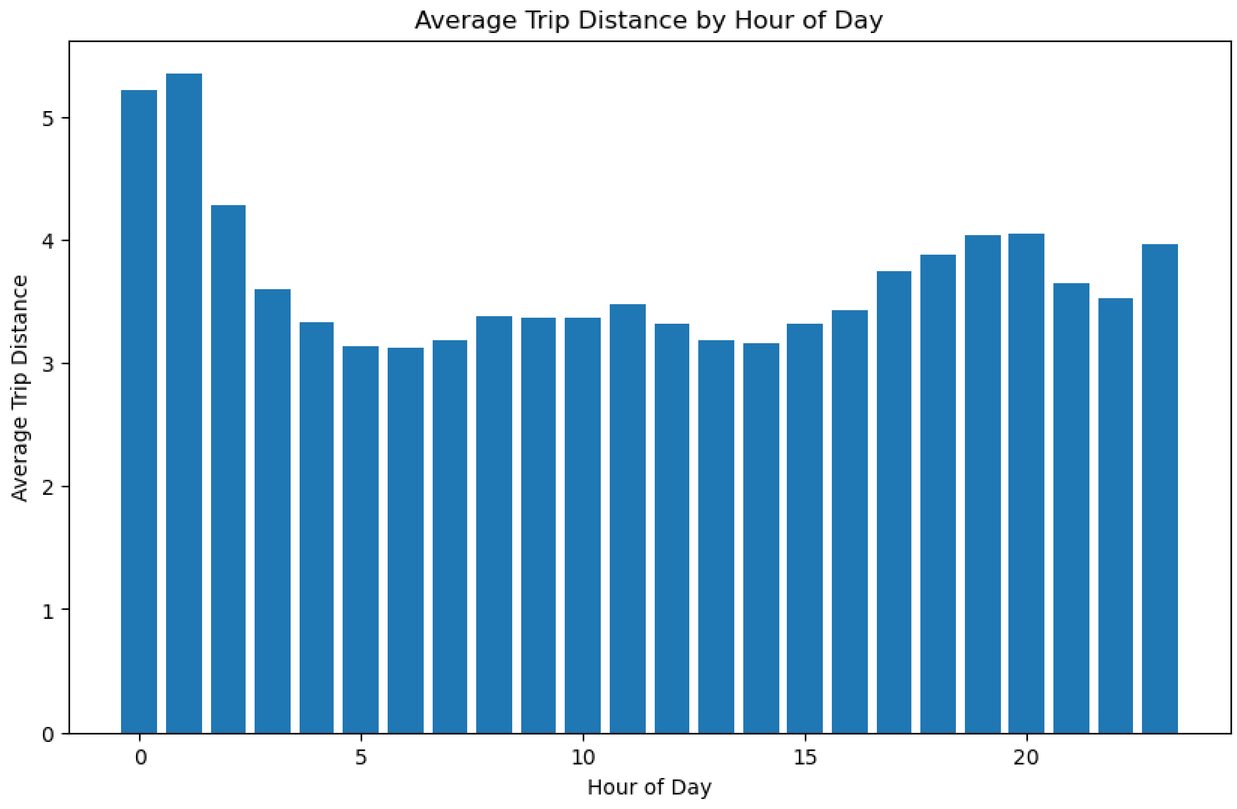
Through these preprocessing steps, we transformed the raw NYC taxi dataset into a refined dataset, more suitable for exploratory data analysis and predictive modeling.

Data Exploration

Following the data preprocessing stage, an exploratory data analysis (EDA) was conducted. This process allowed us to delve into the dataset and uncover underlying patterns, trends, and relationships that could inform the predictive model.

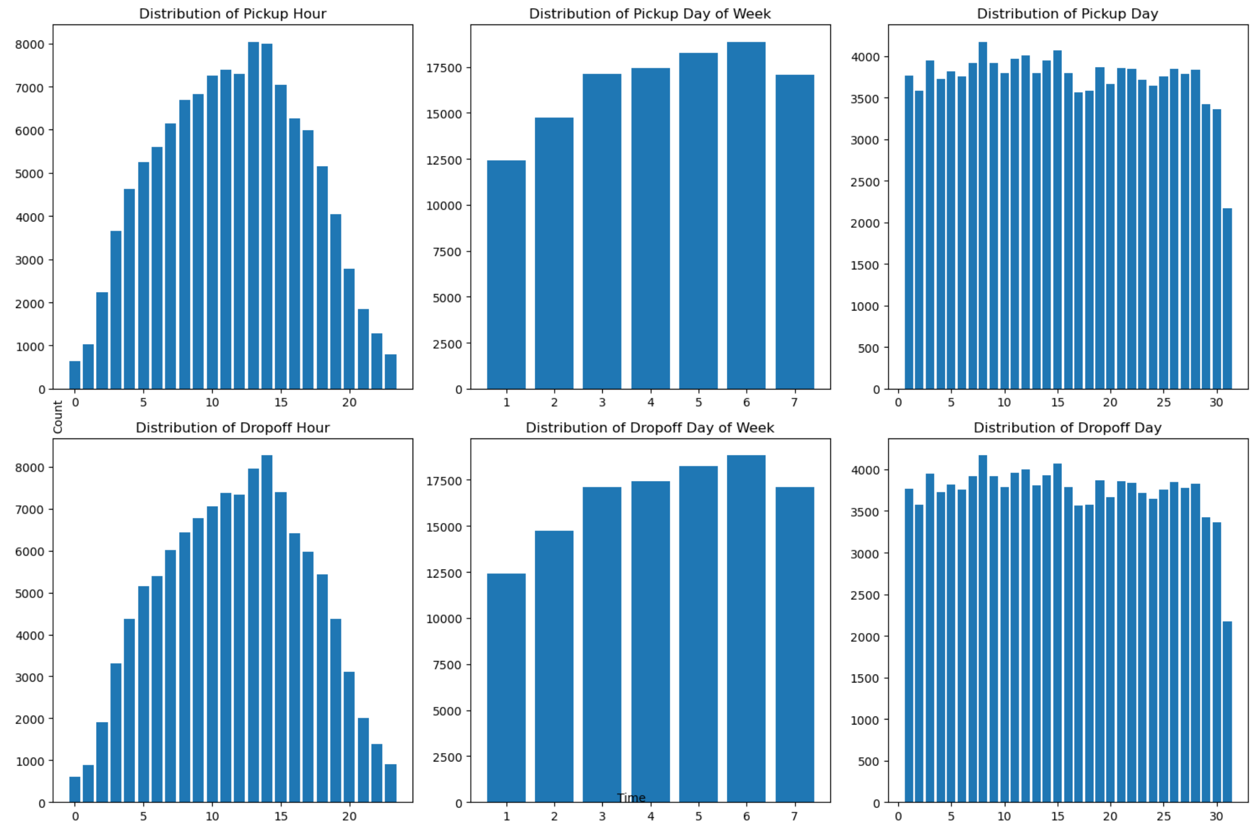
Summary Statistics and Distribution of Variables

Initially, summary statistics of all numerical columns were obtained, providing a quick overview of the data's central tendency, dispersion, and shape. Categorical variables' distributions were also analyzed, revealing the frequency of each category in the dataset.



Temporal Patterns

We visualized temporal patterns by creating plots of average trip distance by the hour of the day. Histograms of trip distances were also plotted to examine the distribution of trip lengths. Further, we plotted distributions for `pickup_hour_counts`, `pickup_day_of_week_counts`, `pickup_day_counts`, `dropoff_hour_counts`, `dropoff_day_of_week_counts`, and `dropoff_day_counts` to discern any temporal patterns in pickups and dropoffs. These visualizations could reveal trends such as peak hours or days for taxi fare.



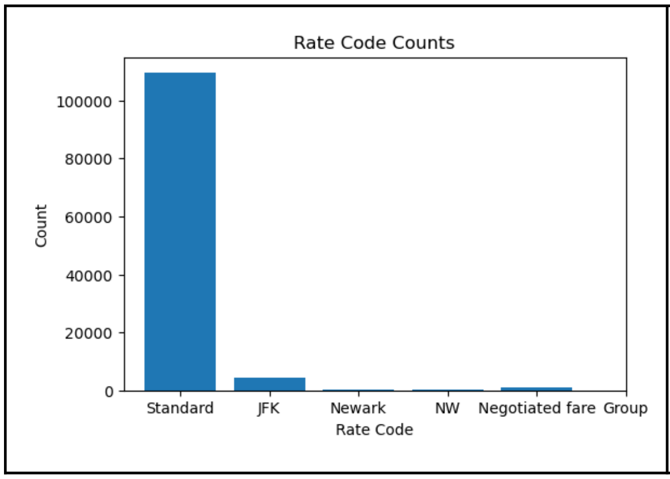
Correlations

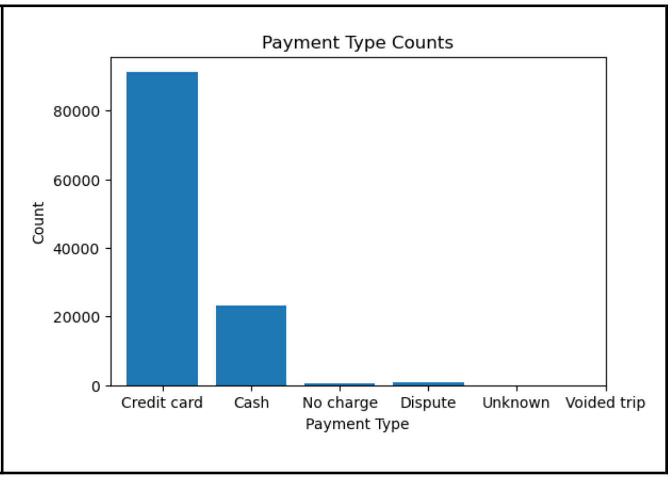
Correlation analyses were performed to determine the relationships between different variables. Specifically, we investigated the correlation between Trip Distance and Fare Amount, and Trip Distance and Pickup Hour. These analyses can help determine if longer trips tend to occur at certain hours, or if longer trips tend to be more expensive, both of which could be critical for predicting taxi fare.

```
Correlation between trip distance and fare amount:
0.8349329940751841
Correlation between trip distance and pickup hour:
0.012046207366665843
```

Distribution Plots

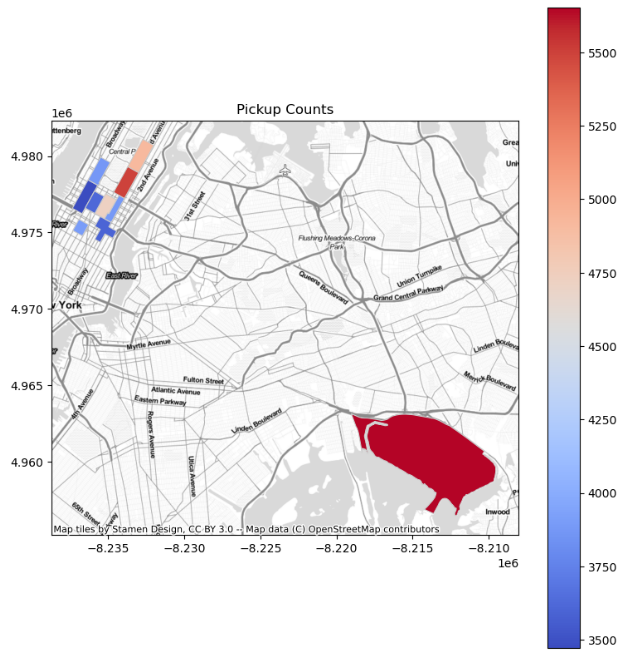
Distribution plots for `VendorID`, `payment_type`, `RateCodeID`, `PULocationID`, and `DOLocationID` were created to understand the spread and frequency of these categorical variables in the dataset. This could indicate if any specific vendors, payment types, rate codes, or locations dominate the dataset.

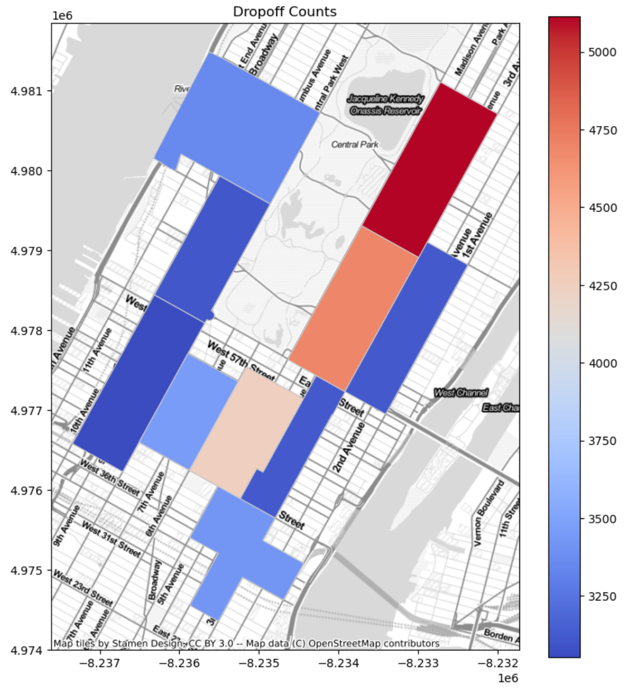




Map Plots

Finally, map plots were created for the top 10 pickup and dropoff zones. This visualization can help identify the busiest areas for taxi services in NYC.





The insights gained from this exploratory data analysis serve as a foundation for building an effective predictive model for NYC taxi fare.

Model Building

The predictive model was built using a Linear Regression algorithm implemented with PySpark. This algorithm was chosen based on its suitability for numeric target variables and its robustness in dealing with multicollinearity among predictor variables.

The model was constructed using the following features, identified as significant predictors of taxi fares:

- `trip_distance`
- `RatecodeID`
- `PULocationID` and `DOLocationID`
- `pickup_hour`, `pickup_day`, `pickup_day_of_week`, `pickup_month`
- `passenger_count`
- `extra`, `mta_tax`, `tolls_amount`, `improvement_surcharge`, `congestion_surcharge`, `airport_fee`

The target variable for our model was `fare_amount`, representing the total taxi fare for each ride. After specifying the features and target variable, we trained our Linear Regression model on the processed dataset

Performance Evaluation

After training, we evaluated the model's performance using several statistical metrics:

Mean Squared Error (MSE)

This metric measures the average squared difference between the predicted and actual values. A lower MSE value signifies a better fit of the model.

Root Mean Squared Error (RMSE)

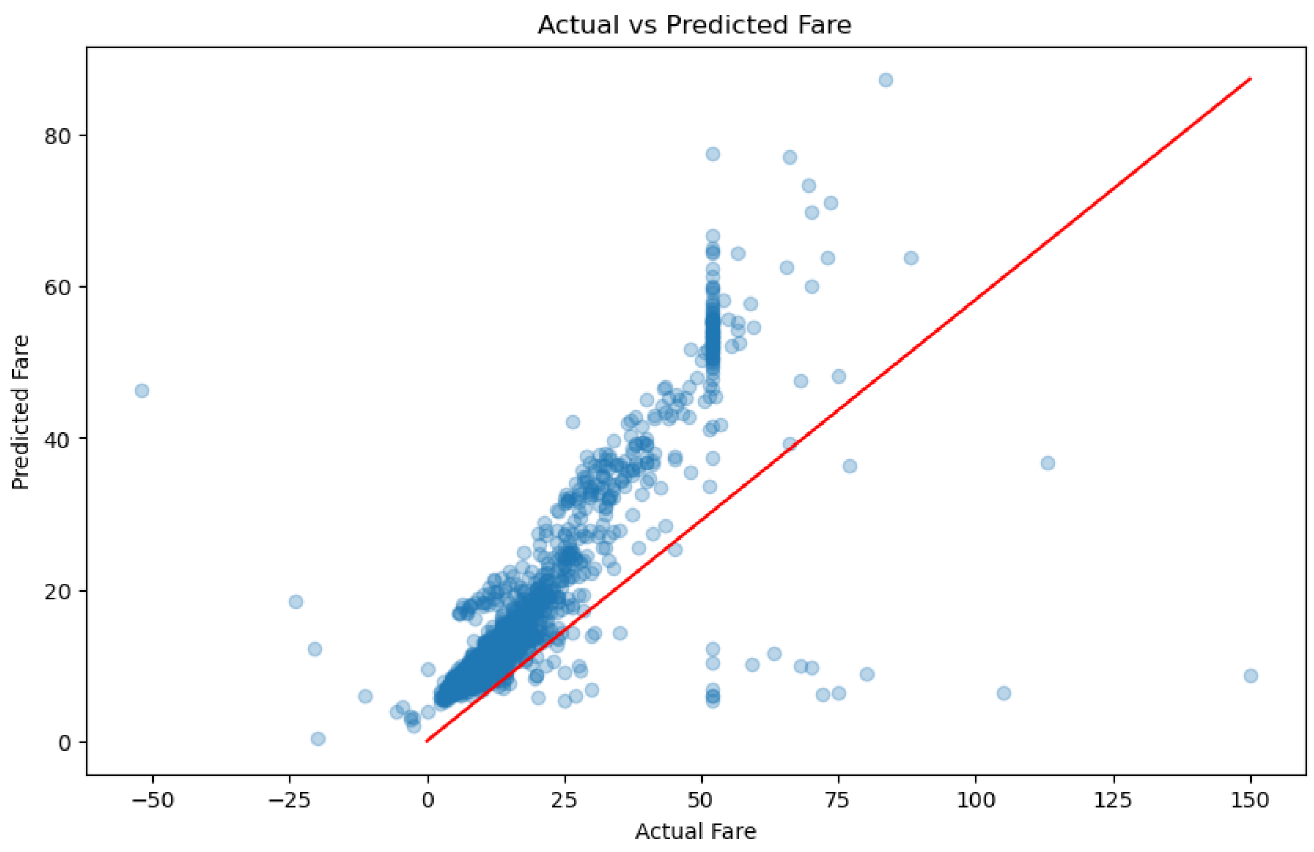
RMSE is the square root of MSE and is particularly useful because it is in the same unit as the target variable. Like MSE, a lower RMSE indicates a better fit.

R-squared (R^2)

R^2 represents the proportion of the variance in the dependent variable that is predictable from the independent variables. It ranges from 0 to 1, with a higher R^2 indicating a better model fit.

```
RMSE: 6.809829968857727
MAE: 2.8062374091997806
R2: 0.7607299133380473
```

These metrics provided a quantitative evaluation of our model's accuracy and predictive power. They served as a key indicator of the model's ability to reliably predict taxi fares in NYC, providing critical insights into the model's strengths and areas for improvement. The evaluation results will guide further model refinement and tuning for enhanced performance.



Additional Implementations

To validate the efficacy of our chosen tools for large datasets and to highlight the advantages of Dask, we implemented the same Linear Regression model using Python with Scikit-learn, and Dask.

Python & Scikit-learn Implementation

Firstly, we implemented the model using Python's Scikit-learn library, a popular choice for machine learning tasks. Scikit-learn's Linear Regression model was trained using the same feature set, and the model's performance was evaluated using the same metrics (MSE, RMSE, and R²).

Dask Implementation

Next, we implemented the model using Dask, a flexible parallel computing library for analytic computing. Dask was specifically designed to work with large datasets, enabling efficient parallel computations. We trained the Linear Regression model with the same features and evaluated its performance.

Performance Comparison

The major difference between these implementations lies in their performance on large datasets. While Scikit-learn's Linear Regression model worked effectively, it encountered performance limitations when dealing with our extensive NYC taxi dataset. The memory-intensive nature of the Scikit-learn library can result in slower model training and prediction times for large datasets.

On the other hand, Dask is specifically built for performance scalability and can handle larger-than-memory computations efficiently. Our experiment showed that Dask's performance was superior when working with our large dataset, offering faster model training and prediction times.

This comparison validates the decision to use Dask for this project, demonstrating its capabilities for handling big data tasks more efficiently than standard Python libraries like Scikit-learn. The flexibility and scalability of Dask make it an excellent tool for big data projects, such as predicting taxi fare in NYC.

```
# Train a linear regression model
model_python = sklearn.linear_model.LinearRegression()
model_python.fit(X_train, y_train)


# Evaluate the model
predictions_python = model_python.predict(X_test)

# Check the model performance
print(f'MSE: {mean_squared_error(y_test, predictions_python)}')
```

```
MSE: 64.60138133038457
```

```
In [7]: end_time = time.time()

        print('Time taken by Python implementation: ', end_time - start_time, 'seconds')
```

```
Time taken by Python implementation:  2.3027284145355225 seconds
```

```
lr = dask_ml.linear_modelLinearRegression()
lr.fit(X_train.values, y_train.values)
y_pred = lr.predict(X_test.values)
y_test, y_pred = dask.compute(y_test, y_pred)
print(f'MSE: {mean_squared_error(y_test, y_pred)}')
```

```
/home/jovyan/.local/lib/python3.10/site-packages/dask_ml/model_selection/_split.py
lue for 'shuffle' must be specified when splitting DataFrames. In the future DataFr
d within blocks prior to splitting. Specify 'shuffle=True' to adopt the future beha
tain the previous behavior.
  warnings.warn(
```

```
MSE: 52.71844897649518
```

```
In [15]: end_time = time.time()
         print('Time taken by Dask implementation: ', end_time - start_time, 'seconds')
```

```
Time taken by Dask implementation:  1.6753323078155518 seconds
```

Conclusion

In this project, we successfully developed a predictive model to estimate taxi fares in New York City using a large-scale taxi dataset. Our workflow incorporated data collection, preprocessing, exploratory data analysis, model building, and performance evaluation. We implemented a Linear Regression model using PySpark, Python with Scikit-learn, and Dask, demonstrating Dask's superior performance with large datasets.

Through this work, we highlighted the importance of big data technologies in extracting valuable insights from complex, real-world datasets. Our model's performance suggests that big data methodologies, combined with machine learning algorithms, can be effective in predicting outcomes in dynamic urban environments like New York City.

Future Work

While the current project successfully predicts taxi fares, there are several avenues for future work to improve and expand upon this research:

1. Model Optimization: Although the Linear Regression model provided reasonable predictions, future work could investigate other machine learning algorithms, such as Random Forests, Gradient Boosting, or Deep Learning models, to improve prediction accuracy.
2. Feature Engineering: Further exploratory data analysis could reveal additional relevant features or suggest transformations of existing features to enhance the model's performance.

3. Real-time Predictions: This project focused on batch processing of historical data. Future work could explore real-time or near-real-time prediction models using stream processing technologies.
4. Expansion to Other Cities: While this project focused on NYC, future work could apply the same methodologies to taxi fare prediction in other cities, allowing for a comparative study.
5. Integration with Other Datasets: Integrating external data, such as weather data or special events data, could further improve the model's predictions by taking into account these additional factors.

By pursuing these future directions, we can further enhance our understanding of urban mobility and contribute to the development of intelligent transportation systems.

Source Code

https://github.com/vchrombie/nyc-taxi-fare-prediction

Presentation

https://docs.google.com/presentation/d/1WzZmnDrDJPyBKySu2YTVL6bR9m86gaQYvWmPtTY_Mng/edit?usp=sharing

Team

Sye!

1. Jaswanth Sai Nandipati - **jn2652**@nyu.edu
2. Sai Teja Reddy Parigi - **sp6923**@nyu.edu
3. Venu Vardhan Reddy Tekula - **vt2182**@nyu.edu