San Francisco State University

Math 448
Introduction to Statistical Learning & Data Mining
Spring 2025

Final Project Progress Report
Taxi Trip Data Analysis and Fare Prediction

Submitted To:
Prof. Tao He

Submitted By:
Harsh Bajpai

⇒ Executive Summary

In New York City, taxi fares are often unpredictable, leaving passengers confused and drivers uncertain about optimal strategies for revenue. This project aims to address that ambiguity by building a data-driven system that predicts NYC yellow taxi fare amounts and final trip charges, using a combination of machine learning and rule-based logic.

I have trained multiple regression models—including Linear, Ridge, Lasso, Decision Tree, and Random Forest—on a rich dataset of over 6 million taxi trips. After careful comparison, a tuned Random Forest Regressor was selected for its strong performance in capturing non-linear patterns without overfitting. I further engineered logic to handle special cases like JFK flat-rate rides and constructed a rule-based pipeline to estimate total charges, including extras like congestion surcharges, MTA tax, and more.

To bring this to life, I developed and deployed a web-based fare estimator that allows users to enter pickup and drop-off addresses and receive transparent, real-time predictions. The backend intelligently fetches trip distance and duration using the Google Maps API and applies the hybrid logic for output. Additionally, I performed a separate revenue optimization analysis focused on how payment method impacts overall fare revenue.

This end-to-end project blends data science, business impact, and user-facing technology—providing a comprehensive, transparent solution for NYC taxi fare estimation, with real-world value for both riders and drivers.

⇒ Introduction: Background & Problem Statement

As a frequent observer of public transportation challenges and digital service gaps, I recognized that NYC taxi passengers often step into cabs without knowing how much their ride will cost. While official fare structures exist, real-world pricing can vary significantly based on trip distance, time of day, pickup and drop-off zones, and additional surcharges. This unpredictability frequently leads to confusion, fare disputes, and a lack of trust in the system.

At the same time, drivers and fleet operators may not fully understand how trip characteristics—especially payment methods—impact their total earnings. Many riders prefer credit cards or digital payments, but whether this affects final revenue outcomes is often unclear. These knowledge gaps create missed opportunities for both transparency and operational optimization.

This project was designed to solve both sides of this issue. First, I aimed to build a predictive system that allows passengers to estimate fares before starting a ride—making the process more transparent and reliable. Second, I wanted to explore the impact of payment methods on revenue, using statistical analysis to determine whether drivers could benefit from encouraging certain types of transactions.

By bridging the gap between raw data, machine learning, and real-world usability, this project addresses a fundamental problem in the urban transportation experience: uncertainty around cost and earnings.

The core objective of this project was to develop a transparent, reliable, and data-driven fare estimation system for NYC yellow taxis. I wanted to give both riders and drivers the ability to better understand trip pricing—before the ride begins—and make smarter, data-informed decisions.

To accomplish this, I set the following goals:

1. **Build a machine learning model** to predict the base fare of a taxi trip using available trip attributes such as distance, duration, pickup hour, weekday/weekend indicator, payment type, and location IDs.
2. **Handle flat-rate airport rides**—specifically JFK to/from Manhattan trips—by integrating rule-based overrides into the predictive pipeline, ensuring fare estimates remain consistent with official regulations.

3. **Develop a hybrid estimation system** that combines machine learning with rule-based logic to predict the full total amount a rider would pay, accounting for standard surcharges like the MTA tax, congestion surcharge, improvement fee, and conditional extras.
4. **Deploy a user-facing web application** where anyone can input their pickup, and drop-off addresses and receive an immediate, itemized fare breakdown.
5. **Analyze the effect of payment methods on total revenue** to uncover whether certain transaction types result in higher earnings and whether behavioral nudges could help optimize driver income without compromising rider experience.

Together, these objectives allowed me to create not just a model, but a real, usable product—grounded in data science, powered by machine learning, and built with practical business implications in mind.

⇒ Dataset Description, Cleaning & Preparation

For this project, I utilized the official **NYC Taxi & Limousine Commission (TLC) Yellow Taxi Trip Records** for **January 2020**. The dataset contains over **6.4 million individual trips**, offering a comprehensive and high-resolution view of passenger taxi behavior in New York City. Each entry includes detailed trip-level attributes such as pickup and drop-off timestamps, trip distance, location IDs, fare-related amounts, payment method, and rate code–used for special trips like flat-rate JFK rides.

A detailed description of all columns and their meanings is provided in Figure below.

Field Name	Description
VendorID	A code indicating the TPEP provider that provided the record. **1= Creative Mobile Technologies, LLC; 2= VeriFone Inc.**
tpep_pickup_datetime	The date and time when the meter was engaged.
tpep_dropoff_datetime	The date and time when the meter was disengaged.
Passenger_count	The number of passengers in the vehicle. This is a driver-entered value.
Trip_distance	The elapsed trip distance in miles reported by the taximeter.
PULocationID	TLC Taxi Zone in which the taximeter was engaged
DOLocationID	TLC Taxi Zone in which the taximeter was disengaged
RateCodeID	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** **6=Group ride**
Store_and_fwd_flag	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	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	The time-and-distance fare calculated by the meter.
Extra	Miscellaneous extras and surcharges. Currently, this only includes the $0.50 and $1 rush hour and overnight charges.
MTA_tax	$0.50 MTA tax that is automatically triggered based on the metered rate in use.
Improvement_surcharge	$0.30 improvement surcharge assessed trips at the flag drop. The improvement surcharge began being levied in 2015.
Tip_amount	Tip amount – This field is automatically populated for credit card tips. Cash tips are not included.
Tolls_amount	Total amount of all tolls paid in trip.
Total_amount	The total amount charged to passengers. Does not include cash tips.
Congestion_Surcharge	Total amount collected in trip for NYS congestion surcharge.
Airport_fee	$1.25 for pick up only at LaGuardia and John F. Kennedy Airports

Figure: Description of NYC Yellow Taxi Dataset Columns

The full dataset originally consisted of **18 columns**. Upon inspection, I found that **about 1.02% of rows** had missing values in fields such as payment type, passenger count, and RatecodeID. Given the overall size, I chose to drop these rows to preserve data integrity without losing meaningful volume.

To enable robust modeling, I engineered several new features:

- **Trip duration** (in minutes), calculated as the difference between drop-off and pickup times.

- **Temporal features** including pickup hour, pickup weekday, pickup month, and a binary is weekend flag.
- Filtered outliers in several numeric columns using domain-informed rules:
    - **Fare amount** capped at $150
    - **Trip distance** capped at 50 miles
    - **Trip duration** capped at 120 minutes
    - **Tolls amount** capped at $30
    - **Extra charges** limited to $5 (to remove clearly invalid entries like $113 or $18)

Additionally, I dropped columns such as tip amount, tpep dropoff datetime, store and fwd flag, and tpep pickup datetime during model training to avoid information leakage and redundancy.

I also verified the validity of passenger count, removing entries with zero passengers and those with counts greater than six, which were either impossible or extremely rare. The categorical features RatecodeID, payment type, PULocationID, and DOLocationID were encoded using one-hot encoding within the modeling pipeline.

This curated, filtered, and feature-enriched dataset served as the foundation for both predictive modeling and exploratory visual analysis. It enabled me to extract reliable insights, model complex relationships, and build a system that reflects real-world fare dynamics in NYC.

⇒Exploratory Data Analysis (EDA)

Before training predictive models, I conducted an in-depth exploratory data analysis (EDA) to uncover the underlying patterns, trends, and relationships within the NYC yellow taxi dataset. The goal was to understand how key features like trip distance, duration, time of day, and payment method influence fare behavior, and to identify any hidden structures or anomalies that could affect modeling accuracy.

To maintain a consistent and visually engaging presentation, I used a customized NYC-themed color palette throughout this phase–featuring vibrant tones like NYC Yellow (#FFD60A), Asphalt Black(#2E2E2E), and Hudson Blue(#295E89). These colors helped draw attention to meaningful data trends while keeping the visual language cohesive and branded.

This section is organized into three main parts: Univariate distributions, bivariate relationships, and temporal and multivariate insights. Each visualization was carefully chosen not only for exploration but also to inform feature engineering and hybrid logic design later in the pipeline. From identifying flat-rate patterns in JFK rides to detecting peak hours and payment preferences, these visual insights laid the foundation for building a transparent and intelligent fare prediction system.

1. Distribution of Fare amount

I began the analysis by examining the distribution of the fare amount variable, which represents the based metered fare for each taxi trip, excluding tips and surcharges. This feature is the primary target for the machine learning model, so understanding its shape and behavior is crucial.

To visualize the distribution, I plotted a histogram overlaid with a KDE (Kernel Density Estimate) curve. The resulting plot revealed a **strong right-skew**, which is typical for fare-related data where most rides are short and low-cost, but a few long trips lead to much higher values. The bulk of the fares are concentrated between **$6 and $20**, suggesting the short-distance, intra-borough rides make up the majority of trips.

A distinct and sharp spike appears near **$52**, which corresponds to the flat-rate fare for JFK Airport trips. This insight was a critical finding during EDA–it confirmed the presence of regulated fares within the data and justified the creation of a **special-case rule in prediction pipeline** for rides involving JFK.

To ensure the model wasn't skewed by extreme outliers, I capped fare amount values at $150, which helped tighten the distribution without losing meaningful variation.

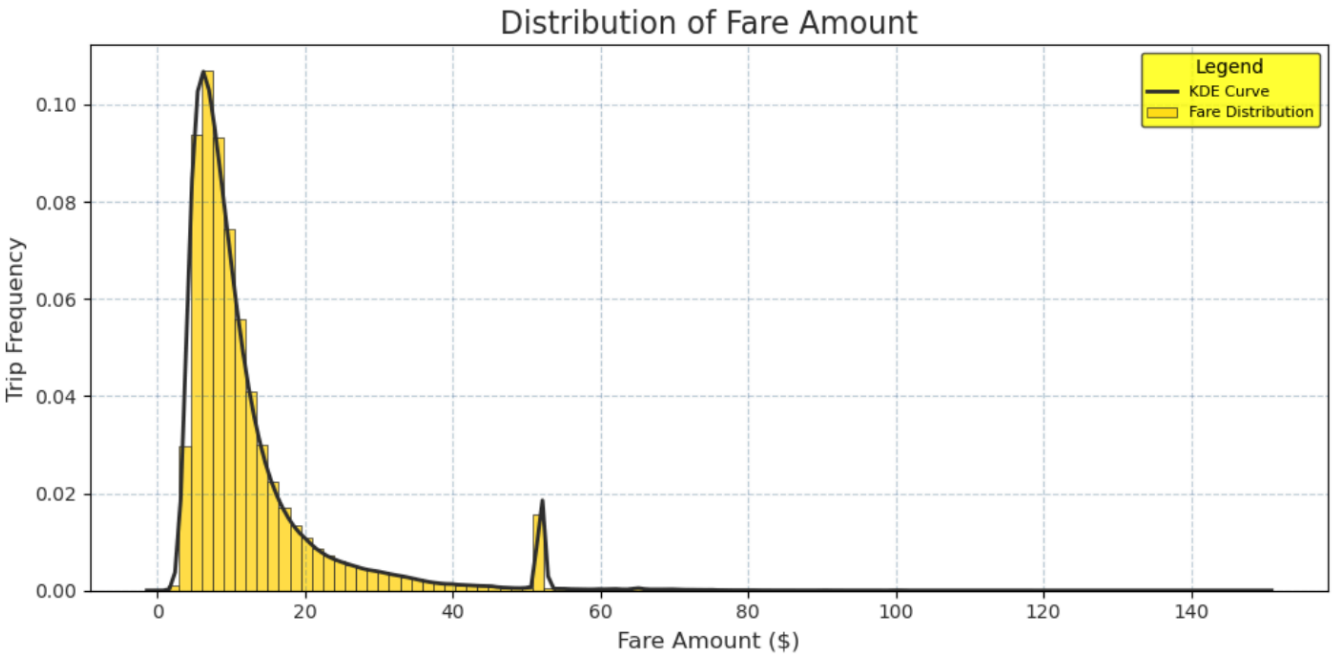

Figure: Histogram of Fare Amount with KDE Overlay

2. Fare Amount vs trip Distance

Next, I explored the relationship between fare amount and trip distance using a scatterplot with a fitted regression line. Since fare is generally distance-based for NYC taxis, this plot was expected to show a strong positive correlation–and it did.

The majority of data points form a clear upward-sloping linear trend, indicating that as trip distance increases, the fare amount also increases in a consistent and predictable way. This supports the intuition behind using distance as a core feature in fare prediction models.

However, one particularly interesting anomaly is the horizontal band of data points at $52, regardless of distance. This cluster represents JFK flat-rate trips, where the fare is fixed irrespective of mileage. These rides bread the usual linear trend and, once again, reinforce the importance of applying rule-based logic for flat-rate exceptions during model training and deployment.

I also filtered the plot to remove outlier trips with extreme distances or fares, which helped focus on the practical fare behavior of 99% of trips.

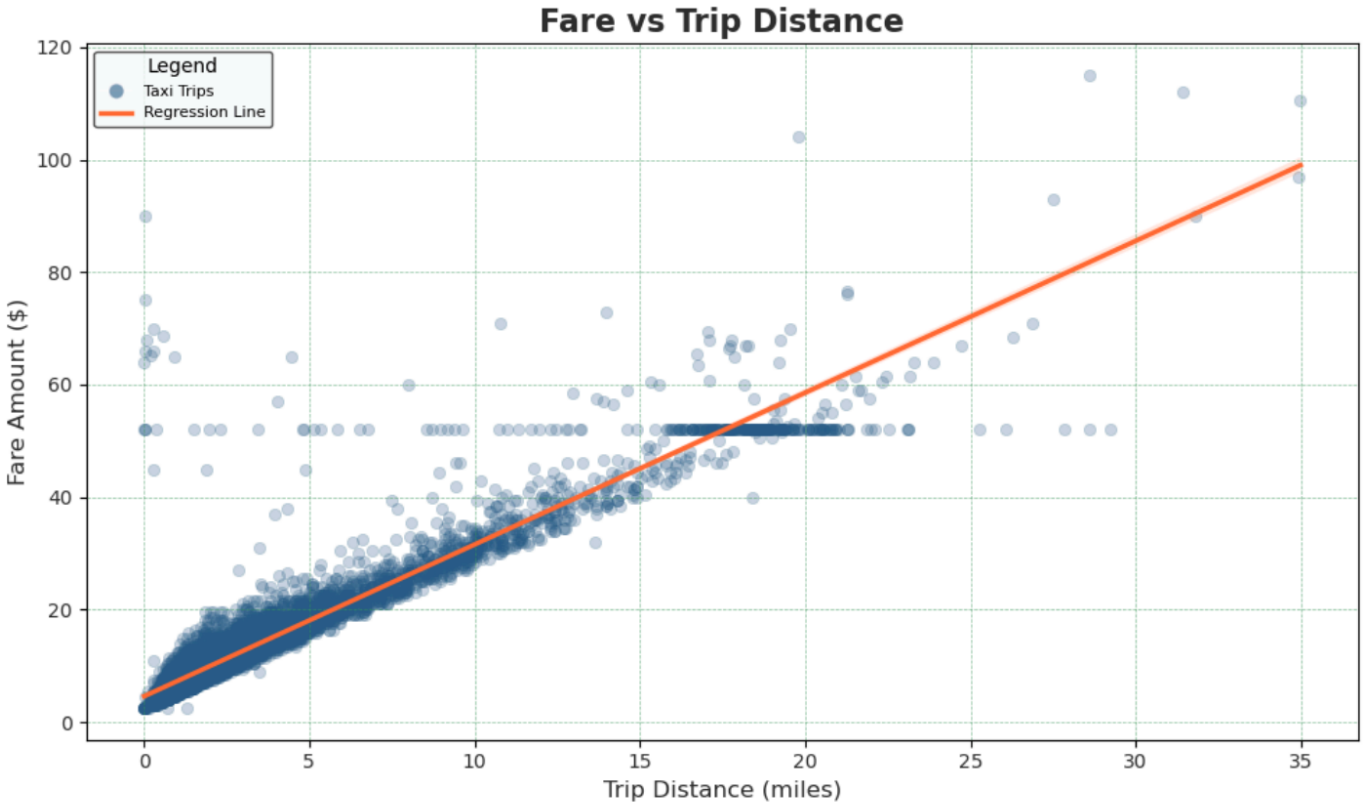

Figure: Scatterplot of Fare Amount vs Trip Distance with Regression Line

3. Average Fare by Pickup Hour

To analyze how fares vary across different times of the day, I grouped the data by pickup hour and calculated the average fare for each hour (from 0 to 23). This was visualized using a bar plot and line plot to capture temporal trends in pricing behavior.

The results revealed two clear peaks:

- A morning peak around 5:00 AM, and
- An evening peak between 9:00 PM and midnight

These peaks align with common real-world phenomena. The **early morning surge** is likely driven by airport-bound passengers catching early flights, especially JFK and LaGuardia. The late-night spike reflects nightlife activity, late-shift workers, and reduced public transit availability during those hours.

This visualization provided not only temporal insight but also strategic input for the hybrid model–reinforcing why pickup hour was included as a feature and why additional surcharges like extra may need to vary depending on the time of day.

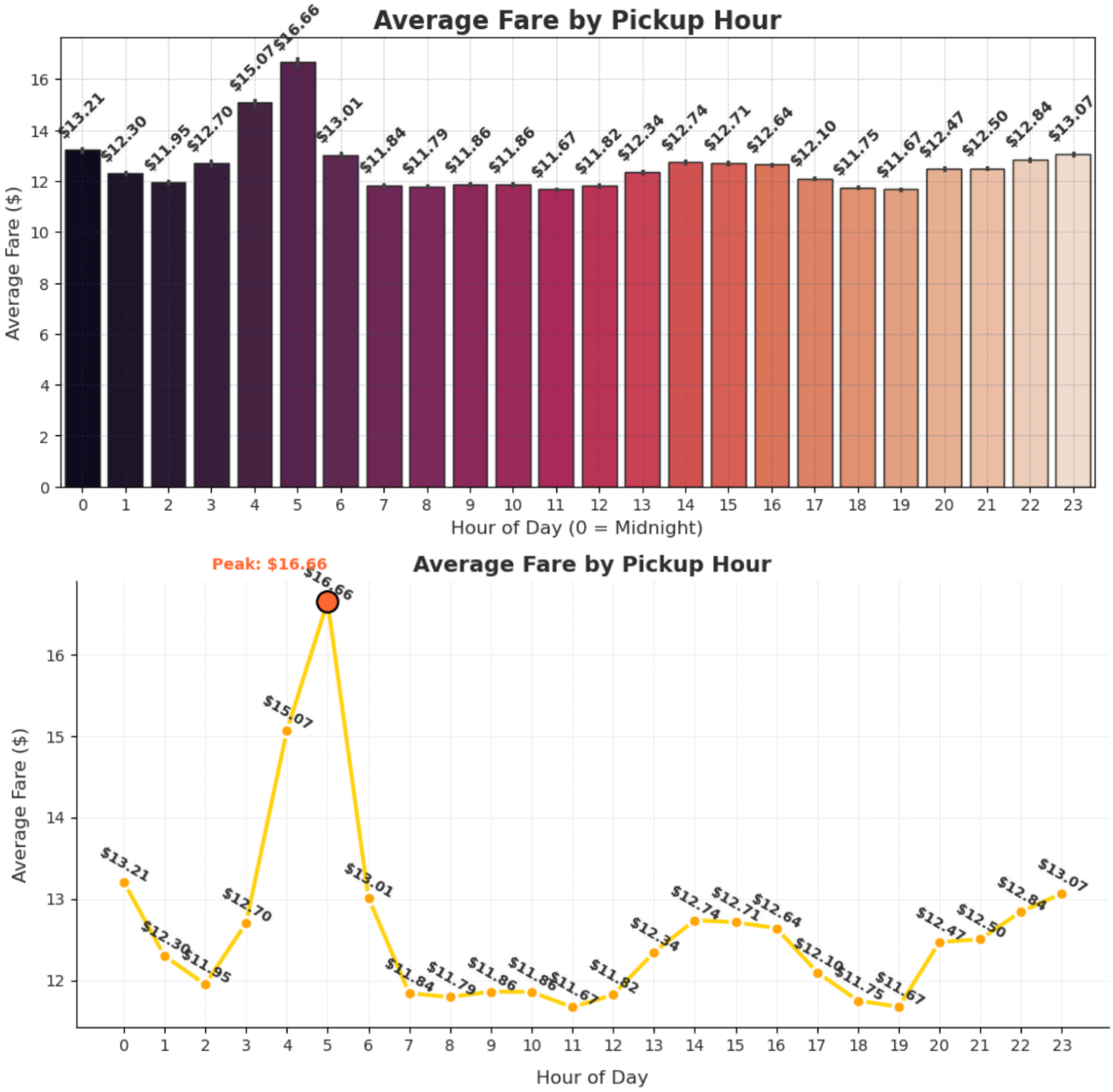

Figure: Bar plot of Average Fare by Pickup Hour

4. Average Trip Distance by Pickup Hour

To understand how the length of taxi trips varies over time, I grouped the data by pickup hour and calculated the average trip distance for each hour of the day. The results were visualized using a single line chart, which provided a smooth and intuitive view of distance trends throughout a 24-hour period.

This line plot revealed some insightful patterns. **Early morning hours (around 4-6AM)** showed notably longer average distances, like reflecting trips to and from airports such as JFK and LaGuardia. This aligns with a earlier observation

that fare amounts also spike during these hours–indicating that these higher fare are, in part, driven by longer trip distances.

During midday hours (10 AM to 4 PM), average distances tend to plateau, reflecting typical short urban rides. Later in the evening, particularly after 9PM, there's another modest uptick, which may correspond to cross-borough trips or return journeys from nightlife districts.

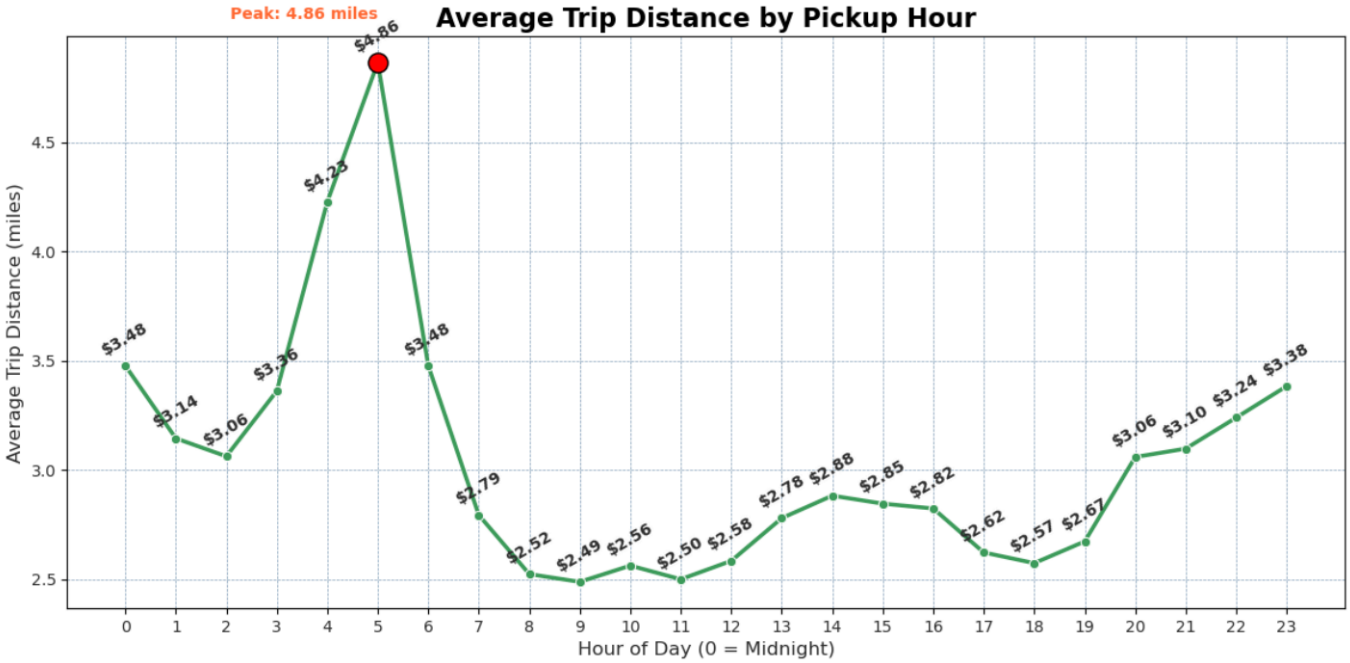

Figure: Line chart of Average Trip Distance by Pickup Hour

5. Average Trip Duration by Pickup Hour

In addition to fare and distance, I explored how the **trip duration** varies throughout the day. By grouping rides by pickup hour and calculating the average duration (in minutes), I generated a line plot to visualize how long taxi rides typically take at different hours.

The analysis revealed an intuitive pattern. **Average trip durations peak during typical traffic-congested hours**, particularly:

- **8:00 AM – 10:00 AM** (morning rush)
- **4:00 PM – 7:00 PM** (evening commute window)

These timeframes coincide with **heavy traffic conditions in NYC**, which can significantly slow down travel even short distances. Interestingly, early morning trips (4-6AM), although longer in distance as seen earlier, tend to have **moderate durations**–indicating **faster travel due to low congestion**.

This plot highlighted a temporal distinction **between trip length and trip time**, both of which are valuable predictors for fare. It also supported the inclusion of both trip duration and pickup hour as features in the final model .

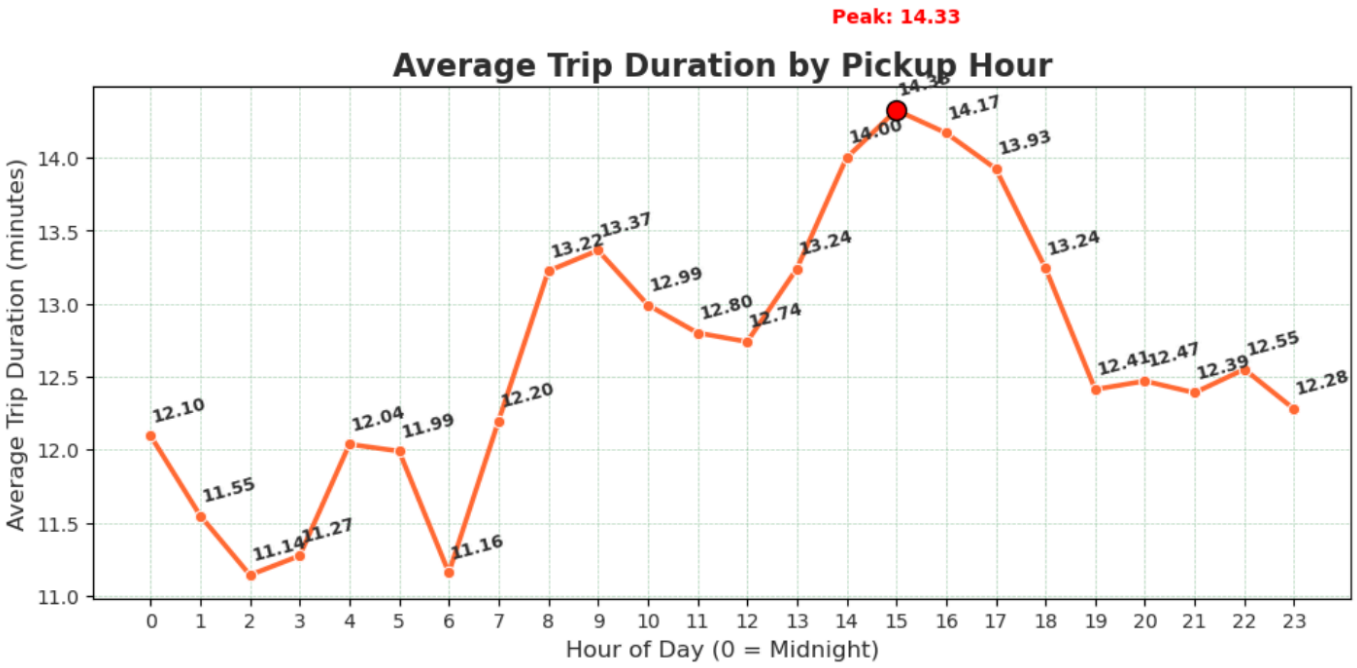

Figure: Line Chart of Average Trip Duration by Pickup Hour

6. Distribution of Trip Duration

To better understand how long NYC yellow taxi trips typically take, I visualized the distribution of the trip duration feature using a histogram. This variable, measured in minutes, was calculated by subtracting the pickup timestamp from the drop-off timestamp and converting the result to total minutes.

The resulting distribution is **positively skewed**, with the vast majority of trips lasting between **5 and 20 minutes**. This makes sense given the urban context of NYC, where most rides are relatively short and localized. There's a long right tail that includes trips over an hour in length; however, to reduce the impact of outliers and potential GPS errors, I capped trip durations at 120 minutes during data cleaning.

A small cluster of extremely short trips (under 1 minute) was also observed. These could represent either genuine ultra-short trips (e.g., around the block), GPS errors, or canceled rides still recorded. Depending on business logic, these edge cases could be handles separately or excluded from feature retraining.

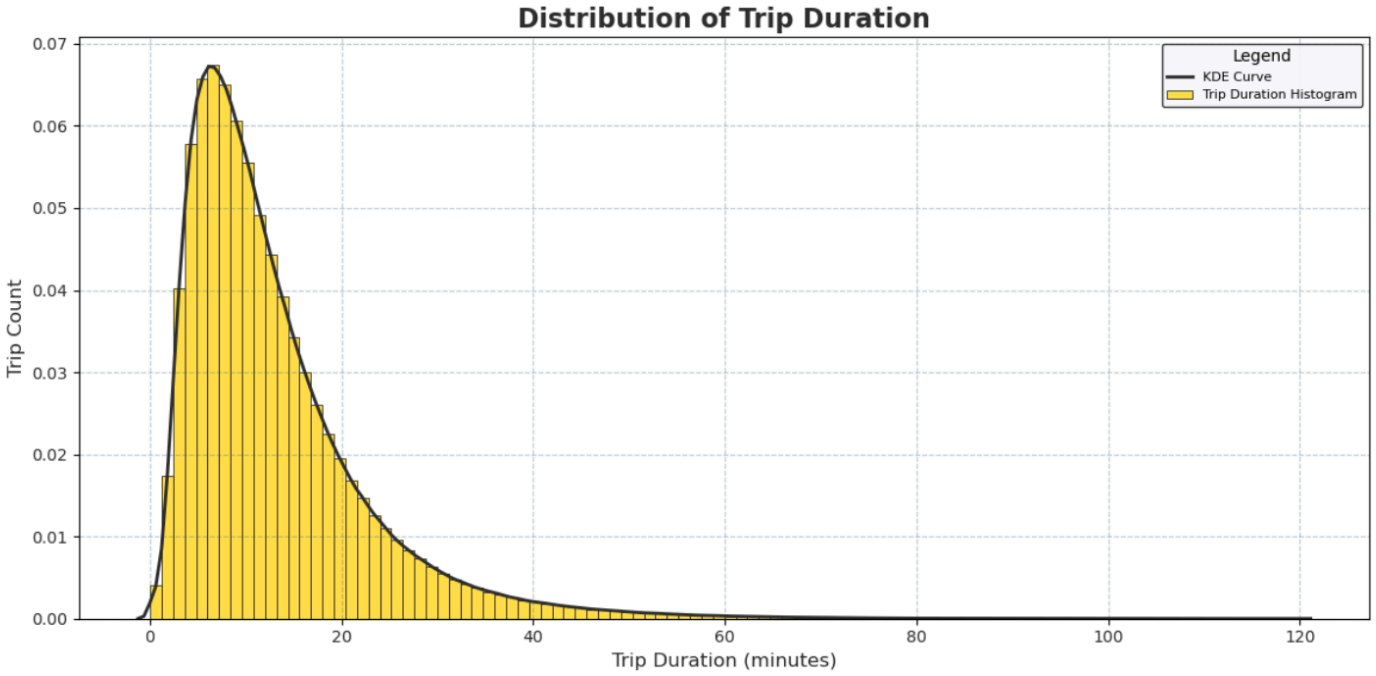

Figure: Histogram of Trip Duration (in Minutes)

7. Fare vs Trip Duration

To deepen the analysis of fare behavior, I examined the relationship between fare amount and trip duration, while also accounting for the impact of RatecodeID, which signifies the type of pricing rule applied to the trip. I visualized this using a scatterplot where each point represents a trip, the x-axis shows duration, the y-axis shows fare, and points are color-coded by RatecodeID.

This plot reveals several key patterns. First, there is a **clear upward trend**–as trip duration increases, so does the fare amount, although with more variability than the fare vs. distance relationship. This reflects how time spent in traffic or idle time affects pricing.

Secondly, the presence of **RatecodeID=2** (JFK flat fare) is immediately visible: these points from a horizontal band at $52, regardless of how long the trip took. This reinforces the idea that JFK trips follow a strict flat pricing rule, independent of distance or time.

Other rate codes (e.g., standard metered rides or negotiated fares) show more natural fare variation, but the clustering still supports the idea that duration is a **moderately strong predictor** of fare, especially when combined with rate code context.

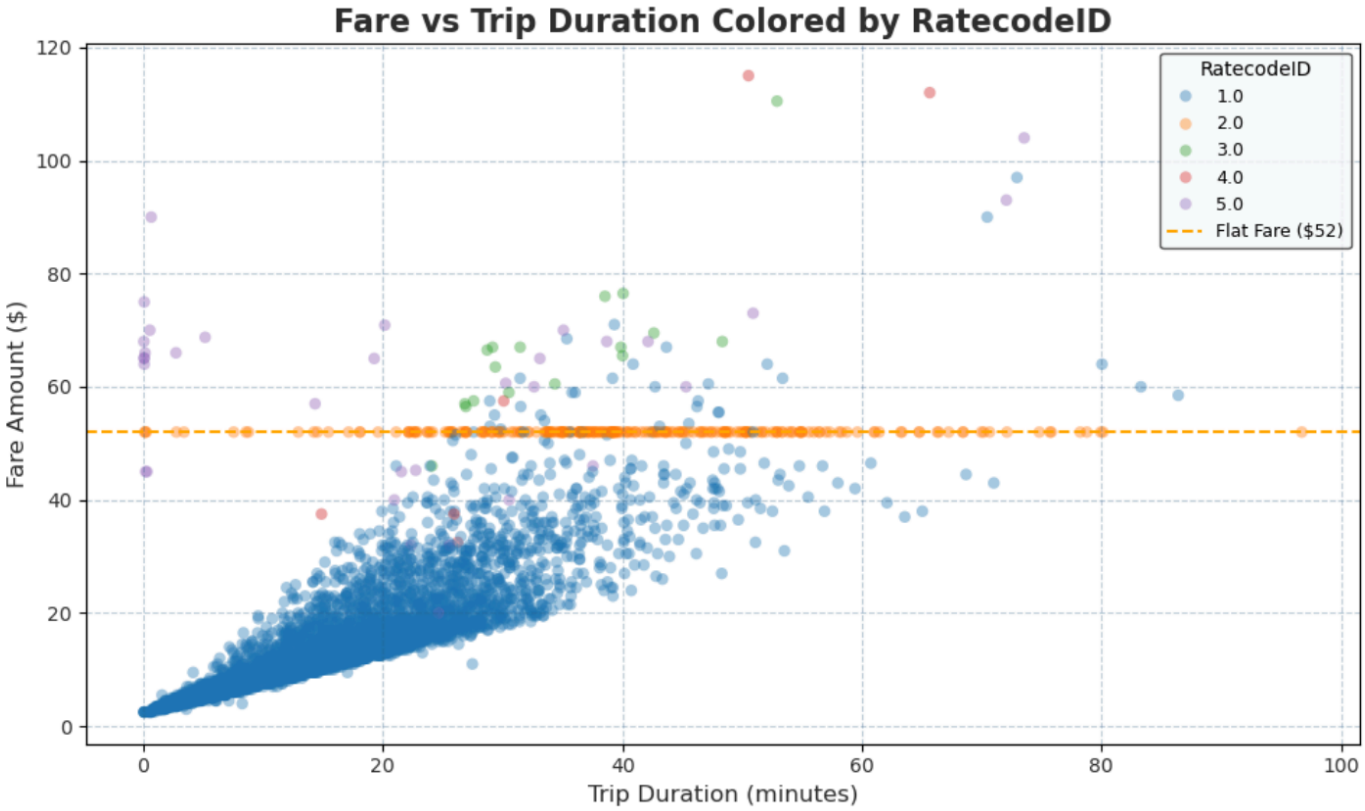

Figure: Scatterplot of Fare vs Trip Duration Colored by RatecodeID

8. Distribution of Payment Types

Understanding how passengers choose to pay is essential not just for fare modeling but also for exploring broader revenue patterns and operational implications for drivers. To that end, I visualized the **distribution** of payment type using a bar chart to show the frequency of each category.

The visualization showed that the vast majority of rides were paid using **credit card** (Payment type 1), accounting for over **60-70%** of the dataset. **Cash payments** (Type 2) were the next most common, followed by a small proportion of disputed or unknown transactions.

This trend indicates a strong rider preference for digital, card-based payments– likely due to convenience and automation. For modeling purposes, this variable was treated as categorical and one-hot encoded. For business analysis, this insight prompted a deeper dive into how **payment method might impact fare revenue**, which is explored later in the results section.

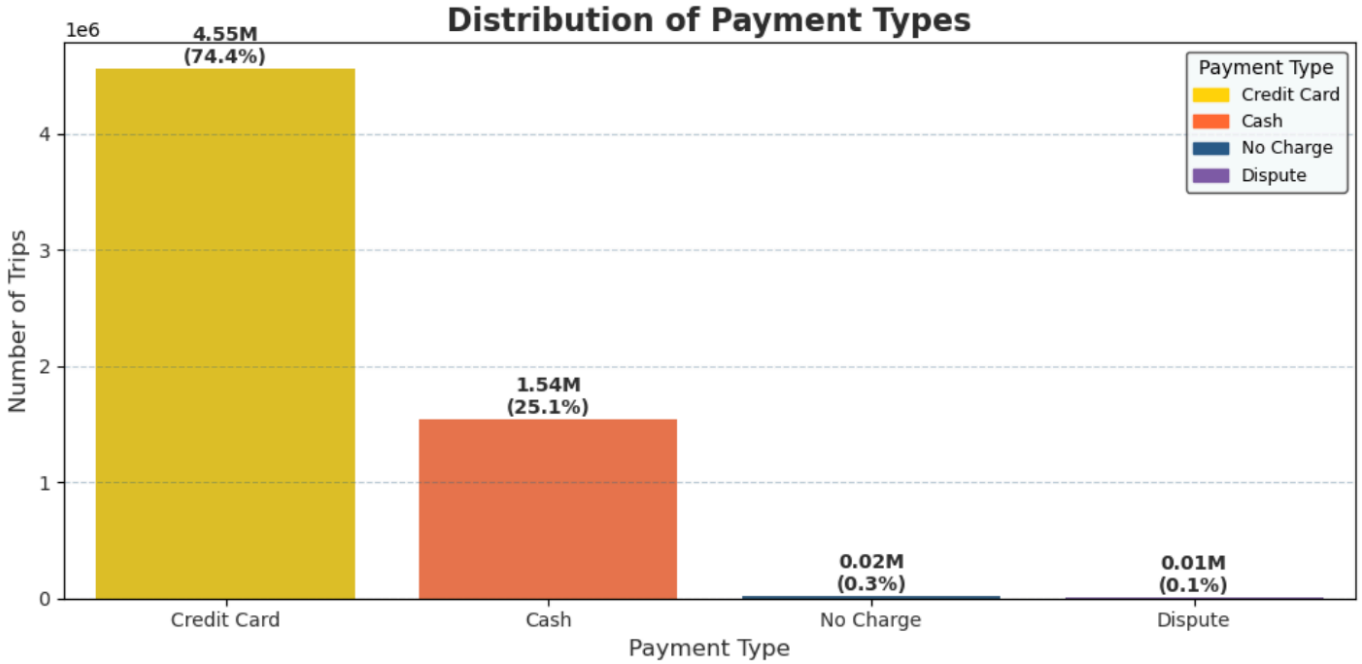

Figure: Bar Chart of Payment Type Distribution

9. Top Correlated Features with Fare Amount

To identify the most predictive variables for modeling fare amount, I computed the **correlation matrix** for all numeric features in the dataset. From the matrix, I extracted and ranked the **top correlated variables** with respect to the target (fare amount), and visualized them using a horizontal bar chart.

Unsurprisingly, the **strongest correlation** was with trip distance, reaffirming that NYC taxi fares are primarily distance-based. **Trip duration** also showed a meaningful positive correlation, reflecting the time component embedded in fare structure (e.g., time spent in traffic or waiting). Features like pickup hour and is weekend had moderate correlations, suggesting temporal effects influence fare but to a lesser degree.

Interestingly, RatecodeID showed low correlation overall due to the fixing pricing effect in only a subset of rides. However, it remained **crucial from a business rules** perspective, especially for JFK rides.

This analysis helped guide feature selection for the Random Forest model, and also validated earlier insights from EDA–particularly the predictive importance of distance, duration and time-based variables.

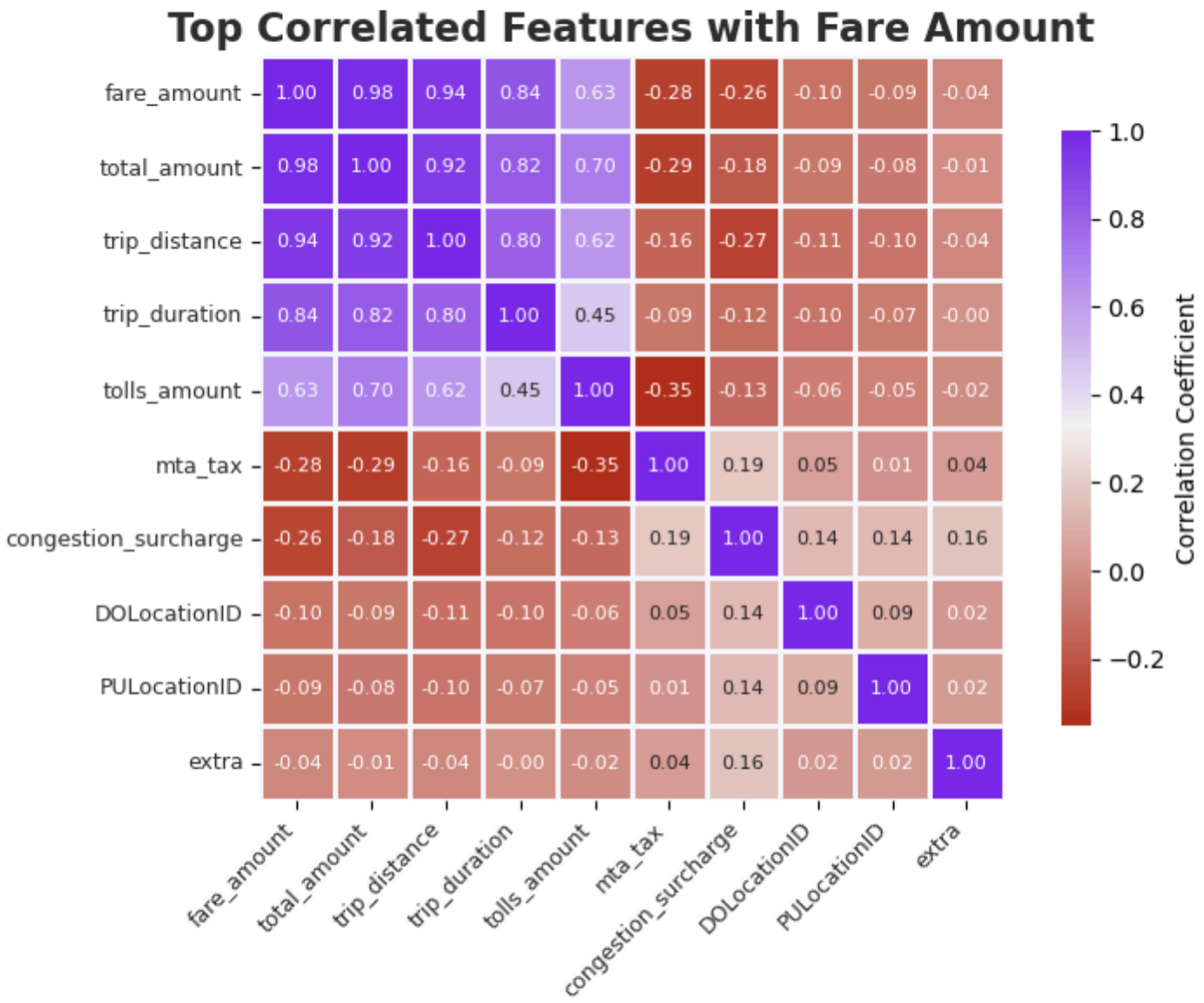

Figure: Bar Chart of Top Correlated Features with Fare Amount

10. Average Fare by Hour and Day of Week

To explore how fare prices vary not just by hour but also across days of the week, I created a **heatmap** that visualized the **average** fare amount for each combination of pickup hour (0-23) and pickup day of week (0=Monday to 6=Sunday).

The resulting heatmap revealed distinct **temporal pricing patterns**. Weekday mornings (especially around 5-6AM) and **late evenings on weekends** consistently showed elevated average fares. These peaks coincide with:

- **Early airport runs** on weekday mornings
- **Nightlife and leisure travel** on Friday and Saturday nights

Conversely, midday hours on weekdays showed the **lowest average fares**, reflecting short intra-city rides during routine hours.

This visualization provided strong justification for including both pickup hour and pickup day of week as features in the predictive model. It also gave further support for **dynamic surcharge logic**–especially for extras during rush hours or late-night windows.

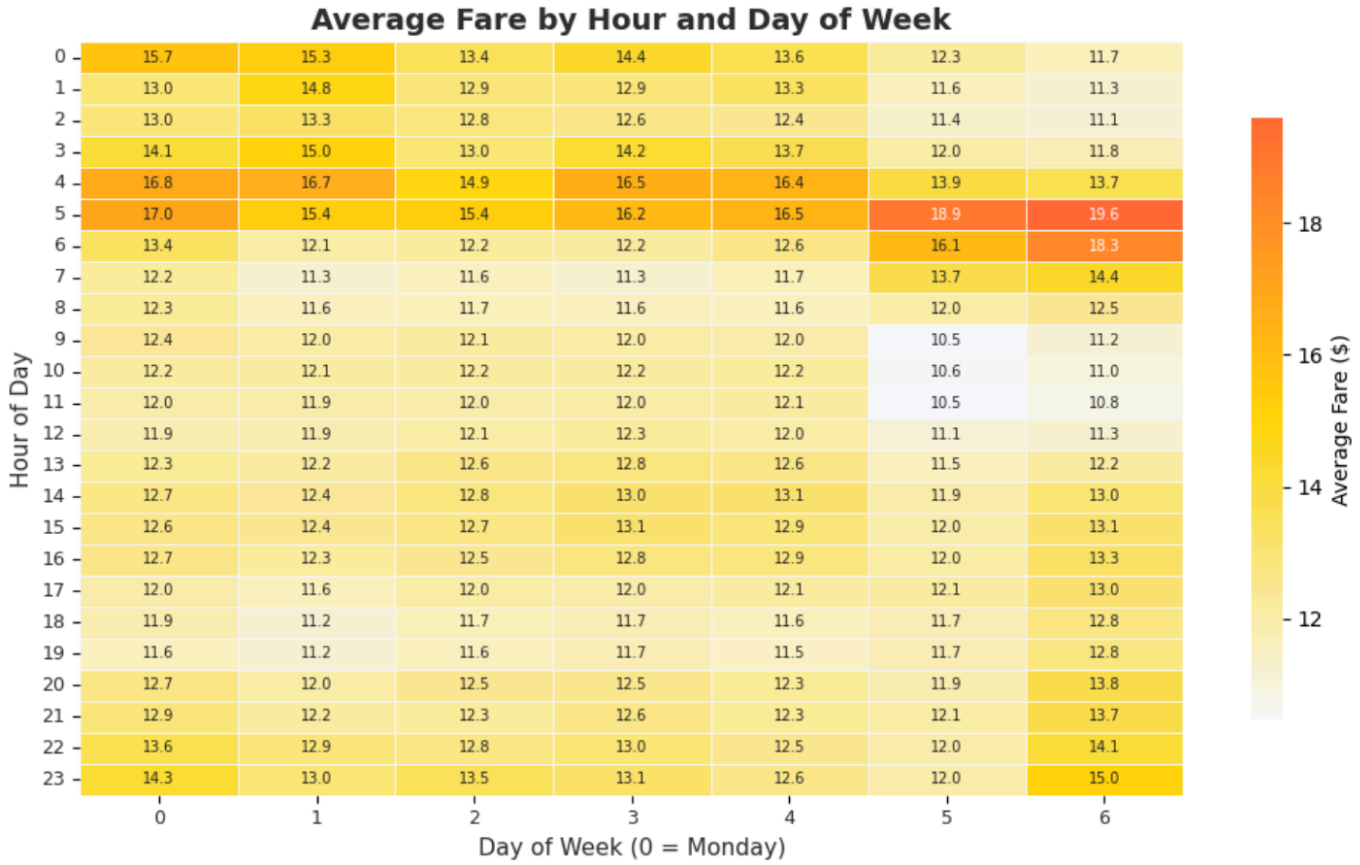

Figure: Heatmap of Average Fare by Hour and Day of Week

11. Trip Volume by Hour of Day

Understanding when NYC taxis are most active can reveal patterns in rider demand and driver availability. To explore this, I grouped the data by pickup hour and plotted a **bar chart** showing the total number of trips taken during each hour of the day.

This plot showed that **trip volume ramps up quickly in the early morning**, peaks between **4:00PM and 6:00 PM**, and gradually tapers off after midnight. This pattern aligns closely with NYC's workday rhythm: morning commutes, lunch breaks, evening commutes, and a slowdown into the night.

Interestingly, even **late-night hours (12-2 AM)** show moderate volume, likely driven by nightlife and hospitality traffic. The **lowest trip volume** occurred around **4:00-5:00 AM**, when both rider activity and taxi availability are at their daily lows.

This visualization was helpful in understanding **demand pressure** across time and reinforced the use of pickup hour for both fare estimation and rule-based logic for extras.

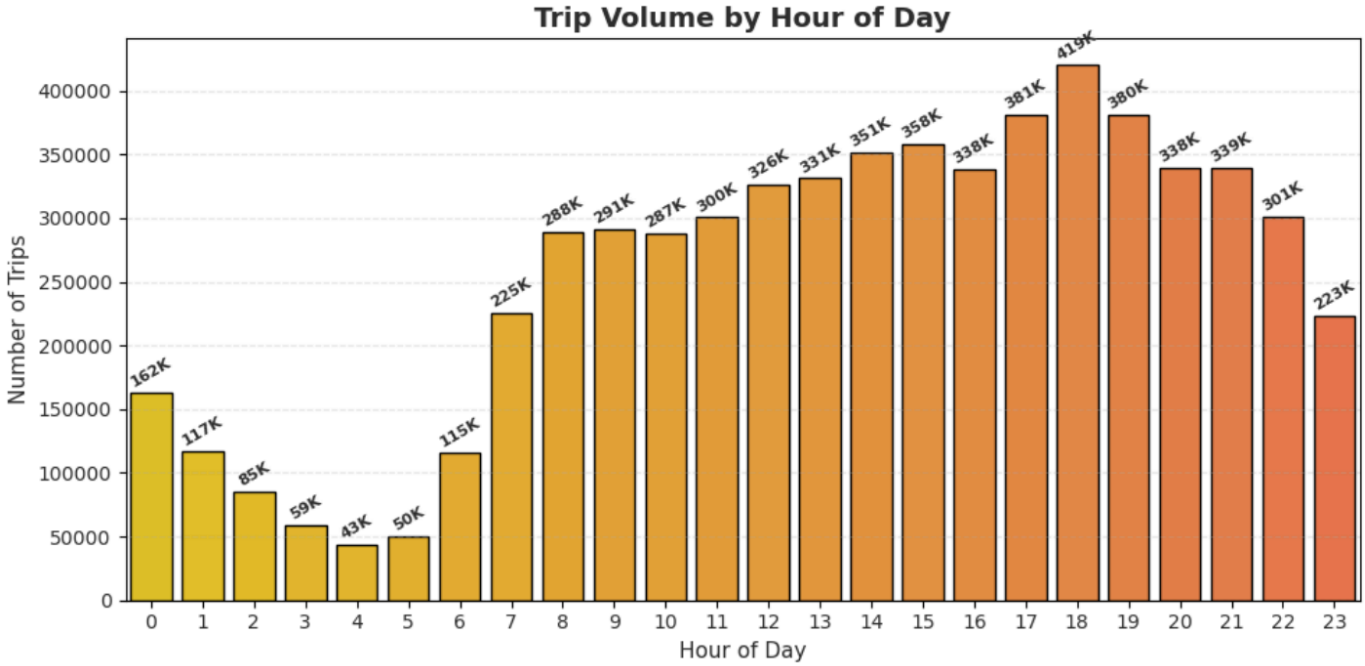

Figure: Bar Chart of Trip Volume by Hour of Day

12. Trip Volume by Day of Week

To understand broader patterns in weekly ride demand, I grouped trips by pickup day of week and plotted a bar chart of total trip volume for each day.

The results showed that Fridays and Saturdays consistently had the highest trip counts, reflecting increased social activity, nightlife, and weekend travel. Mondays and Tuesdays, on the other hand, had slightly lower volume, which aligns with quieter weekday routines. Midweek days like Wednesday and Thursday maintained steady, moderate demand.

Sunday trip volume remained fairly high, likely due to late-night Saturday rides continuing past midnight and travelers returning before the work week. This week-level analysis was essential to contextualize rider behavior and further justified the use of pickup day of week as an input feature in the model.

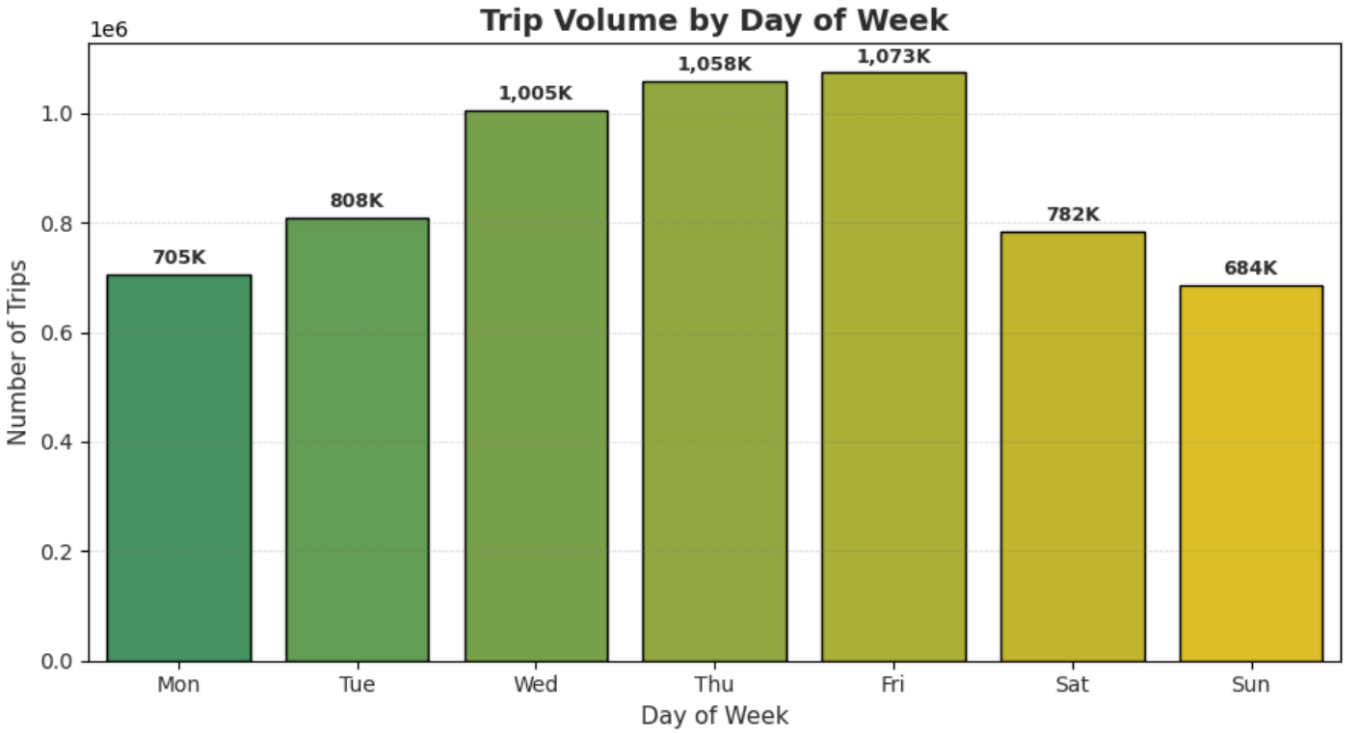

Figure: Bar chart of Trip Volume by Day of Week

⇒ Methodology

To build a reliable fare prediction system, I experimented with multiple supervised learning algorithms–starting from simple linear models to more advanced tree-based approaches. The objective was to identify a model that could capture both the linear and non-linear relationships between trip attributes (such as distance, duration and time features) and the base fare amount.

I began with baseline models like Linear Regression, Ridge, and Lasso, which allowed me to quickly evaluate linear dependencies and understand how regularization impacted feature weighting. While these models were interpretable and computationally efficient, they struggled to fully capture the complexity in fare dynamics, particularly around flat-rate rides and valuable urban traffic patterns.

To better address these complexities, I then moved to tree-based models, specifically the Decision Tree Regressor and finally the Random Forest Regressor. These models offered greater flexibility in modeling non-linear interactions, handling categorical variables, and automatically detecting thresholds. After comparing performances across all models, I selected a well-tuned Random Forest as the final model due to its superior accuracy and generalization on the validation set.

The following sections details the implementation, performance, and takeaways from each model in the order they were tested.

- **Linear Regression**

  I started with Linear Regression as a baseline model to establish a simple benchmark for predicting fare amount. This model assumes a linear relationship between the predictors and the target variable and is widely used for its interpretability and speed.

  For this model, I used features such as trip distance, trip duration, pickup hour, pickup day of week, pickup month, is weekend, and encoded categorical variables ratecodeID, payment type, PULocationID, DOLocationID.

  After one-hot encoding categorical variables and scaling the dataset where needed, I trained the model and evaluated its performance using standard regression metrics:
  - **MAE (Mean Absolute Error) :** $99554.27
  - $\boldsymbol{R^2}$ **Score :** -1286253121544.4221

  Despite its simplicity and interpretability, the model performed poorly in capturing real-world fare dynamics. The underlying assumption of a strictly linear relationship between the predictors and the fare amount did not hold up in practice.

  The R^2 score was very low, and the predicted vs actual scatterplot showed wide variance, especially for mid and high-fare trips. Many predictions were tighly

compressed around the mean, ignoring both short high-fare rides and long low-fare ones. This revealed that the model lacked the flexibility to capture even moderately complex pricing behavior.

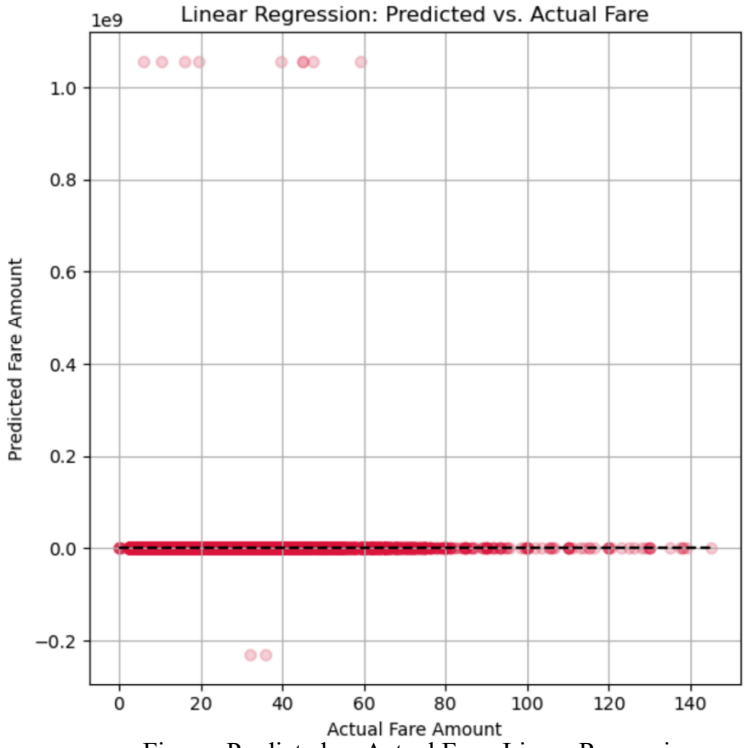

Figure: Predicted vs Actual Fare–Linear Regression

- **Ridge Regression**

  After the catastrophic failure of basic Linear Regression, I turned to Ridge Regression, which adds L2 regularization to the linear model. This approach penalize overly large coefficients, helping stabilize the model in the presence of multicollinearity and high-dimensional categorical variables, especially from one-hot encoding.

  Using the same cleaned and engineered feature set–including trip distance, trip duration, pickup hour, RatecodeID, payment type, and location IDs–I trained the Ridge model with 5-fold cross validation. After hyperparameter tuning, the best model used alpha = 10 and achieved outstanding results on the test set:

  - **Mean Absolute Error (MAE) : $0.43**
  - **R^2 Score: 0.9698**

  These results marked a major improvement over the previous model. The predicted vs actual plot showed points clustered closely around the ideal 45-degree line, and the residual distribution was tightly centered around zero. This confirmed the Ridge Regression was able to capture the fare structure well using only engineered features and proper regularization.

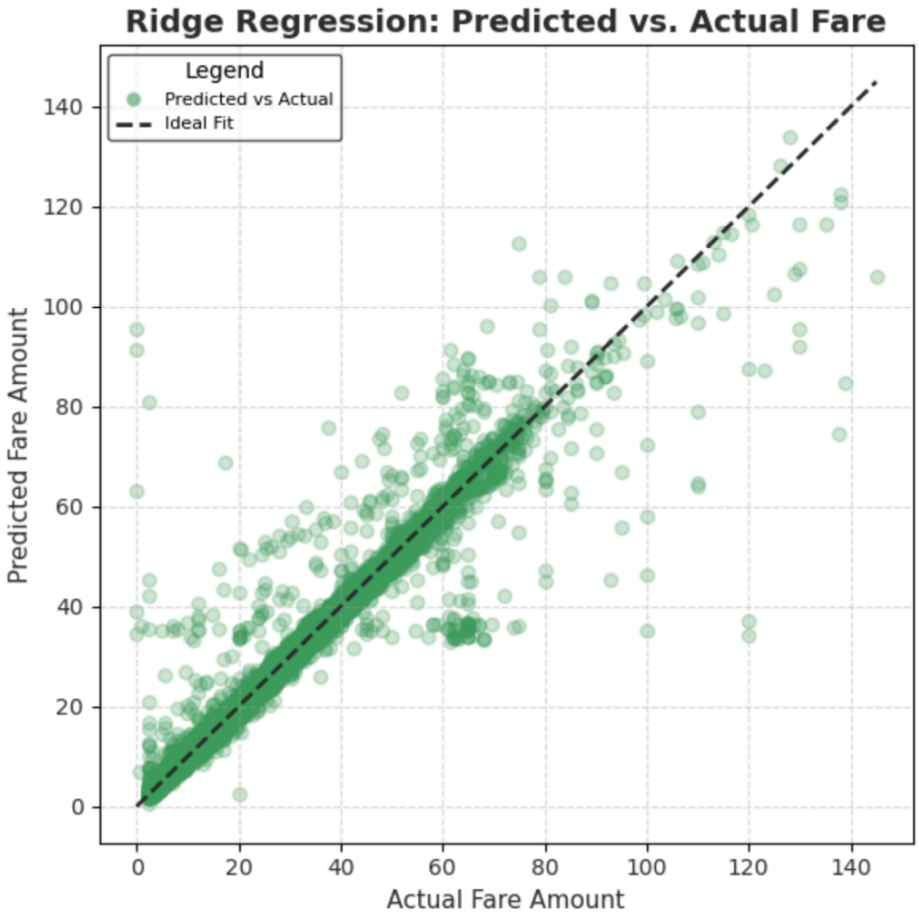

Figure: Predicted vs Actual Fare–Ridge Regression

- **Lasso Regression**

  After evaluating Ridge Regression, I proceeded to implementing Lasso Regression, a liner model with L1 regularization that not only penalizes large coefficients but also shrinks less informative ones to zero, effectively performing automatic feature selection. This makes Lasso particularly useful when interpretability and model simplicity are priorities.

  I used the same cleaned and preprocessed feature set as in Ridge, and tuned the alpha parameter through 5-fold cross-validation. The optimal alpha was found to be 0.01, and the model achieved:

  - **Mean Absolute Error (MAE): $0.44**
  - **R^2 Score: 0.9670**

  These results were nearly on par with Ridge Regression, with slightly lower accuracy but improved model simplicity. The predicted vs actual fare plot showed a strong diagonal alignment, indicating consistent predictions with minor under-or over-estimations on extreme values. More importantly, the residuals were tightly centered around zero, confirming the model's low bias and stable generalization. Additionally, I examined the top coefficients from the trained Lasso model. As

expected, many unimportant one-hot encoded location features were eliminated, leaving behind a cleaner and more interpretable subset. Key predictors included trip distance, trip duration, and specific RatecodeID and payment type categories.

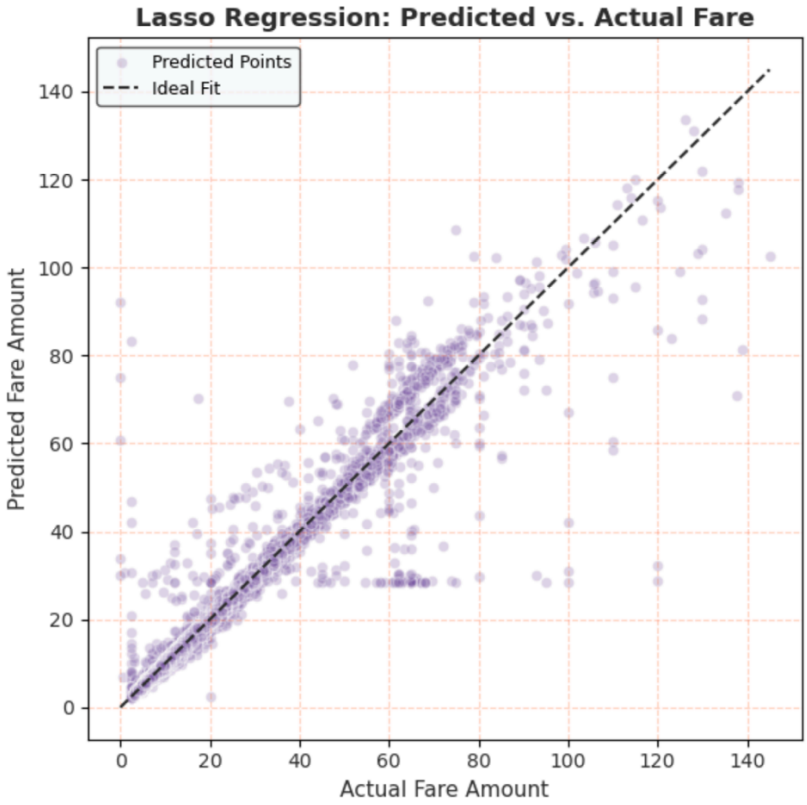

Figure: Predicted vs Actual – Lasso Regression

- **Decision Tree Regression**

  After testing regularized linear models, I transitioned to a non-linear approach using a Decision Tree Regressor. Decision trees have the advantage of modeling complex, rule-based relationships without requiring feature scaling or linear assumptions–making them a strong candidate for fare prediction, where pricing logic includes distance, duration, time, and fixed-rate policies.

  To avoid overfitting, I implemented a pipeline with preprocessing and hyperparameter tuning. A grid search across various max depth values revealed that a depth of 10 produced the best balance between bias and variance.

  The tuned Decision Tree Regressor delivered excellent performance:

  - **Mean Absolute Error (MAE): $0.38**
  - **R^2 Score: 0.9711**

  This marked the best result so far on the sampled dataset, outperforming Ridge and Lasso. The predicted vs actual fare plot showed tight clustering around the ideal line, while the residuals were centered around zero and limited spread–indicating strong generalization without severe overfitting.

In addition, I visualized the top 15 most important features based on the tree's internal structure. As expected, trip distance and trip duration emerged as the dominant predictors, with categorical variables like payment type, ratecodeID, and pickup hour also playing meaningful roles.

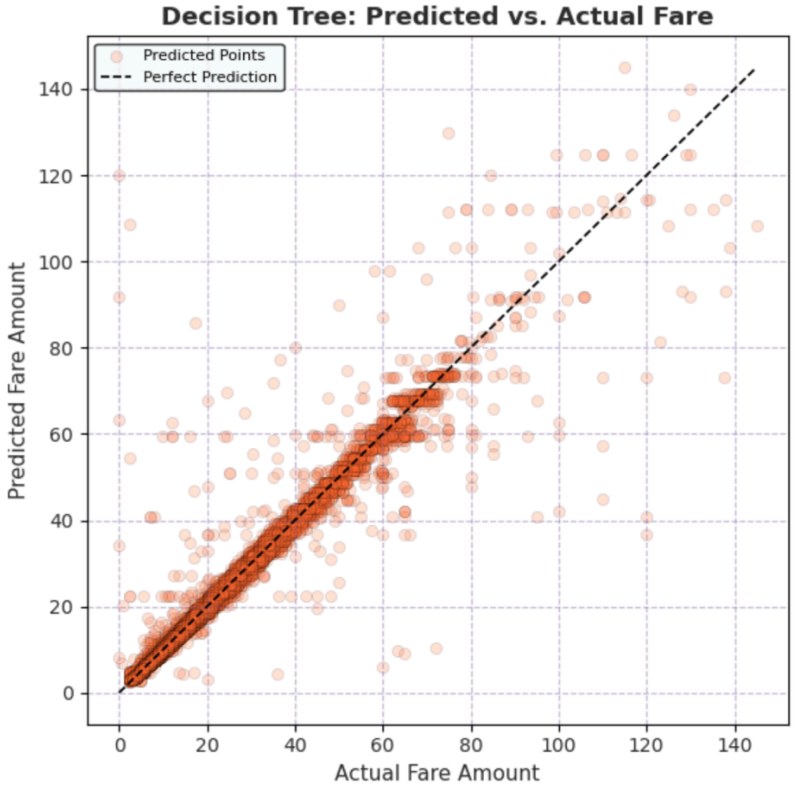

Figure: Predicted vs Actual–Decision Tree Regression

- **Random Forest Regression (Final Model)**

  After comparing linear and tree-based models, I selected Random Forest Regression as the final modeling approach for fare prediction. Random Forests, being an ensemble of decision trees, are highly effective at capturing non-linear relationships, handling mixed feature types, and avoiding overfitting through averaging across multiple trees.

  I initially trained a basic untuned model on 500,000 samples, which already delivered impressive results:

  - Mean Absolute Error: $0.31
  - R^2 Score: 0.9756

  Next, I performed hyperparameter tuning using a grid search over n_estimators and max depth on a smaller subset (100,000) rows to reduce computation time. The best parameters (n_estimators = 50, max depth = 20) were then used to train the model on the full dataset again. The final tuned Random Forest model achieved:

- **Mean Absolute Error (MAE) : $0.33**
- **R^2 Score: 0.9767**

To further evaluate model behavior, I plotted the Predictive vs Actual Fare graph, which showed predictions clustering tightly around the diagonal, indicating consistent accuracy. The residual distribution was symmetric and narrow, with minimal bias. I also visualized the top 15 most important features based on both native feature importances and permutation importance. As expected, trip distance, trip duration, and RatecodeID were the most predictive features, supported by time-of-day signals like pickup hour.

Additionally, I generated a 2D decision surface plot using trip distance and trip duration, which revealed smooth fare gradients and alignment with domain knowledge–longer trips and longer durations yielded higher predicted fares, reinforcing the model's interpretability.

Random Forest captured both non-linear pricing and business rule logic. Final model was saved for deployment.

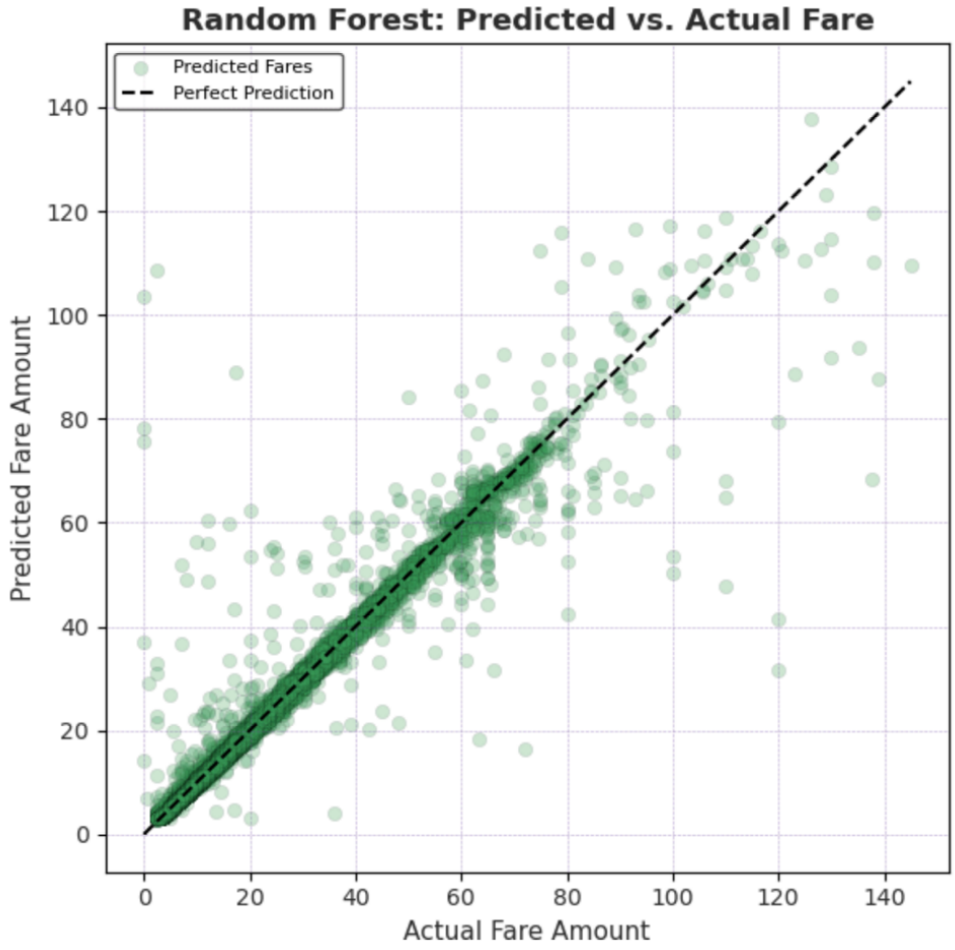

Figure: Predicted vs Actual Fare–Random Forest

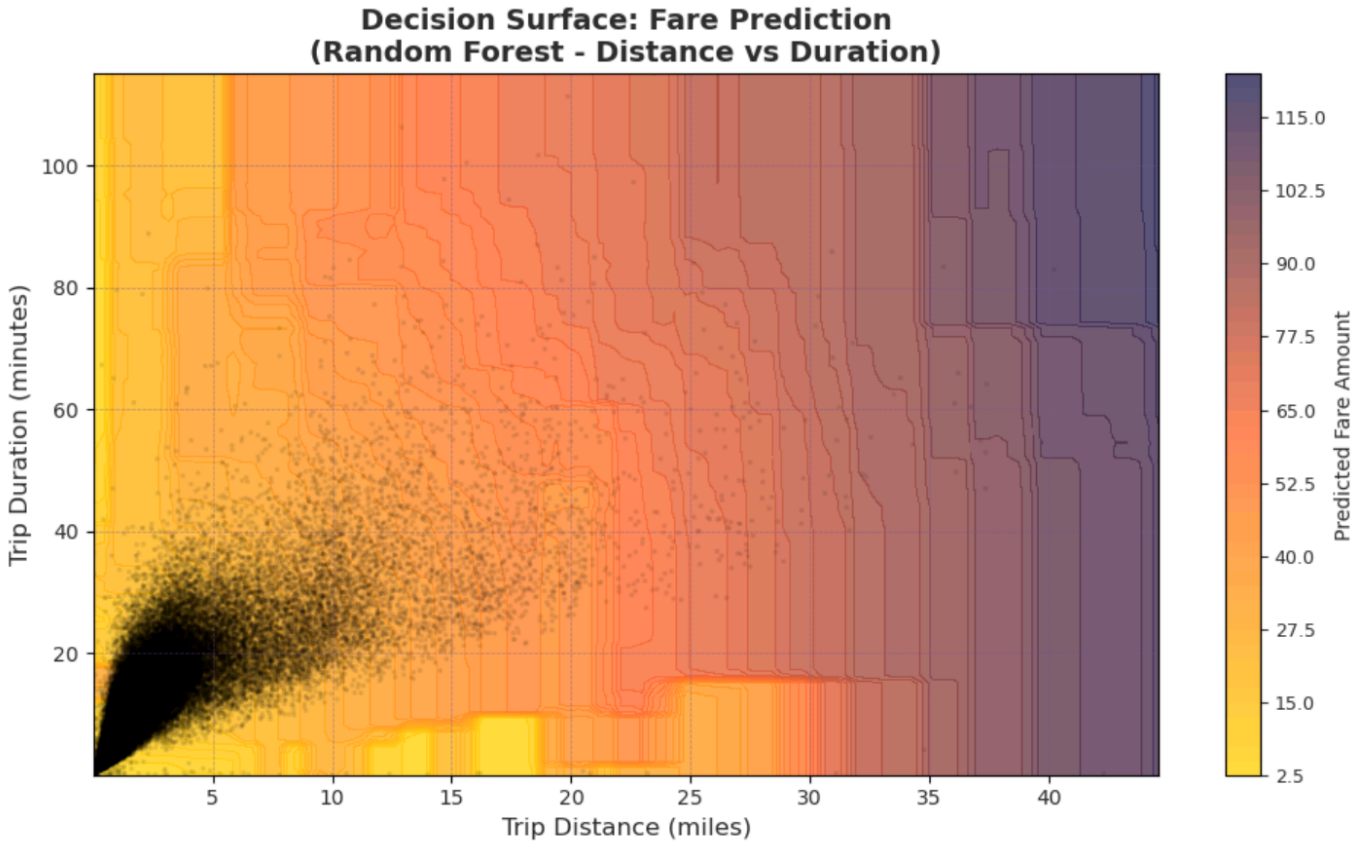

Figure: 2D Decision Surface (Distance vs Duration)

- **Special Case Handling: JFK Flat-Rate Logic**
  While building the fare prediction model, one unique challenge emerged: JFK airport rides, which are governed by fixed fare policy in NYC. Specifically, all trips between JFK Airport and Manhattan follow a flat rate of $52, regardless of trip distance or duration. This rule is represented in the dataset by RatecodeID=2.
  During exploratory analysis, I observed a distinct horizontal cluster at $52 in both the fare amount vs trip duration and fare amount vs trip distance plots. These rides appeared as clear outliers in regression modeling because no machine learning model could predict a flat fare based on varying trip characteristics. Including these points in training led to significant distortion in model behavior.

  To address this, I implemented a hybrid modeling approach. The final fare predictor pipeline applies:
  - A rule-based override for rides with RatecodeID = 2, automatically setting fare amount = 52.
  - The trained Random Forest model for all other metered trips

This ensured that JFK fares were treated with full regulatory accuracy, and it prevented the model from learning a misleading average or misinterpreting flat-rate rides as noisy inputs.

In addition, the fare predictor module was designed to detect JFK trips dynamically based on input conditions, so even if RatecodeID is missing, business rules can still be triggered based on pickup/drop-off zones in future iterations.

⇒ Results

After experimenting with multiple regression models, I compiled a final comparison of their predictive performance using two key metrics: Mean Absolute Error (MAE) and R^2 Score. These results were obtained using a consistent dataset of 500,000 samples (or appropriately subsetted when needed), ensuring fair comparison across all models.

The Linear regression model served as a baseline and performed poorly, severely underfitting the data. Ridge and Lasso offered significant improvements through regularization, with Ridge slightly outperforming Lasso in both accuracy and stability.

The transition to non-linear models showed dramatic gains: the Decision Tree Regressor captured complex patterns with strong accuracy but remained prone to overfitting at deeper levels. Ultimately, the Random Forest Regressor delivered the best balance of performance and generalization, achieving the lowest MAE ($0.33) and the highest R^2 (0.9767).
This model was selected as the foundation for the deployed fare prediction system, and its performance was further enhanced with hybrid logic for special cases like JFK flat fares.

The table below summarized the performance of each model:

Model	MAE ($)	R^2 Score
Linear Regression	99554	-1286253121544
Ridge Regression	0.43	0.9698
Lasso Regression	0.44	0.9670
Decision Tree	0.38	0.9711
Random Forest	**0.33**	**0.9774**

- **Total Amount Estimation via Hybrid Logic**
  While predicting the base fare amount was the primary modeling task, providing a realistic estimate of the total fare charged to passengers required incorporating additional components. These include fixed fees, surcharges, and taxes that are not always predictable based on trip features alone but follow specific business rules enforced by the NYC Taxi & Limousine Commission.

  To solve this, I designed a hybrid rule-based strategy to estimate the total amount by combining the model-predicted base fare with manually calculated components, as outlined below:

Component	Logic/Value
Base Fare	Predicted by Random Forest model
Extra	Time-based logic: $0.5-$3 based on hour and day
Congestion Surcharge	Fixed at $2.50 for eligible Manhattan zones
MTA Tax	Fixed at $0.50
Improvement Surcharge	Fixed at $0.30
Tolls	Set at $0 (placeholder; GPS-based in future)

All logic was encapsulated in the fare predictor module, which applied these rules after predicting the base fare using the trained Random Forest model. For example, extra was dynamically calculated base on pickup time:

- $0.50 at night (8 PM – 6 AM)
- $1.00 for weekday rush hours (4-8 PM)
- $2.50 - $3.00 for weekend evenings and late-night hours

This approach ensured:

- Consistency with NYC fare policies
- Interpretability and breakdown for riders
- Robustness against unpredictable variation in charges like tolls.

- **Web App Deployment**

To make the fare prediction system accessible to users in a real-world setting, I developed and deployed a fully functional web application. The app allows user to enter pickup and drop-off addresses, and it returns a transparent, itemized fare estimate using the trained Random Forest model and the hybrid total amount logic.

The system architecture is modular, lightweight, and designed for scalability.

Frontend: HTML, CSS, JavaScript

- Users interact through a clean and intuitive form interface
- Built-in Google Places Autocomplete API for seamless address entry
- Displays predicted fare and total amount clearly with labeled breakdown

Backend: Flask API

- A Flask application handles incoming requests
- Parses user inputs and interacts with:
  - fare predictor module (hybrid logic)
  - Pre-trained Random Forest model
- Communicates with the Google Directions API to:
  - Fetch trip distance and duration between selected points
  - Provide accurate inputs to the model

Prediction Workflow

1. User enters pickup and drop-off locations
2. Directions API returns estimated distance and duration
3. Fare predictor module checks for special cases
4. Model predicts base fare
5. Rule logic applies surcharges and taxes
6. Total amount is computed and returned to frontend

Deployment & Accessibility

- Application was tested locally and designed for easy hosting (Render)
- Clean separation between presentation layer and logic layer allows for future extensions (e.g., admin dashboard)
- Link to access the live demo (Live Demo)

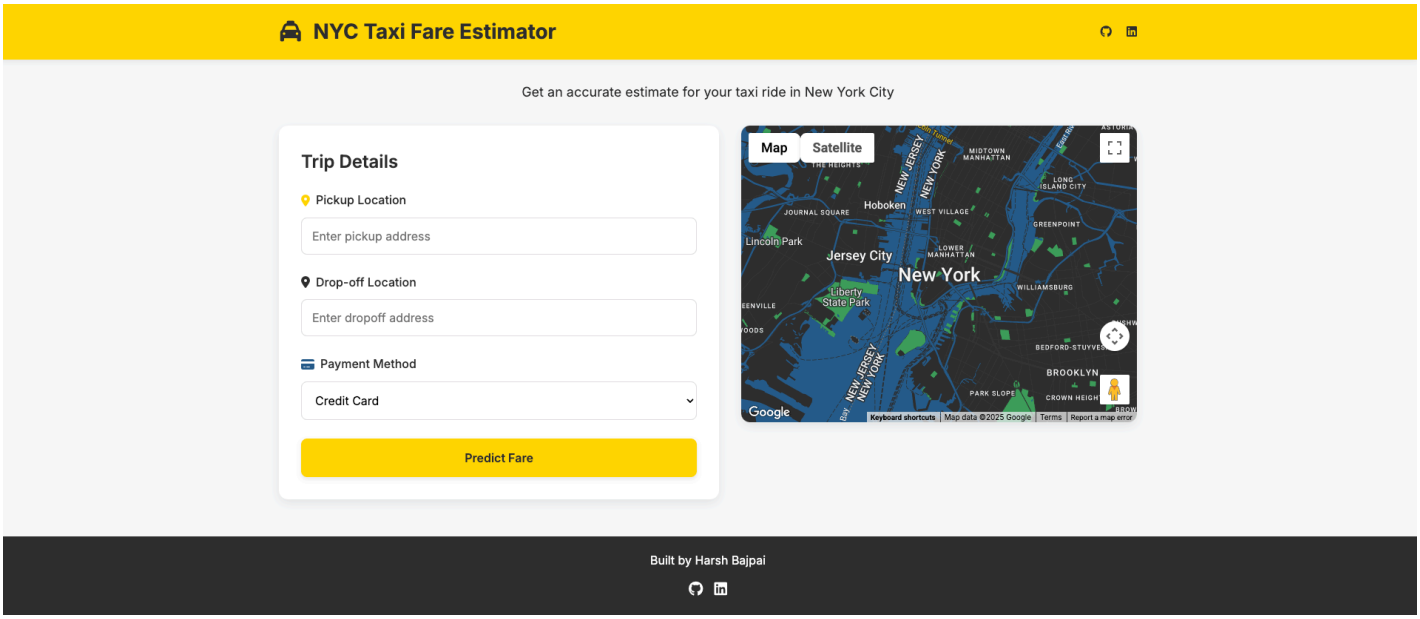

Figure: Screenshot of Web Application UI (Demo)

- **Revenue Optimization & Payment Method Analysis**
  In the second phase of the project, I explored whether payment method influences revenue, with the goal of identifying patterns that could help drivers or operators optimize earnings. Specifically, I compared Credit Card vs Cash transactions to see if one led to higher averages fares or longer trips–and whether customer behavior differed between the two.

  Visual insights: Fare & Distance Distributions
  Two side-by-side histograms were used to compare the distribution of:
    - Fare Amounts by payment type
    - Trip Distances by payment type

  These revealed consistent patterns:

    - Card-paying customers tended to take longer and more expensive rides
    - Cash-paying customers were concentrated in lower fare and distance bands

  This suggests that card users are more likely to travel longer distances, which naturally results in higher fares.

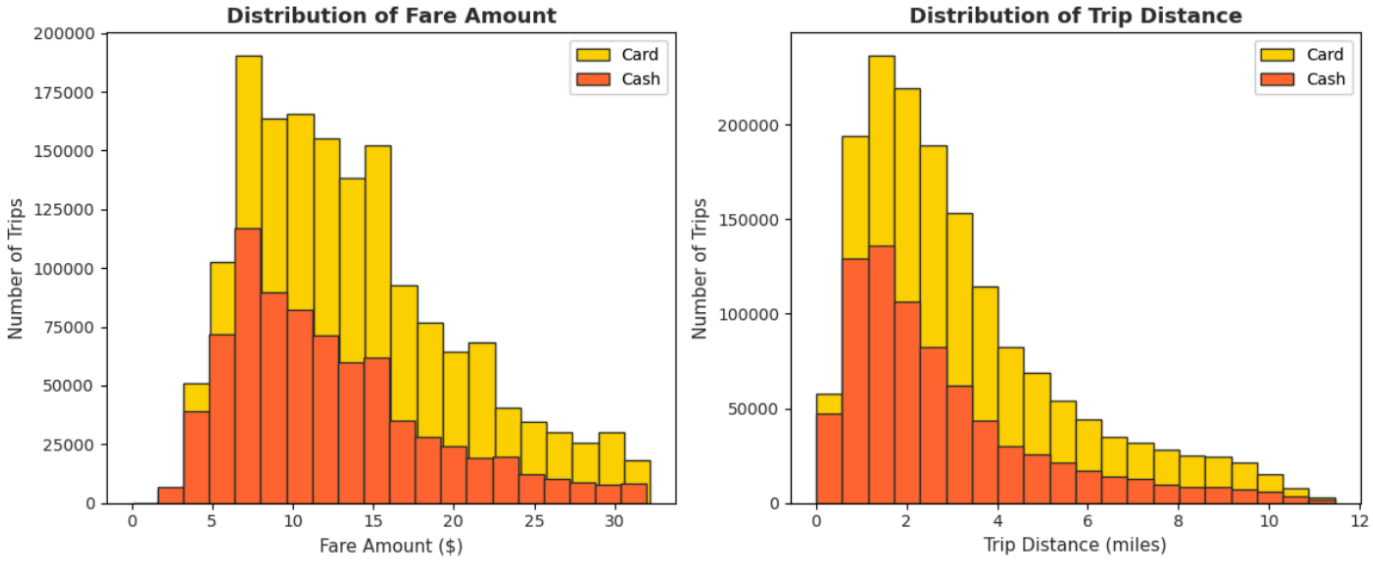

Figure: Fare Amount and Trip Distance Distribution by Payment Type

Statistical Validation: Two-Sample T-Test
To test if these differences were statistically significant, I performed a two-sample t-test comparing the average fare amount for card vs cash rides. The null hypothesis stated that there was no difference between the means.

- T-statistic: 165.60

- P-value: 0.0000

With a p-value well below 0.05, I rejected the null hypothesis, confirming a significant difference in fare amounts on payment method.

Behavioral Insight: Passenger Count vs. Payment Preference
I also examined if group size affects payment method. A stacked bar chart showed:

- Larger groups (3-5 passengers) were more likely to use cash
- Solo travelers leaned more toward credit card payments

This insight could be useful for targeting promotions or adjusting service offerings by passenger demographics.

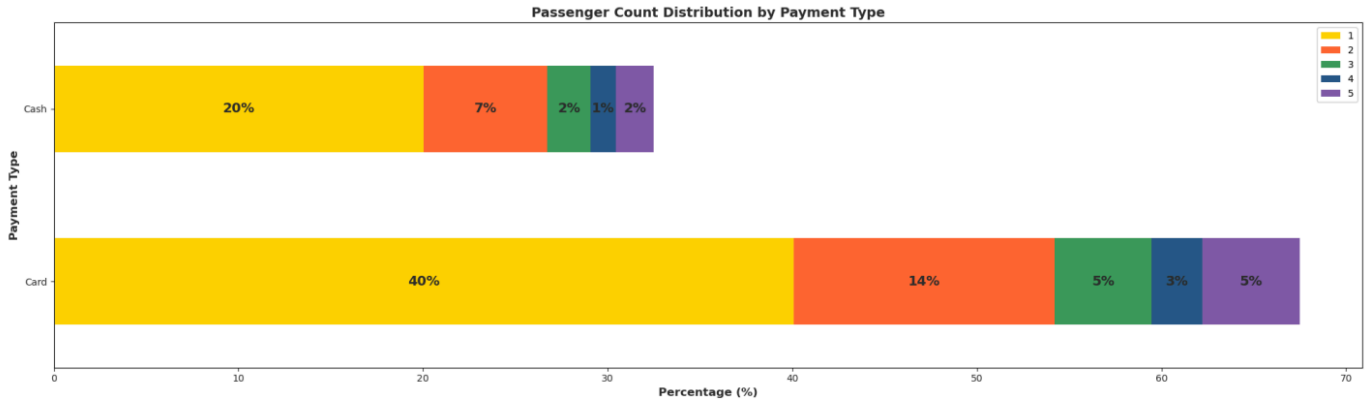

Figure: Stacked Bar Chart- Payment Type by Passenger Count

⇒ Conclusion

This project tackled the real-world challenge of **unpredictable and opaque taxi pricing** in New York City by building a transparent, hybrid fare prediction system. Starting with 6.4 million rides from the NYC TLC dataset, I applied extensive data cleaning, feature engineering, and exploratory data analysis to uncover the true structure behind urban fare dynamics.

I evaluated five regression models, ultimately selecting a **Random Forest Regression** that delivered the highest accuracy **(MAE = $0.33, R^2 = 0.9767).** To handle special cases like **JFK flat-rides**, I embedded rule-based logic directly into the prediction pipeline– resulting in a hybrid system that combines **machine learning intelligence** and **real-world policy compliance**.

Beyond base fare prediction, I extended the pipeline to estimate the t**otal amount paid by passengers**, incorporating extras like congestion surcharges, MTA tax, and late-night fees through deterministic rules. A fully functional web application was then deployed to allow users to get real-time fare prediction by entering origin and destination addresses.

Finally, I conducted a separate revenue analysis based on **payment methods**, revealing that **credit card rides tend to generate more revenue** than cash–an insight backed by statistical testing and visualization.

This project demonstrates not only technical rigor but also **business awareness, real-world deployability, and user empathy**–delivering a tool that could be valuable to passengers, drivers, fleet operators, and even city planners.

⇒ Future Work

While the current system is robust and reliable, there are several opportunities to extend its capabilities:

1. **Toll Prediction Based on GPS Route**
   Integrate real-time toll detection using route geometry from Google Directions API or OpenStreetMap, allowing accurate toll inclusion in total amount.

2. **Weather and Traffic-Aware Fare Adjustments**
   Enhance model accuracy by incorporating real-time weather conditions or congestion indices–factors that can influence fare indirectly through delays or route changes.

3. **Personalized Recommendations**
   Add features to suggest optimal times to travel, best payment method, or highlight surge-prone hours for cost-conscious riders.