OptimalUberFarePricePrediction

PREDICTING OPTIMAL UBER FARE PRICES

Group 33
Dr. M. Mahdi Roozbahani
ML 7641
03 December 2024

INTRODUCTION & BACKGROUND

This project analyzes Uber's fare pricing using various models and a dataset to understand price fluctuations and help users determine the best time to book trips. Chao (2019) analyzes Uber's pricing in New York City to explore the relationship between factors like time of day, weather, and location with surge pricing, offering strategies to minimize costs. Chen and Sheldon (2015) study how surge pricing affects driver behavior, finding it increases driver availability and improves app efficiency. Hira, Khandelwal, and Sawarkar (2021) apply machine learning techniques to predict fares using factors such as time and traffic, helping users book at the most cost-effective time. The dataset used includes information on Uber trips, such as pickup and dropoff locations, passenger count, time, and total cost.

PROBLEM DEFINITION

Uber has transformed transportation throughout cities around the world, but it has also introduced the issue of fare unpredictability. Our project addresses the unpredictability of Uber fare prices and helps users determine the best time to book a ride. This fluctuation makes it harder for users to plan trips effectively and within budgets, which can lead to overspending on basic transportation. It can also help in anomaly detection to identify if users overspent on their trip.

The motivation for solving this issue is to help consumers make more informed decisions about their Uber usage. An accurate fare predictor would allow them to plan itineraries better and determine the optimal time for ordering rides. Additionally, this could increase transparency in ride-hailing services and be extended to other platforms like Lyft.

METHODS

Preprocessing

This project integrates several data preprocessing methods to clean and prepare the data before it is used to train the model. In the preprocessing notebook, data cleaning was utilized to drop rows with missing values or incorrect coordinates. Employing geospatial feature modification, with coordinates close to zero being marked as potentially incorrect, and the latitude and longitude values were switched out for pickup and dropoff locations where they fell out of the expected range, which ensured the location data was trustworthy for the model. For some reason, the dataset had a sizable amount of entries where the latitude and longitude values were swapped which is why this was accounted for in the preprocessing.

Next, the Haversine distance between pickup and dropoff locations was calculated, which gave a solid approximation of the trip's distance, accounting for the curvature of the Earth. Additionally, to find the time-related patterns in the data, the hour of the day, the day of the week, and the month of the year were pulled and turned to cyclic features with sine and cosine (hour, day, month). Weekends and holidays were marked as the demand for Ubers was predicted to be higher during these times, affecting fare prices. The passenger count field was utilized to see if UberXL was used instead of a standard Uber, which would be under the condition of more than four passengers riding.

ML Algorithms / Models

For the unsupervised learning model, this project implements Density-Based Spatial Clustering of Applications with Noise (DBSCAN). The motivation behind this model was that it would help separate the data into inliers and outliers; training the model with faulty data was a large concern, especially as a lot of the dataset was incomplete or had misleading values for some features such as Haversine distance and fare. DBSCAN shines at identifying the outliers and focusing on the clumped data to provide accurate predictions. For these reasons, DBSCAN was actually implemented inside the preprocessing notebook because it would make most sense to run this algorithm before the supervised learning model was run.

Another unsupervised learning model that this project implements is Isolation Forest. Isolation forests is an anomaly detection algorithm that detects and determines unusual data points. A key point about Isolation Forests is that it runs more efficiently than other outlier/anomaly detection algorithms because it analyzes and determines if points are anomalous as it iterates through the data points.

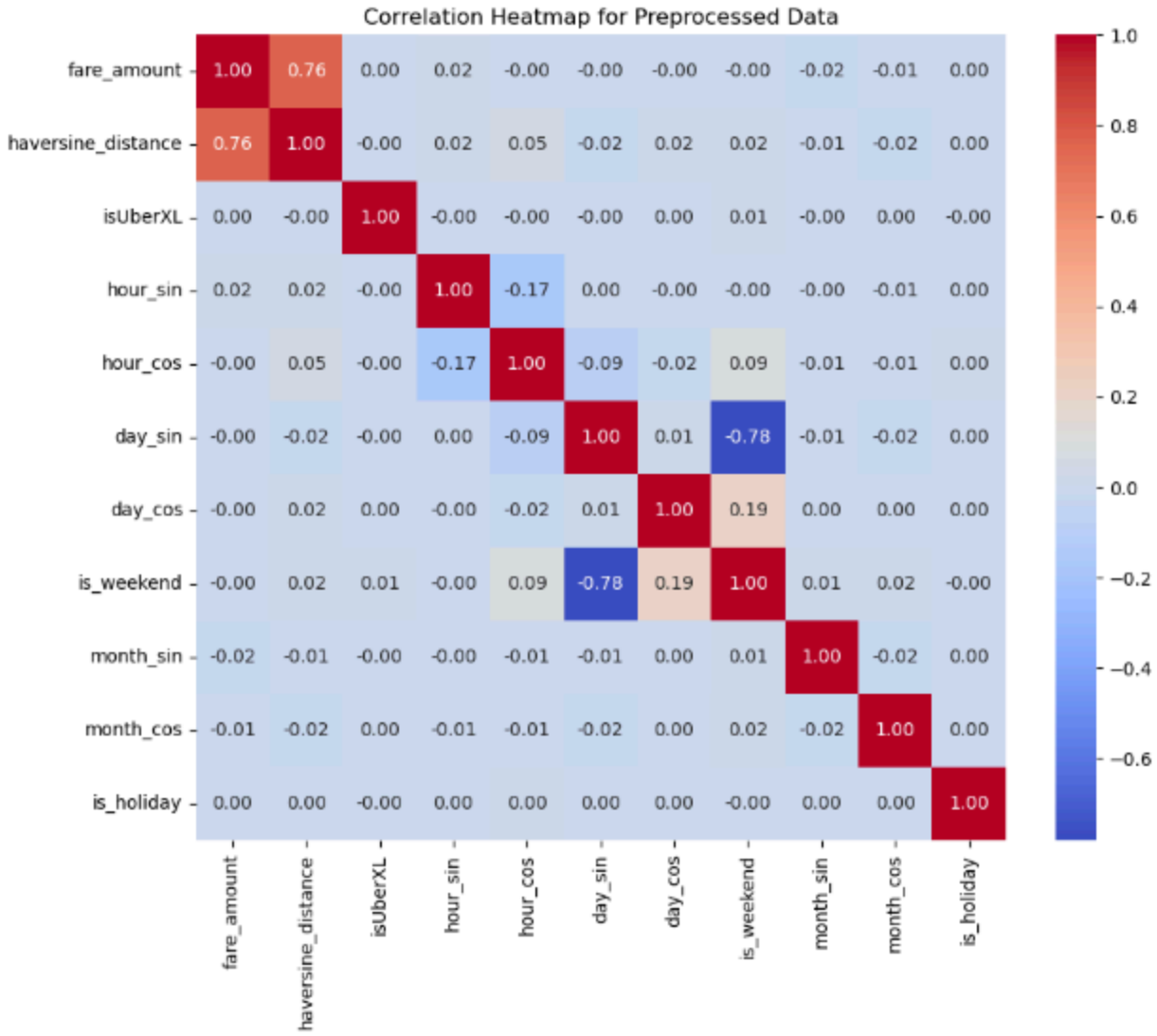
For the supervised learning model, this project also implements Support Vector Regression (SVR). Originally, Multilinear Regression was used, but the RMSE values were high and the SVR implementation using the same data produced significantly lower RMSE values. One of the motivations to use SVR was its response to outliers; a large portion of the data was identified as outliers during the preprocessing, so this model seemed fitting with the Uber dataset that it was

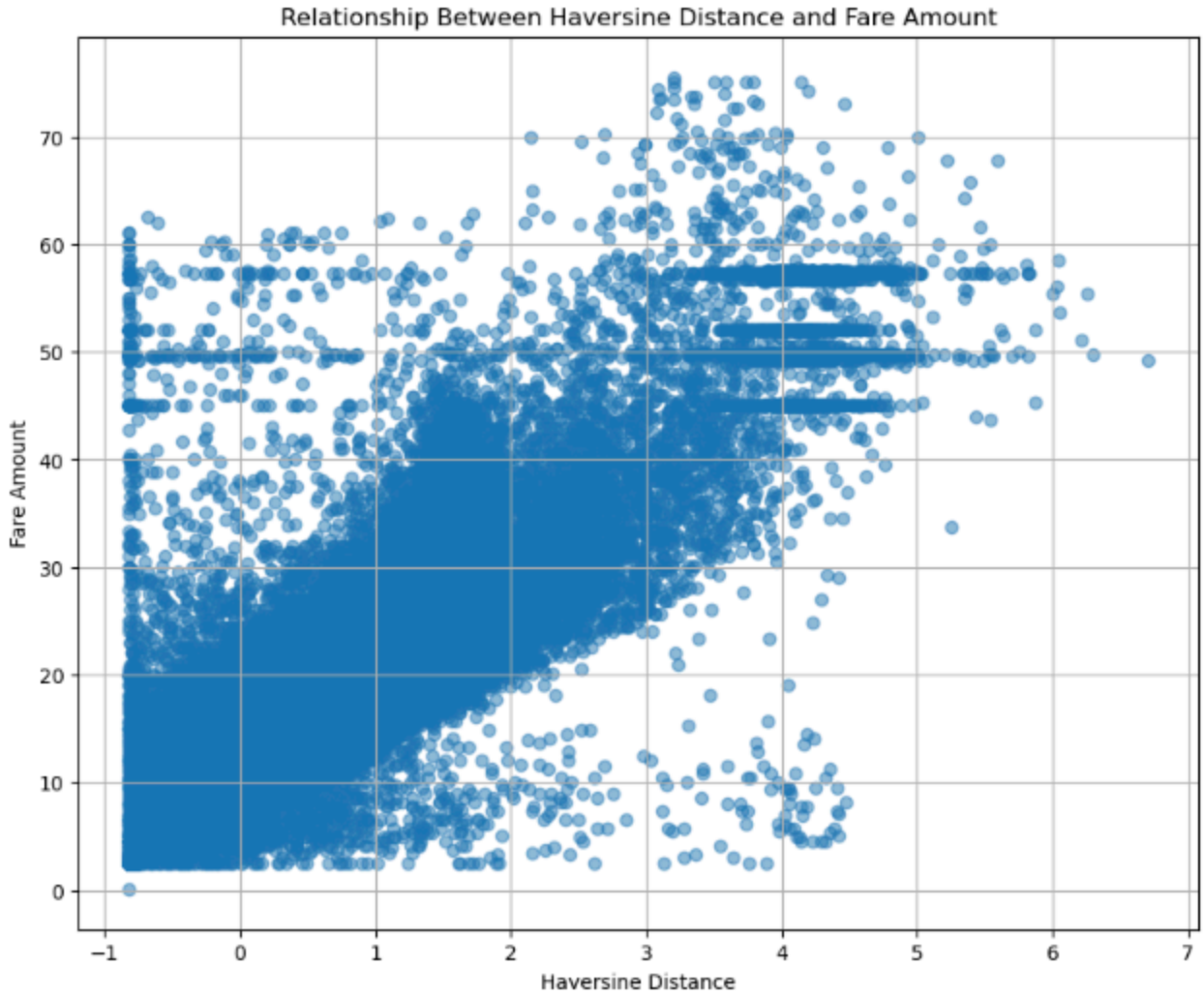
trained on. Additionally, SVR is high performing with non-linear problems, and in this project there is a complex relationship between features such as day of the week, Haversine distance, and the passenger count.

RESULTS AND DISCUSSION

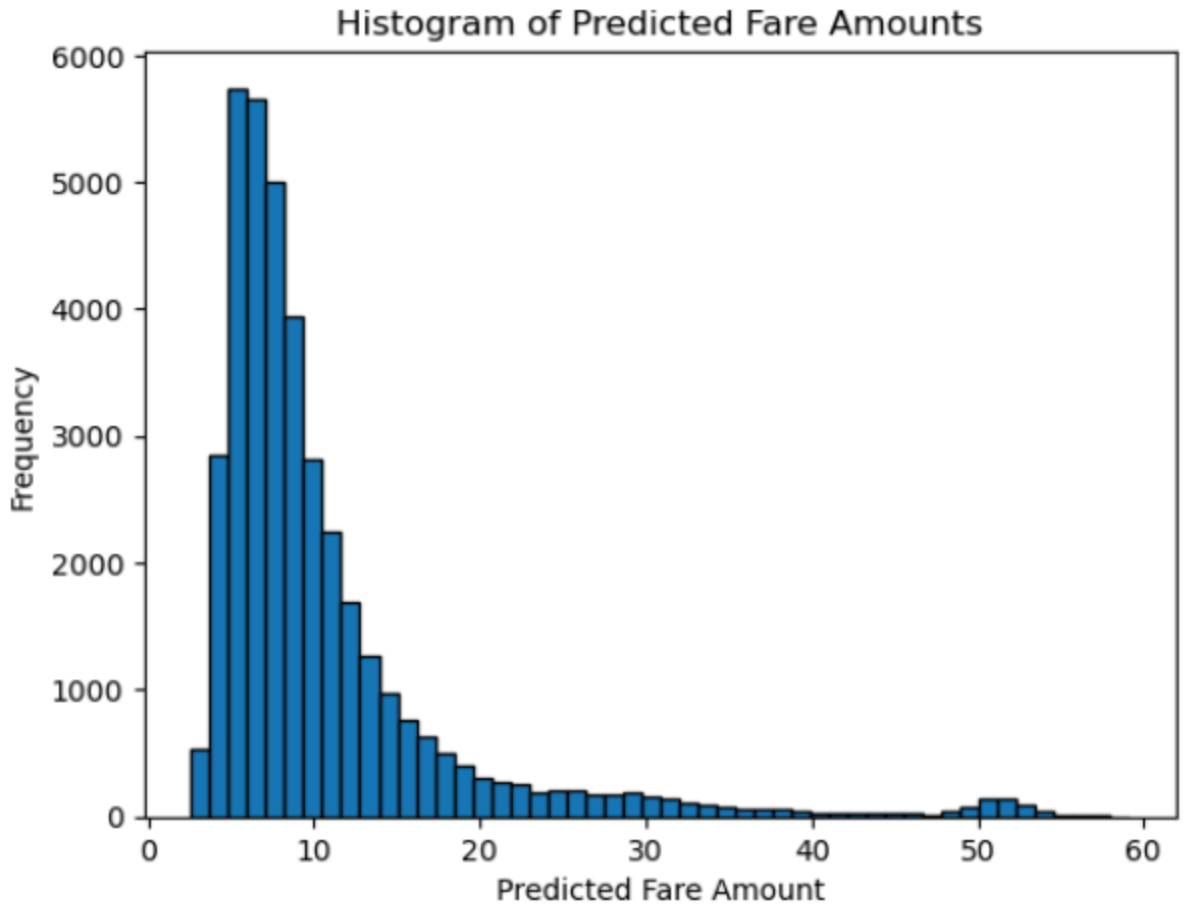
VISUALIZATIONS

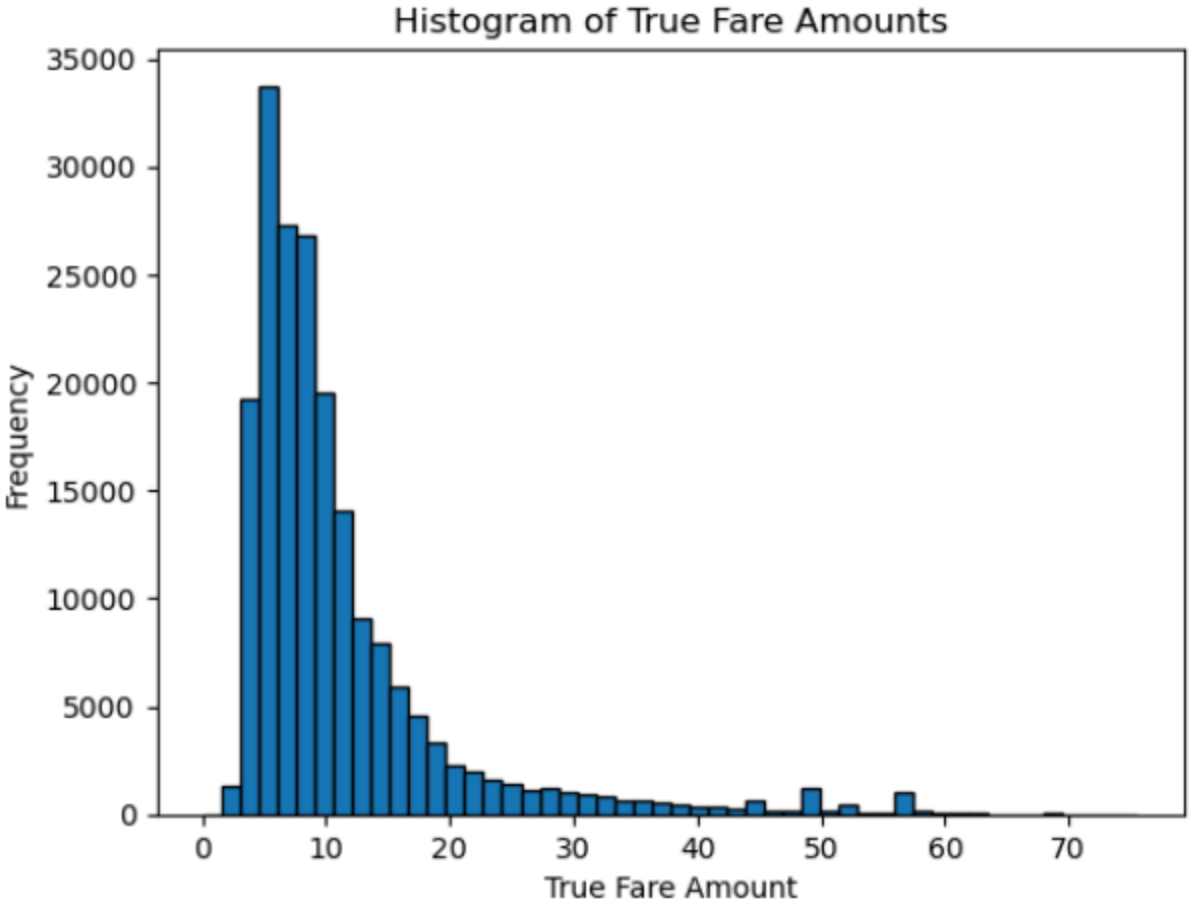
Feature Visualization


Correlation Heatmap for Preprocessed Data
fare_amount 1.00 0.76 0.00 0.02 -0.00 -0.00 -0.00 -0.00 -0.02 -0.01 0.00
haversine_distance 0.76 1.00 -0.00 0.02 0.05 -0.02 0.02 0.02 -0.01 -0.02 0.00
isUberXL 0.00 -0.00 1.00 -0.00 -0.00 -0.00 0.00 0.01 -0.00 0.00 -0.00
hour_sin 0.02 0.02 -0.00 1.00 -0.17 0.00 -0.00 -0.00 -0.00 -0.01 0.00
hour_cos -0.00 0.05 -0.00 -0.17 1.00 -0.09 -0.02 0.09 -0.01 -0.01 0.00
day_sin -0.00 -0.02 -0.00 0.00 -0.09 1.00 0.01 -0.78 -0.01 -0.02 0.00
day_cos -0.00 0.02 0.00 -0.00 -0.02 0.01 1.00 0.19 0.00 0.00 0.00
is_weekend -0.00 0.02 0.01 -0.00 0.09 -0.78 0.19 1.00 0.01 0.02 -0.00
month_sin -0.02 -0.01 -0.00 -0.00 -0.01 -0.01 0.00 0.01 1.00 -0.02 0.00
month_cos -0.01 -0.02 0.00 -0.01 -0.01 -0.02 0.00 0.02 -0.02 1.00 0.00
is_holiday 0.00 0.00 -0.00 0.00 0.00 0.00 0.00 -0.00 0.00 0.00 1.00
fare_amount
haversine_distance
isUberXL
hour_sin
hour_cos
day_sin
day_cos
is_weekend
month_sin
month_cos
is_holiday
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6

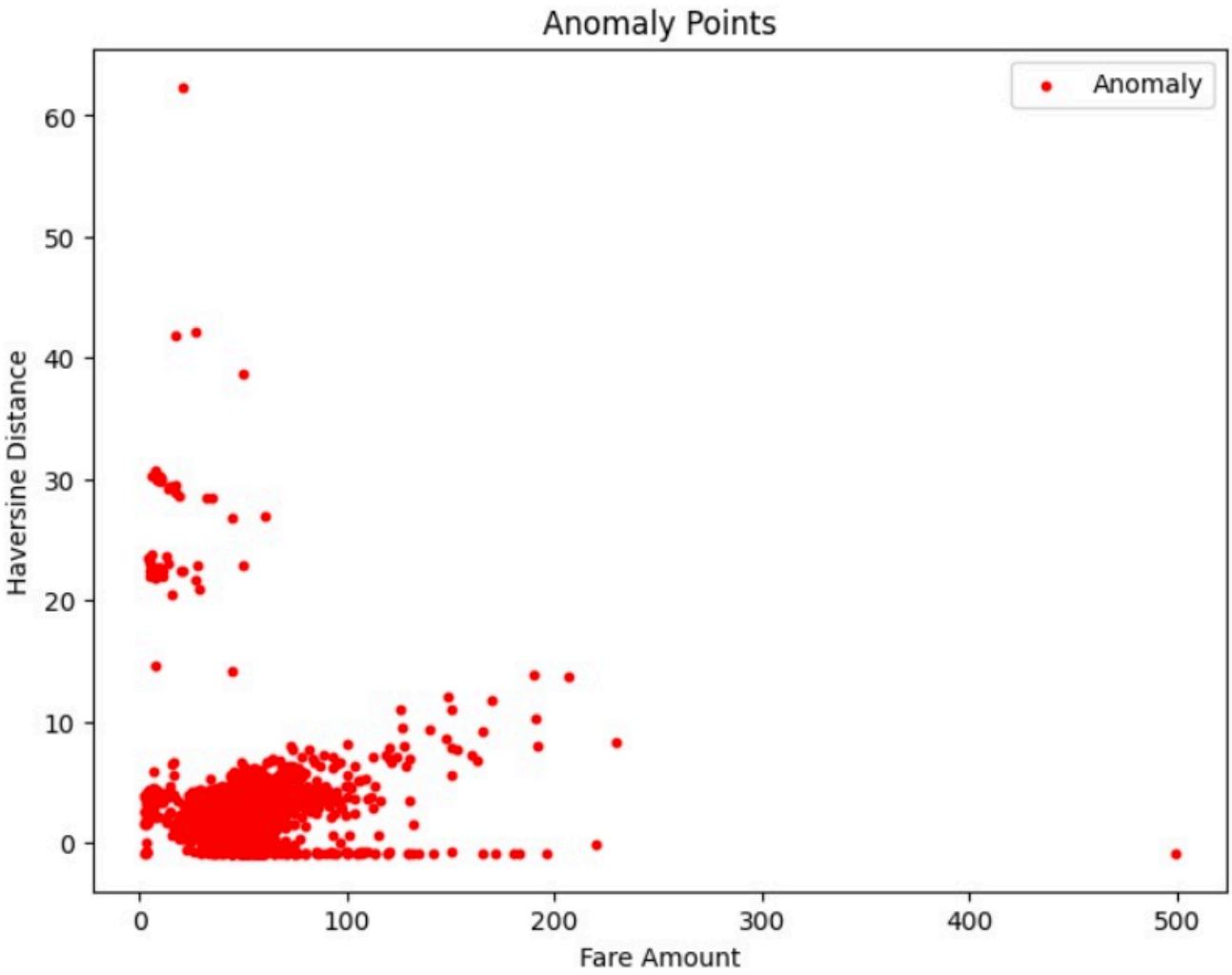


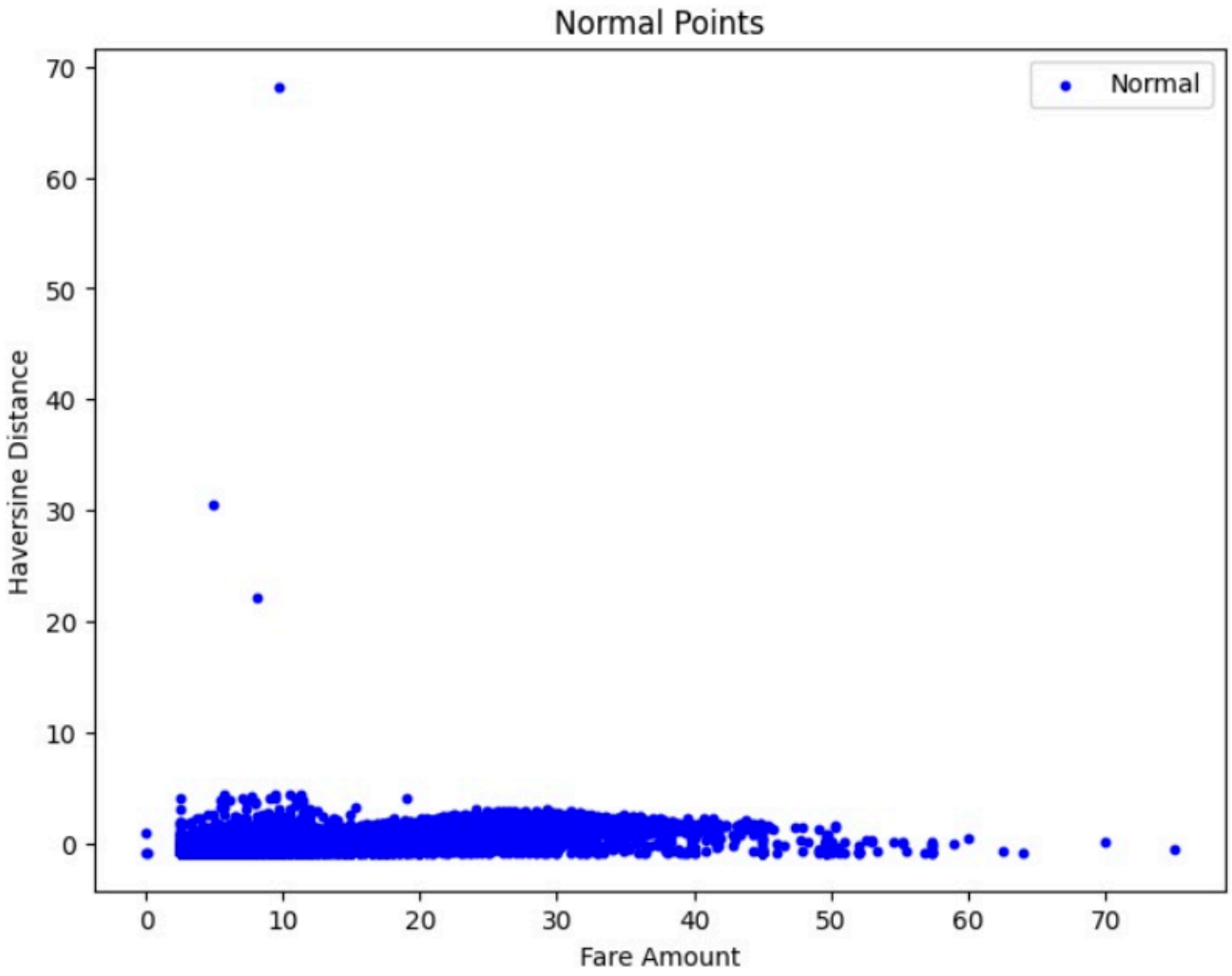
SVR Model Performance


Histogram of Predicted Fare Amounts
Frequency
Predicted Fare Amount
0
1000
2000
3000
4000
5000
6000
0
10
20
30
40
50
60



Isolation Forest Model Performance


Anomaly Points
Anomaly
Haversine Distance
Fare Amount
0
10
20
30
40
50
60
0
100
200
300
400
500



QUANTITATIVE METRICS

DBSCAN

- Inlier count: 192823
- Outlier count: 385
- Outlier percentage: 0.110%

SVR

- Mean Squared Error (MSE): 15.196
- Root Mean Squared Error (RMSE): 3.89820471
- R-squared (R^2): 0.81

ISOLATION FOREST

- Normal Data Points: 183547

- Anomaly Data Points: 9661
- Anomaly Percentage: 5% of Cleaned Dataset

ANALYSIS

The project included the application of DBSCAN (Density-Based Spatial Clustering of Applications with Noise) to identify and separate outliers from the main dataset. As Uber fare data is not free from error, such as implausible distance or car pickup coordinates which do not match between drop-off coordinates, the outliers were readily identified by DBSCAN. DBSCAN groups together points that are closely packed according to the general specifications of a neighborhood, separating regions of varying density. This method helped filter most of the unreliable, non-cohesive data points which improved prediction quality using supervised models. Out of 193,226 events, DBSCAN flagged as noise only 227 (0.117%) data points, and all others were identified as inliers. This left not too many examples of the dataset that can potentially influence model performance negatively if they are not filtered out.

The project also included the application of Isolation Forest to identify anomalies in the dataset. After the dataset was cleaned, the total number of data points was 183547. After implementing the Isolation Forest unsupervised algorithm, the total number of anomalous data points is 9661, resulting in an overall 5% anomaly detection.

We thought that Support Vector Regression (SVR) would be a logical choice for building the model to predict Uber fares because of its ability to deal with many of the nonlinear and robust characteristics contained within the dataset in the previous analysis. The Root Mean Squared Error (RMSE) came out to be 4.04, and since the high R-squared value of 0.81 indicated that SVR has explained around 81% variance in fare prices, it could be used as a reliable predictor for estimation of fare price. And these metrics showed that SVR, being able to handle nonlinearity — such as time of day, local geography, and passenger count — was a good approach compared to relationships of features seen in our dataset.

COMPARISON

DBSCAN and Isolation Forest have a similar objective, but there is a time and place for each. This is because DBSCAN is used more of as a data cleaning algorithm that is only ever used once. Additionally, it takes a long time to process all the inliers and outliers which makes it not a viable option for checking new entries from users. After generating the inliers, this data is what we utilize in our later models as training or test data. That being said, Isolation Forest is better for anomaly detection since it runs in linear time. This means that when a user makes a new row or data point, we can quickly determine whether or not they paid a normal price for their trip. This makes it better than DBSCAN for anomaly detection as that would require us to run the entire costly algorithm again which would end up cycling through all of the points. That being said both of these are unsupervised algorithms so it is not a way to predict prices for a given ride, which is

where SVR comes into play. This is the most time costly algorithm used in this project, in fact it ran for over 20 minutes when we ran it on our end. Although it is so costly, it was necessary for the supervised portion of this project. We were able to use it to predict the price of an uber trip given the parameters which were an integral part of our project. Our R^2 value for this was 0.81 which means it is a good predictor of uber prices based off of our testing data.

NEXT STEPS

For the next steps, one thing we weren't able to do was incorporate weather data as well as traffic data from Google. This is because both of these implementations would require us to implement some sort of API, both of which for a dataset of this size would require us to pay for access. This would require funds that we did not want to spend and additional engineering that we felt was outside the scope of this project.

Another next step that we would want to embark on is expanding our scope beyond New York City. Due to distances being relatively smaller in New York City and certain streets being more expensive due to its location (e.g. Times Square), this could make it harder to interpret the data. A next step would be to include either other large cities, or expanding to suburban and residential areas to give a more comprehensive understanding of fare prices.

REFERENCES

- Chao, J. (2019). Modeling and analysis of Uber's rider pricing. Atlantis Press. https://doi.org/10.2991/aebmr.k.191217.127
- Chen, M. K., & Sheldon, M. (2015). Dynamic pricing in a labor market: Surge pricing and flexible work on the Uber platform. UCLA Anderson School of Management.
- Khandelwal, K., Sawarkar, A., & Hira, S. (2021). A novel approach for fare prediction using machine learning techniques. International Journal of Next-Generation Computing, 12(5), 602–609.

GANTT CHART

Link to Gantt Chart

CONTRIBUTION TABLE

Name	Proposal Contributions
Joey	Visualizations, Video

Name	Proposal Contributions
Rashmith	Video, Report
Rohit	Website, Report
David	IsolationForest
Sai	Haversine Distance

GITHUB REPO LINK

https://github.gatech.edu/rprasanna6/UberFarePrediction