NYC Taxis Fare Prediction Analysis Report

Authors: Han Wang, Jam Lin, Jiayi Li, Yibin Long

Preface

This report was developed as a deliverable for the term project in DSCI 522 (Data Science Workflows), a course in the Master of Data Science program at the University of British Columbia.

The overall objective of this project was to automate a typical data science workflow. This report summarizes the results of a series of automated Python scripts that handle tasks such as data retrieval, data cleaning, exploratory data analysis (EDA), creation of a predictive machine learning model, and interpretation of the results. The report provides a detailed explanation of each step, applying it to the specific context of the dataset in question. It assumes the reader has a basic understanding of machine learning terminology and concepts.

Summary

This report presents a linear regression model developed to predict NYC taxi fare amount. Using data from 30,000 taxi trips in January 2024, we build a linear regression model using trip distance feature, with each additional mile increasing the fare by approximately $3.54. The model was evaluated using a test dataset, where predicted fare amounts were compared to actual fares. While the model performed well overall, some outliers were identified, suggesting the need for further data cleaning and additional features to improve accuracy. Future steps include incorporating more features, experimenting with other regression models like KNN and Lasso regression, and addressing hyperparameters for better generalizability and performance.

Introduction/Background

Taking a taxi in New York City can be overwhelming for first-time visitors, especially for tourists unfamiliar with the city's layout and fare system. With over 200,000 taxi trips taking place daily, yellow cabs remain an essential mode of transportation for both locals and visitors. However, the lack of transparency often leads to concerns among tourists about overcharging or being taken on unnecessarily long routes.

To address these concerns, we use data from 30,000 Yellow Taxi trips recorded in January 2024, provided by the NYC Taxi and Limousine Commission (TLC). Our analysis examines how to predict taxi fare amounts, offering valuable insights to help tourists better understand what they should expect to pay for a taxi ride in NYC. By leveraging data-driven analysis, we aim to provide a clearer and more predictable fare structure, enabling tourists to make more informed decisions during their travels in the city.

Analysis Question

How to make predictions on taxi fare amount?

Dataset

The data set used in this project is the TLC Trip Record Data, which includes taxi and for-hire vehicle trip records collected by technology providers authorized under the Taxicab & Livery Passenger Enhancement Programs (TPEP/LPEP). This data is sourced from the NYC Taxi and Limousine Commission (TLC) and can be accessed from the NYC Taxi and Limousine Commission's website. Yellow and green taxi trip records contain fields capturing pick-up and drop-off dates and times, pick-up and drop-off locations, trip distances, itemized fares, rate types, payment types, and driver-reported passenger counts. Similarly, For-Hire Vehicle (FHV) trip records include fields such as dispatching base license number, pick-up date and time, and taxi zone location ID. It is important to note that the TLC did not create this data and makes no representations regarding its accuracy. FHV trip records are based on submissions from dispatching bases and may not represent the total volume of trips. Additionally, the TLC reviews these records and enforces necessary actions to improve their accuracy and completeness.

Methods and Results

Methods

In order to address our research question, we will begin by selecting the appropriate features from the dataset through exploratory data analysis (EDA) and by consulting the data dictionary to better understand the instances within the dataset (dataset information is linked in the references). Since this is a linear regression modeling problem, we will use the LinearRegression model as our model of choice. Linear regression is a fundamental and widely used method for predictive modeling. The dataset used in this study includes daily fare amounts and trip distance data from

January 2024, covering 30,000 observations. The data will be split into training and test sets, with 70% allocated for training and 30% for testing.

The analysis will be conducted using the Python programming language [Van Rossum and Drake, 2009], with the following Python packages: numpy [Harris et al., 2020], pandas [McKinney, 2010; VanderPlas, 2018], scikit-learn [Pedregosa et al., 2011], and altair [VanderPlas et al., 2018].

EDA

In [1]:
```
import pandas as pd
import altair as alt
import numpy as np


data_set_link = "https://d37ci6vzurychx.cloudfront.net/trip-data/yellow_trip

# Use only a smaller, random subset (30,000 rows) of the data
df = pd.read_parquet(data_set_link).sample(30000, random_state=123)
df.to_csv('data/yellow_tripdata_2024-01.csv', index=False)
df.head()
```

Out[1]:

	VendorID	tpep_pickup_datetime	tpep_dropoff_datetime	passenger_count
2688292	2	2024-01-30 17:40:12	2024-01-30 17:47:05	2.0
2956481	2	2024-01-29 12:25:03	2024-01-29 12:53:32	NaN
2203446	2	2024-01-25 16:21:38	2024-01-25 16:36:41	1.0
2436695	2	2024-01-27 20:06:28	2024-01-27 20:41:05	1.0
2063582	2	2024-01-24 09:47:54	2024-01-24 10:03:31	1.0

We'll split the dataset into training set and test set.

In [2]:
```
from sklearn.model_selection import train_test_split

train_df, test_df = train_test_split(df, test_size=0.3, random_state=123)
X_train, y_train = train_df['trip_distance'].values.reshape(-1,1), train_df[
X_test, y_test = test_df['trip_distance'].values.reshape(-1,1), test_df['far
```

Now, we'll perform a summary of the data set that is relevant for exploratory data analysis related to our regression analysis. We'll check out the summary statistics for each column in the dataset.

In [3]:
```
train_df.describe()
```

Out[3]:

	VendorID	tpep_pickup_datetime	tpep_dropoff_datetime	passenger_count
count	21000.000000	21000	21000	19973.000000
mean	1.752190	2024-01-17 01:47:00.986333	2024-01-17 02:02:19.647285	1.333250
min	1.000000	2024-01-01 00:06:10	2024-01-01 00:10:03	0.000000
25%	2.000000	2024-01-09 17:18:02	2024-01-09 17:37:35.750000	1.000000
50%	2.000000	2024-01-17 11:00:56.500000	2024-01-17 11:16:55	1.000000
75%	2.000000	2024-01-24 18:57:54	2024-01-24 19:12:33.250000	1.000000
max	6.000000	2024-01-31 23:56:14	2024-02-01 00:15:56	8.000000
std	0.433951	NaN	NaN	0.826226

Based on the summary statistics, it shows that the mean trip distance is 3.20 miles, while the median is 1.68 miles, which may point to a right-skewed distribution. As for the fare amount, the mean fare amount is 18.1 USD, while the median is 12.8 USD, which again points to a right-skewed distribution for this column.

Now, we'll create a visualizations for exploratory data analysis. First, we want to confirm that there are no missing values in the columns that we are performing regression on.

In [4]:

```python
# !pip install "vegafusion[embed]>=1.5.0"

alt.data_transformers.enable("vegafusion")
```

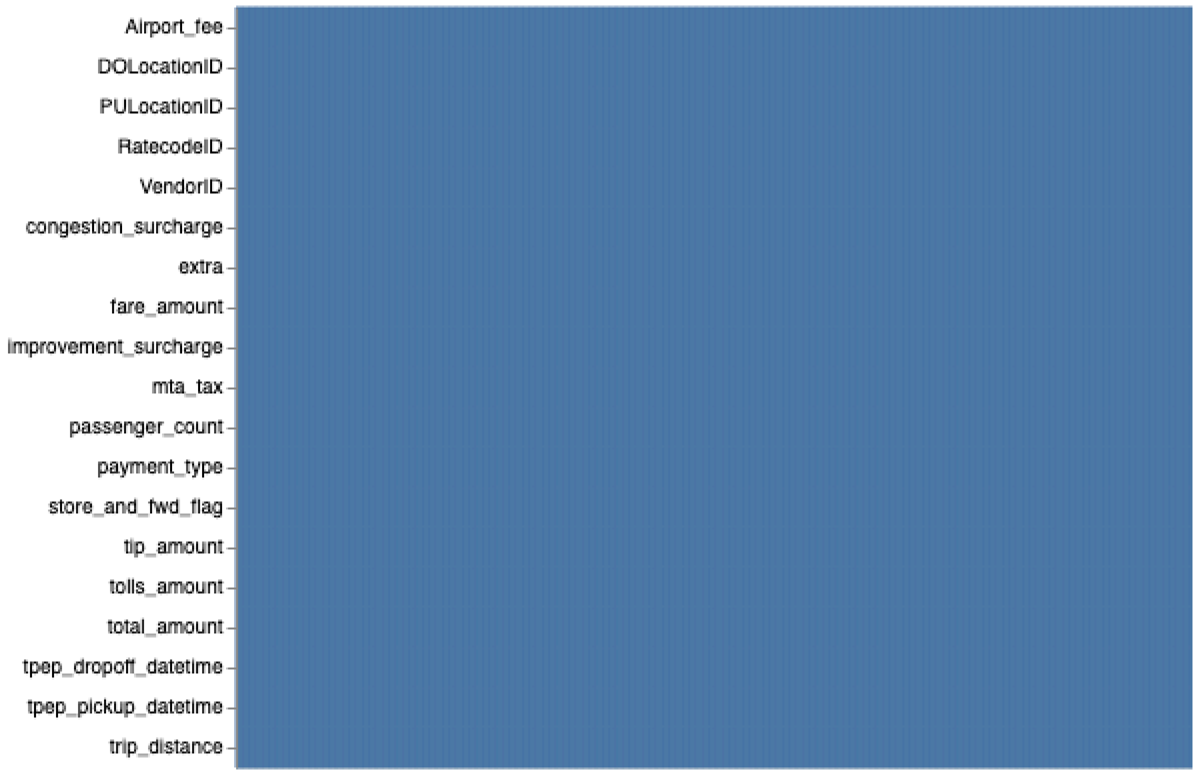
Out[4]:

```
DataTransformerRegistry.enable('vegafusion')
```

In [5]:

```python
# Visualize missing values

alt.Chart(
    train_df.isna().reset_index().melt(
        id_vars='index'
    )
).mark_rect().encode(
    alt.X('index:O').axis(None),
    alt.Y('variable').title(None),
    alt.Color('value').title('NaN'),
    alt.Stroke('value')
).properties(
    width=df.shape[0]
)
```

Out[5]:

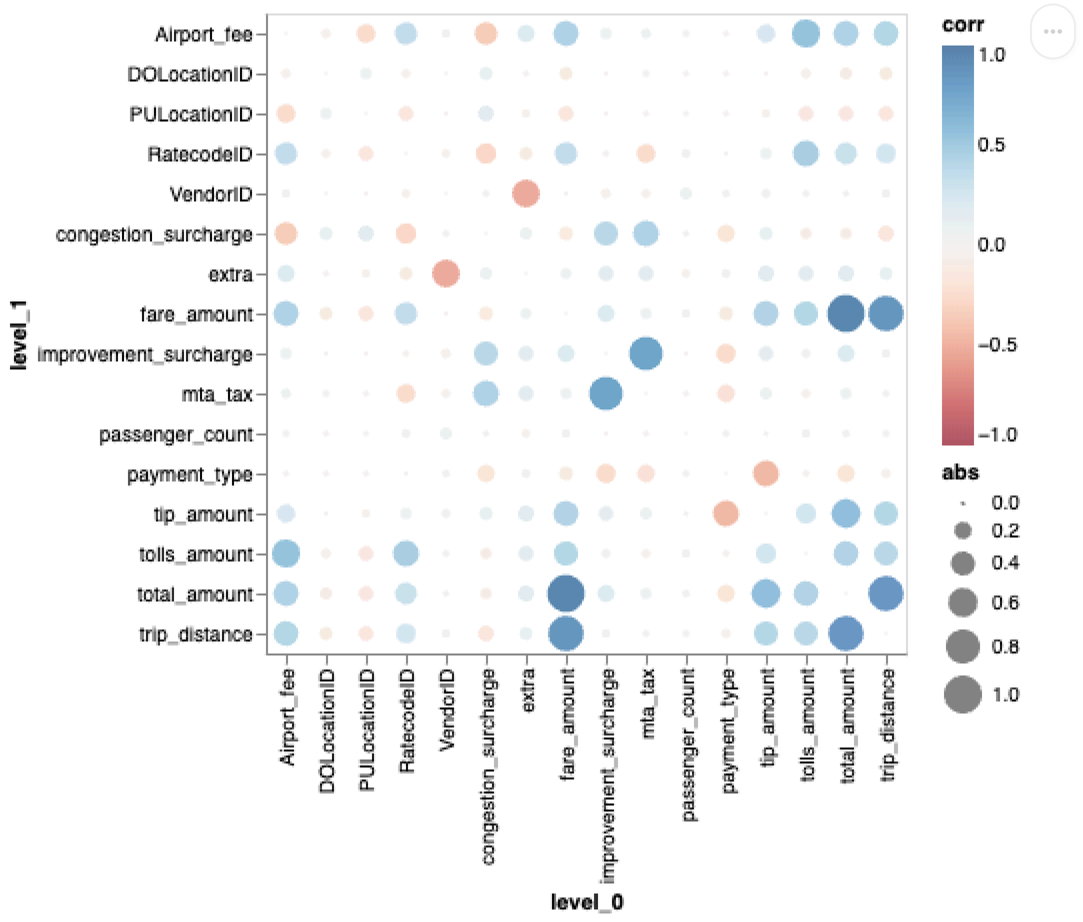


As evidenced by the chart, there are no NaN values.

Next, we'll create a correlation plot of all of the columns against one another, to check out the strength and direction of associations between columns.

In [6]:
```
# Correlation Plot

corr_df = train_df.select_dtypes('number').corr('spearman', numeric_only=Tru
corr_df.loc[corr_df['corr'] == 1, 'corr'] = 0  # Remove diagonal
corr_df['abs'] = corr_df['corr'].abs()

alt.Chart(corr_df).mark_circle().encode(
    x='level_0',
    y='level_1',
    size=alt.Size('abs').scale(domain=(0, 1)),
    color=alt.Color('corr').scale(scheme='redblue', domain=(-1, 1))
)
```

Out[6]:



Based on the correlation plot, we see that "trip_distance" and "fare_amount" have a fairly high positive correlation, which may indicate that they are fairly positively associated with each other.

Modeling

As discussed in the Methods summary, we will now build and test our Linear Regression Models.

In [7]:
```
#!pip install scikit-learn

from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score, mean_absolute_erro
```

In [8]:
```
model = LinearRegression()
model.fit(X_train, y_train)

intercept = model.intercept_
slope = model.coef_[0]

test_predictions = pd.DataFrame({
    'trip_distance': X_test.flatten(),
    'fare_amount': y_test,
    'y_pred': model.predict(X_test)
})

print(f"The regression line formula is: y_hat = {slope:.4f} * trip_distance
```

```
The regression line formula is: y_hat = 3.5369 * trip_distance + 6.7623
```

Finally, let's calculate some error metrics and perform a visualization of the result of the regression model in the form of a scatter plot with the regression line.

In [9]:

```
test_predictions
```

Out[9]:

	trip_distance	fare_amount	y_pred
0	3.14	16.3	17.868009
1	0.41	4.4	8.212379
2	17.19	70.0	67.560908
3	1.19	10.7	10.971130
4	1.08	12.1	10.582075
...	...	...	...
8995	1.10	8.6	10.652813
8996	1.44	10.0	11.855345
8997	0.90	7.2	9.945440
8998	0.65	7.2	9.061225
8999	0.94	10.7	10.086915

9000 rows × 3 columns

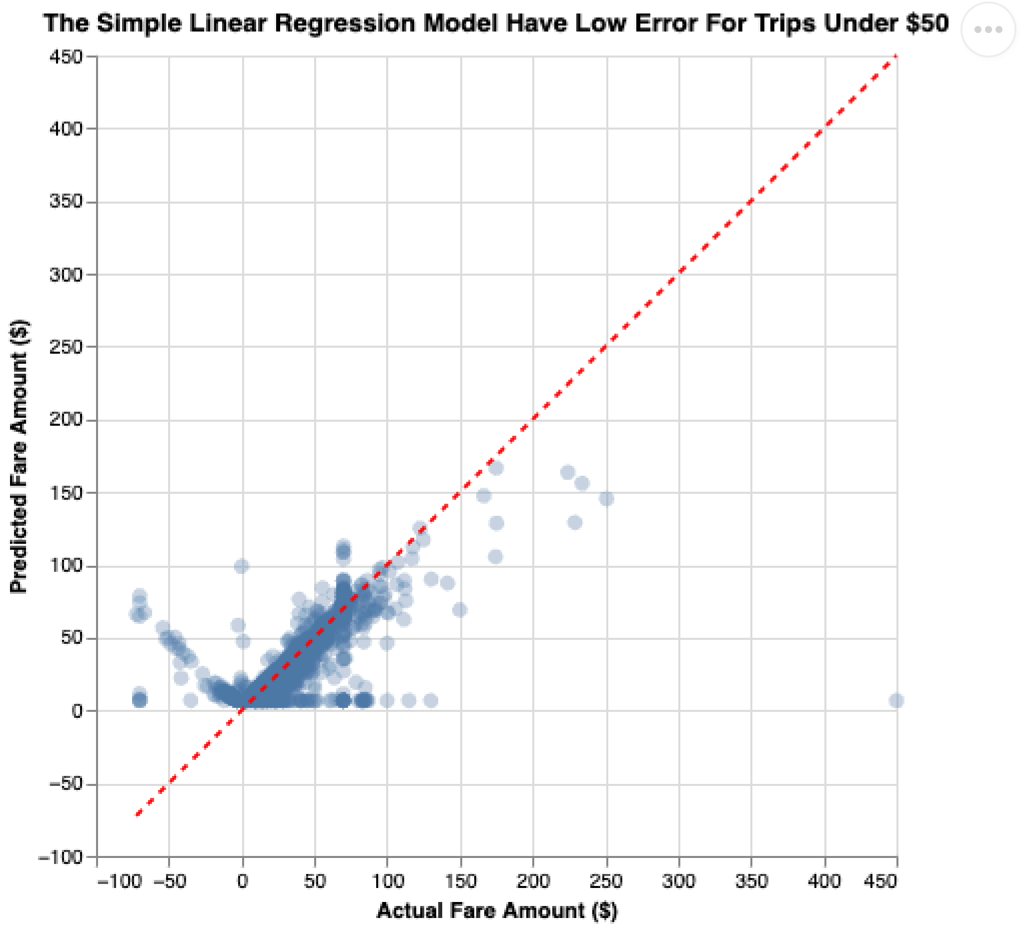
In [10]:

```
# Calculate error metrics
y_true = test_predictions['fare_amount']
y_pred = test_predictions['y_pred']
test_predictions['residuals'] = y_true - y_pred
test_predictions['abs_residuals'] = np.abs(test_predictions['residuals'])

metrics = {
    'RMSE': np.sqrt(mean_squared_error(y_true, y_pred)),
    'R²': r2_score(y_true, y_pred),
    'MAE': mean_absolute_error(y_true, y_pred)
}

print("\nRegression Performance Metrics:")
print(f"RMSE: ${metrics['RMSE']:.2f}")
print(f"R²: {metrics['R²']:.3f}")
print(f"MAE: ${metrics['MAE']:.2f}")
```

```
Regression Performance Metrics:
RMSE: $10.01
R²: 0.701
MAE: $3.61
```

In [11]:
```
error_scatter = alt.Chart(test_predictions).mark_circle(size=60, opacity=0.3
    x=alt.X('fare_amount', title='Actual Fare Amount ($)'),
    y=alt.Y('y_pred', title='Predicted Fare Amount ($)'),
    tooltip=['trip_distance', 'fare_amount', 'y_pred', 'residuals']
).properties(
    width=400,
    height=400,
    title='The Simple Linear Regression Model Have Low Error For Trips Under
)
error_diagonal = alt.Chart(pd.DataFrame({
    'x': [test_predictions['fare_amount'].min(), test_predictions['fare_amou
})).mark_line(color='red', strokeDash=[4, 4]).encode(
    x='x',
    y='x'
)

error_scatter + error_diagonal
```

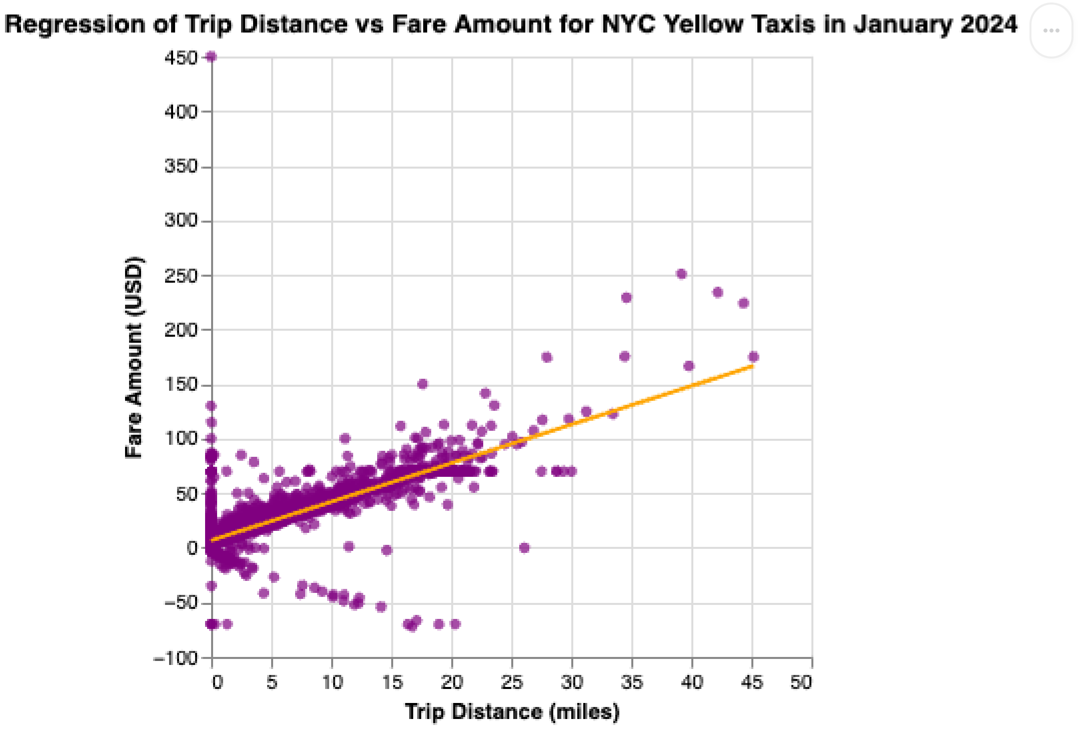
Out[11]:



In [12]:
```
scatter_plot = alt.Chart(test_predictions).mark_circle().encode(
    x=alt.X('trip_distance', title='Trip Distance (miles)'),
    y=alt.Y('fare_amount', title="Fare Amount (USD)"),
    color=alt.value('purple'),
    tooltip=['trip_distance', 'fare_amount']
).properties(
    title="Regression of Trip Distance vs Fare Amount for NYC Yellow Taxis i
)
```

```
line_plot = alt.Chart(test_predictions).mark_line(color='orange').encode(
    x='trip_distance',
    y='y_pred'
)

chart = scatter_plot + line_plot

chart
```

Out[12]:



Results

The objective of the analysis was to predict the fare amount using a linear regression model. The regression plot, generated from the model, demonstrates a positive linear relationship between trip distance and fare amount for NYC Yellow Taxis in January 2024. This confirms that as the trip distance increases, the fare amount also increases, as expected.

The linear regression model shows moderate predictive power with an R-squared value of 0.701, which explains about 70 percent of fare variance. The low MAE of USD 3.61 indicates good accuracy for typical rides, although the higher RMSE (USD 10.01) suggests sensitivity to outliers. The visual analysis of the errors reveals the model performs best for the USD 10-50 range, and underestimates fares over USD 100. The actual NYC taxi fare is calculated through initial fare, per-mile charge, and additional fees for peak hours, nighttime, tolls, etc that this model cannot predict, but can be built further using more complicated multilinear regression or ML models.

Based on the estimated result from the regression line formula derived from the model, this suggests that for each additional mile traveled, the fare increases by approximately

USD 3.54, and the base fare (when the trip distance is zero) is around USD 6.76.

To evaluate the model's predictive performance, we applied it to the test data, which was split from the original dataset (70 percent training, 30 percent testing). The predicted fare amounts were compared against the actual fare amounts in the test set. The predicted values were calculated using the formula derived from the model, where trip distances from the test set were used as input to generate the corresponding fare predictions.

A scatter plot of the actual fare amounts versus trip distances, along with the regression line, was generated to visualize the results. The plot showed that the model fits most of the data well, but there were some outliers where the predicted fare deviated significantly from the actual values. These discrepancies could be due to errors or special cases in the fare data.

Limitations and Next Steps

Overall, our model may be useful in the initial analysis of tourists interested in making informal predictions about NYC taxi fare amounts. However, there are several areas where this work can be improved. In this analysis, we did not include hyperparameters like the regularization parameter (α), which is essential for controlling overfitting and improving model generalizability. Furthermore, the model currently uses only a single feature (trip distance), which may result in underfitting and bias. In the next steps, we plan to incorporate more features into the model and refine the approach to improve its precision and predictive power. Additionally, experimenting with other regression models, such as KNN regression for non-linear relationships and Lasso regression for linear relationships, could be valuable to see if these models improve performance.

References

Charles R Harris, K Jarrod Millman, Stéfan J van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E Oliphant. Array programming with NumPy. Nature, 585(7825):357–362, 2020. URL: https://doi.org/10.1038/s41586-020-2649-2, doi:10.1038/s41586-020-2649-2.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research, 12:2825–2830, 2011.

Jake VanderPlas. Altair: interactive statistical visualizations for python. Journal of open source software, 3(7825):1057, 2018. URL: https://doi.org/10.21105/joss.01057, doi:10.21105/joss.01057.

New York City Taxi and Limousine Commission. TLC Trip Record Data. Retrieved from https://www.nyc.gov/site/tlc/about/tlc-trip-record-data.page, 2024. Wes McKinney. Data structures for statistical computing in python. In Stéfan van der Walt and Jarrod Millman, editors, Proceedings of the 9th Python in Science Conference, =51 – 56. 2010.

Wes McKinney. Data structures for statistical computing in python. In Stéfan van der Walt and Jarrod Millman, editors, Proceedings of the 9th Python in Science Conference, =51 – 56. 2010.

```
In [13]: !jupyter nbconvert --to webpdf yellow_taxi_analysis.ipynb
```

```
[NbConvertApp] Converting notebook yellow_taxi_analysis.ipynb to webpdf
[NbConvertApp] Writing 169802 bytes to yellow_taxi_analysis.pdf
```