NYC Taxi Data Analysis

Milan Chheta
Master of Computer Science
Indiana University

Jay Madhu
Master of Data Science
Indiana University

Biraj Parikh
Master of Data Science
Indiana University

Abstract:

Through this project, we sought to better understand the dynamics of taxi vehicles like commuting patterns, traffic, tipping patterns, taxi fares (and more) in New York City. What kind of trips are made in cabs? Where do those trips occur? What are the predominant costs and locations of taxi trips, and what are the planning implications of these findings? We hope to answer these questions by performing analysis on TLC (Taxi and Limousine Commission) taxi data which is publicly available[1].

1. Introduction:

Analyzing commuting patterns in urban communities has become a key research area lately. Such analysis reveals critical results about a city's transport services. The New York City Taxi and Limousine Commission (TLC), created in 1971, is the agency responsible for licensing and regulating New York City's medallion (yellow) taxicabs, for-hire vehicles (community-based liveries, black cars and luxury limousines), commuter vans, and paratransit vehicles. TLC has released a staggeringly detailed historical dataset covering over 1 billion individual taxi trips in the city starting from January 2009 through December 2019. Taken as a whole, the detailed trip-level data is more than just a vast list of taxi pickup and drop off coordinates: it's a story of how New Yorkers move around the city. How bad is the rush hour traffic? What are the hotspots for pickup and dropoff location? How much tip do drivers get? The dataset helps in addressing all of these questions and many more.

This dataset is of high value for fetching useful insights that may help policy makers and city administration to make effective policies towards curbing traffic and reducing the number of disputes in public transport. And hence, The purpose of our analysis is to extract useful insights so that the solutions are developed from both, the customer's perspective as well driver's perspective.

2. Methodology and Implementation:

In this section we will walk you through the steps taken by us in implementing this project, i.e, methodology and implementation details, right from data preprocessing to data visualization to Machine Learning.

2.1 Data Exploration and Preparation:

As mentioned earlier the data for this project has been taken from TLC's official data which has been made available to the public. Given the humongous size of the data, we have only used the data for 1 month only, i.e, December 2019. With a size of 900MB, the dataset has over 6.8 million taxi trip records for yellow and green cabs. While analyzing only 1 month of data doesn't seem sensible, we are

more interested in behavioral patterns rather than spatial patterns. Moreover, the dataset we've downloaded is for yellow taxis. The dataset has 19 features which include VendorID, PickupDateTime, DropOffDateTime, PassengerCount, TripDistance, PickupLongitude, PickupLatitude, RateCodeID, Store FWD Flag, DropOffLongitude, DropOffLatitude, PaymentType, FareAmount, Extra, MTA Tax, ImprovementSurcharge, TipAmount, TollsAmount, TotalAmount. However, we are using the features that are relevant to our analysis, the details of which are given below.

Data is downloaded from the source using the urllib and zipfile libraries in python. The code snippet to achieve that is given below. The code snippet below directly downloads the '.csv' file which contains the data we for the month and the year we want. The code is flexible enough to let us download the dataset for any month and any year we specify. We wrote a special python code for this, which can download any type of taxi data based on the parameters you give.

```
for month in range(1,2):
    urllib.request.urlretrieve("https://s3.amazonaws.com/nyc-tlc/trip+data/"+ \
                               "yellow_tripdata_2019-{0:0=2d}.csv".format(month),
                               "data/nyc_2019-{0:0=2d}.csv".format(month))
```

Once we have the '.csv' file download, we access that file using pandas library in python to create a dataframe so that we can perform data pre-processing. In data pre-processing, we extract the features which are important to our analysis after which we clean the data by removing outliers and dropping the null values. Moreover, we also add 3 new features called pickup_time, dropoff_time and pickup_dow (day of the week) to our data. In addition to that, we've also included the latitude and longitude for New York city, given we have location based data, useful for plotting maps for data visualization. These features are calculated using the existing features. Once data preprocessing is done, we dump the data into SQLite, our choice of database for this project. The data is dumped into the database using SQLAlchemy which is an open-source SQL toolkit and object-relational mapper for Python. A database is created called "nyc_database.db" which has a table called "*nyc_yellow_2019_1*". Details of the table are given below:

Column Name	Column Description
tpep_pickup_datetime	Holds the pickup datetime information for a trip
tpep_pickup_datetime	Holds the dropoff datetime information for a trip
passenger_count	Number of passengers for a trip
trip_distance	Distance of trip done in miles
PULocationID	TLC Taxi Zone in which the taximeter was engaged
DOLocationID	TLC Taxi Zone in which the taximeter was disengaged
payment_type	Type of payment in number 1= Credit card, 2=

	Cash, 3= No charge, 4= Dispute 5= Unknown, 6= Voided trip
tip_amount	Tip amount for a trip (Cash tips are not included)
total_amount	Total amount for a trip (without tips)
weekday	Weekday for initiated trip
pickup_hour	Pickup hour for a trip
dropoff_hour	Drop off hour for a trip

Table 1: *nyc_yellow_2019_1*

2.2 Use of Queries:

One of the best things about python is it's support for other languages like SQL. The pandas library in python allows us to write SQL queries in a python environment. The query passed in *read_sql_query()* parse it to give the required output as a dataframe. The data frame generated is then used for plotting and analysis using the plotly library. Given the type of use case, most of our queries used aggregate functions like count() and avg(). Below, we show you an example from our code which results a dataframe which is used for plotting:

```
df_day_pc = pd.read_sql_query('SELECT pickup_hour as Hour, count(*) AS passenger_count \
                               FROM nyc_yellow_2019_12 \
                               GROUP BY pickup_hour', nyc_database)

df_day_pc.iplot(x="Hour", kind="bar", title="Number of Passengers travelling per hour",
                xTitle='Hour', yTitle='Count', color = "orange")
```

2.3 Predictive Analysis (Machine Learning model):

Up until now, all our analysis has helped us answer various questions regarding NYC taxi trips. This analysis will also prove as the basis for the predictive analysis that we are going to perform now. Through the above plots and visual analysis that we've done so far, we can find out which feature is important for our model which one is not.

We have built a machine learning model to predict the total fare for taxi trips. Here we have built a multivariate linear regression. While the use of this model might be questioned, The reason for selecting this model is:

- It is more interpretable compared to other models like random forest, svm especially for such huge data
- Linear regression assumes that the variables are independent
- Many time complex models give poor results compared to simple models as they over learn, leading to the problem of overfitting.

As it is done for most of the models, we divide the data into train-test sets with a ratio of 75:25. The linreg method in the sklearn module of Scikit-Learn library in python is used to train the machine

learning model. Two metrics are used: MAE (Mean Absolute Error) and RMSE(Root Mean Squared Error). Accuracies for the model are given below.

Metric	Value
Mean Absolute Error	~5.0
Root Mean Squared Error	~9.0

2.4 Tools and Technologies:

Jupyter Notebook: Our whole project is implemented in Jupyter Notebook. Jupyter Notebook is an open-source web application that allows us to create and share codes and documents. It provides an environment, where you can document your code, run it, look at the outcome, visualize data and see the results without leaving the environment. This makes it a handy tool for performing end to end data science workflows – data cleaning, statistical modeling, building and training machine learning models, visualizing data, and many other uses[4].

SQLite: Our choice database for this project is **SQLite.** SQLite is a relational database management system contained in a C library. In contrast to many other database management systems, SQLite is not a client–server database engine. Rather, it is embedded into the end program that uses it. SQLite is a relational database management system contained in a C library. In contrast to many other database management systems, SQLite is not a client–server database engine. Rather, it is embedded into the end program. Although it might appear like a "simple" DB implementation, SQL is used in SQLite. SQLite is meant to be great for both developing and testing and offers more than what is needed for development. Given such advantages and the kind of data we are dealing with, SQLite becomes our obvious choice.

SQLAlchemy: It is an open-source SQL toolkit and object-relational mapper for the Python programming language. SQLAlchemy is used to create a database and connect that database with our python environment. It is designed for efficient and high-performing database access, adapted into a simple and Pythonic domain language.

While above mentioned were the major tools that we used in this project, below are the list of python libraries which were equally useful and deserved a special mention. The python modules used are:
urllib - used to download the dataset from the data source
pandas - used for handling the dataset, and performing data manipulation and analysis.
seaborn, plotly nd cufflinks - used to create interactive visualisation plots
Scikit-Learn: Scikit-learn is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.

3. Results and Conclusion:

In this section we'll be presenting some of our important results with a conclusion. The analysis of NYC yellow cab taxis tell us a lot more about their passenger distribution, fair distribution and their tips distribution. Following are the conclusions we can draw from the analysis:

- The analysis successfully shows us that the passenger books the taxi more during holidays and peak hours, the tips given by passenger varies according on hours and daily basis apart from other uncaptured data. The number of pickups increases during the evening period of the day in NYC and so does the tips given by passengers, and it is inferred from this data that evening life in NYC proves to be busy where more demand for yellow cab is required by passengers.
- Linear model was selected after performing a statistical analysis wherein we check whether our data follows the assumption of linear model. It was found that the assumptions were met and hence we move ahead with this particular model. The linear model produces an RMSE of ~9, which is great but it is still good and has a scope of a lot of improvements.

<u>Plots:</u>

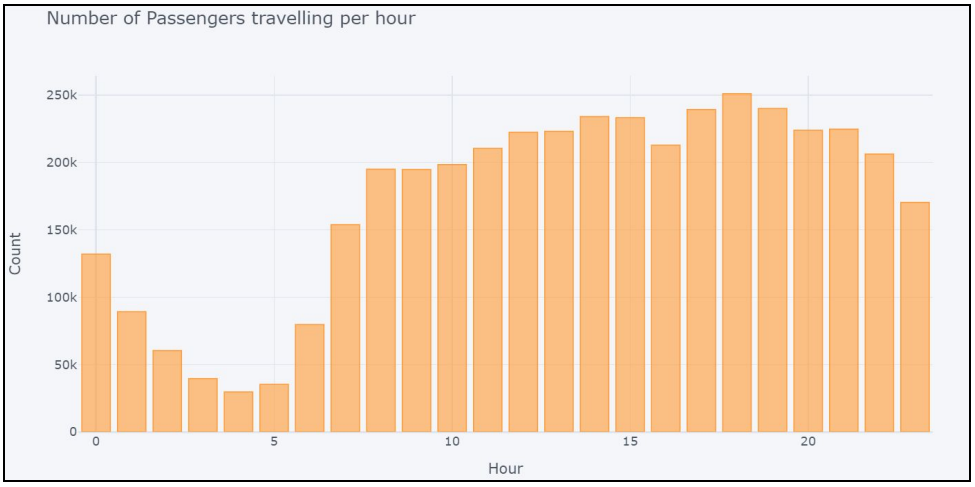

Figure 1: From the above graph it is clearly seen that the demand for yellow cab is experiencing more number of bookings in the evening during peak hours.

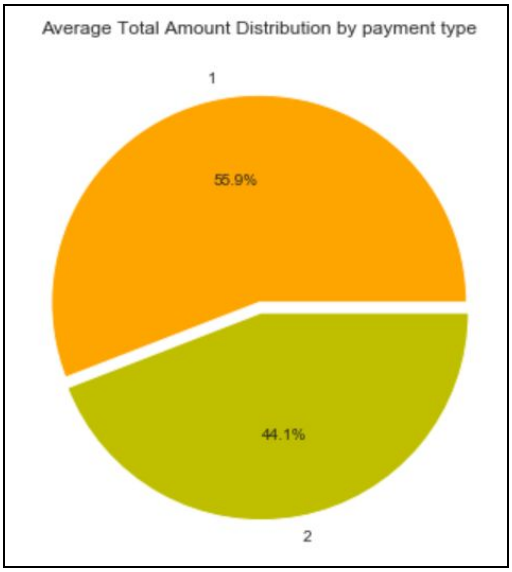

Figure 2: type 1 = Cash, typ e= Cards

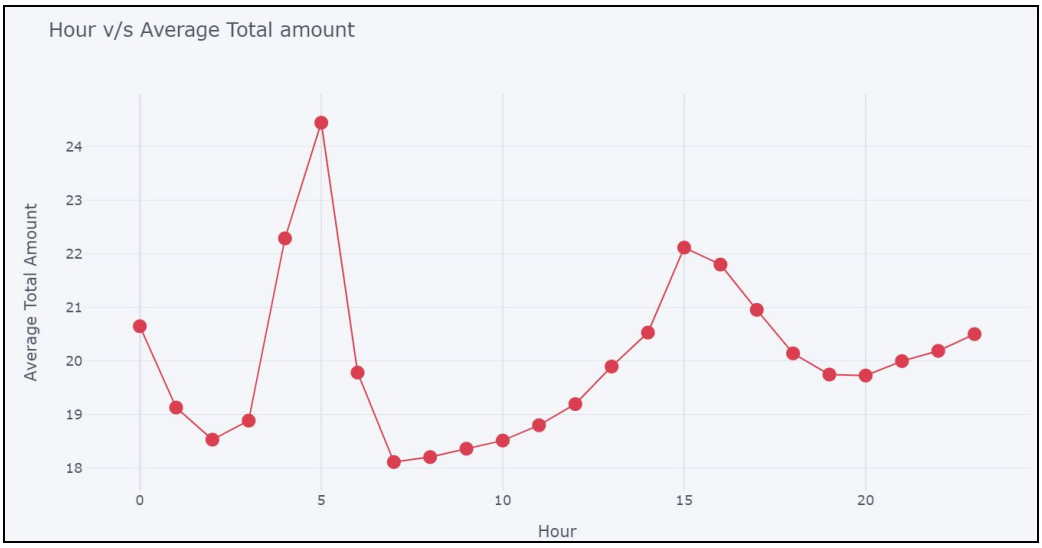

Figure 3: Total Fare according to hour of the day

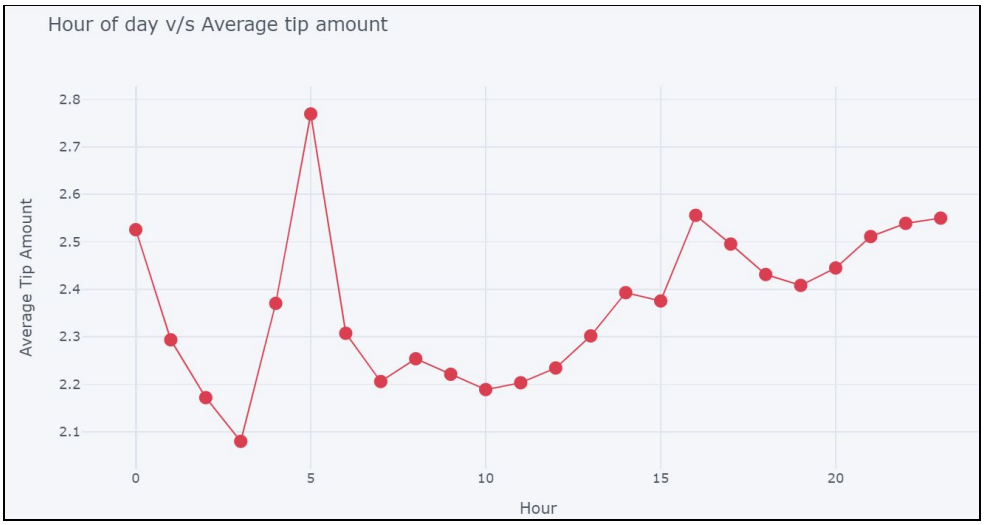

Figure 4: tip according to hour of the day

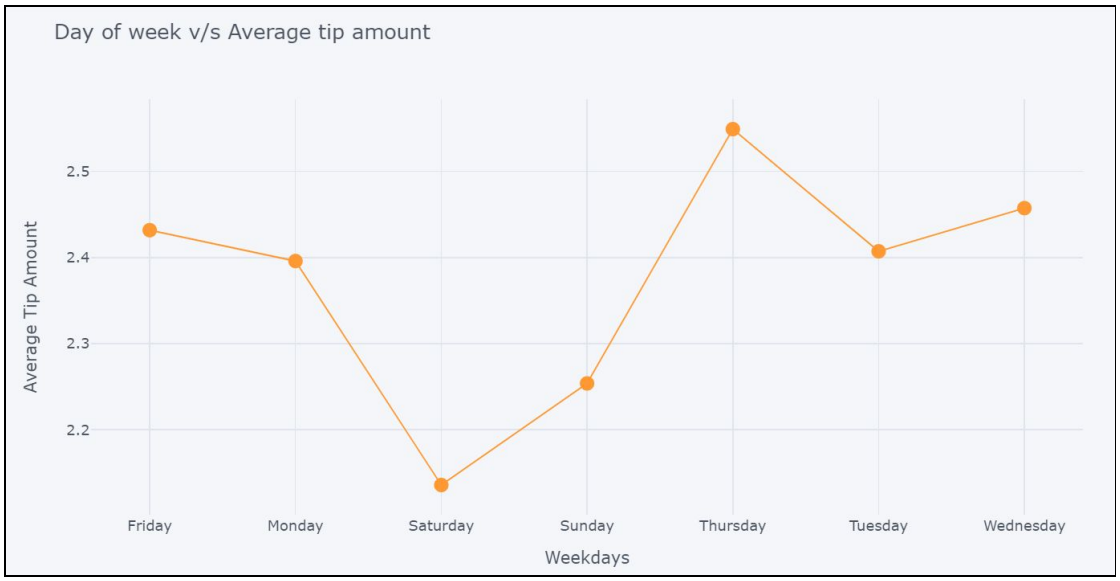

Figure 5: Total Fare according to hour of the day

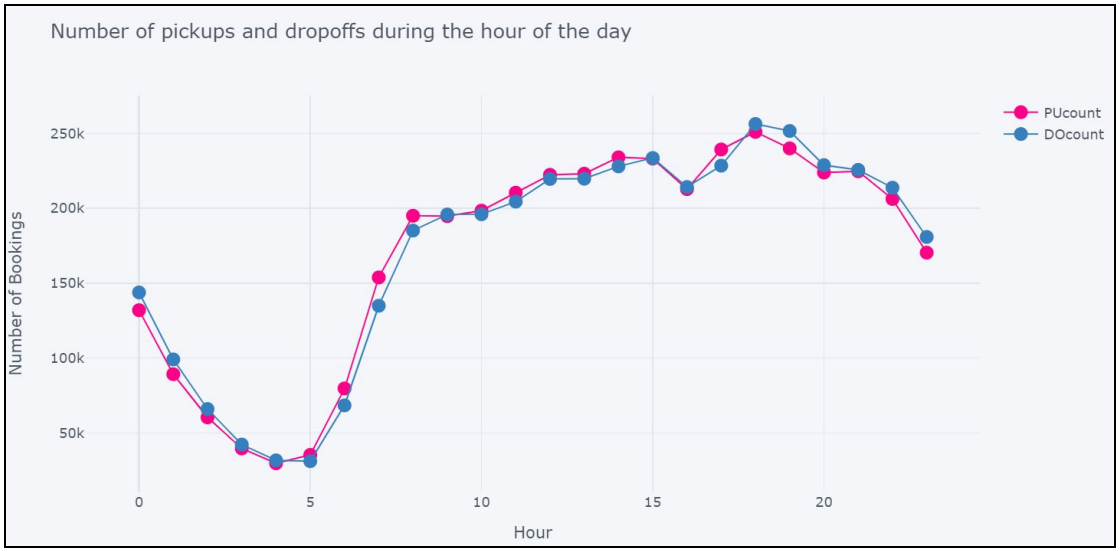

Figure 6: No of pickup and dropoff the hour of the day

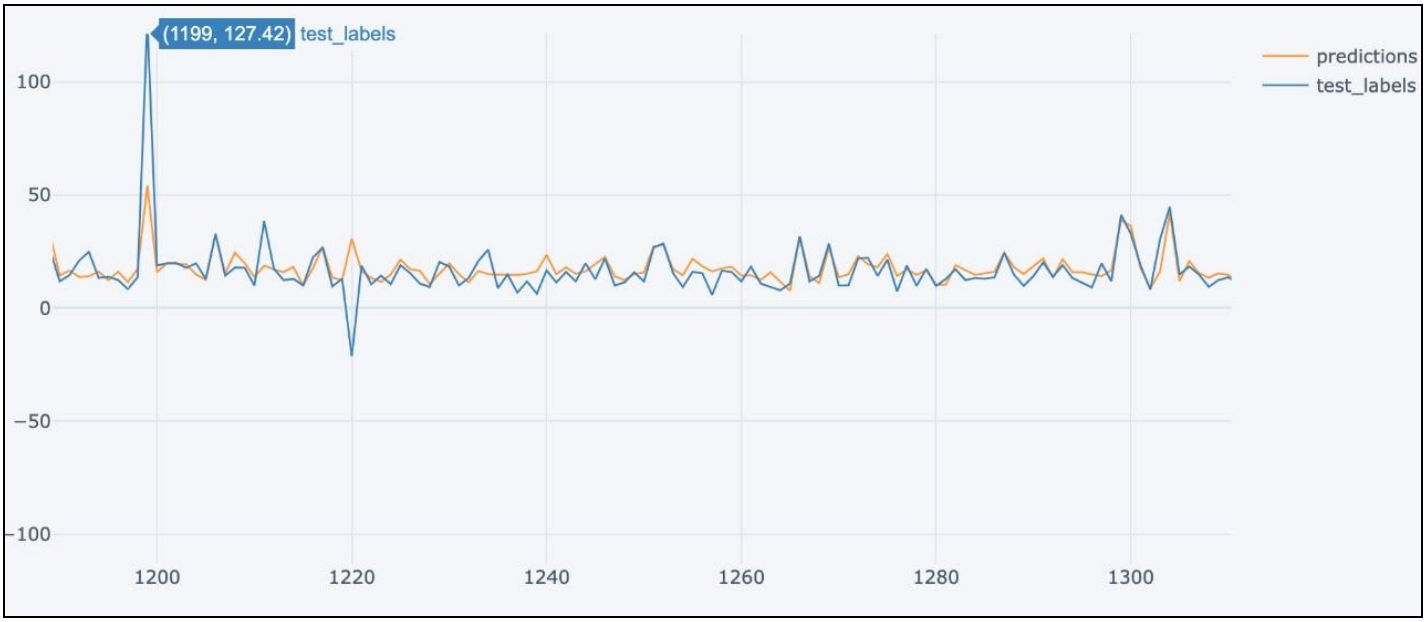

Figure 7: Actual vs Predicted

Note: Figure 7 is a 'zoomed in' version of the original plot

Contribution:

Milan Chheta	Research, Queries, Visualization (+ code), Documentation and Presentation
Biraj Parikh	Research, Queries, Visualization (+ code), Documentation and Presentation
Jay Madhu	Research, Queries, Visualization (+ code), Documentation and Presentation

References:

[1] https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page
[2] https://toddwschneider.com/posts/analyzing-1-1-billion-nyc-taxi-and-uber-trips-with-a-vengeance/
[3] https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page
[4] https://www.analyticsvidhya.com/blog/2018/05/starters-guide-jupyter-notebook/