Summary Report

New York City Taxi Trip Duration Report

Omar Ayman Tawfiq Bakr

Abstract—In this report, I will discuss my findings in New York City Taxi Trip Duration Data from Kaggle, which can be found here. We will go through insights that I gained from exploring the data, what I have done and what could be improved

Keywords— *EDA, data analysis, machine learning, AI, data exploration*

1. Introduction

New York City Taxi Trip Duration was originally a dataset held in Kaggle to predict the trip duration of a trip based on these factors 1. pickup longitude, pickup latitude of the trip 2. dropoff longitude, dropoff latitude of the trip 3. Time of the start And pretty much that's it! You can say this problem was focused on the feature engineering part, aka coming up with features that could help the model to correctly predict the trip duration.

2. Abstract

I have come up with what I think are some decent features, as follows:

1. Time-related features: `is_weekend`, `is_rushhour`, features directly extracted from the datetime object of the pickup, and also some cyclic time features (a fancy name for applying sin and cos functions for each feature).
2. Distance-related features: Haversine distance, horizontal distance, and vertical distance.
3. Location features: At which Borough the pickup is, and also for the dropoff.
4. OSRM API features: I used the OSRM API to predict the time and distance for the trip between pickup and dropoff points.
5. Some ad-hoc features to capture simple interactions between variables.
6. Traffic volume-related characteristics: What is the average traffic volume for each Borough?
7. Weather-related features: What was the weather on that day (temperature, snow, rain, etc.)?

3. New York Trip Duration insights

3.1. Data types and Null values

- The data was clean and all data types were almost correct except for the 'pickup_datetime' which needed to be casted,**note that we have decided to drop the 'dropoff_datetime' from the data to make the problem a bit challenging**

3.2. pick up date time

- checking the date range, we found that the trips were almost uniformly distributed between January, February, March, April, May, June, and July. I guess that other months were held for testing, as you can see in
- trips are also nicely distributed, as you can see in Fig. 1

3.3. Time of the day

There is a noticeable increase in trip length for late hours and almost the opposite in early hours, which is to be expected! which you can see in Fig. 2

3.4. Weekend or Not

We can see that the trips during weekends tend to be shorter than the ones on working days, as you can see in Fig.3

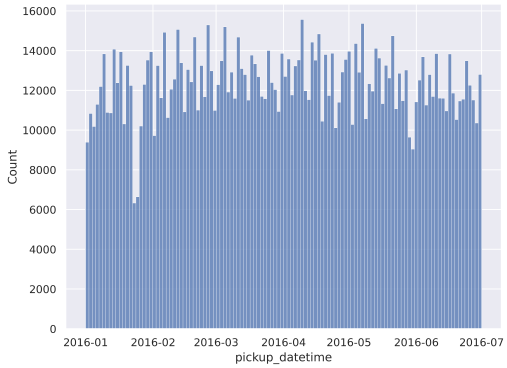

Figure 1. distribution of trips across different dates

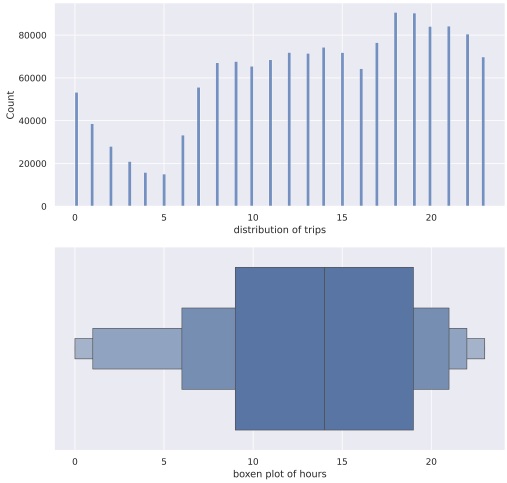

Figure 2. Enter Caption

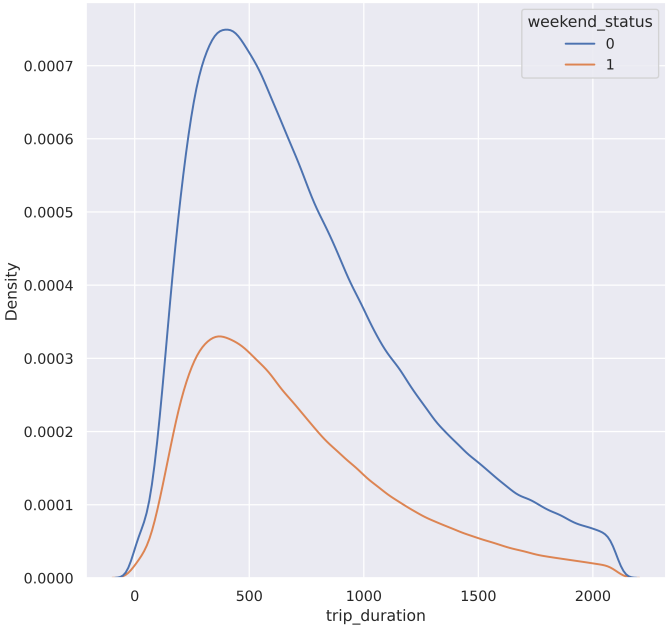

Figure 3. Density of trip duration over working and non-working days

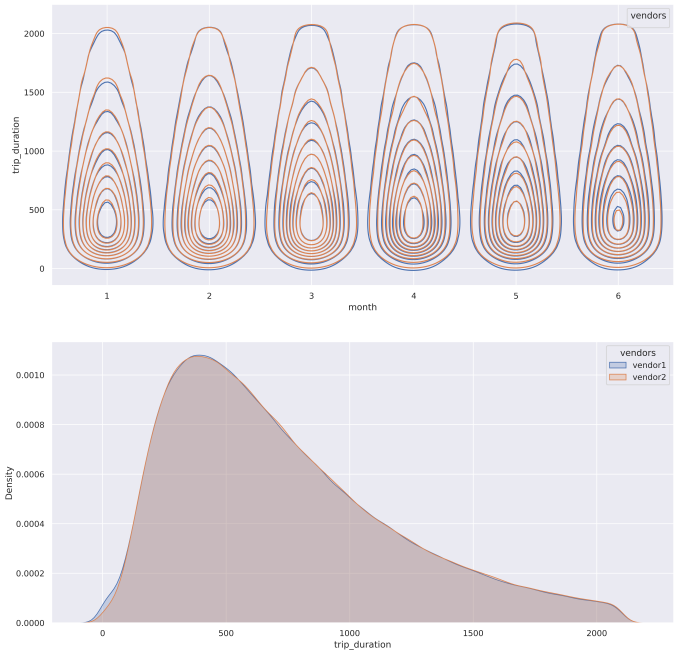

Figure 4. trip Distribution For Each Vendor

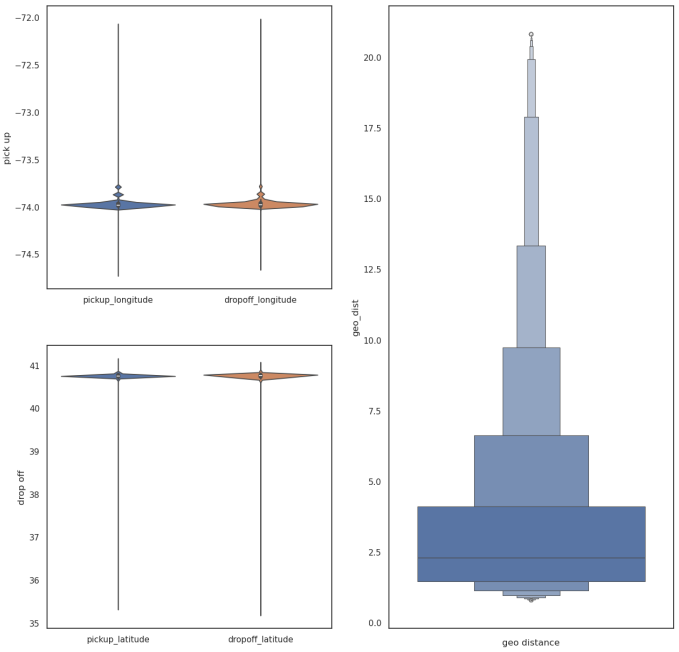

Figure 6. Distribution of trip distance

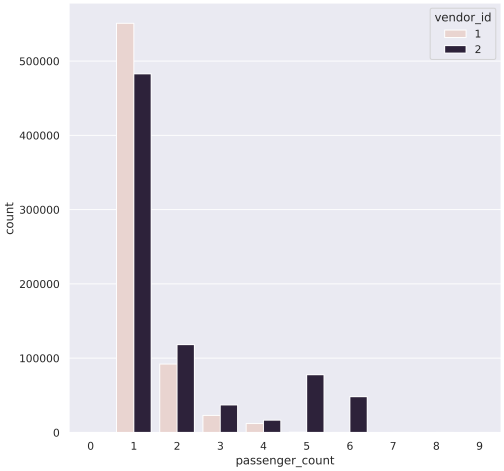

Figure 5. number of passengers per vendor

3.5. vendor id

I had a suspicion about this column from the very beginning, so my judgment could be biased, but I will try to convince you why it is useless.

- first, there are 2 vendor IDs, which indicate that there are 2 separate entities or companies
- checking the average trip duration, we find that 845.4 for vendor 1 and 1058.64 for vendor 2 (a difference! Yes, but is it noticeable? Probably not !)
- If you check the trip distribution for each vendor, you probably won't notice any difference for both the overall distribution and for each month! as you can see in Fig 4

3.6. Passenger count

- Here we can find a noticeable difference between the two vendors, vendor 2 takes a very large number of passengers, up to 6, as you can see in Fig 5

3.7. geo distance and longitude and latitude

- some anomalous tips are >100 Km could reach 1200 Km, which is not normal at all
- just filtering 0.01% of the data, we get a nicely distributed trip distance
- you will see that longitude and latitude are almost focused on a single point, this is ok because the difference between each line is around 111 km

3.8. osrm api

- OSRM (Open Source Routing Machine) is a high-performance routing engine for shortest paths in road networks
- you can download the data you need locally and build a Docker image, then query that image as you want. I have included instructions in a separate file.
- I have used it to predict the **distance** between pickup (longitude, latitude) and drop off (longitude, latitude) and **time** needed for the trip
- Note that since the data is from 2016, the estimate from the OSRM api is a bit odd since it is using the 2025 latest data about the roads

4. geographic information

4.1. Borough information

If we map each trip (pickup and dropoff) into a Borough neighbourhood, we will find that Manhattan has the biggest share among all five regions, and you can double-check this in Fig. 7

- i have used the data here about the New York Borough regions
- each Borough is defined by a 'MultiPolygon'shape

5. Traffic Volume

According to the data from here we can find that 'Queens ' is the busiest neighbourhood of all of them, and you can see this in Fig 8

- note that we have added 'unknown' territory that we will assign for any unknown neighborhood

6. Weather Data

data can be found in this link, which is pretty clean. We took all the data available, which I think has a huge impact on the trip duration, like

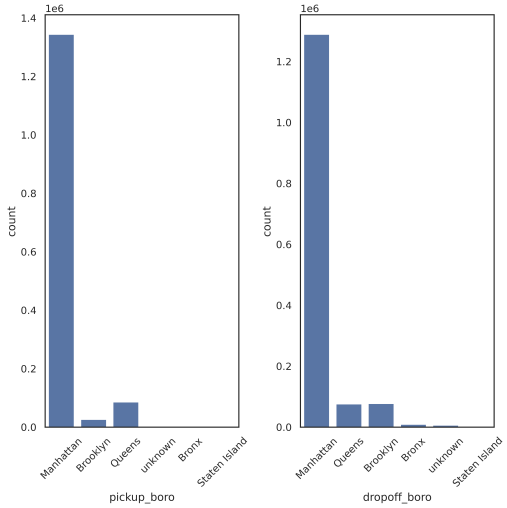

Figure 7. total number of pickups and dropoffs for each Borough region

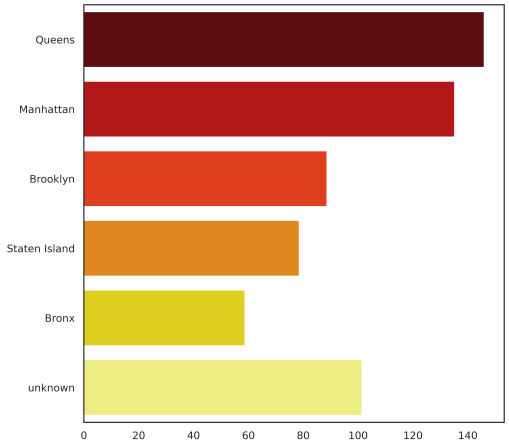

Figure 8. Traffic volume for each neighborhood

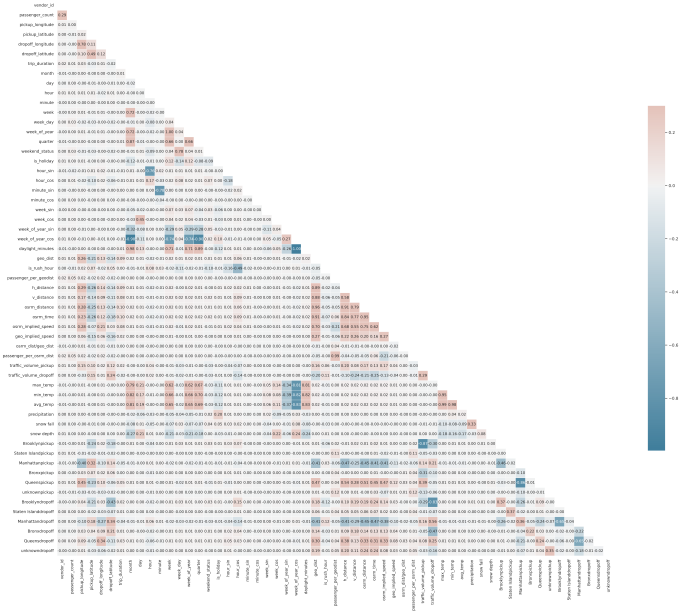

Figure 9. Correlation matrix for all the features, you can see that we have lots of 0(s) but this does not mean that there is no relationship between these features and the output, **this only means that there is no linear relationship between them** but you will see that we got a reasonable performance in the modeling section

- max temperature
- min temperature
- snow depth
- snowfall

7. Correlation matrix for all the features

You can check the correlation matrix in Fig. 9

8. Modeling

This work was for a closed context, the goal was to focus on feature Engineering, part of the problem not so much on modeling. We agreed to set the model to be 'sklearn.linear_modelRidge(alpha =1)' to compare our results with each other, and here are my results

- I have used 'PolynomialFeatures (degree=2)' **don't increase the degree over, you will overfit the data, at least this is what I got**

You can find the results in this Table1

Table 1. scores over Train and Test Data

Data	rms error	R^2 score
Train	0.3973	0.7146
Test	0.4010	0.7106

Note: These results were tested over Kaggle-provided data

9. What could be Enhanced

- Better traffic volume data; extract avg. time/distance per borough day-segment (e.g., **Qns-Man morn.**).
- More granular weather (e.g., wind/trip direction); more explicit variable interactions (not just PolynomialFeatures).
- Feature selection (e.g., top 500 from model).
- Explore automatic feature engineering tools.