Big Data Management

Project nr 1: Analyzing New York City Taxi Data

Team members: Anna Maria Tammin, Maria Anett Kaha, Eidi Paas, Pirjo Vainjärv

Provided dataset

For the geographical boundaries of the different boroughs of New York City we used the nyc-boroughs.geojson file provided by the course organizers in Moodle. The trip dataset itself that contains data about each ride was downloaded from https://www.andresmh.com/nyctaxitrips/ Trips section. The dataset contains 12 files, trip_data(1-12).csv. Since the files are large, we decided to continue with only using trip_data_1.csv file (2.4GB). In the csv file, each row represents one taxi ride information, except for the first row, which is the header. Due to the large size of the file, we could not attach it to the Git repository nor zip file uploaded to the Moodle (100MB per file), it has to be manually downloaded from the previously provided link or we have also compressed it with 7z and uploaded it to a Google Drive. https://drive.google.com/file/d/1ZdUWt0ATyJrQbJEXSge7GNACusphhJoL/view (347MB).

NB! The analysis of queries results were done on trip_data_1 .csv file.

Data processing and transformations

All of the trip_data_X.csv files are read into a single Spark DataFrame. The processing begins by defining the schema to optimize the reading and parsing of the taxi trip data. It ensures that each field in the CSV files is correctly recognized and cast to its appropriate data type.

Instead of creating a separate dataframe to isolate the zero-passenger trips, we filtered them entirely out from the taxi_df_og Dataframe, ensuring that the dataset only retains valid trips.

Transformations done on the taxi dataframe:

1. Trips where the trip_distance is zero and both the pickup and dropoff locations are identical and trips with zero passengers are excluded.
2. Additional UNIX timestamp columns are added and the duration of each trip is calculated based on them.
3. Trips with duration ≤ 0 seconds or > 4 hours are excluded from the dataset.
4. Finally, only relevant columns are selected for further analysis, and any rows with missing values (null) are dropped.

After these steps, we are left with 14,335,835 rides.

Addition of Borough Data

The borough data is loaded from the GEOJSON file. This data is then broadcasted across all worker nodes in a Spark cluster. A dictionary is built, where keys represent the borough codes and values are lists of polygons that the borough contains. After the polygons are gathered, each list of borough polygons are sorted by area in descending order. This ensures that the largest polygon for each borough is used for spatial queries.

A User-Defined Function (UDF) called get_borough_udf is created to determine which borough a given set of trip pick up and drop off coordinates (longitude and latitude) belongs to. The function is applied to the taxi drip data to determine the pickup and dropoff boroughs for each trip. New columns, pickup_borough and dropoff_borough, are added to the DataFrame. These columns store the borough codes.

Selected parameters

hack_license – A unique identifier for the taxi driver or vehicle operator.

pickup_latitude – The latitude coordinate of the location where the passenger was picked up.

pickup_longitude – The longitude coordinate of the location where the passenger was picked up.

pickup_datetime – The timestamp when the taxi ride began.

dropoff_latitude – The latitude coordinate of the location where the passenger was dropped off.

dropoff_longitude – The longitude coordinate of the location where the passenger was dropped off.

dropoff_datetime – The timestamp when the taxi ride ended.

Calculated parameters

pickup_ts – The pickup timestamp represents the time the ride started, in seconds since the Unix epoch. It's calculated by converting the pickup_datetime to a Unix timestamp

dropoff_ts – The dropoff timestamp represents the time the ride ended, in seconds since the Unix epoch. It's calculated by converting the dropoff_datetime to a Unix timestamp.

duration – The ride duration is the difference between the dropoff and pickup times (in seconds).

pickup_borough – The pickup borough is the area where the ride started, determined using the pickup location's longitude and latitude.

dropoff_borough – The dropoff borough is the area where the ride ended, determined using the dropoff location's longitude and latitude.

Queries

Query 1 - Computing utilization, idle time per taxi

The dataset was processed to calculate the idle time for each taxi driver. Repartition was used to get all trips of the same driver. The window function was used to get the previous drop-off time for each driver. The idle time was calculated by subtracting the previous drop-off time from the pick-up time. Then the data frame was filtered based on the idle times (times greater than 4 hours were removed), also we excluded cases where pickup time was earlier than the prev_dropoff time, just to avoid overlapping. Finally, the sum of the idle time of each taxi driver was calculated. Then the total_idle_time, total_ride_duration, and total_time were used to compute utilization_rate.

Top 10 rows :

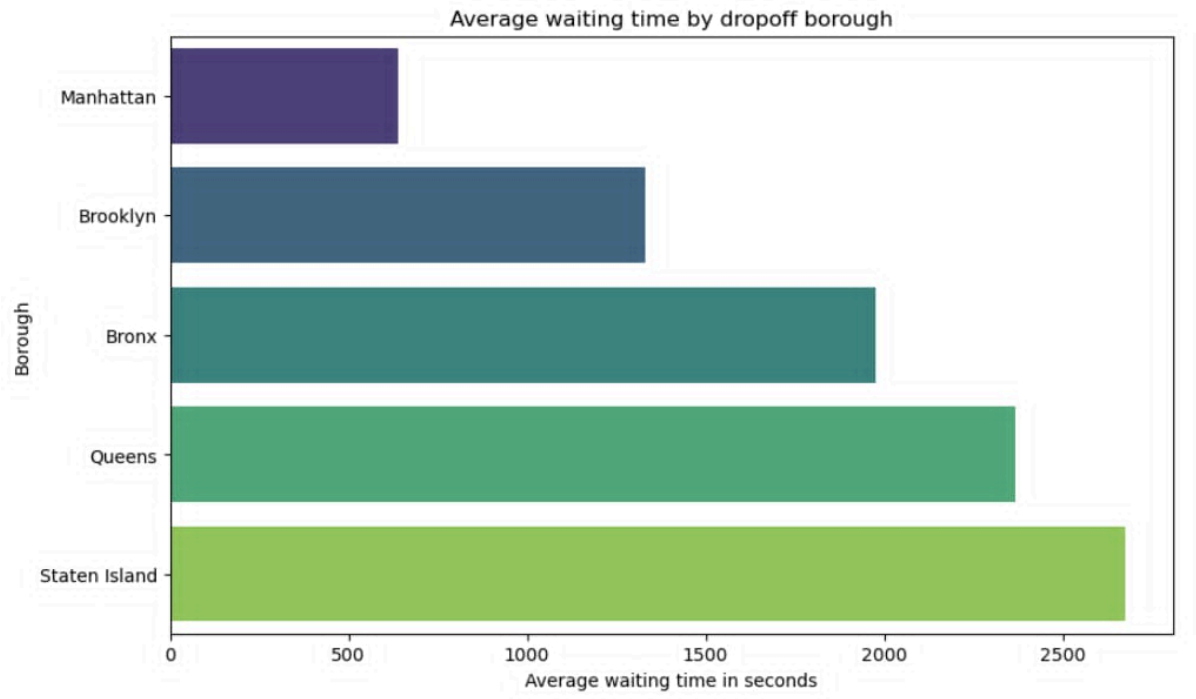
hack_license	**|total_idle_time**	**total_ride_duration**	**total_time**	**utilization_rate**
001EEDEA00E57988E...	293027	308115	601142	0.5125494475514937
01606C9E10D8D0B19...	277259	300871	578130	0.5204210125750264
02548BECEDACA82F0...	205033	97009	302042	0.3211771872785904
02856AFC22881ABCA…	425100	353220	778320	0.4538236201048412
02E3C1D2FE5D53C22…	85142	71026	156168	0.4548050817068798
03A2D28F831C5C3E5...	393780	402300	796080	0.5053512209828158
069B5562096AF7684...	312240	242460	554700	0.4371011357490535
0DC7C87D535512AF8...	19593	17017	36610	0.46481835564053536
0E087EC55DCBE7B2D...	10554	414	10968	0.03774617067833698
0FBF11956EE14B253...	374160	294240	668400	0.4402154398563734

*** We also computed an average utilization rate which is 0.4724117287691575.

Query 2 - The average time it takes for a taxi to find its next fare(trip) per destination borough

This query calculates the average time a taxi waits for its next fare after dropping off a passenger in each borough. First, each trip is ordered by pickup time for each driver (hack_license), and the next pickup time is retrieved using a window function. The waiting time is then calculated as the difference between the next pickup and the previous dropoff time. To clean the data, rows with missing values are removed. To find waiting time, we look at the dropoff time and the next pickup time. If the difference is more than 4 hours, we do not consider them as an idle time, but as a time off. The average waiting time is then calculated for each dropoff borough, and borough codes are mapped to their names. The results are visualized in a bar plot, showing boroughs on the y-axis and their average waiting times in seconds on the x-axis.

Graph:



This bar chart shows the average taxi waiting time by drop-off borough, based on trip_data_1 file.

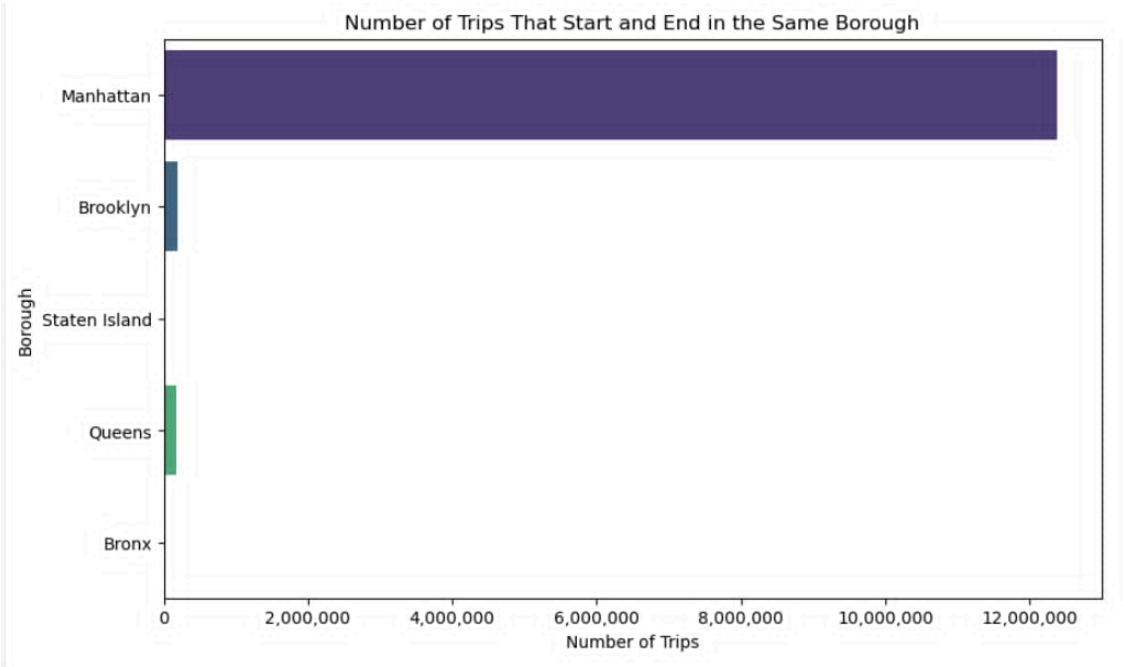
- Staten Island has the longest waiting time, higher than other boroughs. This likely means fewer taxis and lower demand.
- Manhattan has the shortest waiting time, which makes sense because of its high taxi availability and frequent ride requests.
- Bronx, Brooklyn, and Queens fall in between, with Queens having slightly longer waits. This may be due to different demand levels and taxi availability.

Overall, the data suggests that busier areas like Manhattan have shorter waiting times, while quieter areas like Staten Island experience longer gaps between rides.

Query 3 - The number of trips that started and ended within the same borough

In order to calculate how many taxi trips started and ended within the same borough, we filtered the given dataset to keep only trips where the pickup and dropoff borough were the same. Then data was grouped by the pickup borough to count the number of trips within each borough. First the boroughs were in the form of numerical codes, we created a DataFrame borough_names_df to give the borough the official names (Manhattan, Brooklyn, Queens, Bronx, Staten Island).

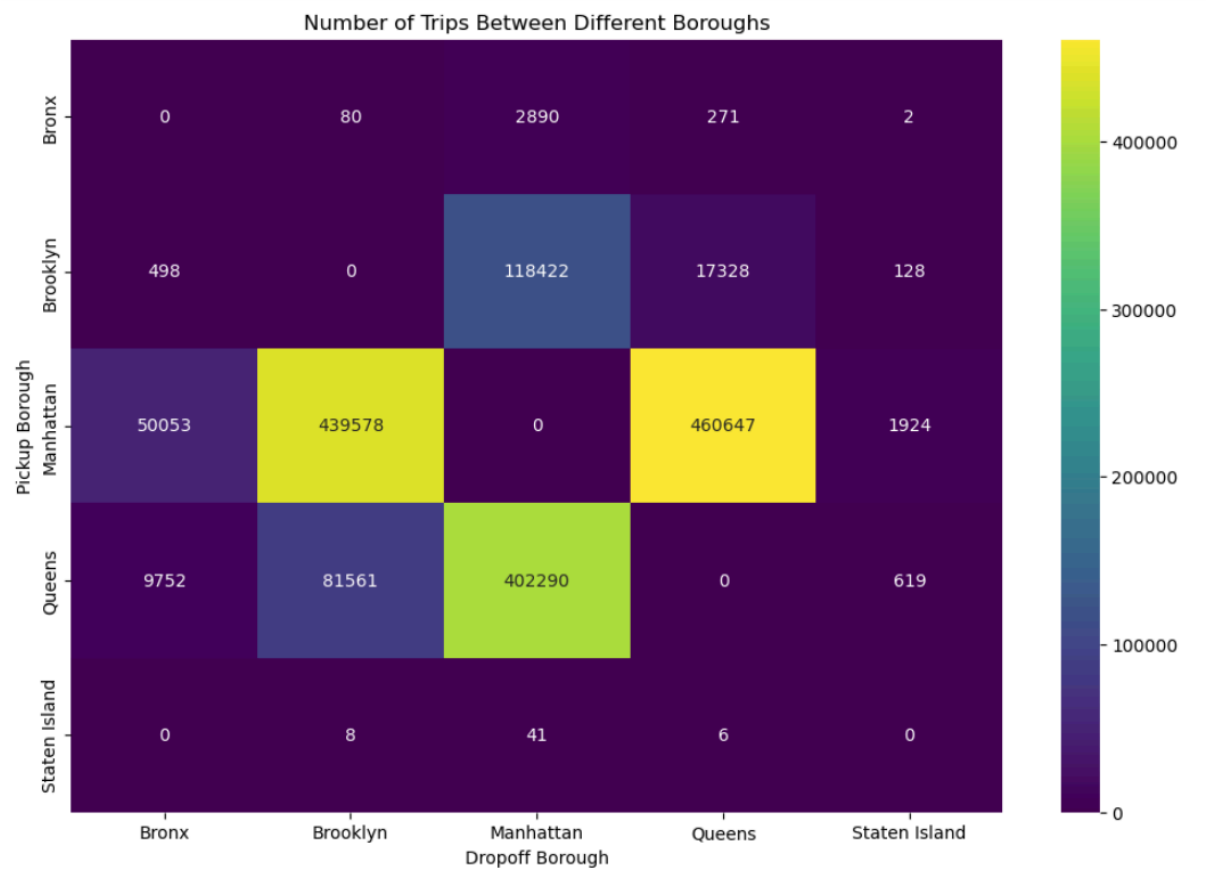
Graph:



From these results, based on the trip_data_1 file it is shown that the most same-borough trips are dominating in Manhattan, Staten Island and Bronx have the lowest number of same-borough trips. This probably does not describe all the data, because we have taken a little of our whole dataset.

Query 4 - The number of trips that started in one borough and ended in another one

To analyze the trips between different boroughs, the dataset was filtered to count the number of trips where the pickup and dropoff locations were in different boroughs. We also grouped the data by pickup and dropoff borough and summed the number of such trips in every possible variation. The result, stored in different_borough_trips_df, represented the numbers of such trips. We again gave the boroughs the names instead of the numeric code. The result on the number of trips that started in one borough and ended in another one resulted in 1,586,098 trips based on the trip_data_1 file which had a total of 14,335,835 trips.



This heatmap is visualizing the number of taxi trips between different boroughs in New York City. It is shown that the majority of inter-borough trips involve Manhattan, making it the central hub of travel. Most of them are 460,647 trips from Manhattan to Queens. Brooklyn and Queens also have significant inter-borough trips. Staten Island has the lowest number of inter-borough trips - this may be due to different demand levels and taxi availability.

Conclusions

The project successfully analyzed New York City taxi data by applying structured data transformations and borough mapping. The results showed that the average taxi utilization rate was 47.24%, with an average idle time of 1798.42 seconds (approximately 30 mins). Manhattan had the shortest waiting time, while Staten Island had the longest, with an average idle time being 4 times longer than in Manhattan (638 vs 2674 seconds). Based on trip_data_1.csv file most same-borough trips occurred in Manhattan, while Staten Island and Bronx had the fewest. Inter-borough trips followed expected patterns, with high traffic between neighboring boroughs. These insights provide a clearer understanding of taxi movement and demand distribution in New York City.