Big Data Project Report

Group member:
Name: Jiaming Zhang N#: 17556219 Email: jz1137@nyu.edu
Name: Mengdi Wang N#: 19722179 Email: mw3265@nyu.edu

Google Doc Link:
https://docs.google.com/document/d/1V8YWxsA0M0yG02-m4VjbLQmGlaBUTPwLDGKC0FR7QH4/edit
Github Repository: https://github.com/benjamin21st/bigdata_2016

Introduction

Taxi data is more than a log for transportation, because of its flexible nature and lack of rigid routine, taxis can go to places where public transportation cannot, hence its trip data are way more complex than that of subways or buses. Also, taxi data tells stories about the city where they are operated: not every neighborhood gets equal amount of taxi visit, nor does every neighborhood share the same amount of pickups. This may serve as evidences for gentrification and social-economic studies.

Comparison of taxi data is even more intriguing and insightful. Why do people take more taxis over weekends than weekdays? Why are April and August almost always the lowest over the years in terms of monthly taxi trips? Why did the trip count increase/decrease over last year at the same time? If we are able to answer these questions, not only can we tell the lifestyle of the people living in this city, but also provide valuable implications for businesses in related industries.

The taxi raw data itself is also worth studying. Given the flexibility of taxis, there are tens of data attributes for each single trip, together it forms an enormous dataset, which is perfect for studies in big data.

Using our knowledge in mapreduce and relational databases, we aim to unravel the stories and myths in these data by developing algorithms to clean up the large raw dataset, serving it efficiently through a web server API, and creating an intuitive graphic user interface.

Data source

Taxi Trip Data

The original data source for taxi fact book is offered by http://www.nyc.gov/ including yellow taxi trip records and green taxi trip records in year 2015. The entire data set contains 19,233,777

records for green taxi and 141,620,046 records for yellow taxi. The total size of the data is about 30GB.
Green taxi data contains 20 columns, and detailed description can be found here: http://www.nyc.gov/html/tlc/downloads/pdf/data_dictionary_trip_records_green.pdf. However, the actual data has two more columns (Ehail_fee) than the definition in this document, hence in our study, the actual columns number is 21.
The definition in the columns falls into the following sections:

1. Date time: pickup datetime, drop off datetime
2. Location information: pickup longitude, pickup latitude, dropoff longitude, pickup latitude.
3. Categories: store_and_fwd_flag, VenderID, RateCodeID, Trip_Type, Payment_Type
4. Scalar: the remaining columns are all scalars.

Similarly, the yellow taxi data has 19 columns. And the definition can be find from this link: http://www.nyc.gov/html/tlc/downloads/pdf/data_dictionary_trip_records_yellow.pdf
The definition in the columns falls into the following sections:

1. Date time: pickup datetime, drop off datetime
2. Location information: pickup longitude, pickup latitude, dropoff longitude, pickup latitude.
3. Categories: store_and_fwd_flag, VenderID, RateCodeID, Trip_Type, Payment_Type
4. Scalar: the remaining columns are all scalars.

New York Neighbourhood Data
New York Neighbourhood Data is provided by zillow, we used for spatial analysis for taxi trips. The data was downloaded in shapefile format. We wrote python script to extract the neighbourhoods only in the five boroughs of New York City, and saved as the location data. The location data consists of the following columns:

- Name, example: [‘NY’, ‘Greenwich Village’ ,’New York City-Mahattan’]
- Bound Box, the outer box of the neighbourhood
- Polygon coordinates list

This data have 128 objects, and is used for the spatial data processing and heat map rendering.

NYC Taxi Factbook
Based on existing NYC Taxi Factbook, we extracted the content that our current dataset allows to generate, and developed a web service API that returns data matching a given query, and created data tables and interactive graphs for visualization.

The web service API talks to a database powered by MySQL and hosted on our local machine. The database schema contains three tables tripsStats, tripsSpatialStats, tripsPolygonStats. Each table has fields that corresponds to the data attributes generated by data processing, the attribute names and their types are listed below:

tripsStats	**tripsSpatialStats**	**tripsPolygonStats**

id (Integer(),) datetime (DateTime(),) taxi_type (Integer(6)) rate_code (Integer(6)) total_cnt_vendorID_1 (Integer(10)) total_cnt_vendorID_2 (Integer(10)) total_trip_time (Decimal(12,) total_pass_cnt (Integer(10)) total_trip_dst (Decimal(12,) total_sfflag_Y (Integer(10)) total_sfflag_N (Integer(10)) total_fare_amount (Decimal(12,5) total_extra (Decimal(12,5) total_mta_tax (Decimal(12,5) total_tip_amount (Decimal(12,5) total_ehail_fee (Decimal(12,5) total_imprv_srchg (Decimal(12,5) total_total_amount (Decimal(12,5) total_payment_1 (Integer(10)) total_payment_2 (Integer(10)) total_payment_3 (Integer(10)) total_payment_4 (Integer(10)) total_payment_5 (Integer(10)) total_payment_6 (Integer(10)) total_trip_type_0 (Integer(10)) total_trip_type_1 (Integer(10)) total_trip_type_2 (Integer(10)) total_tolls_amount (Integer(10)) total_record_cnt (Integer(10))	id(Integer()) datetime(DateTime()) taxi_type(Integer(6)) rate_code(Integer(6)) action(Integer(6)) total_record_cnt(String(2048)	id(Integer()) datetime(DateTime()) taxi_type(Integer(6)) rate_code(Integer(6)) action(Integer(6)) PolygonId(Integer(6)) Count(Integer(10))

Table 1. Factbook Database Schema

The web service API itself is built using Flask, a minimal python framework. And queries are completed using SqlAlchemy ORM (Object Relational Model) instead of raw SQL queries. But raw SQL queries are executed as a testing procedure to ensure our data is correct. The data we are interested to acquire for the factbook are: total traveling distance of both yellow taxi and green taxi, average trip count of both taxi types in a number of intervals (yearly, monthly, weekly, and daily), passenger counts for both taxi types in the same intervals mentioned earlier, and trip count distribution grouped by month and by weekday.

In terms of how each of these queries' results are computed. The Total Traveling Distance was the sum of all data row's individual traveling distances for each taxi type; Average Trips was the sum of all data row's individual total count, divided by the interval, 1 for yearly, 12 for monthly, 52 for weekly, and 365 for daily; Passenger Counts were calculated in the same fashion as trip count; Trip distributions were calculated by aggregating trip counts in a hash table, where key is

the integer encoded timerange (January is 1, Febuary is 2, etc. Similarly, Monday is 1, Tuesday is 2, etc.).

The web based frontend contains 3 data tables for the first three queries mentioned above, and two interactive graphs for the distribution. Graphs are generated using d3.js , a data visualization javascript library. This frontend allows us to visualize our data in a more intuitive manner, and allows us to easily answer the essential information in NYC factbook:

- NYC Factbook gives out the mileage a typical taxi travels within a year, but unfortunately, we do not have the medallion info of each taxi, therefore we calculated the total distance all taxis travelled within this year. For yellow taxi, this number is over 750 million miles, which is about 4 round trips from earth to the sun; For green taxi, this number is over 55 million miles, which is about halfway from earth to the sun.

Total Traveling Distance (miles)

Yellow	Green
752619054.03	55198038

Table 2. Total Traveling Distances

- For average trips per day, our result shows 327,614 miles for yellow taxi, which is less than that of 2014 based on the 2014 NYC Taxi Factbook.

Average Trips

Time Interval	Yellow	Green
Year	119579474	19164472
Month	9964956	1597039
Week	2299605	368547
Day	327614	52505

Table 3. Average Trips

- Regarding passenger count, in 2014, there are an average of 600,000 passengers per day for yellow taxis, our record shows 550,781 in 2015. Which also indicates a decline.

Passenger Counts

Time Interval	Yellow	Green
Year	201035209	26346814
Month	16752934	2195567
Week	3866061	506669
Day	550781	72183

Table 4. Passenger Counts

- The NYC Taxi Factbook also has a *Daily Trips* section that shows the trend of aggregated trips over a range of time at a monthly interval. We grouped data by month instead of by day, and visualized through d3.js. From the graph below, we find the following:
    - Although significantly less in quantity, green taxi trips count follows the same trend as the yellow taxies
    - April and August are the lowest points, this is true in previous years too. This tendency may have been caused by comparatively changeable weathers during these months, which may affect road condition and encourage travellers to take public transit like subway that are not easily affected by weather.
    - October peaks in our dataset for 2015, but not so much in previous years, it would be interesting to find out what event in October, 2015 has caused the shift

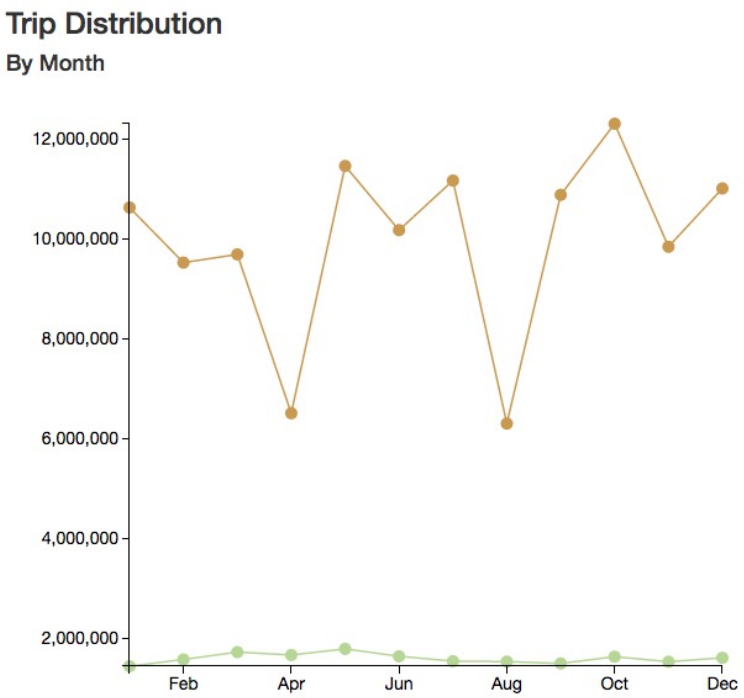

Figure 1. Trip Distribution by month

- We are also interested in the distribution over weekdays throughout the year. And through Figure 2, we have a few interesting insights:
    - Yellow taxis and green taxis shows two different trends, changes in yellow taxi trip count appears to be fluctuating between a range, while green taxi counts shows a steady growth as the week goes near weekends, reaching a peak on Saturday.

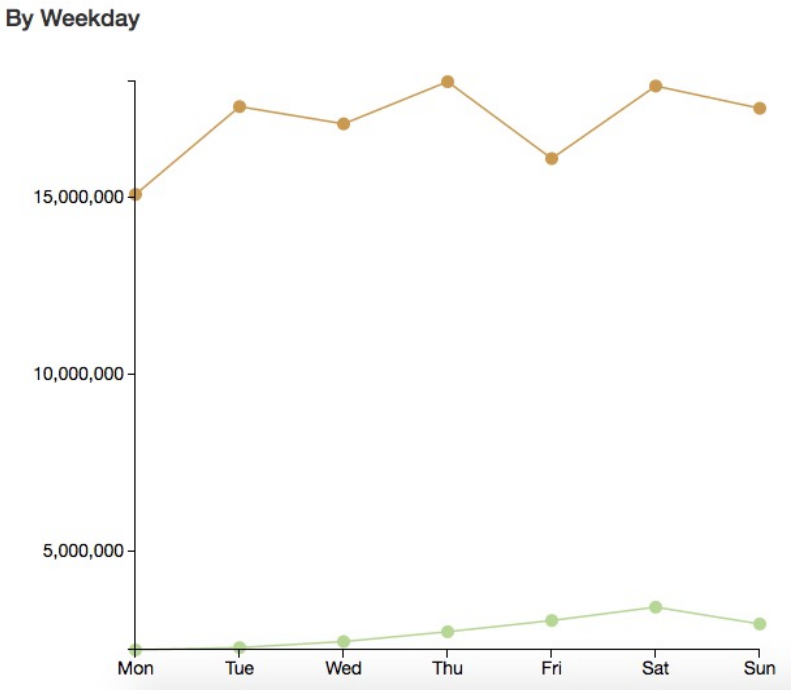

Figure 2. Trip Distribution By weekday

For next steps of our Factbook implementations, we would streamline the data import process, ideally having a web app that allows the Factbook publisher to select raw data file (can be uploaded or linked to a public data storage), then the system automatically performs data processing, database entry, and front end UI generation. We could also enable user to pick historical data for comparison, and allow better navigation in our data visualization such as panning and zooming.

Data Processing

The input was consited of both the yellow trip data and green trip data. The output is designed as follows:

(Date_time, taxi_type, rateID), [total_value1, total_value2, …]

The key of the output consists of three parts:

- Date_time: is the date time of each output, the unit length is 1 day
- taxi_type: indicates the taxi type of the record, 1: green taxi, 2: yellow taxi;
- rateID: shows the rate id of the record, the definition of the value is the same as in the raw data.

The value of the output is a list, where each item is the total values for all the trips with the corresponding key.

Hadoop stream is used for data processing, the map and reduce design is as follows:

Map:

The input of map is each row of the data.

The output key value pair is:

((Date_time, taxi_type, rateID), [value1, value2, value3…])

Algorithm: For the input, we simply split the input data by lines as output

Reduce:

The input of reduce is:

((Date_time, taxi_type, rateID), [value1, value2, value3…])

The output out reduce is:

((Date_time, taxi_type, rateID), [total_value1, total_value2, total_value3…])

Algorithm: For the scalar value columns, we simply sum all the the values as the total value for each trip. For the categories type, we count the occurrence of each value (e.g. if the input data consists of 3 record and their trip_type values are 1,1,2, the output would be total_triptype_1:2, total_triptype_2:1).

Spatial Analysis

In this section, we will discuss our work on the spatial analysis for nyc taxi trips. We have two purposes, the first one is to find statistic results for taxi usage in each neighbourhood. The second is to find some origin-destination analysis result.

Data Processing

The input were both the yellow trip data and green trip data. And the New York Neighbourhood data is used for polygon detection.

For purpose 1, the output is designed as follows:
(Date_time, taxi_type, rateID, action), [value1, value2, …]
The key of the output consists of three part:

- Date_time: is the date time of each output, the unit length is 1 day
- taxi_type: indicates the taxi type of the record, 1: green taxi, 2: yellow taxi;
- rateID: shows the rate id of the record, the definition of the value is the same as in the raw data.
- Action: indicates the action in this neighbourhood, 0: pickup in this neighbourhood, 1: dropoff in this neighbourhood.

The value of the output is a list, where each item is the total number of trips in this neighbourhood with the corresponding key.

Hadoop stream is used for data processing, the map and reduce design is as follows:

Map:

The input of map is each row of the data.

The output key value pair is:

((Date_time, taxi_type, rateID), (pickup_neighhourhood, dropoff_neighbourhood))

Algorithm: For each record, the map side algorithm finds the neighbourhood for pickup location and drop off location. This is a 2 phase algorithm. Given a location with longitude and latitude, In the first phase, the algorithm iterates all the neighbourhood bound box data, and return all the

neighbourhoods the location may be located. Because the length of the neighbourhood list is 128, and each bound box checking needs 2 comparison. Hence this process is very fast. If one location is both identified by two neighbourhoods. The second phase will be applied on this location. In the second phase, the algorithm checks all the potential neighbourhoods by calculating the location and all the polygon coordinates of the neighbourhood. This phase is time consuming, however it occurs rarely. From our observation, some location still belonged to 2 neighbourhoods. The reason for this is that some neighbourhood is a true subset of another neighbourhood. For this situation, we just output all the neighbourhoods as a result.

Reduce:
The input of reduce is

((Date_time, taxi_type, rateID), (pickup_neighhourhood, dropoff_neighbourhood))

The output out reduce is:

(Date_time, taxi_type, rateID, action), [value1, value2, …]

Algorithm: In reduce side, we used a hashmap H to store all the keys, the value of the hashmap is a list with 128 items, each item at i'th position is the total count for the i'th neighbourhood. Given an input with key k and value (i, j), the algorithm increased i'th position value in H[(k, 0)] by 1, and increased j'th position value in H[(k, 1)] by 1. If i, or j is a list, the algorithm will increase all the neighbourhood in the list by 1.

For purpose 2, the output is designed as follows:
(origin_poly_id, dest_poly_id, taxi_type, rateID), count
The key of the output consists of three part:

- origin_poly_id: neighbourhood id for origin of trip
- dest_poly_id: neighbourhood id for destination of trip
- taxi_type: indicates the taxi type of the record, 1: green taxi, 2: yellow taxi;
- rateID: shows the rate id of the record, the definition of the value is the same as in the raw data.

The value of the output is the total count of trips from origin neighbourhood to destination neighbourhood.

Hadoop stream is used for data processing, the map and reduce design is as follows:
Map: We used the same map in purpose 1.

Reduce:
The input of reduce is

((Date_time, taxi_type, rateID), (pickup_neighhourhood, dropoff_neighbourhood))

The output out reduce is:

(origin_poly_id, dest_poly_id, taxi_type, rateID), count

Algorithm: Because the number of the neighbourhood is 128, then the total key pairs is at most O(10^4). Thus we can use a hashmap H to store all the count. Given an input as (DT, taxi_type, rateID), (i,j), the hashmap item with key (i,j,taxi_type, rateID) is increased by 1.

Application

Two applications were built based on the data generated from the hadoop. They are listed as follows.

Taxi usage rank

Form the trip spatial analysis, we basically get a rank for the usage of taxi between different neighbourhoods. The following tables shows the top 10 neighbourhoods in NYC that are better served by taxi. Midtown is the most popular neighbourhood, and the neighbourhoods on the list have high median household income among NYC. (household map http://project.wnyc.org/median-income-nation/?#13/40.7420/-73.9924). The rank is expendable, we can get other ranks by adding different filters, like taxi_type = Green or date time.

rank	Neighbourhood	Amount
1	Midtown, New York City-Manhattan	45009718
2	Upper East Side, New York City-Manhattan	37231528
3	Upper West Side, New York City-Manhattan	23548857
4	Gramercy, New York City-Manhattan	21513593
5	Chelsea, New York City-Manhattan	18138800
6	Garment District, New York City-Manhattan	15162592
7	Greenwich Village, New York City-Manhattan	11753453
8	Murray Hill, New York City-Manhattan	9522897
9	East Village, New York City-Manhattan	9423773
10	Clinton, New York City-Manhattan	9031628

Table 5, Top 10 Taxi Usage (dropoff, pickup)

NYC taxi usage heat map (site: http://cims.nyu.edu/~mw3265/NYC_Taxi_Map/index.html)

Besides, we built a heatmap to visualize the taxi usage by using d3. The screenshot are shown as follows.

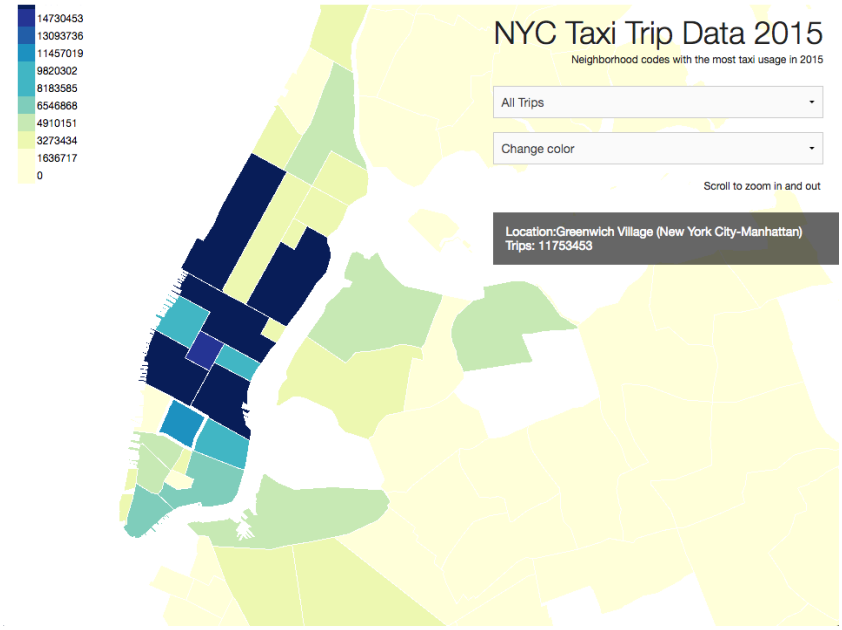

Figure 3. NYC 2015 All Taxi Trips heat map

We can visualize all the taxi trips, and select the type of different taxi types. From figure 4, green taxi are mainly served outside manhattan, and through figure 5, yellow taxi rarely go outside of manhattan.

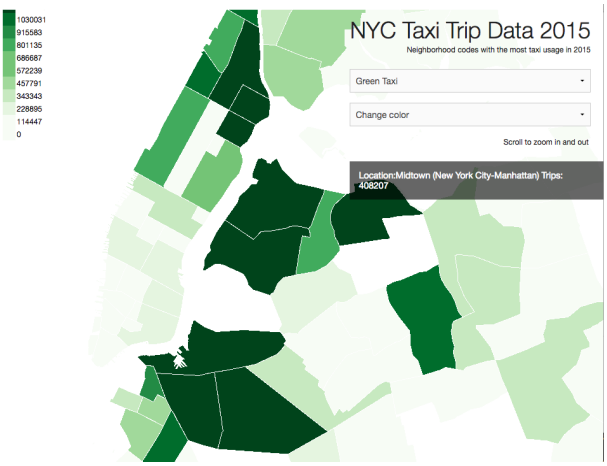

Figure 4. NYC 2015 Green Taxi Trips heat map

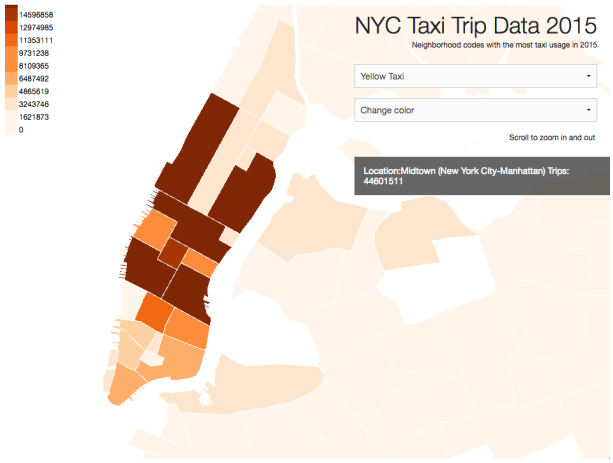

Figure 5. NYC 2015 Yellow Taxi Trips heat map

We can visualize dropoff and pickup separately as well.

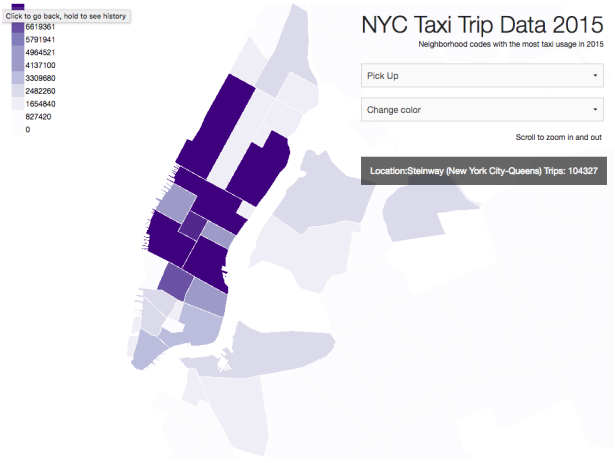

Figure 6. NYC 2015 Taxi Pickup heat map

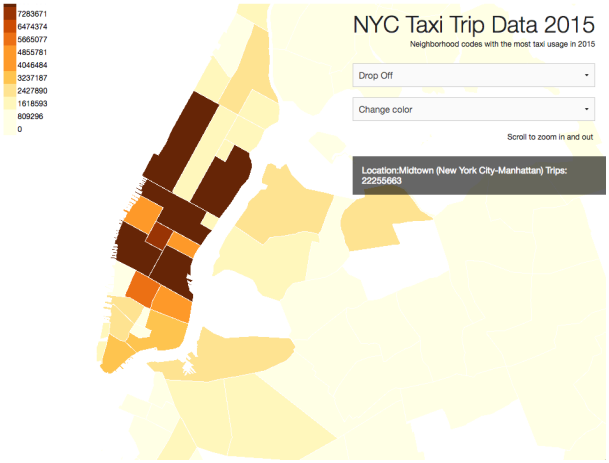

Figure 7. NYC 2015 Taxi Drop off heat map

Rank of Trips between neighbourhoods

From the origin-destination analysis, we can get popular tip list as follows:

rank	Origin	Destination	Amount
1	Upper East Side	Midtown	6516892
2	Midtown	Upper East Side	6516892
3	Midtown	Garment District	3764784
4	Garment District	Midtown	3764784
5	Midtown	Upper West Side	3678148
6	Upper West Side	Midtown	3678148
7	Midtown	Gramercy	3560040
8	Gramercy	Midtown	3560040
9	Upper East Side	Upper West Side	3392189
10	Upper West Side	Upper East Side	3392189

Table 6. Trip Rank

The trips between Midtown, Upper East Side, Upper West Side, Garment District, Gramercy are of the highest frequency.

Trip Confusion Matrix between neighbourhoods in NYC

Beside the Trip rank between different neighbourhoods, we created some confusion matrices for trips. The confusion matrix gives a direct idea about the taxi usage between neighbourhood. The following figure is the confusion matrix for taxi usage in Manhattan, each row stands for the origin neighbourhood, and each column is the destination neighbourhood. In the following figure, we found that the diagonal is significant, which implies for most trips, the origin and destination are in the same neighbourhood. For these trips, the number of the trips in midtown, upper east, upper west, gramercy ranked top. For the trips between different neighbourhoods,

both row and column for midtown are the most significant. This implies that people either in midtown or go to midtown would more likely to use taxi.

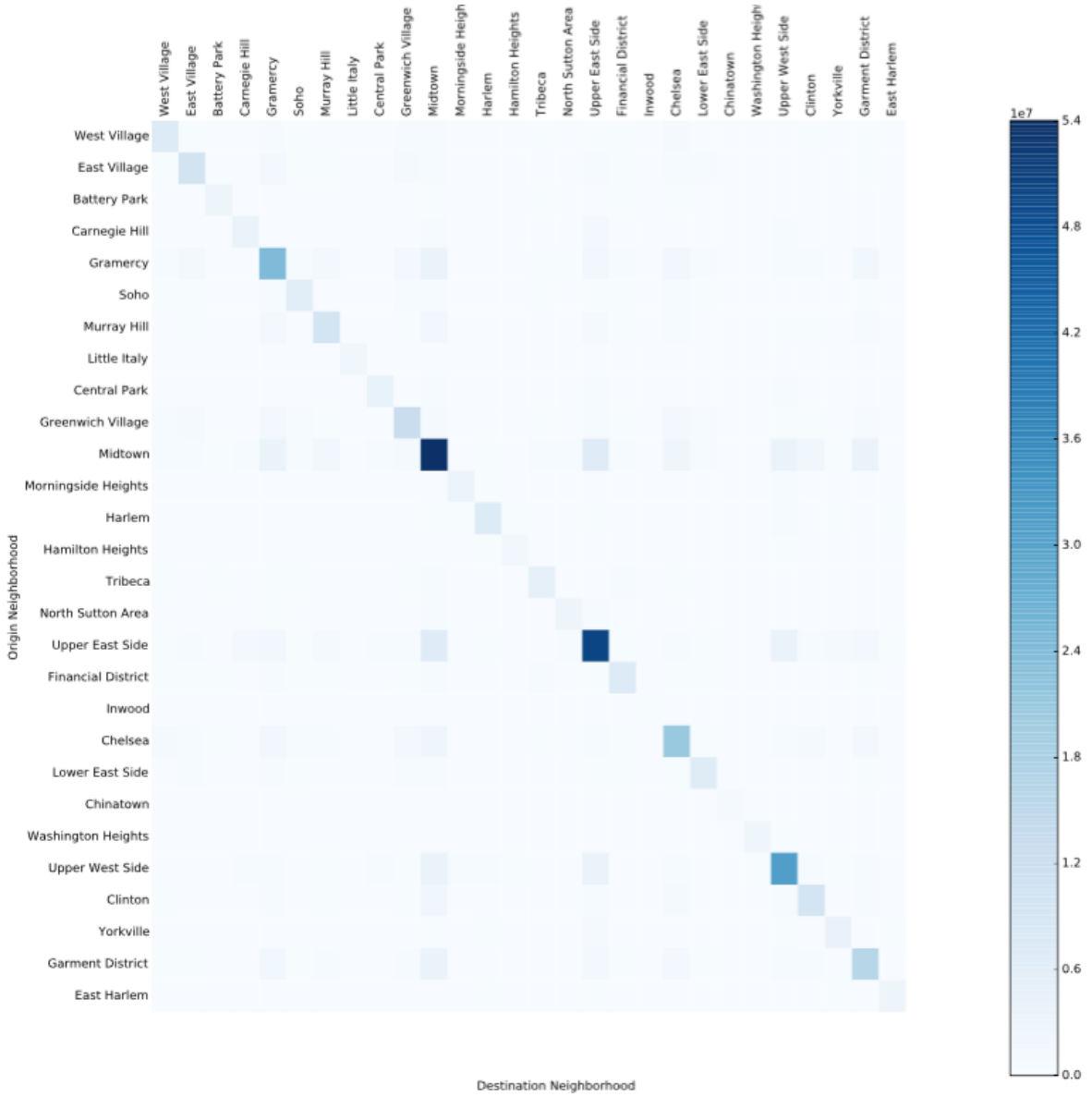

Figure 8. Confusion matrix for trips in Manhattan

Data Quality

We found some dirty data during our processing.

Dirty data:

- In yellow and green data, rate_type = 99 appears several times in the data, which is beyond the description
- In green data, before June, there are two empty columns in the data.
- Some trip has too small trip time like 20 seconds with 1.8miles distance.

Conclusion

In our recreation of NYC Taxi Factbook, we obtained valuable insights in taxi trips' volume, type, and distribution through our REST API server and responsive charting front end. We also developed a conceptual framework for automatically generating NYC taxi data and efficiently serving them on the fly.

During the data processing steps, we used hadoop streaming for data statistics and polygon detection. All the hadoop tasks can be completed for about 15 minutes.
For the spatial data analysis, we found the most popular neighbourhoods and most popular taxi trips. Besides, We built an heatmap website for visualizing the taxi usage in NYC and confusion matrices for visualizing the trips between neighbourhoods.

Future Work

However, our Factbook has its limitations. Firstly, the source data did not contain medallions info, therefore we cannot produce data analysis on each individual taxis; Secondly, we did not get a chance to leverage trip fares data, which may contain yet more insights; Thirdly, we did not provide a solution for interactively choosing historical data, or making any complex data visualization or analysis using our web GUI. As is mentioned in the Factbook section, our efforts in the next steps will be focused on these aspects. Our heatmap can be more variation and selected between different dates.