Big Data Analytics Project

Tianhao Li & Jiaxiang Lin

1 Overview

This project analyzes yellow taxis, green taxis, Uber, and Citi-bikes data in NYC. We will focus on the following questions:

1. Has the green taxis better served regions where are underserved by yellow taxis?
2. When is the morning/evening peak time? Where will people go during the peak time?
3. If it has, which borough is negatively affected most/least by Uber the trips for yellow/green taxis?
4. Which Citi-bike station is the most popular start/end station?

In order to answer these questions, we used several Mapreduce tasks to processing the data in the AWS platform. After get the data we want, we used D3.js to visualize the result and then analyze it based on diagrams and data.

To complete this project, our members both actively participated in.
Tianhao Li:
Designed the analytics tasks. Collected all data used, including different kinds of geographical data. Visualized all the result data using D3.js. Helped with technique problems encountered, for example, gave the solution for the low efficiency of judging points and polygons intersection algorithms.

Jiaxiang Lin:
All data processing tasks, including: building AWS clusters, preprocessing raw data, and writing Mappers and Reducers for each task and running them on AWS.

2 Data

2.1 Analytics object data

Taxi data (yellow and green)
- http://www.nyc.gov/html/tlc/html/about/trip_record_data.shtml
Uber Data
- https://github.com/fivethirtyeight/uber-tlc-foil-response
Citi-Bike Data
- https://www.citibikenyc.com/system-data

2.2 Geographic Data

Neighborhood Tabulation Areas (NTAs)
- http://www1.nyc.gov/site/planning/data-maps/open-data/dwn-nynta.page
Community Districts
- http://data.beta.nyc/dataset/nyc-community-districts

3 Environment configuration and Data preprocessing

3.1 Environment configuration

All the data processing are implemented on AWS EMR cluster, and judge the region based on json file using *shaply* library in Python3.5. There are many polygons in the json file, each of them represents a certain region of NYC.
The following is some tips to install needed package when creating cluster in EMR and how to fetch the json file when running.

Installing of *shapely* library on Ubuntu

Sudo apt-get install python-pip
Sudo apt-get install libgeos-dev
Pip install shapely

Installing of *shapely* library on EMR:

When create cluster, add the following script into Bootstrap actions, which can be found under Advanced options Step 3:

#!/bin/bash
sudo yum -y install libgeos
sudo pip install shapely

Uploading json files to S3

When running streaming job, in arguments window write:

cacheFile s3://mydirectory/myfile.json#reference

Loading json files in mapper

In mapper, write the following in arguments:

```
import json
with open("reference") as f:
    js = json.load(f)
```

3.2 Data preprocessing

For all yellow taxis data we used, some records put longitude in latitude field, and put latitude in longitude field. So we corrected them first, and picked the longitude range in -80 to -70. Also in the mappers, we discarded records with longitude or latitude equal to 0.

For the geographic files, the original json files have 15 digits for each coordinates, which would increase the running time of algorithm. We truncated the digits to avoid this problem

4 Data processing

4.1. Has the green taxis better served regions where were underserved by yellow taxis?

Data sets used:

Yellow (2013-5 ~ 2013-10) Green(2013-8 ~ 2013-10)

MapReduce tasks:

(1)Task1

YellowRoundMap.py/GreenRoundMap.py:

Function: Because it will cost a lot of time to judge millions of point one by one, so we first hash them to several buckets, by round the Pickup_longitude and Pickup_latitude by 4.
Input: Green(2013-8 ~ 2013-10)
Output: (key, value) = ((Pickup_longitude,Pickup_latitude,month), revenue)

YG_RoundReduce.py:

Function: Count the total trips and revenue in each bucket. Both GreenRoundMap.py and YellowRoundMap.py can use this Reducer.
Input: Output of GreenRoundMap.py or YellowRoundMap.py.
Output: (key, value) = ((longitude,latitude), (month,current_trips,current_revenue))

Performance:
For Green Taxi data: 1 min with 1 Master cluster and 9 Core clusters.
For Yellow Taxi data: 4 min with 1 Master cluster and 9 Core clusters.

(2)Task2

YG_FinalMap_AreaID_Month_trips_revenue.py:

Function: Each bucket generated by YG_RoundReduce.py has key (longitude,latitude). This mapper judge which polygon the key belongs to based on nta_simplified.json file. Both Yellow and Green Round result can use this Mapper.
Input: Output files of YG_RoundReduce.py.
Output:(key, value) = ((region_id,month), (trips,revenue))

YG_FinalReduce_AreaID_Month_trips_revenue.py:

Function: Several bucket may in the same region, so calculate the total trips and revenues for each region.
Input: Output of YG_FinalMap_AreaID_Month_trips_revenue.py.
Output:(key, value) = (region_id, (month,trips,revenue))

Arguments:
-cacheFile s3://jl7379-ds1004-sp16/Project/YellowGreen/nta_simplified.json#reference
Performance:
For Green Taxi data: 3 min with 1 Master cluster and 9 Core clusters.
For Yellow Taxi data: 81 min with 1 Master cluster and 9 Core clusters.

(3)Task3

Count_YG_By_nta_Map.py:

Function: Get the region_id, month, and trips for Reducer to count. Both Yellow and Green Final result can use this Mapper.
Input: Output files of YG_FinalReduce_AreaID_Month_trips_revenue.py.
Output:(key, value) = (region_id, (month,trips))

Count_Yellow_By_nta_Reduce.py/Count_Green_By_nta_Reduce.py:

Function: Calculate total trips of Green Taxi from May to July and from Aug to Oct for each nta region.
Input: Output of Count_YG_By_nta_Map.py.
Output:(key,value) = (region_id, (type,trips_5to7,trips_8to10)),type has two values: yellow and green.

Performance:1 min with 1 Master cluster and 9 Core clusters.

(4)Task4
BetterServed_Map.py:
Function: No operation. Notice that we need to put the output files of Count_Green_By_nta_Reduce.py and Count_Yellow_By_nta_Reduce.py together.
Input: Output files of Count_Green_By_nta_Reduce.py and Count_Yellow_By_nta_Reduce.py
Output:(key,value) = (region_id, (type,trips_5to7,trips_8to10))

BetterServed_Reduce.py:
Function: Calculate the total trips of Yellow and Green from May to July and from Aug to Oct for each nta region, then calculate the index(Green(8-10)/(All(8-10) - All(5-7))) of each nta region.
Input: Output of BetterServed_Map.py
Output: (key, value) = (region_id, index), print to output files.

Arguments:
-D mapreduce.job.reducers=1
Performance:1 min with 1 Master cluster.

4.2 When is the morning/evening peak time? Where will people go during the peak time?

Data sets used:
Yellow(2015-6) Green(2015-6) Uber(2015-6) Citi-bike(2015-6)
MapReduce process:
(1) Task1
Peaktime_201506_Map.py:
Function: For each trip in the input files, analyze it is a weekday trip or weekend trip and the hour and minute of its pick_up time.
Input: Yellow(2015-6) Green(2015-6) Uber(2015-6) Citi-bike(2015-6)
Output:(key, value) = ((Type,Daytag,Hour,Minute), 1), Type has four values: yellow, green, uber, or citibike, Daytag has two values: weekday or weekend, Minute has two values: 0 or 30

Peaktime_201506_Reduce.py:
Function: Count the total trips for each key generated in Mapper.
Input: Output of Peaktime_201506_Map.py.
Output:(key, value) = ((Type,Daytag,Hour,Minute), trips),Type has two values: yellow, green, uber, or citibike,Daytag has two values: weekday or weekend,Minute has two values: 0 or 30

Performance:2 min with 1 Master cluster and 2 Core clusters.

(2)Task2
For the second question, we take the morning peak time (8:00-9:00) and evening peak time (18:00-19:00) during the weekday to observe the difference. We only consider the region in Manhattan.

Weekday_201506_89_YG_Map.py:
Function: For the trips during 8:00-9:00 from Monday to Friday, get the pickup region based on pickup longitude and pickup latitude, and get the drop-off region based on dropoff longitude and drop-off latitude. Json file is community_district_simplified.json.
Input: Yellow(2015-6) Green(2015-6)
Output:(key, value) = (Pickup_region_id,Dropoff_region_id,1)

Weekday_201506_1819_YG_Map.py:
Function: Same as Weekday_201506_89_YG_Map.py, except the time is during 18:00-19:00.
Input: Yellow(2015-6) Green(2015-6)
Output:(key, value) = ((Pickup_region_id,Dropoff_region_id), 1)

201506_Anytime_YG_Reduce.py:
Function: Count the total trips from Mapper.
Input: Output of Weekday_201506_89_YG_Map.py or Weekday_201506_1819_YG_Map.py
Output: (key, value) = ((Pickup_region_id,Dropoff_region_id), trips), print to output files.

4.3 If it has, which borough is negatively affected most/least by Uber the trips for yellow/green taxis?

Data sets used:
Yellow (2014-8) Green(2014-8) Uber(2014-8)
For the data above:
UberTrips=829275
GreenTrips=1344943
YellowTrips=12688877
UberTrips/AllTrips=829275/14863095=0.0558
MapReduce process:
(1)Task1
uber_round_Map.py:
Function: Because it will cost a lot of time to judge millions of point one by one, so we first hash them to several buckets, by round the Pickup_longitude and Pickup_latitude by 4.
Input: Yellow (2014-8) Green(2014-8) Uber(2014-8)
Output:(key, value) = ((longitude,latitude,type), 1),Type has three values: yellow, green, uber

uber_round_Reduce.py:
Function: Count the total trips in each bucket.
Input: Output of uber_round_Map.py.
Output:(key, value) = ((longitude,latitude,type), trips), print to output files.

Performance: 1 min with 1 Master cluster and 9 Core clusters.

(2)Task2
uber_final_Map.py:
Function: Each bucket generated by uber_round_Reduce.py has key (longitude,latitude). This mapper judge which polygon the key belongs to based on nta_simplified.json file.
Input: Output files of uber_round_Reduce.py.
Output:(key, value) = (region_id, (type,trips)), print to uber_final_Reduce.py

uber_final_Reduce.py:
Function: Count the total uber trips and uber trips for each nta region, then calculate the index(UberTrips/AllTrips - NYCUberTrips/NYCAllTrips). UberTrips/AllTrips has already been calculated.
Input: The output of uber_final_map.py.
Output:(key, value) = (region_id, index)

Arguments:-cacheFile s3://jl7379-ds1004-sp16/Project/Uber/nta_simplified.json#reference
Performance:32 min with 1 Master cluster and 9 Core clusters.

4.4 Which Citi-bike station is the most popular start/end station?

Data sets used:
Citi-bike(2015-1~2015-6)
MapReduce process:
citibike_2015_in_Map.py:
Function: For each trip from input, get end_station_id and end_station_name. End_station_longitude and end_station_latitude are also been put in value, in case for future use.
Input: Citi-bike(2015-1~2015-6)
Output:(key, value) =
(end_station_id,(end_station_name,end_station_longitude,end_station_latitude))

citibike_2015_out_Map.py:
Function: For each trip from input, get start_station_id and start_station_name.
Start_station_longitude and start_station_latitude are also been put in value, in case for future use.
Input: Citi-bike(2015-1~2015-6)
Output:
(key, value) = (start_station_id, (start_station_name,start_station_longitude,start_station_latitude))

citibike_2015_inout_Reduce.py:
Function: Count for total trips end in each station or start in each station. Both citibike_2015_in_Map.py and citibike_2015_out_Map.py can use this Reducer.
Input: Output of citibike_2015_in_Map.py or citibike_2015_out_Map.py.
Output:(key,value) = (end/start_station_id,
(end/start_station_name,end/start_station_longitude,end/start_station_latitude,trips)), print to output files.

Performance:
For citibike_2015_in_Map.py: 1 min with 1 Master cluster and 2 Core clusters.
For citibike_2015_out_Map.py: 1 min with 1 Master cluster and 2 Core clusters.

5 Analysis and Visualization

5.1 Have the green taxis better served some regions?

Green taxis was launched around the beginning of August, 2015. In order to answer this question above, we can calculate the index below.

Index of green taxis' impact: Green(8~10)/(All(8~10) - All(5~7))

For each nta region, using the increasing of green taxis trips versus increasing of all taxis trips between 3 months before and after the launch of green taxis. Based on the data we calculated, we display the index on the map.

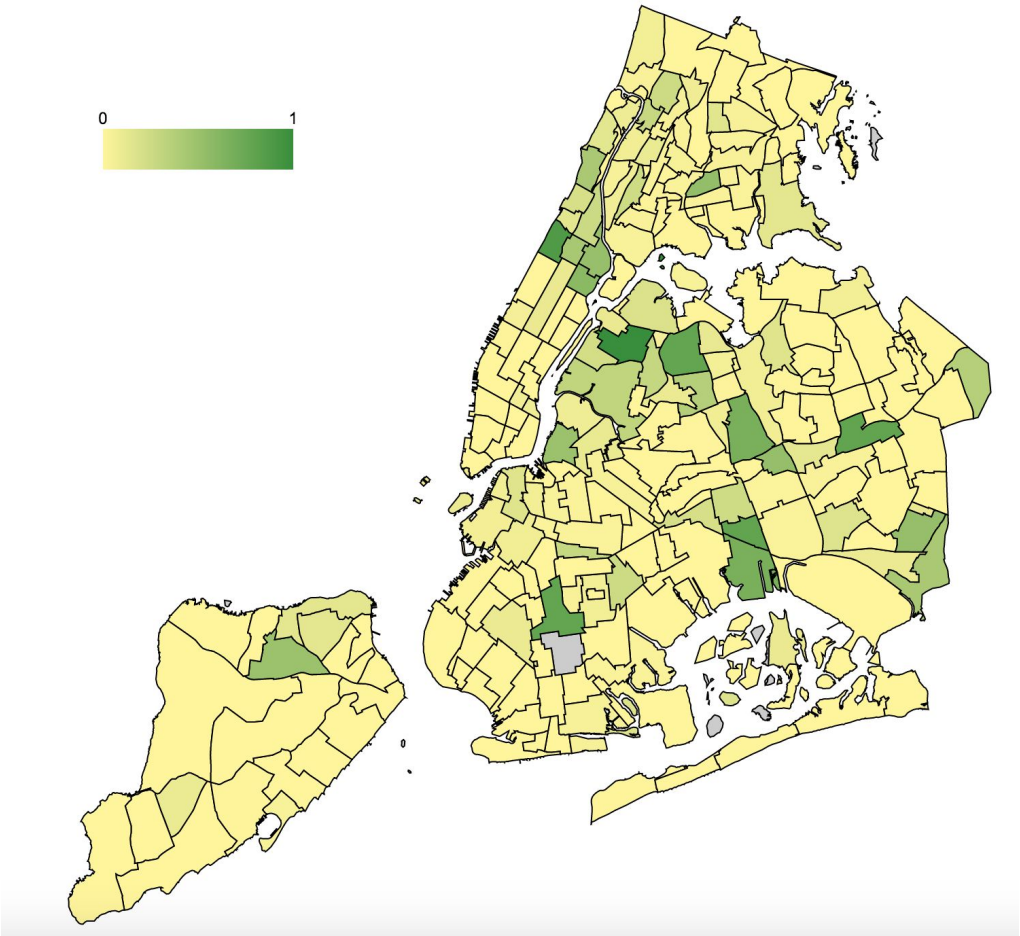

Figure 5.1 Index of green taxis' impact

From the diagram, we can find that Astoria(Queens), Morningside Height(Manhattan), Ozone park(Queens) are the most better served regions. But the majority of the out-boroughs are still under the domination of yellow taxis.

Besides, it also shows that regions near the green taxis' forbidden regions, i.e. north Manhattan, Williamsburg and Long Island City have higher index. This map shows that even green taxis can only run at restricted regions, they are still around the city's center part.

5.2 Where is Uber well served?

We use the index to evaluate the service of Uber. The darker of a region, the more Uber takes up among all kinds of trips in that region(yellow, green, uber).

Index: UberTrips/ReginalAllTrips - NYCUberTrips/NYCAllTrips

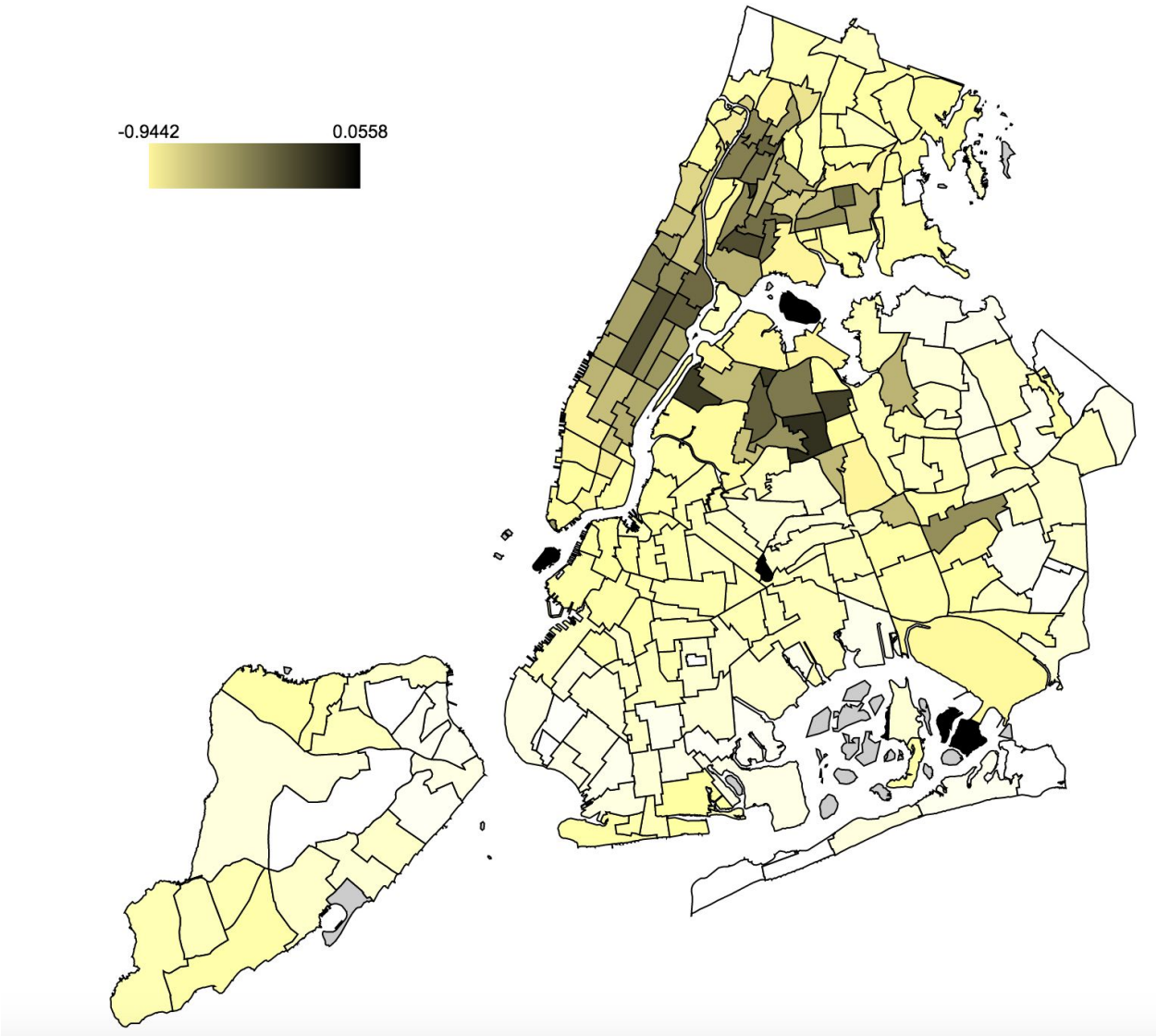

Figure 5.2 Index of Uber's service

From the diagram, we can see mos black regions focus on the upper side of Manhattan and northern queens where the subways not served very well.

5.3 Morning and Evening Peak for weekdays and weekend

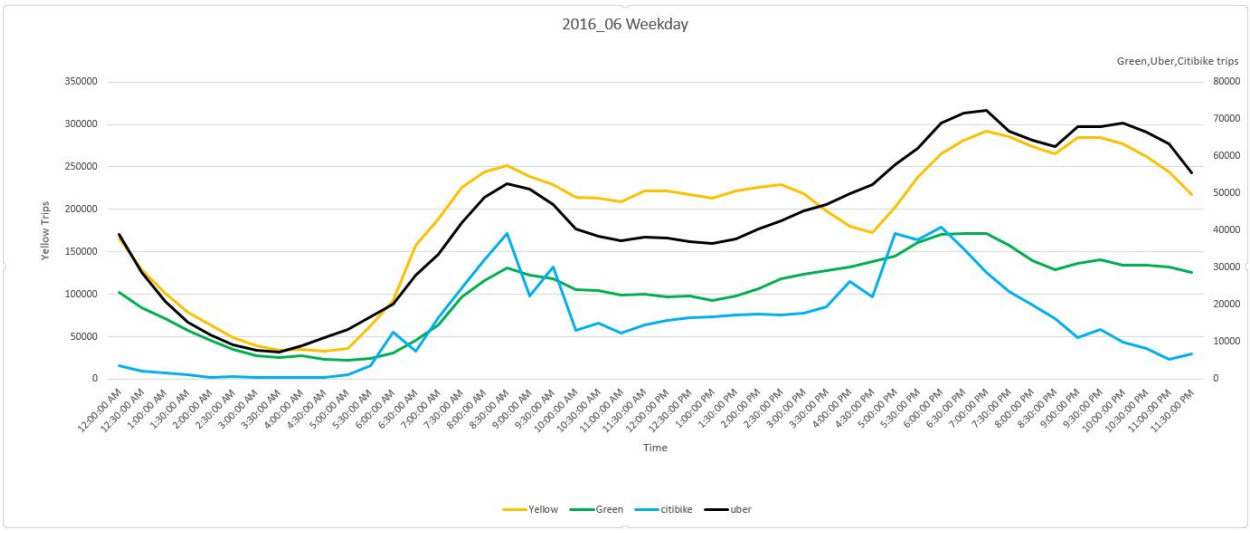

Figure 5.3 The trips of yellow taxis, green taxis, uber and citibikes in weekdays

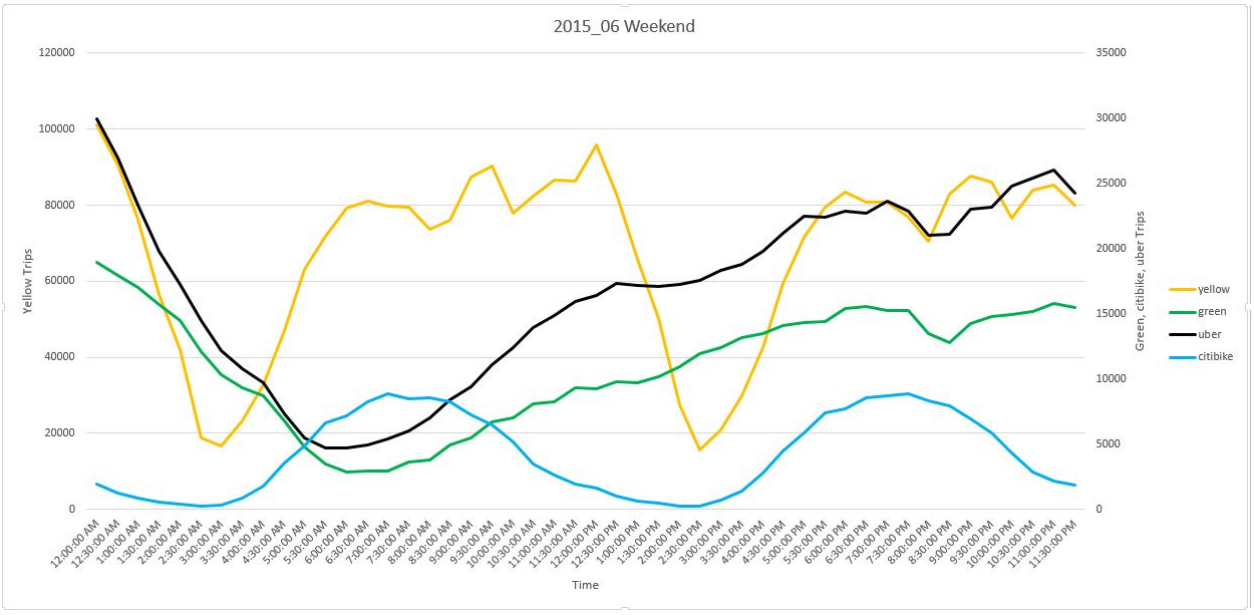

Figure 5.4 The trips of yellow taxis, green taxis, uber and citibikes in weekend

From the first diagram, we can find that the morning peak time in weekdays is obviously around 8~9am. This is the time people going out for work or school. But for evening peak, it's more complicate. Citi-bike has the earliest peak hour around 5:30~6:30pm. And for taxis and ubers, their peak is around 7pm. Later at 9pm there is another weak peak time for them. The reason may be 9pm is the time for those who dining out or working extra hours going back home.

From the second diagram, we can find that Citi-bikes are still a good sensor for peak times. But for other three, there is no obvious morning peak. For green taxis and ubers, there is a obviously accumulate growing until midnight.

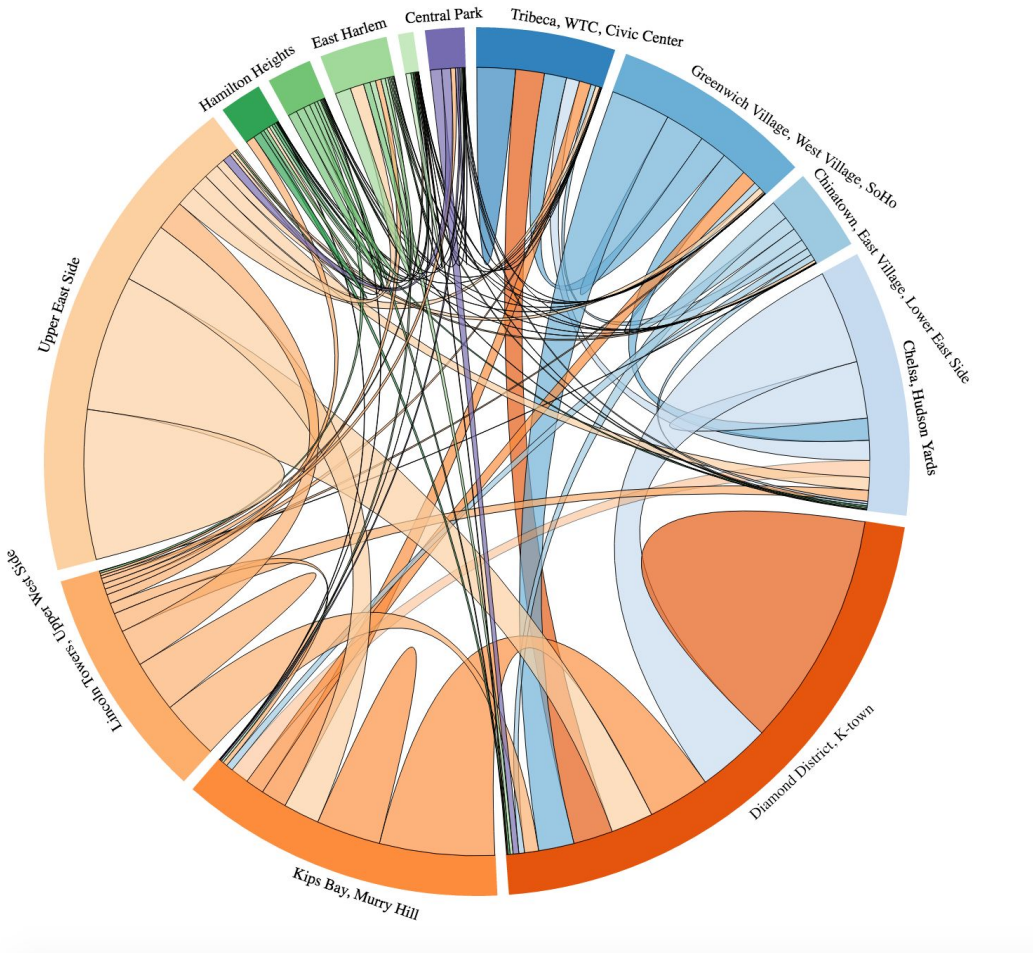

Figure5.5 Taxis Flows in Manhattan at Weekdays' Morning Peak(8~9am) of June 2015

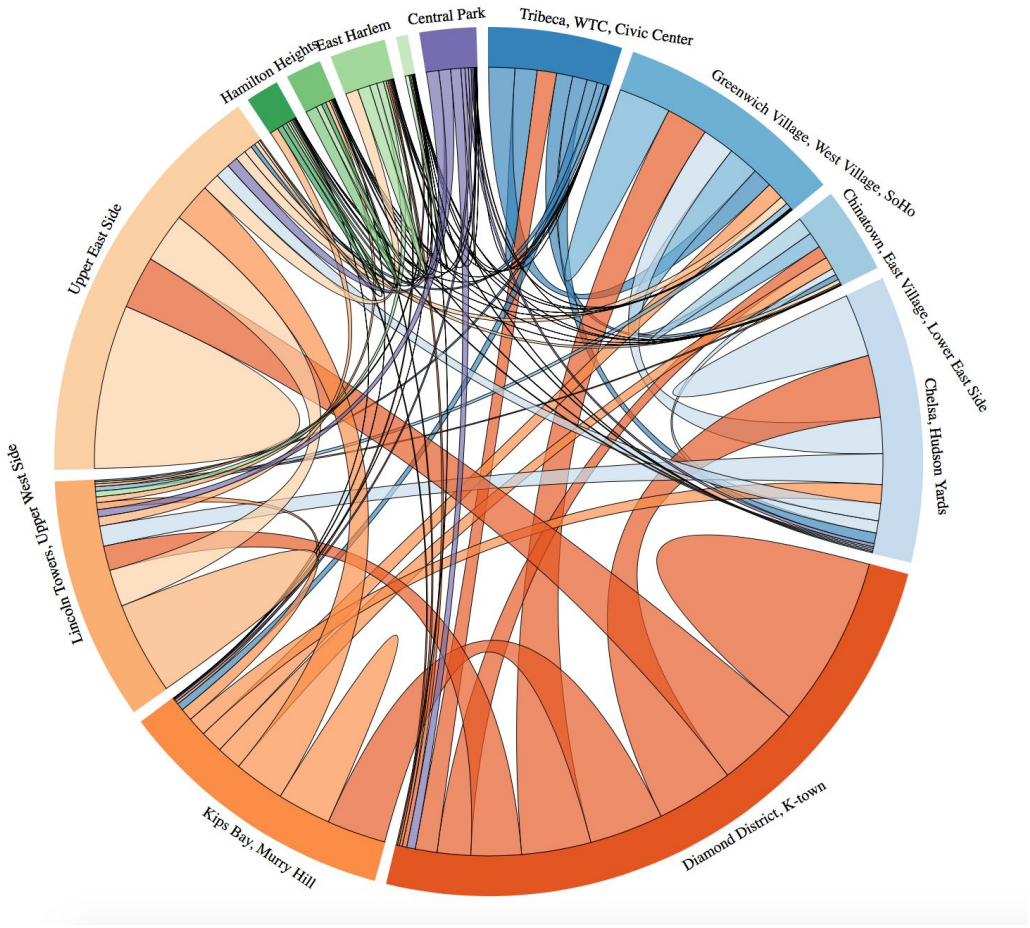

Figure5.5 Taxis Flows in Manhattan at Weekdays' Evening Peak(18~19pm) of June 2015

Above are two chord diagrams, the length of the arcs implies the pickups happened in the district. The chord starting from a arc shows the taxis flow from this district to each districts including itself. The chord diagrams show that, during the weekdays' night peak time of June 2015, the most pickups happened in the Diamond District including Midtown, K-Town and FlatIron. Also, the taxis flow between them is also the largest one among all interdistrict flows. The trips from Diamond District dominated the flows to many districts which is illustrated by more red chords here. But when it comes to morning peak, this domination disappeared.

5.4 Citi-bikes

After the data processing, we get the trips of each Citi-bike station including both trips starting and end at this station, also the coordinates of each station. For this task, we use the Voronoi diagram to illustrate the number of bikes starting or ending at each station. Voronoi diagram partition the boroughs into small regions based on the distance to the centroid i.e. each Citi-bike station.

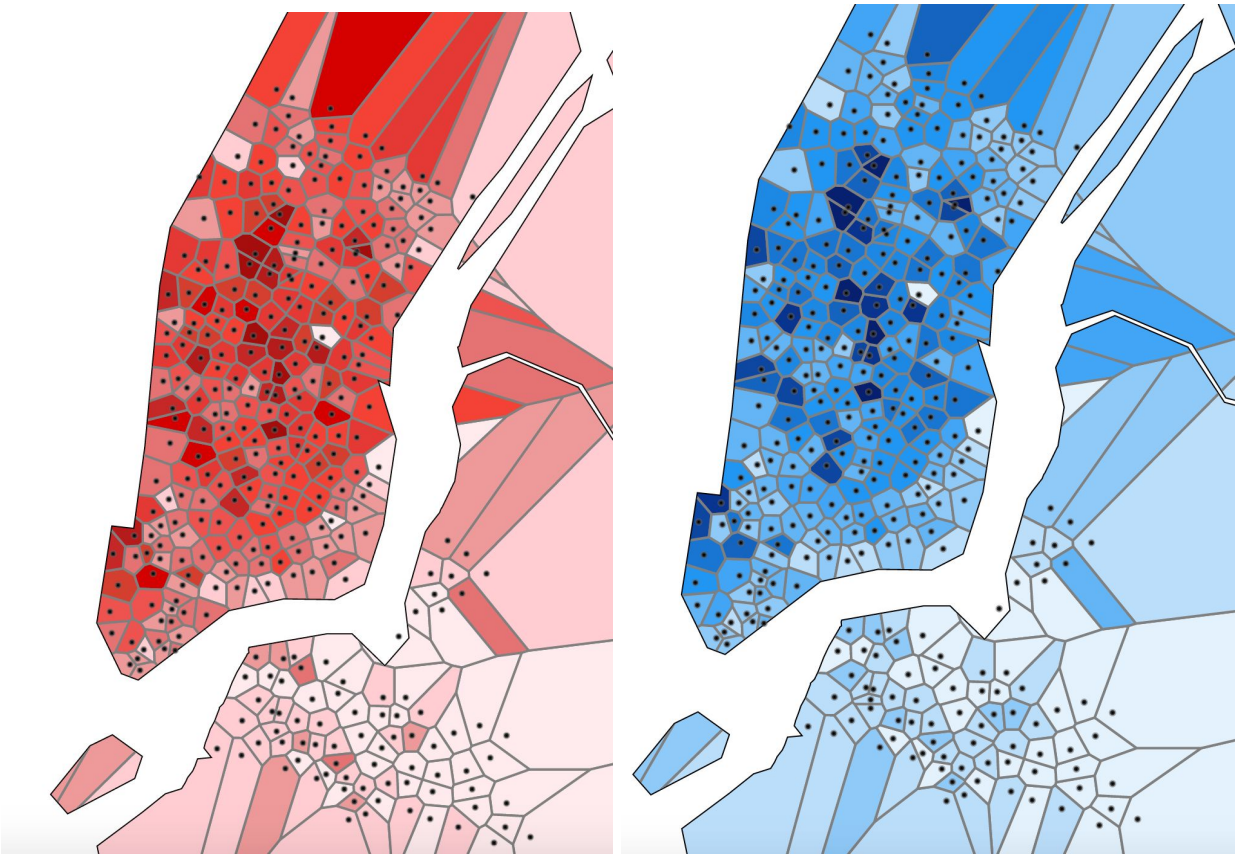

Figure5.6 The Voronoi Diagram of Citi-bike stations with number of bikes going in(left) and out(right) bikes (Jan~ June 2015)

The diagrams show that the most busy station for both biked going in and out are near Penn Station, Port Authority, Grand Central, Bryant Park and Tribeca. Not surprisingly, these places are all transportation hubs. Penn Station, Port Authority, Grand Central all have large crowd flow. Many visitor may start their NYC trips there, Citi-bike is a good way if they don't like subways. Bryant Park is the center of city, also a large hub for work commuters in the city. Tribeca is similar because of Jersey City.

Using the number of bikes going in minus number of bikes going out, we can have the diagram below. From this diagram, we can find the trend of bikes. It is interesting that, for the largest several sinks, there are also a sources accompanying with it.

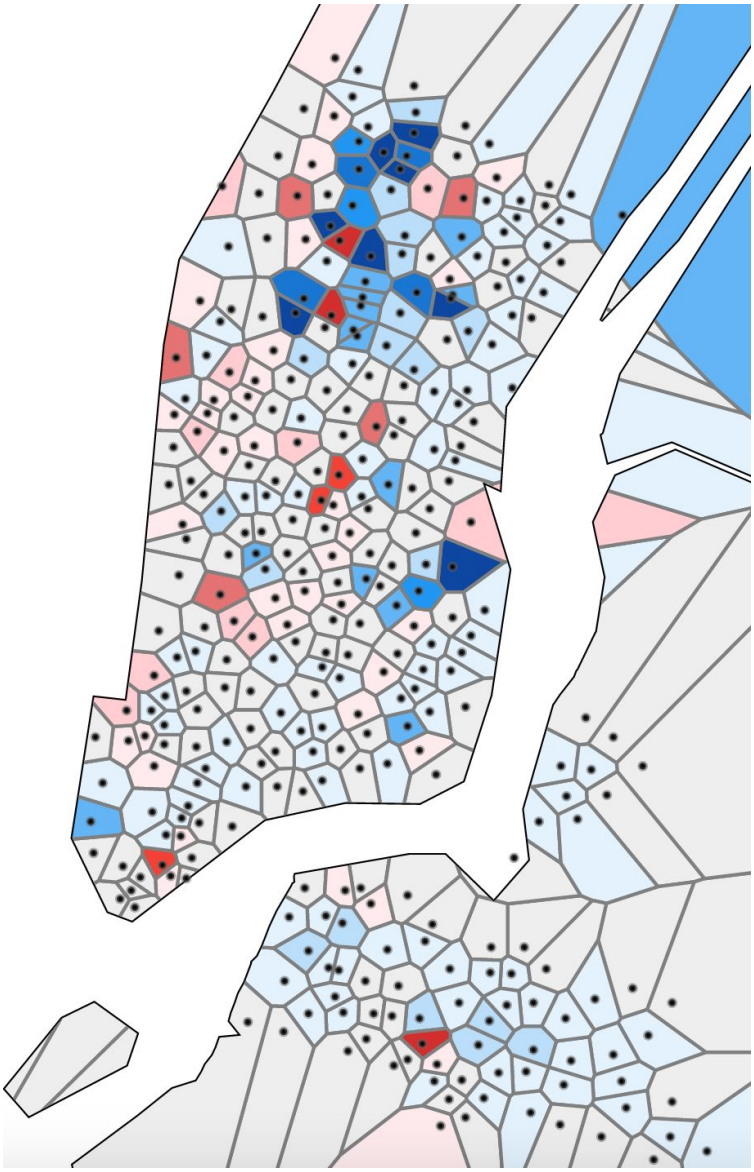

Figure5.7 The Voronoi of Citi-bike station with the number of Ending minus number of Starting(Red implies “in” is greater than “out”, blue vice versa)