Introduction

We decided to create a factbook to compare New York City green taxis and Chicago taxi operations that took place through all of 2016. In order to create this factbook we will be using big datasets containing a of bunch data that include different descriptions of multiple taxi cab rides from both cities. This data will be extracted using various methods from Spark in Python language. Some of our comparisons will include average fares earned per ride, taxi rides per hour, and frequently visited areas in each city. Some methods used in Spark include reduce, map, and reducebykey to gather results needed for these comparisons.

About the dataset

We used two datasets for taxi rides, Chicago Taxi Rides 2016 from kaggle and Green Taxi Trip Data 2016 from New York City open data website. They were both roughly two gigabytes in size. For spatial we used the Borough Boundaries dataset from New York City open data website and Boundary Community Areas from Chicago open data website. They were fairly small.

Running locally

In order to extract data from these datasets we turned our datasets into rdd objects and dataframes. RDD short for resilient distributed dataset can handle big datasets, in otherwords when processing big data an rdd does not impose any restrictions on what data can be stored into the rdd partitions. This allowed us to extract the information easier by using built in functions for rdd and dataframes. However, running the scripts on the whole datasets took extremely long and would often cause pyspark to crash on our computers. Our computers don't have enough RAM to read and write data, so the computation required to grab and filter information from a large dataset can be extremely slow. The required time to successfully compile and run a test locally on our computer took around two to three hours which was extremely inefficient. In order, to avoid this time extensive test our scripts were converted into a .py file and ran on a cluster. Test runs made the script run entirely faster taking around twenty five minutes to finish filtering and retrieving the data we needed to compare to the two cities.

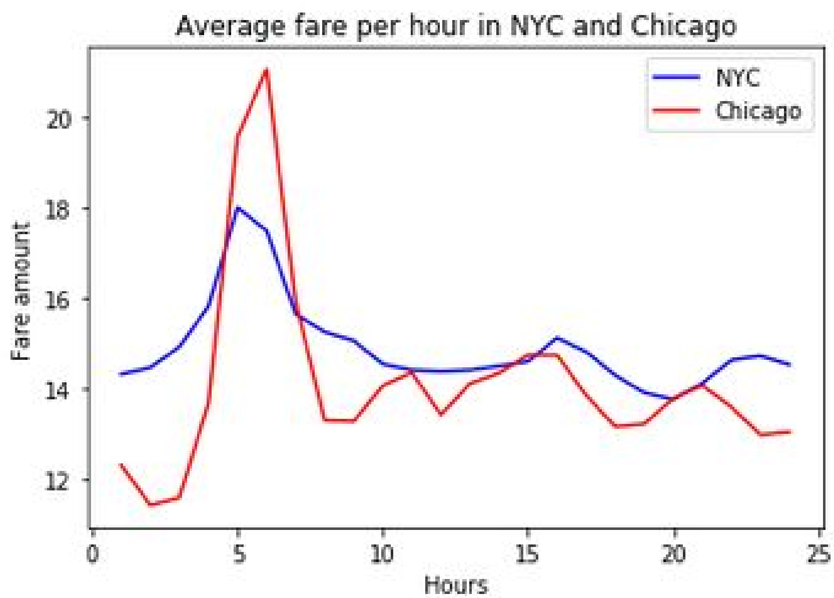
Problems (challenges)

Using loops and dictionaries to represent filtered data proved be a headache while running our .py files onto the cluster, as it would cause the server to timeout, thus causing the job to fail on the cluster. In order fix this challenge, changes were made by replacing the use of dictionaries and loops by mapping the selected row and reducing by key to create a total count for each key within an rdd.

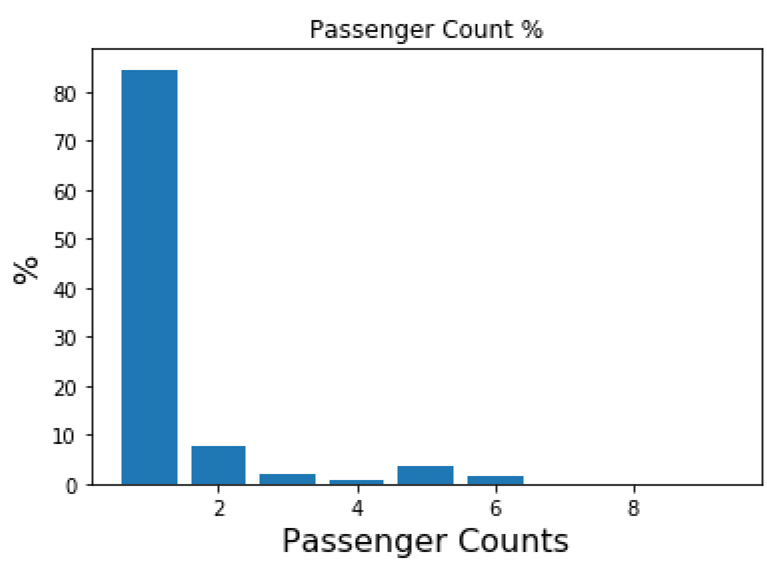
Another problem was trying to do spatial on the cluster. The job will fail saying that fiona couldn't find the shapefile, even when it exists locally and on the HDFS.

Results

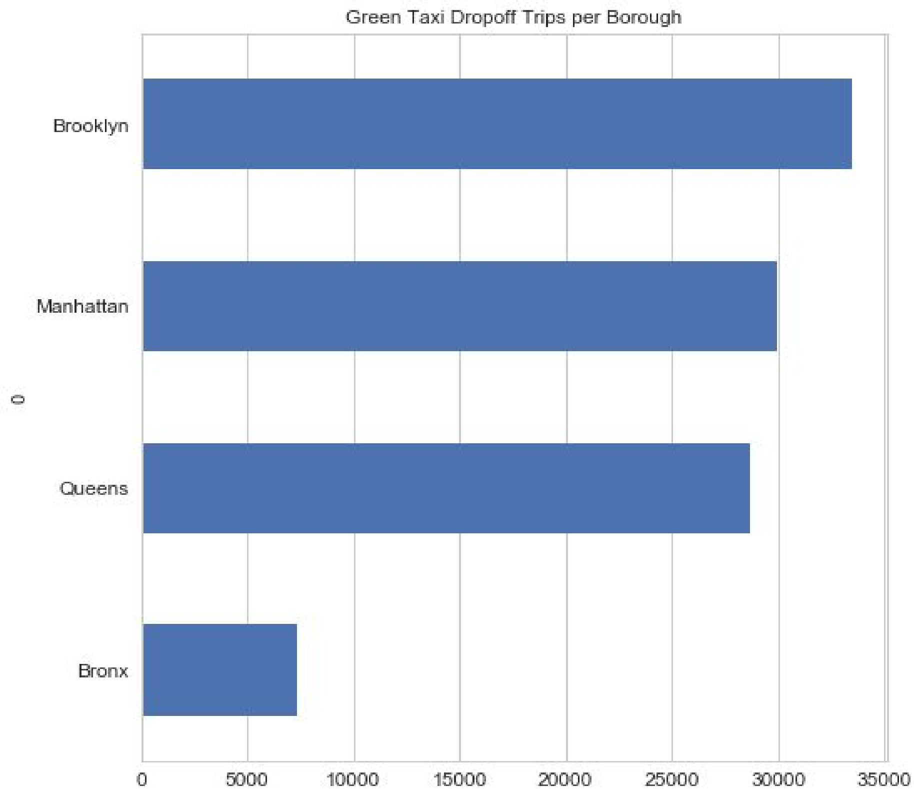
We calculated the percentages of people who paid cash or credit in each city by simply creating a filter function that extracted the row that contained payment types. This function was than used as an iterator function when we created a new rdd. Using the built-in function from spark called x.mapParitionsWithIndex(function), which outputs a list of items that we want filtered out. The function serves as an iterator function that selects only what is specified or wanted. In this case, a list containing only cash or a list containing only credit. The total count of cash and total count of credit were then divided by the sum of both to get the percentages.

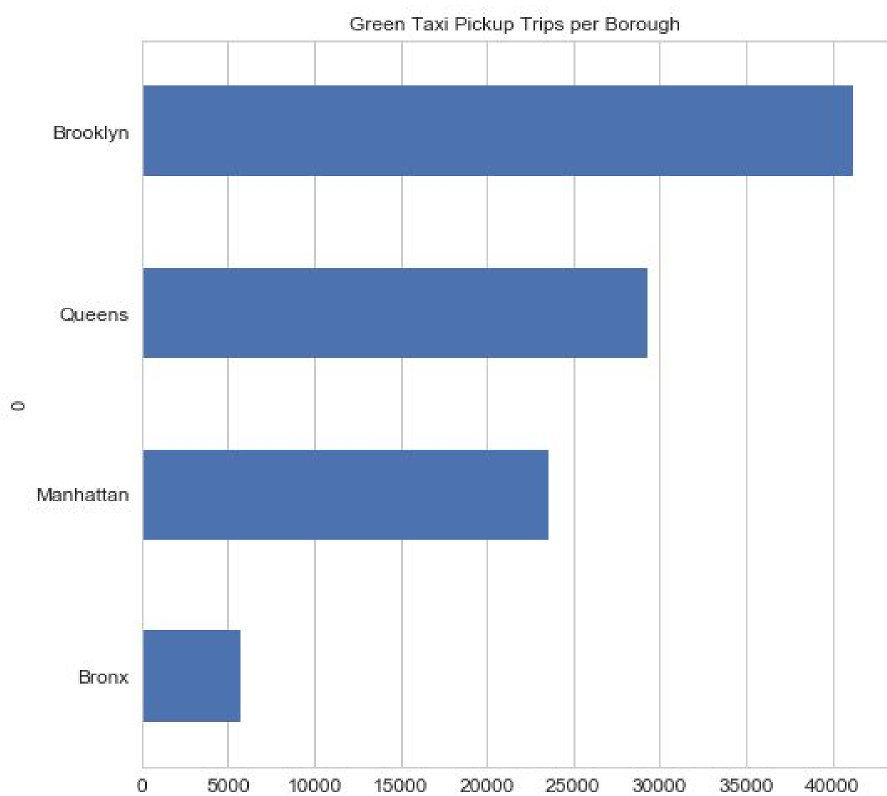


The graph above represents the average fare per ride a taxi cab driver earned during each hour of the day. The result was calculated by taking the total gross fare including tips and extras and divided that by the total amount of taxi rides that took place in each hour. In terms of code the reduceByKey function was used to calculate the gross fares per hour. There are some similarities shown here between the two cities. We see that taxis earned more during the morning hours when people are going to work and constant fare throughout the rest of the day.

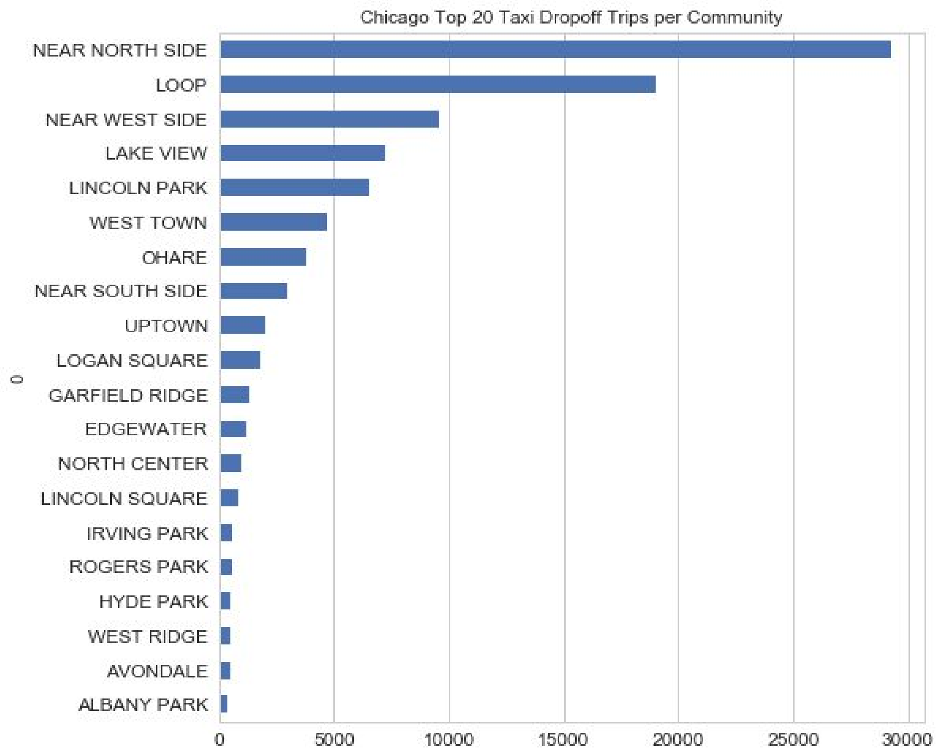


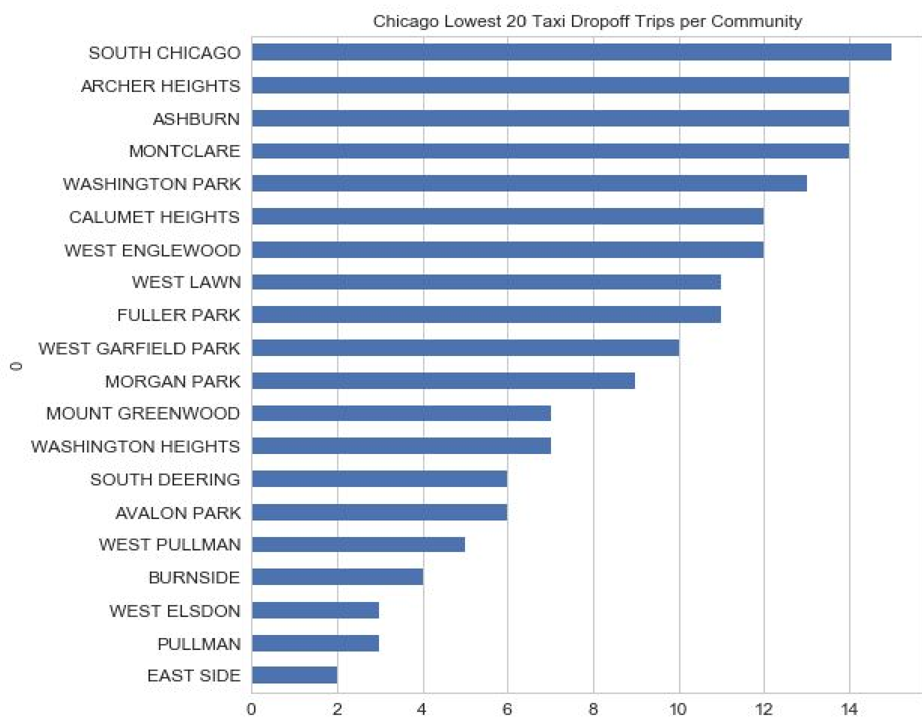
This graph is for NYC's Passenger Count %. This data was obtained by doing the count of passenger count divided by the sum of counts. We see that when green taxis are taken, there are only one passenger. Passenger counts greater than 6 is almost non existent because the only time you can have that much people is when you have children sitting on adult's lap in the rear.



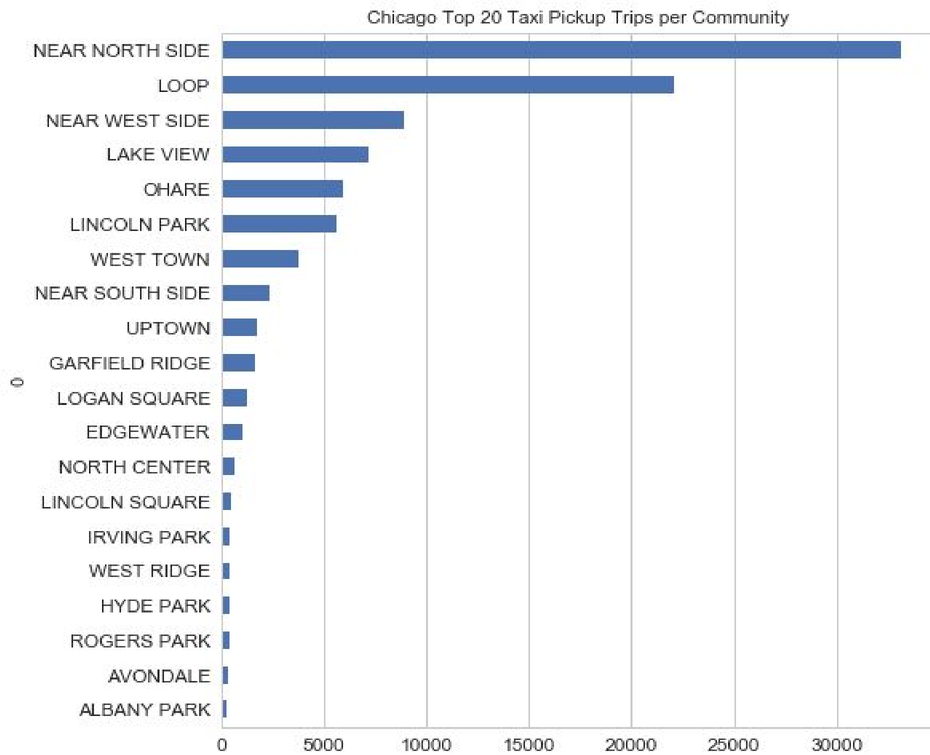


The two graphs to the left are the # of green taxi drop-offs and pickups in each borough. This is done by using geopandas and rtree to find where the pickup and drop-off locations were based on boroughs. This is based on 100,000 green taxi rides. As we can see from the graph, Brooklyn is the most popular pickup and drop-off location. The reason Manhattan isn't rank 1 is because green taxis aren't allowed to be in mid and downtown Manhattan. Green taxis in Staten Island is basically non-existent. From this graph we can also infer that people tend to go from Brooklyn to either Manhattan or Queens.

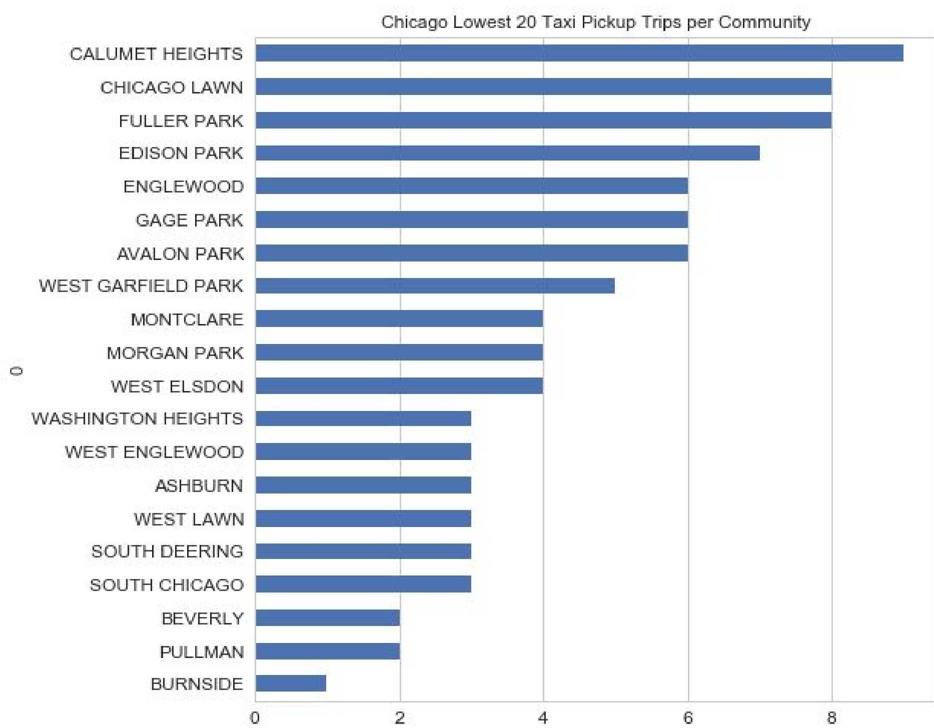




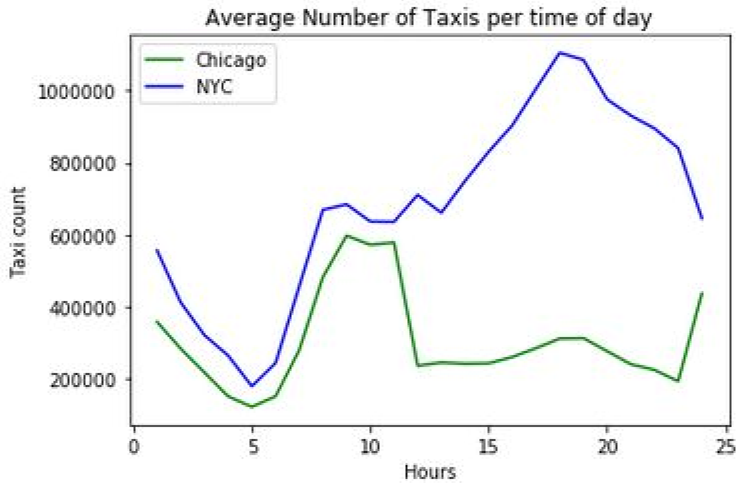
These two graphs shows the top 20 and lowest 20 drop-off trips in Chicago communities. This is also done using geopandas and rtree. We see that most of the taxi rides drop people off at Near North Side, Loop, Near West Side, Lake View, and Lincoln Park. This means that these are probably some good places to visit when you go to Chicago.The most uncommon drop-off places are East Side, Pullman, West Elsdon, Burnside, and West Pullman.



These two graphs shows the top 20 and lowest 20 pickup trips in Chicago communities. This is made using geopandas and rtree as well. We see that the top 4 pickup communities are the same as the top 4 drop-off places. Ohare managed to beat Lincoln Park in picking up people for taxi rides.

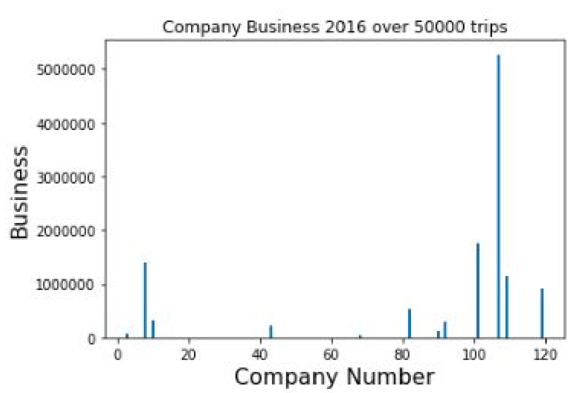


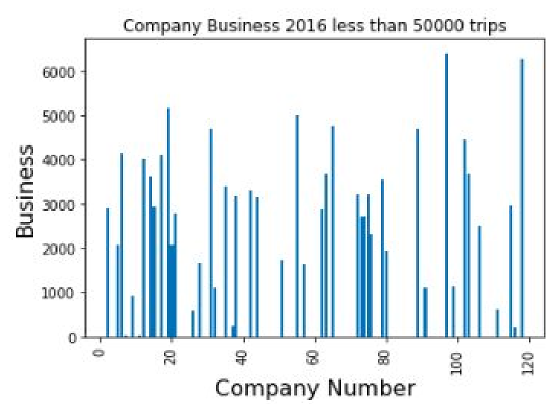
The bottom 20 pick up communities are basically the same as the bottom 20 drop-off communities, but in different order. People probably rarely see a taxi around these communities.



This graph shows a comparison between the total number of taxi rides taken during each hour of the day. The two cities start off with a similar pattern, however the pattern takes a dip down since for Chicago during the afternoon to evening hours. Code wise, in order to retrieve the data used to make this comparison the first two digits of the hour were filtered out and put into a new rdd and we simply used reduceByKey to create a total count for each key representing an hour phase throughout the day.

When retrieving the number of unique vehicle active, a filter function was used to take the row containing all taxi vehicle numbers from the Chicago taxi dataset and place into an rdd. The built-in function .distinct() was used to gather all unique vehicle numbers. The .count() function was than used to get the total number of unique vehicles in the rdd.



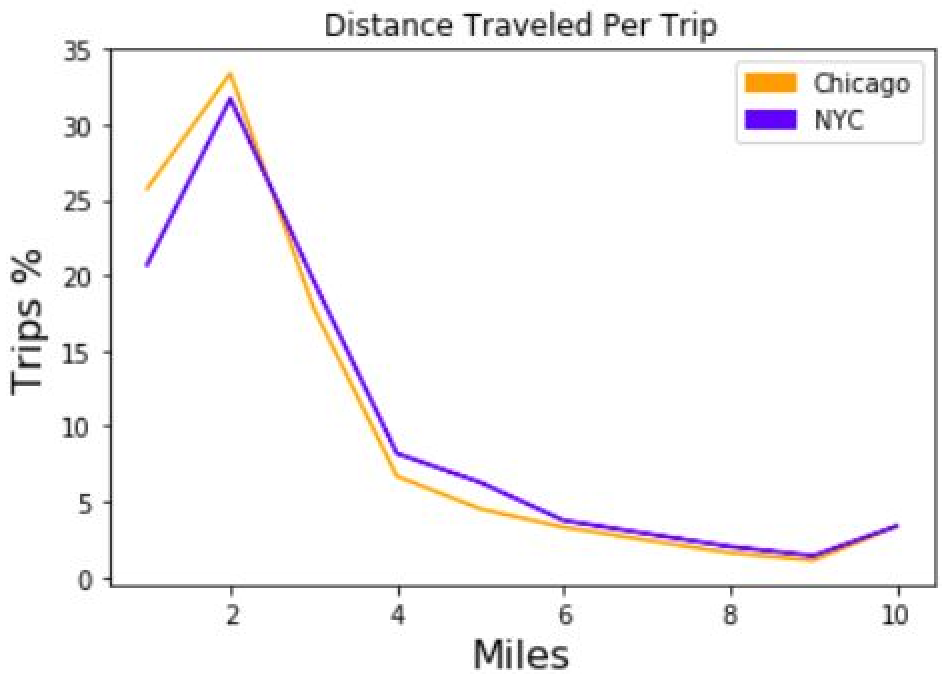


These two graphs shows the chicago company business in 2016. The left graph displays companies with over than 50 thousand trips and the right graph displays companies with less than 50 thousand trips. Our dataset recorded 119 companies. The company number is just a replacement of the actual company name. We did not want to include the actual company names in our graphs because it will be very messy.

We provided a dictionary to map what company number correspond to what company name below:

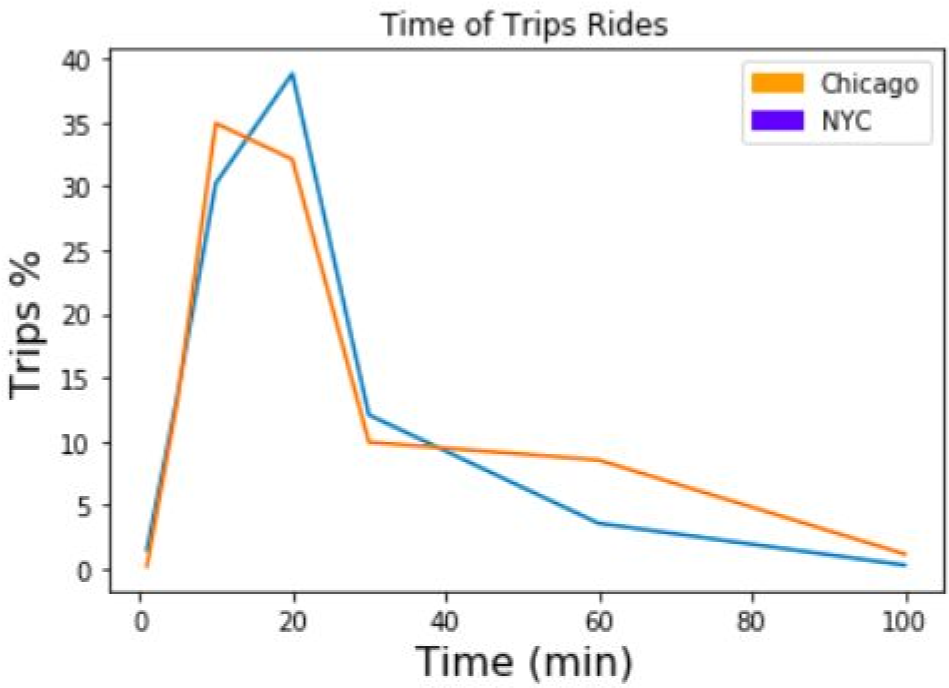
```
0: 3623-Arrington Enterprises
1: 5874 - Sergey Cab Corp.
2: 5874 - 73628 Sergey Cab Corp.
3: Chicago Medallion Management
4: 3011 - JBL Cab Inc.
5: 5997 - 65283 AW Services Inc.
6: 3152 - 97284 Crystal Abernathy
7: 4732 - Maude Lamy
8: Blue Ribbon Taxi Association Inc.
9: 2241 - 44667 Manuel Alonso
10: KOAM Taxi Association
11: Blue Cab Co
12: 6747 - Mueen Abdalla
13: 3141 - Zip Cab
14: 3094 - 24059 G.L.B. Cab Co
15: 4787 - 56058 Reny Cab Co
16: 4053 - Adwar H. Nikola
17: 5724 - 75306 KYVI Cab Inc
18: 3385 - Eman Cab
19: 0118 - 42111 Godfrey S.Awir
20: 3201 - C&D Cab Co Inc
21: 2192 - 73487 Zeymane Corp
22: 5864 - 73614 Thomas Owusu
23: Park Ridge Taxi and Livery
24: 4787 - Reny Cab Co
25: 2733 - Benny Jona
26: 3669 - 85800 Jordan Taxi Inc
27: 1408 - 89599 Donald Barnes
28: 6057 - 24657 Richard Addo
29: 5129 - Mengisti Taxi
30: 5724 - 72965 KYVI Cab Inc
31: 3011 - 66308 JBL Cab Inc.
32: 2823 - 73307 Lee Express Inc
33: 2241 - Manuel Alonso
34: 0118 - Godfray S.Awir
35: 2809 - 95474 C & D Cab Co Inc.
36: 3620 - David K. Cab Corp.
37: 6488 - 83287 Zuha Taxi
38: 3591 - 63480 Chuks Cab
39: 0694 - Chinesco Trans Inc
40: 6743 - Luhak Corp
41: 3094 - G.L.B. Cab Co
42: 2733 - 74600 Benny Jona
43: Chicago Medallion Leasing INC
44: 5074 - 54002 Ahzmi Inc
45: 6488 - Zuha Taxi
46: 5724 - KYVI Cab Inc
47: 3253 - 91138 Gaither Cab Co.
48: 3556 - RC Andrews Cab
49: 1247 - Daniel Ayertey
50: 3253 - Gaither Cab Co.
51: Patriot Trans Inc
52: 2809 - 95474 C&D Cab Co Inc.
53: 4615 - Tyrone Henderson
54: 3897 - Ilie Malec
55: 3141 - 87803 Zip Cab
56: 2092 - Sbeih company
57: 3620 - 52292 David K. Cab Corp.
58: 3385 - 23210 Eman Cab
59: 5776 - Mekonen Cab Company
60: 5074 - Ahzmi Inc
61: 3152 - Crystal Abernathy
62: 3201 - CD Cab Co Inc
63: 585 - 88805 Valley Cab Co
64: 3201 - CID Cab Co Inc
65: 1085 - 72312 N and W Cab Co
66: 5062 - Sam Mestas
67: 3591- Chuk's Cab
68: T.A.S. - Payment Only
69: 5437 - Great American Cab Co
70: 4623 - Jay Kim
71: Chicago Elite Cab Corp.
72: 0694 - 59280 Chinesco Trans Inc
73: 4615 - 83503 Tyrone Henderson
74: 3623 - 72222 Arrington Enterprises
75: 4623 - 27290 Jay Kim
76: 3319 - CD Cab Co
77: 585 - Valley Cab Co
78: 1085 - N and W Cab Co
79: 5006 - 39261 Salifu Bawa
80: 4053 - 40193 Adwar H. Nikola
81: DTA Test
82: Northwest Management LLC
83: 3385 - 23210 Eman Cab
84: 3319 - C&D Cab Company
85: American United Cab Association
86: 5864 - Thomas Owusu
87: C & D Cab Co Inc
88: 0118 - Godfrey S.Awir
89: 6574 - Babylon Express Inc.
90: Suburban Dispatch LLC
91: 5062 - 34841 Sam Mestas
92: Top Cab Affiliation
93: 2823 - Seung Lee
94: 5006 - Salifu Bawa
95: 5997 - AW Services Inc.
96: 5129 - 98755 Mengisti Taxi
97: 6743 - 78771 Luhak Corp
98: 3201 - C & D Cab Co Inc
99: 3556 - 36214 RC Andrews Cab
100: 2767 - Sayed M Badri
101: Dispatch Taxi Affiliation
102: 2092 - 61288 Sbeih company
103: 4197 - 41842 Royal Star
104: 2192 - Zeymane Corp
105: 3591- 63480 Chuk's Cab
106: 1247 - 72807 Daniel Ayertey
107: Taxi Affiliation Services
108: 3385 - Eman Cab
109: Choice Taxi Association
110: 3669 - Jordan Taxi Inc
111: 2823 - 73307 Seung Lee
112: 3897 - 57856 Ilie Malec
113: 3591 - 63480 Chuk's Cab
114: 1408 - Donald Barnes
115: 6742 - 83735 Tasha ride inc
116: 2241 - 44667 - Felman Corp Manuel Alonso
117: 4197 - Royal Star
118: 5129 - 87128
119: Chicago Elite Cab Corp. (Chicago Carriag
```

We got our data by just adding the count of each company number using map and reducebykey to get a list of [company number, count_of_trips] for each company. Using that, we were able to plot our bar graph.

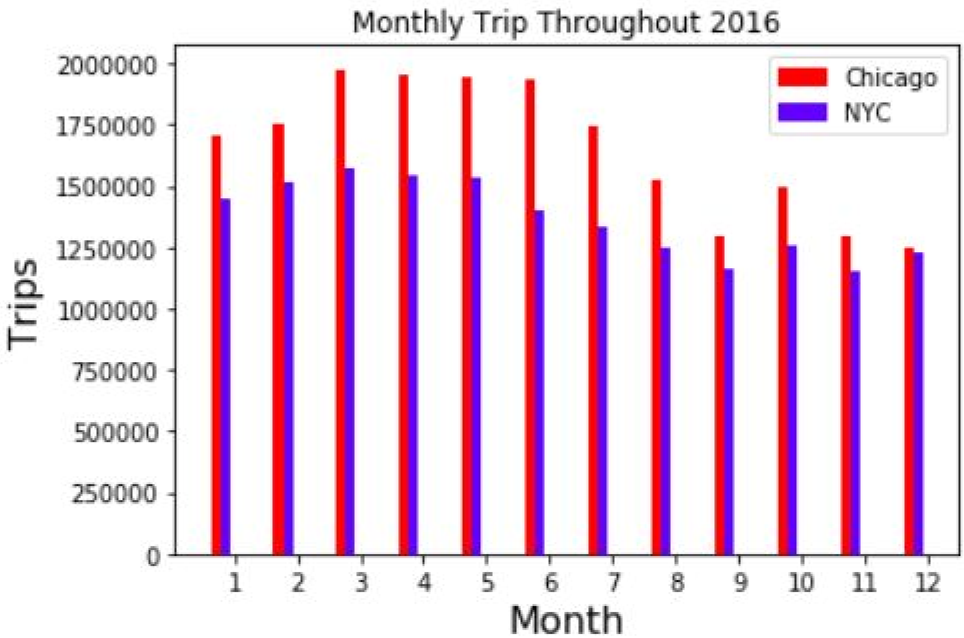


This graph shows the distance traveled per trip for both NYC and chicago. As you can see, they are very similar. Most trips are about one to two miles.

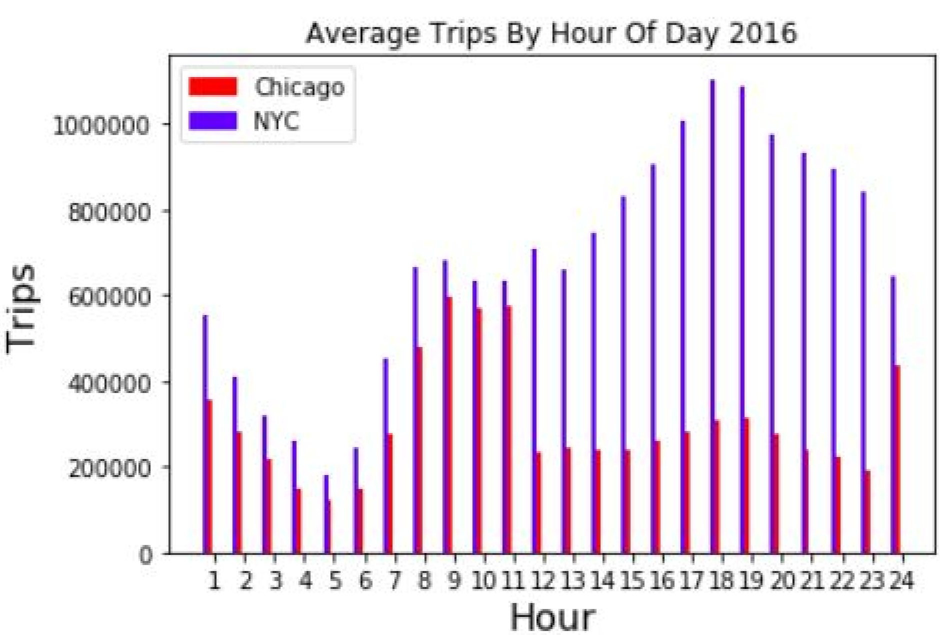
Using streaming, we got the count of each mile from 1 mile all the way to 10 miles. The 10 miles include everything over 10 miles as well. Using map and reducebykey, we were able to get a list of [mile. Count] for each trip. We then converted the trips into percentage to see how many percentage of the trip belongs in which mile bracket. Using these data, we were able to plot our line chart.



This is very similar to the previous chart above, but instead, this chart shows the time of the taxi rides by percentage. As you can see, majority of the rides are around the 20 minute mark. To get this data, we used streaming to retrieve data from both the nyc and chicago taxi csv file. We then used map and reducebykey to get a list of [time, count]. Using the data we acquired, we were able to plot our graph.



This graph compares Chicago and NYC monthly trips throughout 2016. As you can see, the two states are very similar in term of trend. We used streaming to get every row from both the Chicago and NYC csv dataset as each individual row represents an unique trip. We then use map and reducebykey to get a list of [month, count] which helped create the graph.



This graph shows the average trip by the hour of day. From the graph, it shows that the most popular time is during 9,10 and 11th hour of the day, and from the 12th hour to 23th hour, the average amount of trip decreases dramatically. We were able to retrieve our data by using streaming techniques on both the Chicago and NYC csv datasets. We then used map and reducebykey to get a list of [hour, count], which helped us create our graph.

The charts: Company business 2016, Distance traveled Per Trip, Time of Trip Rides, Monthly Trip Throughout 2016 and Average of Trip by Hour of Day 2016, all used the scripts chi.py and nyc.py to extract the data needed. “Chi.py” took 19 minutes and 9 seconds to run on the cluster and “nyc.py” took 42 minutes and 45 seconds to run on the cluster. The bd_project.py file took 27 minutes and 50 seconds.