Team DAAL CS398 Final Report:

Members: Alexandre Geubelle, Lawson Probasco, David Raskin, Anmol Nigam

New York City is known for its transportation problems. Our project's goal was to explore common modes of transportation in this large city. We tried to understand how millions of people manage to move throughout the city everyday by using graphs, maps and machine learning models. We transformed data pertaining to parking tickets, taxis and bikes in hopes to better understand the dynamics of one of the world's largest cities.

What dataset(s) did you use?

Our project was based on a visual analysis of a group of datasets that all relate to transportation in New York City. We ended up focusing on four datasets, namely Street Centerline, Parking Violations Dataset, City Bike Dataset, and TLC Trip Record Data. The street centerline dataset provided us with information about 100,000+ street segments, their location, and addresses within that segment. The parking violations dataset was a set of 4 csv files, each with 10M+ rows, that supplied all parking tickets from 2014-2018. The City Bike dataset was expansive and contained rows upon rows of starting and ending locations of bike trips in NYC. Finally, TLC Trip Recode Data was extensive information (pickups, drop offs, fares, etc.) on the hundreds of millions of taxi trips in NYC.

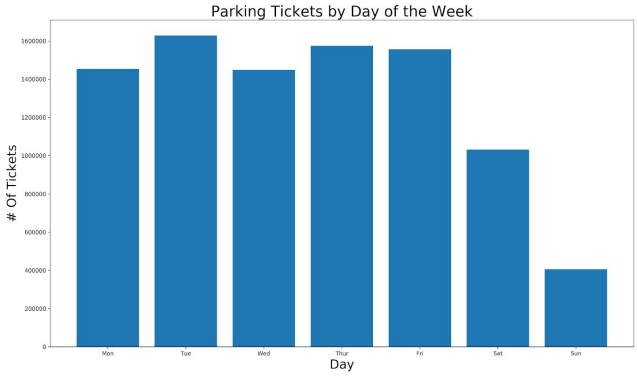
What frameworks did you use?

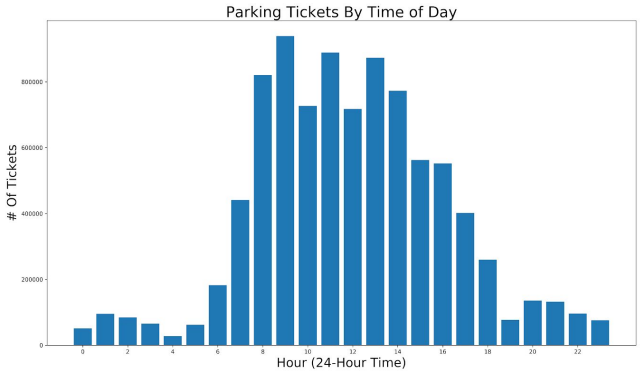
We performed computation on both small samples, as well as the entire dataset using Spark, SparkSQL, and SparkML. The core of our analysis involved first sampling data in order to run small spark jobs to test code, gathering some analytics; be it aggregate statistics, co-ordinate points, or features and labels, and then finally we would run our code on the larger dataset (after ensuring all the data was clean and easy to work with, typically done by filtering out bad data), and finally visualize whatever results we gained from our analysis.

What types of knowledge did you extract from the dataset?

In the case of the parking ticket dataset, we were provided with a large number of columns with miscellaneous information about the type of ticket (meter time expired, parking in a restricted area, etc.) as well as some information about the vehicle it was written to. The first information that we were able to extract from the dataset pertained to what we could learn about the work cycle of ticketing employees. We had predicted that tickets would be issued mostly on the weekdays and we also predicted that during the day, we would see a bell curve during the hours of 9am-5pm. The dataset provided us with two important columns that allowed us to determine the date and time of each ticket. These columns were "Issue Date" (ex. 06/24/2017)

and “Violation Time” (ex. 0625A). A couple of functions were applied to these datasets that allowed us to transform the date into a python datetime variable and the violation time was transformed into military time. From this point, we were able to use spark to condense our giant dataset into two simple visualizations that would show us which days of the week and what times of day the parking officers were most active. These graphs are shown below, and we were able to check our hypothesis. We were even able to see small dips during normal lunch times.





The majority of what we wanted to learn about parking in NYC was based around location. We felt that visualizations on a map would be the best way to quickly condense the millions of rows into a quickly understandable format, one in which we could quickly make deductions about New York City car transportation. Some of the observations we hypothesized that we could make included finding high traffic areas throughout all of NYC and how those high traffic areas changed over time.

We knew that map visualizations would be important for our project of extracting as much as possible from this dataset, but unfortunately there were no latitude or longitude columns. Thus, we combined this dataset with the Centerline New York City Street dataset in order to map our tickets to a specific location. The Centerline dataset is relatively small with only 100,000+ rows, but it had very useful columns including latitude, longitude, geometry of the road, street name, and a range of house numbers. Since the parking ticket dataset also provided us with a house number/street name and we were using spark dataframes, just a simple join (in which we joined based each parking ticket to a street segment if the street match and the house number was within the range of that segment) was required. We had now extracted a new feature allowing us map parking tickets in New York City.

An important consequence of this join was that once we had grouped by street segment and counted the number of tickets, we had effectively reduced tens of millions of data points to just over 20,000 street segments. This means that we had successfully partitioned the majority of the computational effort to the cloud and now we could quickly visualize our map of New York city where each street had a value associated with the number of tickets given at that point. Some of these visualizations are shown below.

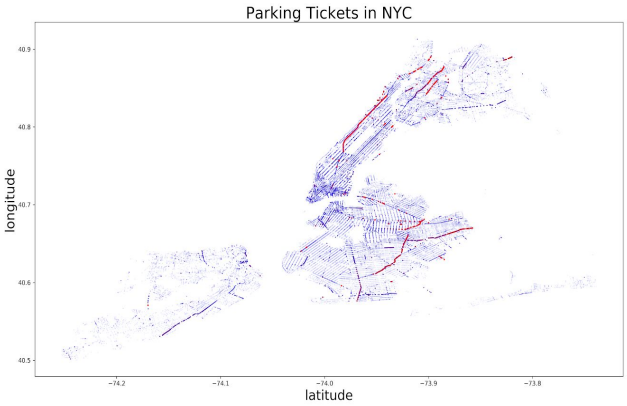


Figure 1

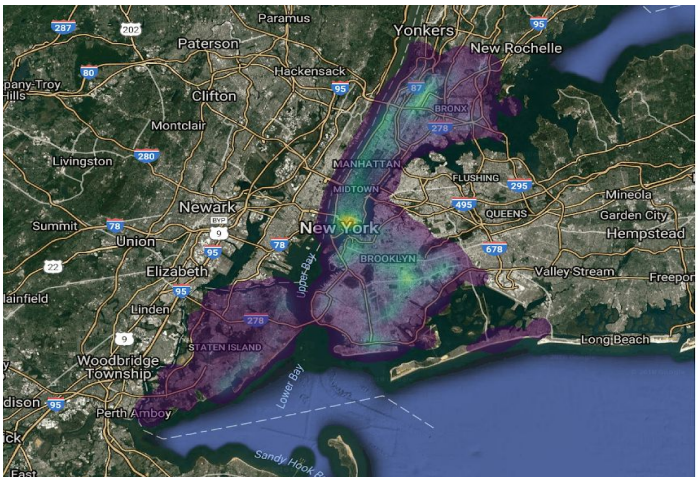


Figure 2

Figure 1 was one of the first visualizations we created. It is a simple scatter plot of the latitude and longitude of each street segment where the color of each point is based on the number of tickets given at that location. We used this simple map to determine which streets had the highest concentration of tickets and this also helped give us an idea of the busiest sections of New York City in terms of car traffic. Figure 2 came later and we felt it was very important as it helped to cleanly display the sections of New York that had the most tickets. Unlike Figure 1, this does not depend on the size of each street segment and gives us the area with most car traffic and tickets rather than the specific streets.

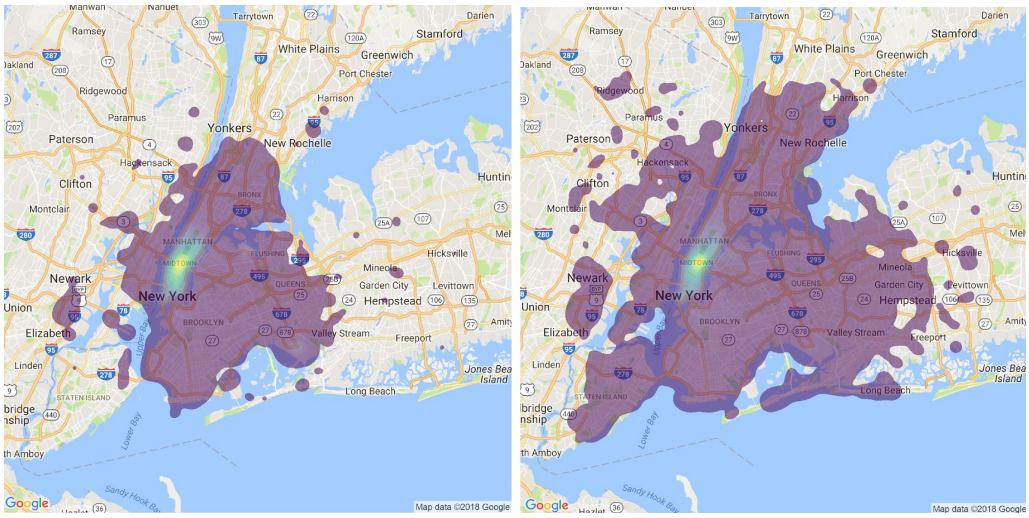
The next data we worked to extract was more geared towards an everyday application. We directed our study towards meter expiration tickets which provided us with many different avenues to applications. Our main idea was to provide smaller scale, specific visualizations that showed parking meters near a given location and also displayed the number of times a ticket had been issued to a vehicle at that meter. We felt that this is an especially applicable visualization in that it could be used by a variety of different users. For example someone could use it to try and find the meters near them that are least likely to be ticketed if they choose not to pay. On the other hand, the officials that are giving out tickets could search along their routes to see if there are meters on other streets that might not get checked very often.

Again we had the issue of not being provided a definitive latitude and longitude value for each parking ticket, but we did have a meter number. We solved our problem by finding a dataset of all parking meters in the NYC area. This dataset provided us with 15,000+ meters with information about their latitude, longitude and meter number. We were able to, again using spark, quickly join these datasets and count the number of tickets at each meter. We also used spark and the cluster to quickly filter by proximity to the central location, so that visualization with the output data would be as simple and quick as possible. An example visualization is shown below where the star is the central location and red dots are meters which have numerous tickets associated with them.

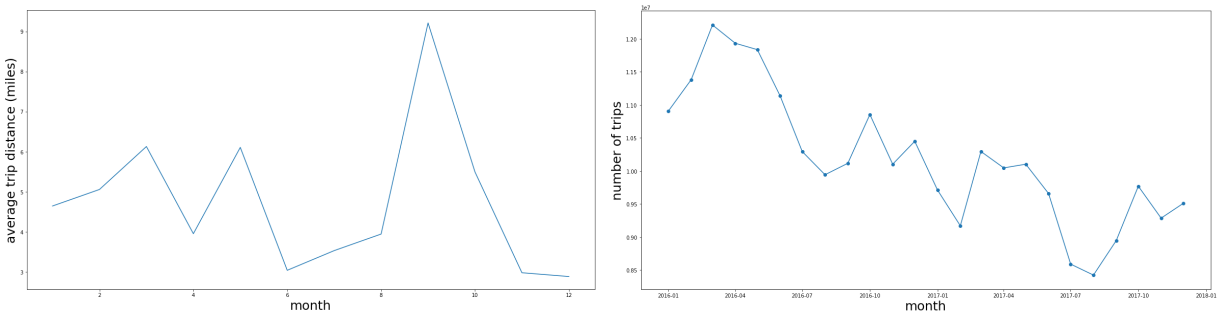
Figure 3

In the figure above, we can quickly see the roads in which a user might want to check for meters and which are less likely to be checked by parking officials. We can also see the streets that are more often checked, and it is not surprising that the area meters around the hospital are busy and that many tickets were written in that area.

For the taxi data, we were mostly interested in analyzing both aggregate trends over time (from 2016 - 2017), as well as visualizing the general flow of taxis in New York City. For the former task, we began our analysis by first filtering out bad data - this was a necessary step, especially for some of the more recent 2017 data, due to a lot of inconsistencies in the way things were filled out (there we some particular discrepancies between how 2016 and 2017 labeled location data, for example). After this initial cleaning step, we gathered aggregates across the dataset; trips per month, average trips per day, average trip distance per day, and average trip distance per month. We also collected all the location data from a sample of the dataset and plotted them on a heatmap, which used filtered location data based on time of day, as well as pickups vs. drop offs.



Using this data, we were able to clearly visualize where the most frequent taxi pickups and drop offs occur, some interesting demographics of the yellow taxi such as the bounds of its pickups and drop offs, and also the decline in taxis over the past two years, largely attributed to an increase in the use of uber. The two above figures show a very clear visualization of the very frequent midtown Manhattan area drop offs and pickups, and you can quite easily see the overall borough coverage of the yellow taxi by pickups. On the right, the drop offs heat mapping shows a further range of coverage, due to the yellow taxi servicing in-borough, but dropping off out of borough.



The figure on the left depict changes of average trip distance over the course of a year (on a per monthly basis), and on the right we see another similar month-to-month type graph, but this one clearly shows the decline of uber as stated above. The spikes and dips on the left seem to be somewhat strange, and we attribute some of it to the changing of the seasons throughout the year. While this might explain a portion of this, the abnormal data (especially when compared to the 2017 average trip distance) seems significant in its volatility, but we couldn't exactly pinpoint the sole reason or reasons for this.

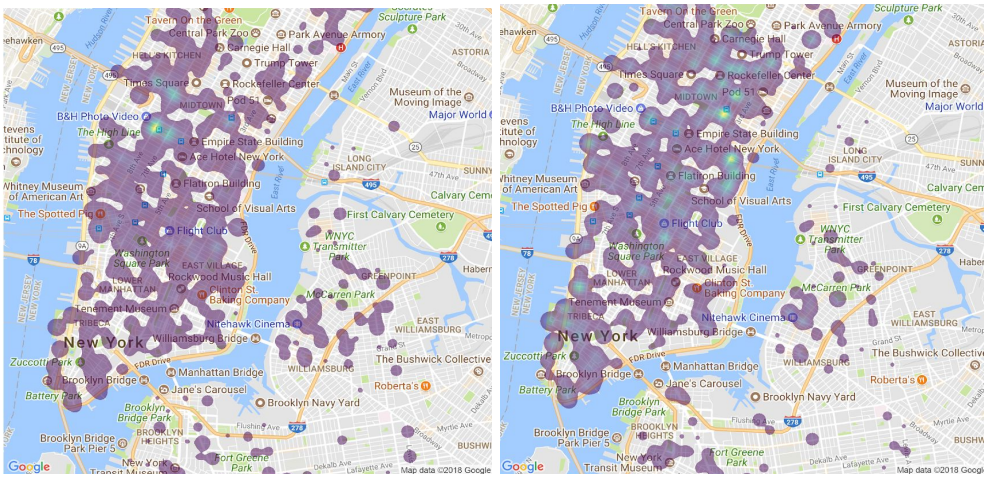
The city bike data is similar to the taxi data set, except with some additional features included within the dataset, that potentially could be insightful. Among the most influential columns we use in our analysis are the columns with information pertaining to the trip duration, start & end time, start & end stations. Other potentially interesting data within the city bike data was information pertaining to the gender and age of the individual using the bike, while certain data points also contained data about the location of bike throughout the day, which may data gathered from certain bicycles that are tracked throughout the day.

The first information we extract is the locations of bike pickups and dropoffs as this information allows us to make comparisons with both the parking data and taxi data sets, as well as motivates some unique visualizations about the distributions of bikes throughout New York City which we predict will be heavily concentrated around subway stations and the commercial areas of New York in the morning (7am - 12am), while in the evening areas of nightlife and tourism will have higher concentrations of bike drop offs.

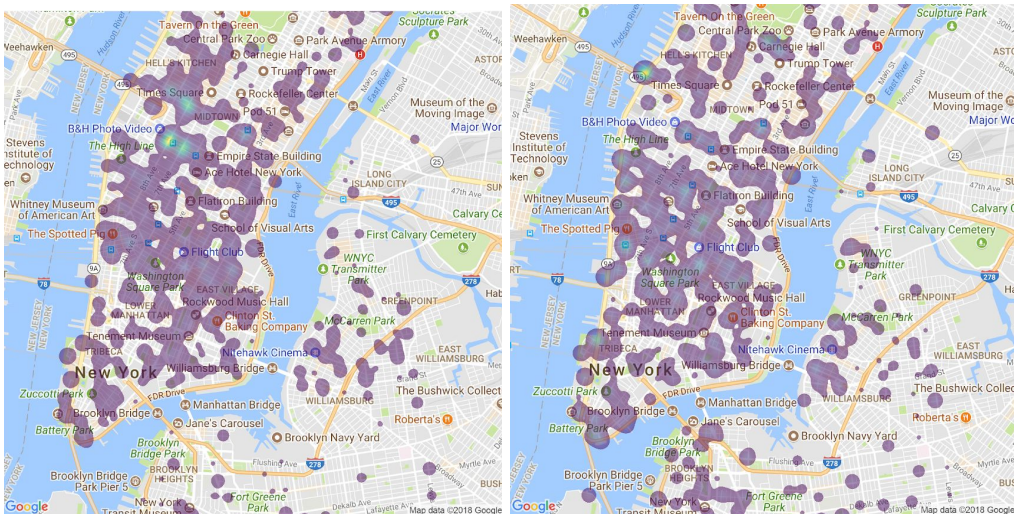
The first set is to combine the datasets that we have, since our data is within compressed zip files for every month of 2013 - 2017, we aggregate these months into a single large csv with all the data points and then separate it into different files based of certain features we want to analyze such as year, season and time of day. In addition, we decided to remove data points that contain invalid values as since we have several million rows losing a couple thousand does not significantly change our analysis. Finally we also remove some of the features in the dataset that are not needed for our analysis. These include; trip duration, bikeid,

usertype, birth year and gender. These columns do not allow us to make comparisons across datasets, hence they are superfluous and make our file size far larger than it needs to be.

To do the preceding steps we use our good friend Spark except for the initial data aggregation because we were unable to load the individual files into HDFS properly so we did it locally to perform the previous aggregations as well as filters on the data to reduce the overall size of the dataset. The dataset does provide with the time at which the bike was picked up and dropped off, but does not give an exact longitude and latitude for each station, so we make use of the Centerline dataset in order to extrapolate the approximate latitudes and longitudes of the stations for use from the station id and street locations. After extrapolating these values we transform them to make the uniform for ease of use with our visualization tools, using proper datetime variables, conversions from GeoJSON and decimal formats. After doing the following tasks we visualize our interpretations for the data analyzed, we start with drop offs and pickups by time of day. We use the same map visualizations that we use for the taxi and parking dataset, but in order to make sure that we can actually see the important features on the map, we decided to randomly select ~10000 datapoints of morning and evening pickups to make sure that our entire map is not incomprehensible (and so that the visualizations don't take an inordinate amount of time to run).

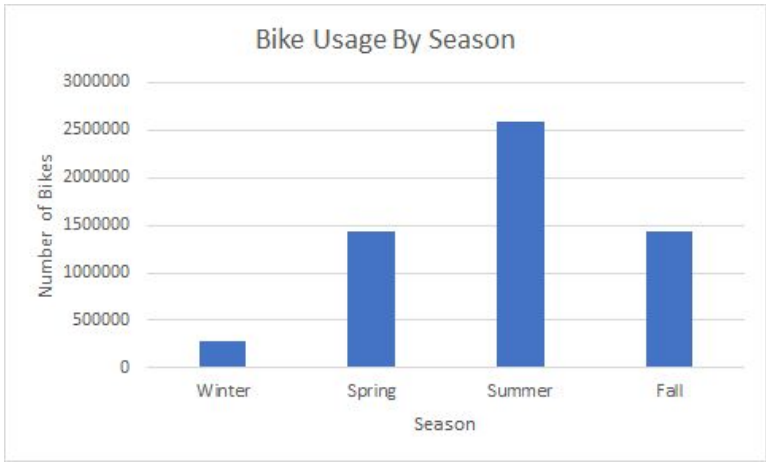


On the left we see the taxi pickups before 9am and on the right the drop offs before 9am, we notice that the drop off spread of bikes is far less widespread than the pickups, which makes sense as people tend to congregate towards the center of the city for work, but their residences are more widespread.



Here on the left we have the morning dispersal of bike pickups and on the right we have the evening dispersal of drop offs. We notice some key points here: first that the evening spread is more dispersed then the morning which makes sense as people return home, they are farther out from the inner city. In addition, we notice that on the top left corner of the evening drop offs near Times Square there is a high concentration, akin to our prediction about their being late night activity near popular places like Times Square, with the location specified being one of the closest stations to Times Square.

Next we analyzed the relationship between season of the year and the usage of the city bike system, we would expect that in the summer months (May - August) bike usage will be at its peak, while winter months (November - March) will have less usage, and the months of March and October will be somewhere in between, as seen in the graph below.



We confirm our suspicions that Summer is the most common month for riding bikes, and spring and fall are slightly behind and winter occupies a miniscule proportion of total bike rentals.

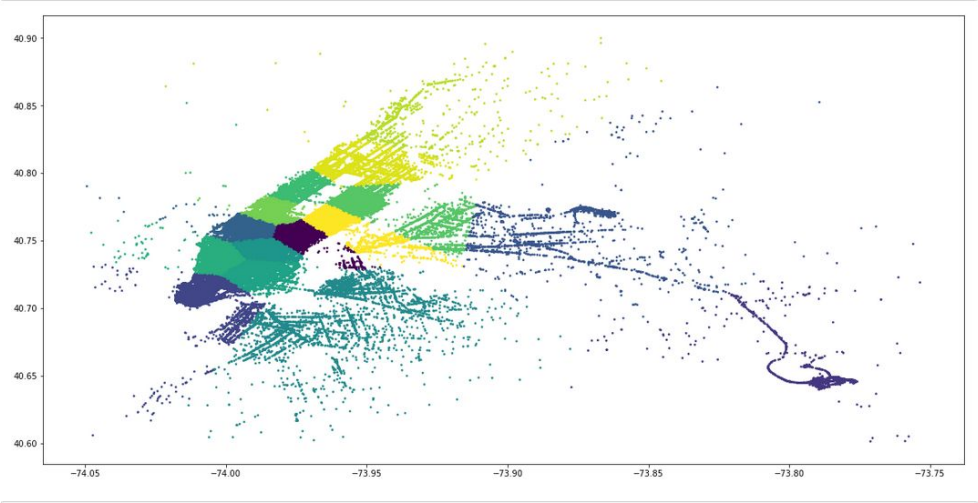
Finally although we did make visualizations for the bike pickups by year as well as plotting individual bike routes that individuals take during the course of their day, these visualizations do not lead to any meaningful conclusions, and look near random amalgamations of data points, without any discernible patterns, hence for the sake of brevity they are excluded from this paper, but are included within the group repository.

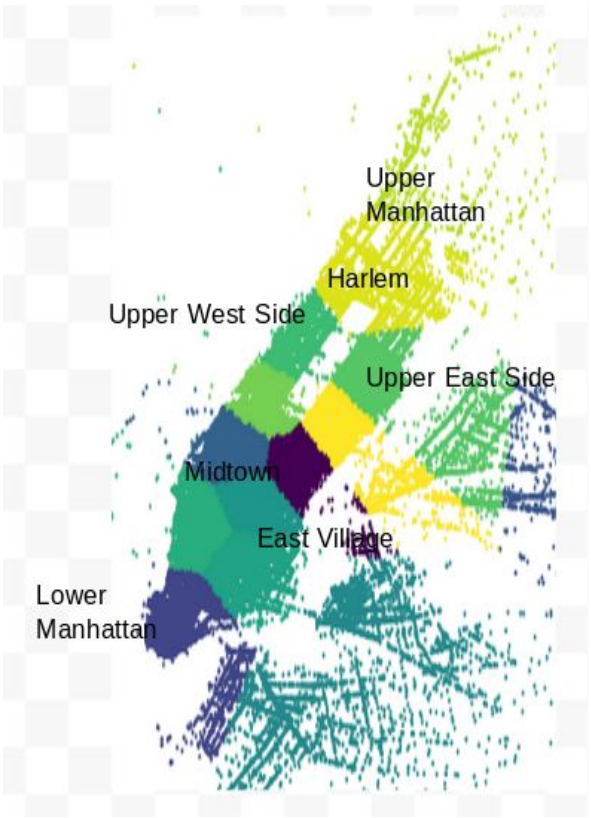
With performing machine learning on the data we hoped to predict some properties of transportation data so that it would be possible to inform the The goal of a clustering based on latitude and longitude was to see if the volume and relative position and volume of taxi pickups and dropoffs could accurately map major sections of New York City. In this analysis we used New York taxi data from January 2015 which had 2.8 million rows.

K-Means Clustering:

Before clustering the data we had to eliminate outlier values. There were many data-points that had 0 values for both pickup latitude and pickup longitude, or that had positional values far outside New York City. Also because of the scarcity of taxi data outside of Manhattan and its surrounding area, we chose to limit the area of analysis to between -73.75 and -74.05 for longitude, and 40.6 and 40.9 for longitude.

For clustering we used a K-Means model. This model clusters data points based on the grouping with the nearest mean. In this case we found a good K value to be 15 based off of error with K. The clustering gives a solid depiction of various section of Manhattan including Harlem, Upper East Side, Upper West side and Lower Manhattan. It also has clusters for Brooklyn and Queens, as well as a tight cluster around John F. Kennedy Airport. The results are shown below.





Linear Regression:

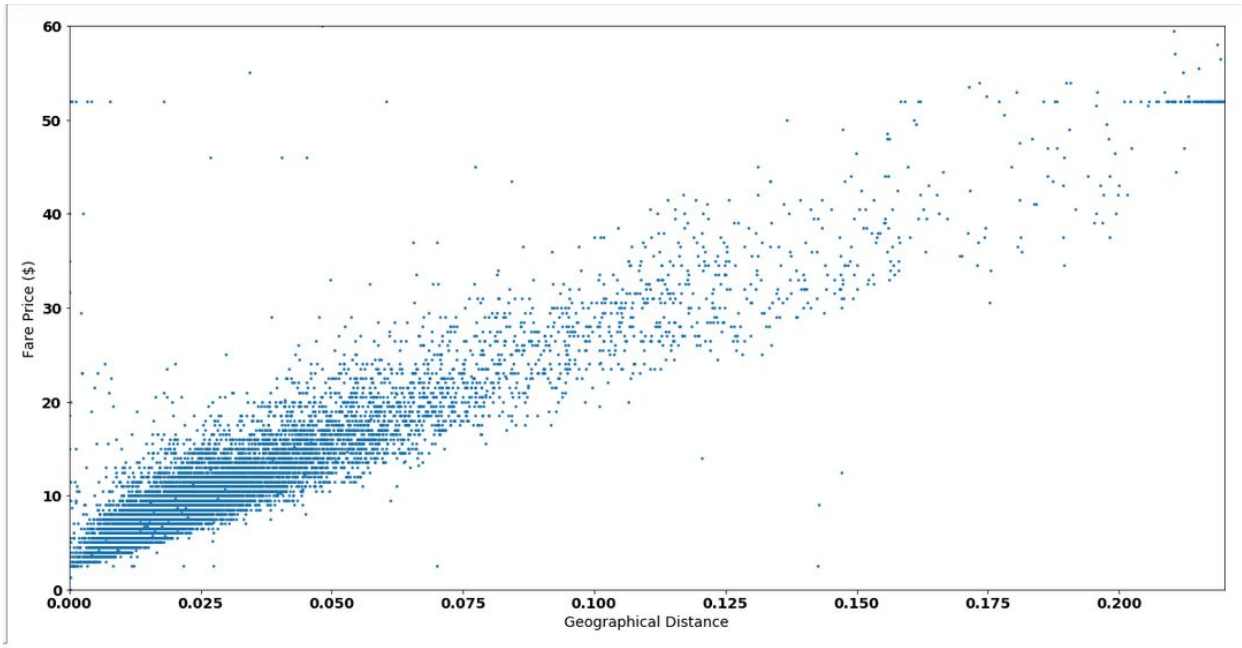
For thie regression we looked to predict the fare amount based off of starting point and ending point. To perform the linear regression we used Pyspark's mllib LinearRegressionWithSGD class. To analyze the regression we used Pyspark's mllib Regression Metrics class.

At the outset we tried putting the latitude and longitude as features for the regression and get a prediction of fare amount based off that. This was not successful because there was no clear ordering to the data and led to an average explained variance value of around 0.08, but an RMSE value greater than 10. To combat this issue with ordering a good solution seemed to be to bunch the data into regions and then do a regression based on region. To do this we first performed a K-Means clustering on the pickup latitude and longitude and a K-Means clustering on the dropoff latitude and longitude, each with 20 clusters. Hence, each row of data was assigned a value for a pickup and dropoff cluster. Then we performed a regression using the dropoff and pickup cluster values as features. This in an explained variance of around 0.4 which is fairly good, but the RMSE value was even higher at 11. This is most likely because having only 20 clusters to predict a specific fare amount would never account for the variability in fair price.

Next we wanted to try using straight line distance from pickup point to drop off point to get a regression on fare amount. This seemed like the best option since taxi companies presumably base fares off of distance traveled. For features in the regression model we used

the euclidean distance calculated from pickup longitude and latitude to drop off longitude and latitude as the only feature. Unexpectedly this resulted in a variance of 0.05 which was even far lower than before. We soon realized that the problem was that if the “intercept” field is not set to true for the Linear Regression class, the regression equation will have a y-intercept of 0.

Once this was fixed the resulting regression had an explained variance of 0.6 and the RMSE was still 10.1 dollars. Even though there seemed to be a clear relationship between fare amount and distance as seen below:



This is most likely because there were so many more data points at the lower end of the price spectrum that it skewed the regression equation to have a lower slope and a worse RMSE since most of the predicted values hovered between 11 and 15 dollars.

To solve this issue we used the QuantileDiscretizer class to split up the data into 100 equally spaced regions for both distance and Fair Price and took the mean of the data points in each region. This solved the problem of having too many data points at one end of the spectrum. The final RMSE was 5 dollars and the explained variance was 0.82.

What issues did you encounter in working with the dataset?

In terms of issues we encountered, a lot of if had to do with data being significantly messier than what we may have hoped for, as well as not always having the information available to us that we wanted to use. A good example of this would be the latter half of the taxi data. Whereas the 2016 taxi data has incredibly simple coordinate points used to mark locations in specific fields, the 2017 data used some sort of two three-digit codes to denote location. That being said, we weren’t able to find the proper dataset to join with the 2017 data in order to get coordinates from this different system for location.

Similarly for the bike data, because the dataset did not provide exact locations, we had to derive them from another dataset, while corrupt and missing data was a significant issue in the bike data, as the data was rather difficult to work with, and we did communicate with the maintainers of the data that their data is unwieldy. (Update 05/01/2018: they rolled out an update that fixed the corrupted data sets)

Some other issues we ran into, much like other groups, had to do with migrating too much data onto the cluster, certain jobs taking an extraordinarily long time, and coordinating jobs between groupmates.

How might your work be applicable to a real situation in-industry?

An interesting application of the taxi data would be to use the heatmaps and visualizations to determine where to service taxis to. You could analyze trends in taxi pickups and dropoffs to determine, for example, during evening rush hour you need most of your taxis servicing the midtown Manhattan area. Also, while the fact is sort of widely known, this data could have been used in foresight to predict the fall of taxi usage, in order to try and implement better strategies to either negate the effects of uber entirely or cushion the blow to business, because the decline is actually quite significant. This taxi data could also be used by businesses to find the more densely populated areas by time of day in order to adapt better business models and practices.

In the case of the parking tickets data, there are a couple of obvious real life situations in which our analysis could be used. The first is that the data could be used by ticketing officials of New York City. Because we have taken the data and visualized it onto a map, it is much easier to see which sections of the city are being focused on more by the ticketing officials, and they might be able to change their routes to cover more ground. Another application for our parking data is when it is used in conjunction with the parking meter dataset. With the ability to find parking meters near a user and then output them onto a map with colors representing how many tickets are given at that station, users could easily search for parking with the best chance of not getting caught. This same data could also be used by the police though as they search their routes to see which side roads are seldom checked.

In total, the results of our data can give city planners a better idea of how transportation in New York City functions and how commuters are making use of the various facilities available to them to traverse the city. For example during the summer months bikes are heavily used, while they are sparse in the winter. We see this correspond to a change in taxi usage potentially allowing an inference about relationships between the two.

Performance Metrics

Taxi data seemed to perform rather well (in terms of total job time) when not dealing with aggregation - parsing and writing coordinate points into a CSV for roughly 12 million entries finished in under 2 minutes, performing at upwards of 90k entries per second. With aggregates

however, and a much larger total number of entries at once, we're looking at a roughly 130k entries per second for a total runtime of nearly 30 minutes on the entire dataset, which contained nearly 240 million entries.

Parking ticket data is similar in performance to the taxi data. In the case of our initial aggregations during which we split up the parking ticket data by hour of the day and day of the week, we saw short execution times (only 15 seconds). In one case, when grouping by day of the week, the function ran in 13.14 seconds on a dataset of approximately 3GB meaning we had an average of approximately 215 MB/s. When we moved onto more complicated transformations of the data, namely joining two datasets together, we saw a significant slowdown. When joining the parking tickets (3GB) and the centerline (50MB) datasets, execution time increased to 73.28 seconds which means and average of only 40 MB/s.

Bike data performed similar to the taxi and parking data after the initial uploading of the giant dataset to HDFS, as mentioned before the initial combining of monthly datasets was done locally due to an inability to upload to HDFS, while the remaining filtering and separating of files was done completely on spark. In terms of runtime discounting cluster waiting, the files ran at ~150mb/s to run with the size of the dataset ranging from 1GB to 12 GB depending on whether the job was being run using the smaller yearly/season datasets or the big complete dataset.