Via Data Challenge

Anna Guidi • 09.28.2019

Part 1 - Aggregation

Overview

1. Propose a metric and/or algorithm to assess the potential efficiency of aggregating rides from many vehicles into one, given the available data. Make realistic assumptions and any necessary simplifications and state them.

2. Implement your proposed method and evaluate Manhattan's overall efficiency using yellow taxi data from the first full week (Monday-Sunday) in June 2016. Discuss how your method would scale with more data; in other words, discuss the complexity of your implementation.

3. Based on your implementation in the previous question, use visualizations to show how efficiency varies with time and location. Discuss any potential business implications based on your findings.

Overview

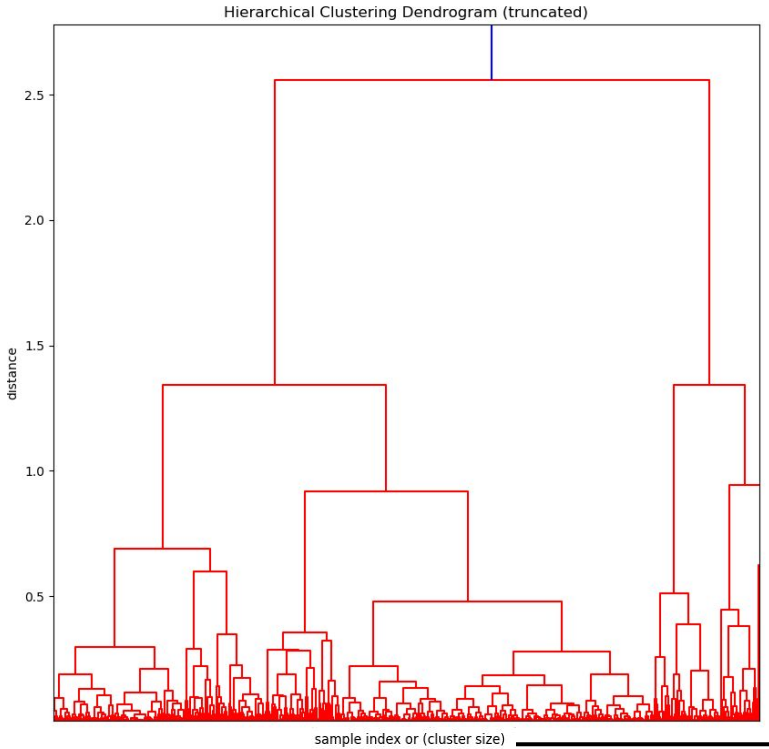
Approach chosen:

- Hierarchical Clustering
  - Bottoms-up approach, fast compared to alternative (such as K-Means variation)
  - ‘ward’ used in linkage method, Max Distance threshold set to .01 degrees (corresponding to ~1 km)
  - 4 features: pickup_lat, pickup_long, dropoff_lat, dropoff_long

Simplifications / Assumptions:

- Not trajectory based - rather based on which points have similar origin and destination points
- Used ***distance*** (~1 km maximum) as ***proxy for customer waiting time.***
- Clustered rides together based on pickup_time grouped in 5 minute bins rather than rolling time window
  - Made assumption that if you are up to 5 minutes into your ride, a rider would not be too upset to turn around to pick up another passenger, and additionally the driver has not driven too far from the pickup point yet, which is at most 1 km away from the first pickup location.
- Did not take into consideration traffic, weather, or cost of ride

Hierarchical Clustering



I used scipy linkage() together with fcluster() to see how Via could cluster rides together based on similar pickup and dropoff points in a 5 minute frame.

The scipy linkage() function uses a specified method ('ward', as it minimizes variance and tends to produce even sized clusters) and metric (.01 degrees) to calculate distances between clusters (which start off as 1 point consisting of pickup latitude, pickup longitude, dropoff latitude and dropoff longitude). Based on this calculation, each iteration will merge the two clusters which have the smallest distance according the selected method and metric.

It is unlikely that in this case many clusters will be merged (as can be seen in the image on the left), but it is important to know that this is a characteristic of the algorithm.

Methodology & Data Cleaning

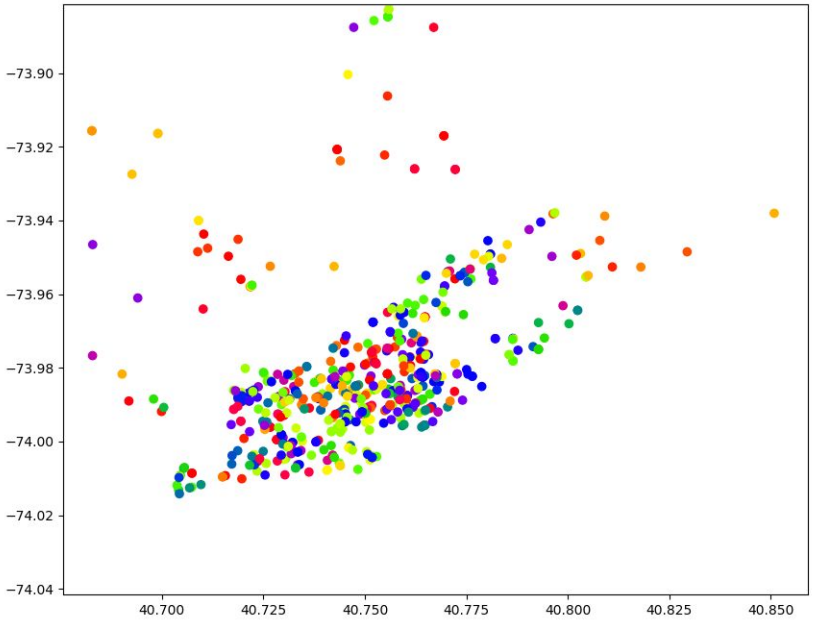
- Deleted Rides < 90 seconds
- Deleted Rides where pickup_lat and pickup_long == dropoff_lat and dropoff_long
- Deleted Rides where Passenger count == 0
- Checked that no rides had latlong or pickup/dropoff time blank
- Transformed pickup and dropoff to datetime
- Disregarded all rides where rider_count > 6. This is because a party of 6 or more likely ordered a yellow cab in advance, much like they would with Via, and I will make the assumption that both Taxi Cabs and Via vehicles have a **capacity of maximum 6 riders**, meaning there is no possibility of adding a 6th + rider along the way. In addition I will assume that a party of 6+ people would prefer a direct ride as opposed to rideshare even on the Via app. Therefore, we can assume that for 6+ riders, max efficiency is reached, and these rides will be excluded from any of the calculations.
    - In a future, more sophisticated analysis, it would be possible to optimize which type of car (capacity of 4, 6 or 6+) to send out in order to achieve maximum efficiency. This could also not be a static decision, but fluctuate during the day based on demand (discussed more in later slides).

Complexity of Implementation

Scalability:

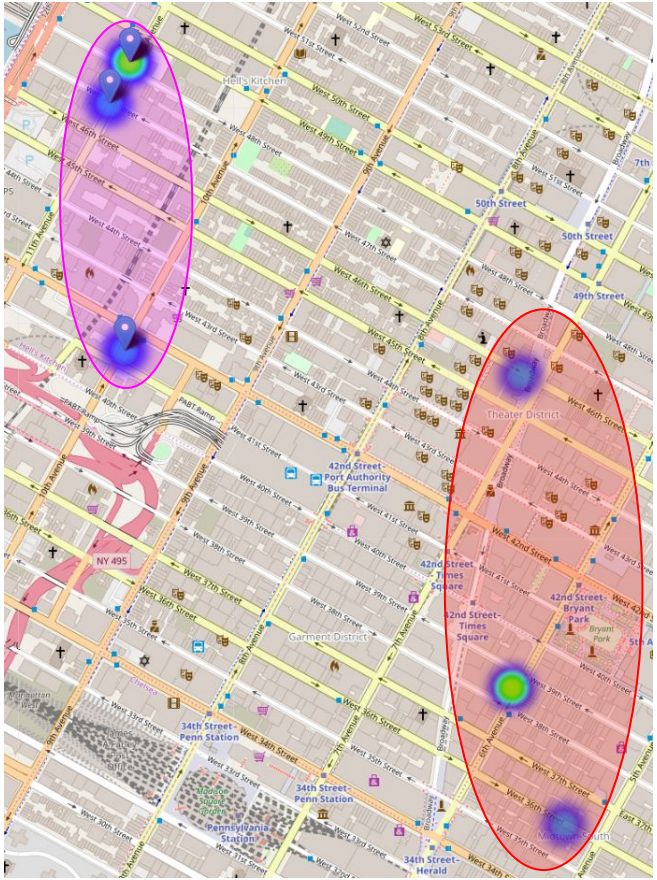
- Scipy linkage has O(n^2) complexity
- With ~750 data points it had a runtime of 30.7 ms ± 736 µs per loop, meaning that it could scale significantly before approaching slow run times.
- This has the advantage that we could run the same piece of code several times with different parameters such as vehicle capacity or other factors, and find an optimal solution more easily without having to worry about computational cost.

fcluster output - Overall view



The image to the left is the output of the algorithm for a 5 minute interval. We can clearly see the Manhattan outline and even traces of Manhattan's famous grid system.

In this view, it is difficult to make sense of the clusters as there are so many. However when we zoom in and look more in detail (next slide), we have an opportunity to look at what pickup and dropoff points constitute an individual cluster.



fcluster output - Verification single example

- We see what were **3 trips** (individual, individual, and group of 3 as can be seen by the green dot in the heatmap) that were grouped into **1 trip with 5 riders** by the algorithm
    - Pickup in pink (with the pins), Dropoff in red
    - Inter-distance for pickup and dropoff seems reasonable

Efficiency metric(s)

avg_rider_capacity_Taxi = avg number of riders based on yellow cab taxi data / Via max capacity (6)
avg_rider_capacity _Via = avg number of riders in car after / Via max capacity (6)

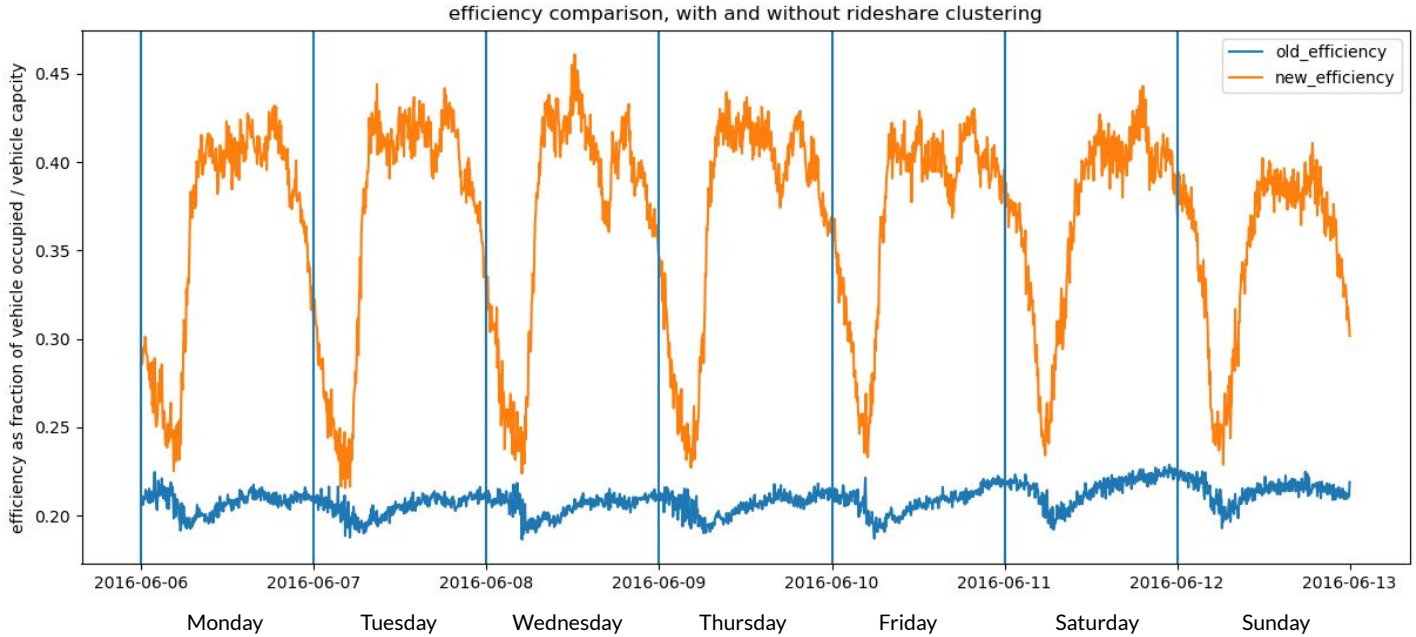
where we have clusters with > 6 riders, we assume we have to send multiple Vans with a capacity of 6 people, which will diminish the overall efficiency. For the sake of the calculation, we will take the average number of people in the van for the minimum amount of vans we need to send in order to accommodate that cluster (Ex. 8 individuals in one cluster would mean we have to send 2 vans and our average number of rider capacity will be 4/6)

The % change in efficiency can be calculated as:

avg_rider_capacity _Via - avg_rider_capacity_Taxi / avg_rider_capacity_Taxi

Additionally, I have also compared the number of unique clusters, meaning the number of Via vehicles needed to fulfill demand in a given time frame, to the number of taxicabs that were needed to satisfy the same demand in the same time frame.

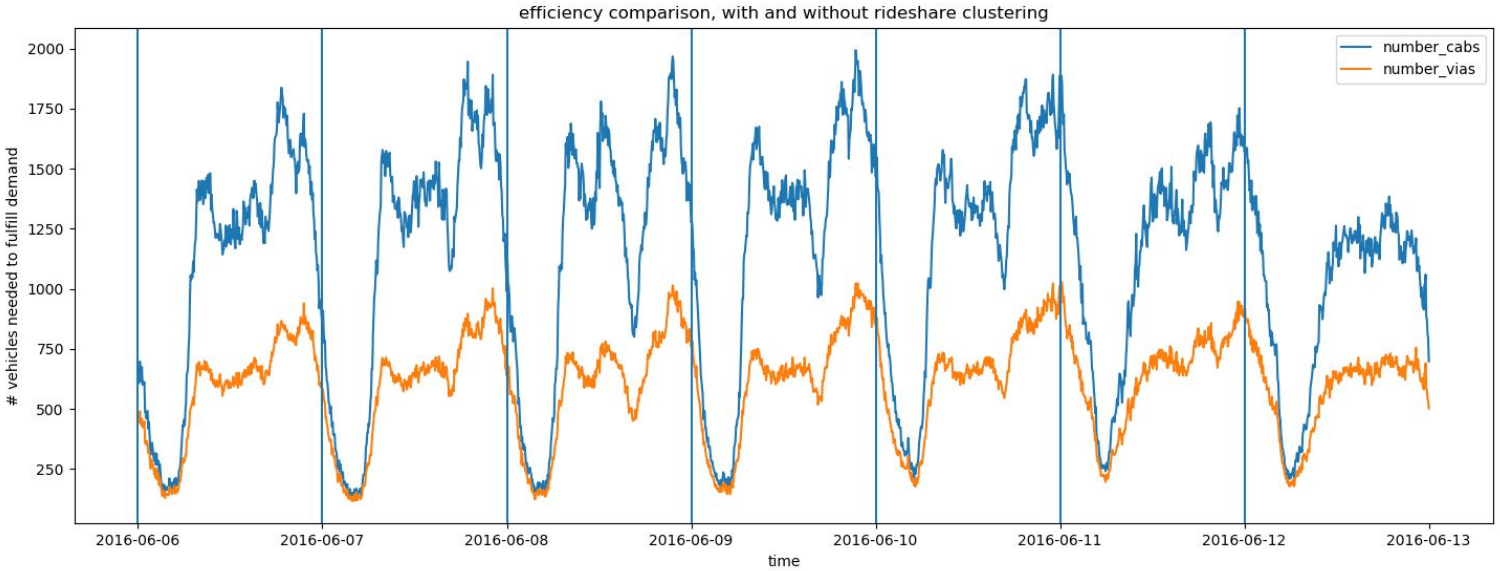
Efficiency over a week



What we are looking at:

This graph shows the % of a vehicle occupied (assuming each vehicle can take 6 people) over time for current taxicabs and projected efficiency if Via were to use the clustering algorithm to group riders in their vehicles.

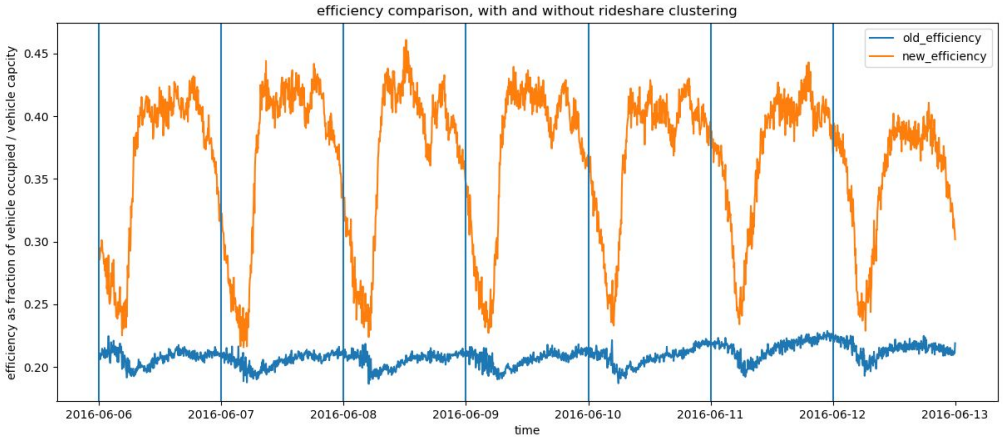
Efficiency in # of vehicles needed



What we are looking at:

This graph shows the number of vehicles needed on the road to satisfy demand, with current taxi cabs in blue, and Via vehicles in orange (again with the assumption of a maximum capacity of 6 people). **This could be a very useful metric for persuading cities such as New York and others, which are constantly preoccupied with fighting congestion and traffic, to allow Via to operate in these cities, as fewer vehicles means less traffic and congestion.**

Results - Efficiency over a week

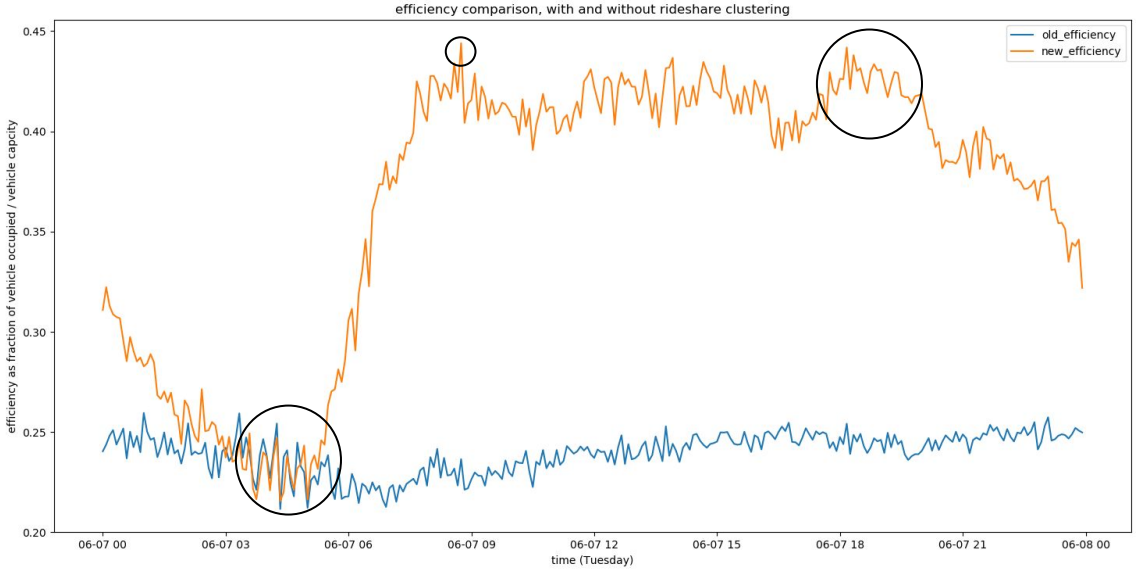


Findings & Business Implications:

As the graph shows, there is huge potential for increase in efficiency with ride-sharing throughout almost all hours of the day - the closest the ridesharing and regular taxi efficiency overlap is would be in the early AM hours (around 4 AM, more on next slide), when very few people are travelling in general and density of demand is lower.

We can see a generally repeatable pattern day to day which loosely corresponds with what we know about the average working person's daily activities, with Monday through Thursday looking more alike than Friday, Saturday and Sunday which have their peaks and troughs of efficiency at different points in time in the day (more on next slide).

Efficiency over time (24h in 5 min intervals)



What we are looking at:

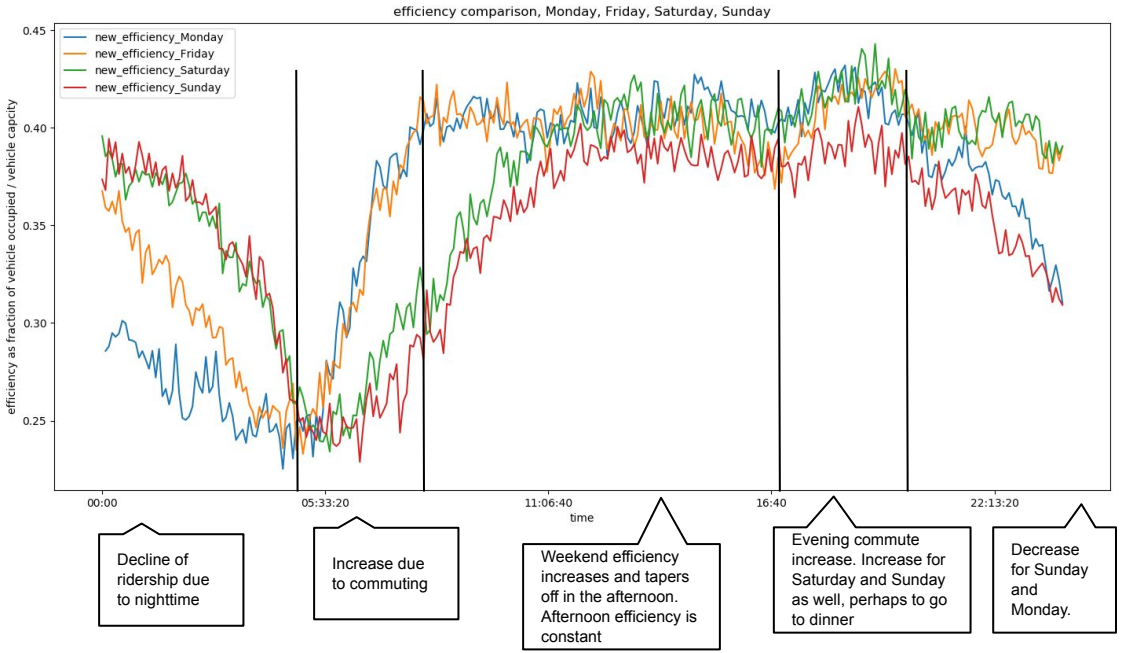
I have visualized Tuesday as it is a regular workday- as expected, the trough in demand and therefore in ridesharing efficiency is between 4 and 5 AM, when there is little reason to travel aside from perhaps needing to get to a flight or other rare occurrences such as emergencies. The only time the new_efficiency with clustering is lower is during this time. This is because (as discussed more in the next slide), capacity vehicle capacity of 6 passengers, means there is only 1 passenger to pick up and the resulting efficiency will be ⅛ as opposed to sending a smaller vehicle with a capacity of 4 people, the resulting capacity would be higher at ¼. However, a larger vehicle + clustering algorithm allows for greater efficiency by a wide margin during all other times of the day.

We see efficiency increase exponentially between 5 AM and 8 AM (8 AM being the time during with we can see peak efficiency), when commuters need to get to work. This of course is a great opportunity for Ride sharing and larger vehicle sizes.

After the morning commute we see a mild decrease in clustering efficiency whereas old_efficiency related to taxicabs climbs but is still much lower - this makes sense as in a commute we can expect a larger number of people to have the same destination (such as FiDi or Midtown), making it more efficient to cluster riders together. Post lunch, trips will be more personal and individualized, making them harder to cluster together.

We see another spike in the evening, although it is more sporadic than the morning spike- this makes sense as people generally leave the office during the same time frame, between 5 and 7 PM, but there is much more variance than arrival time at the office, when there is pressure to be on time.

Weekday vs. Friday vs. Weekend Comparison



What we are looking at:

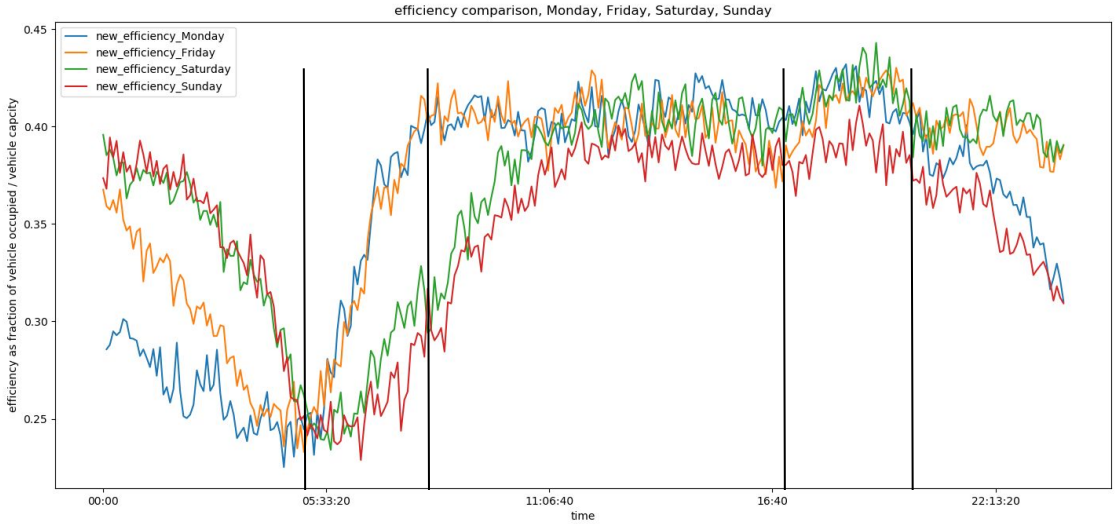
Here we are benchmarking efficiency of Monday (representing a “typical” workday) vs. Friday, Saturday, and Sunday.

The efficiency fluctuates as we would expect if we think of demand - between midnight and 5 AM is where we see some of the biggest differences in efficiency across the 4 days.

While the efficiency line graph across all 4 days follows mostly the same patterns, the approximate slopes for what we can think of as “segments” between the days vary, and troughs and peaks are offset depending on the day. For example, on Monday between midnight and 4 AM night we have low efficiency relating to low demand, as people start the workweek the next day and are not out and about. In contrast, Saturday and Sunday we see similarly declining, but much higher efficiency during the same time period, likely because of higher density of demand from riders coming home from a night out on Friday and Saturday the night before. The efficiency for Friday is somewhere between Monday and the weekend, which also makes sense since Thursday evening is still a relatively popular time to go out, if not as popular as the weekend.

We can see that Monday and Friday have their troughs between 4 AM and 5 AM, whereas Saturday and Sunday the troughs happen a little later as there might still be a few more people out partying. As the day continues, we see a much sharper increase in efficiency on Monday and Friday, as riders aim to get to work by a certain hour, whereas demand for Saturday and Sunday increases much more gradually, and reaches its peak much later in the day.

Weekday vs. Friday vs. Weekend Comparison (Continued)



What we are looking at:

The demand is relatively consistent between 11 AM and 5 PM, which could be an advantage for Via as this makes resource planning more flexible throughout the week. On weekdays, starting around 5 PM we see a 2 -3 hour window where there is a spike in efficiency due to evening commutes before all 4 days hit a trough around 8.30 PM (perhaps a lull in demand due to dinner).

After ~8.30 PM is where we see the biggest difference between the days: On Monday and Sunday evenings we see a steady decline in efficiency as people prepare for the workday, whereas on Friday and Saturday we see a lot of fluctuation in the remaining hours of the day, and it is harder to pinpoint when the decline in ridership starts happening - this of course is due to people heading out to celebrate the weekend at different points of time in the night, and pickups to go out overlapping at some point with pickups to go home.

The last observation is that efficiency is lower on Sunday overall, as more people likely prefer to stay put.

Summary

Findings & Business Implications:

- The efficiency fluctuates according to what we know about a working individual's daily habits and routine, meaning that demand is never completely unpredictable. Weekdays differ from weekends, especially at night.
- Commuter spike at 8 AM, steady demand between noon and the 5PM, Commuters go home between 5 and 7 PM, demand then decreases (with speed of decline depending on weekday vs weeknight), with a trough around 4 AM
- The business implications: whereas initially we have assumed a static capacity vehicle capacity of 6 passengers, which could backfire at certain times of the day- say at 4 AM when there is only 1 passenger to pick up and the resulting efficiency will be ⅙ as opposed to sending a smaller vehicle with a capacity of 4 people, the resulting capacity would be higher at ¼. Based on these visualizations, not only can Via optimize for an the best mix of vehicle types depending on vehicle capacity, Via can employ drivers with different car types at different times of the day, so that more vans are deployed during rush hour and Saturday nights, and smaller vehicles at times during the day when demand is low and there are more single-rider clusters. Via could also possibly offer a commuter pass to entice new riders to travel via Via shuttles to work, or other passes during maximum demand time to and from popular locations.

Efficiency by Location

Approach:

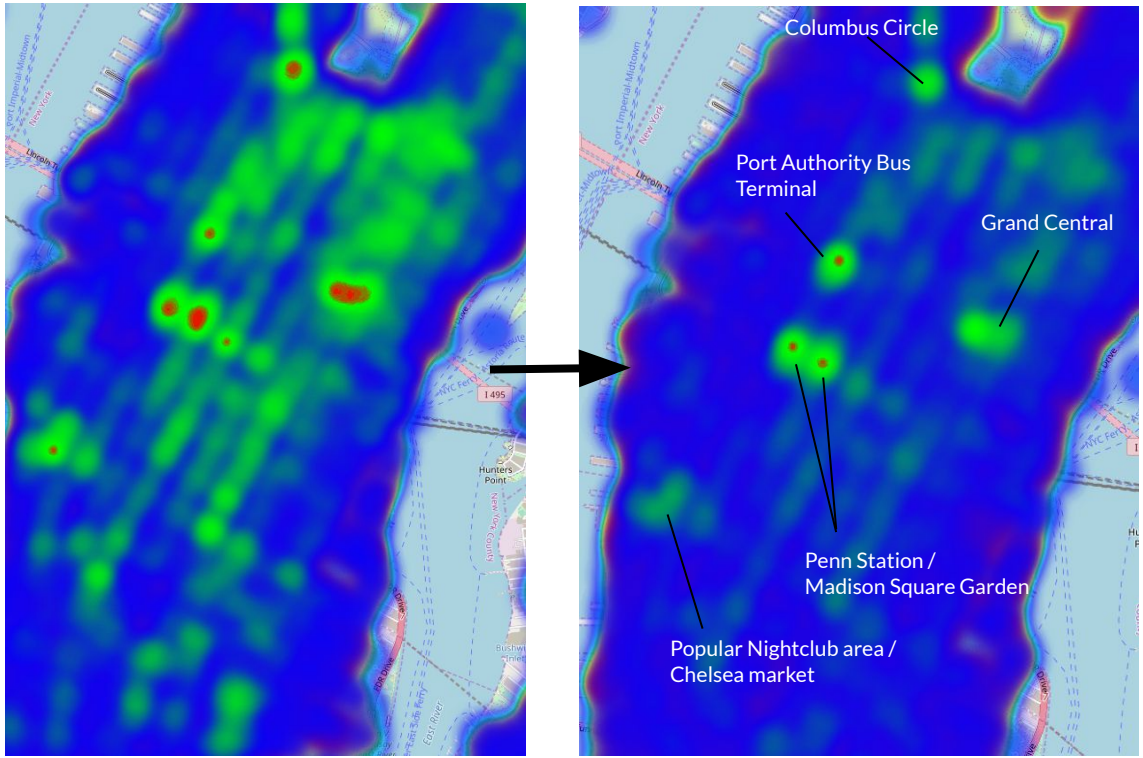
- Initially, I attempted to select a 10% sample size from the dataframe and compare efficiency metric by pickup Zip Code - however mapping latlong to Zip Code (via geopy) takes time and moreover not all points can be easily tied to a Zip Code.
- Another approach that could be taken is to map latlongs to Manhattan neighborhoods (using the Shapely library and a jeojson file made available by the city of New York), and compare efficiencies by neighborhood via a choropleth perhaps. However, iterating over all neighbourhoods in order to classify a latlong point takes time as well.
- Since we know that efficiency is linked closely to demand density, the easiest way to get a rough idea of efficiency by location is by understanding what demand looks like by plotting a heat map based solely on the latlong of a given rider's pickup point. This is the final approach that I chose.

Efficiency by Location

Observations:

- Based off of a random sample of 10% of the data, using the Folium library I created a heatmap of what are demand points based off of the pickup latlong (parameters can be tinkered with to change how the heatmap flags dense demand areas).
- On first glance, we can see that the majority of the demand is concentrated in Manhattan, in particular what seems to be below 94th street.
- We can also see 2 “blobs” that correspond to LaGuardia’s 2 terminals and JFK.
- Because we area analyzing yellow cab data, it makes sense that the demand in Brooklyn and Queens is lower as there are green taxis to satisfy demand there as well.

Efficiency by Location



Findings & Business Implications:

- Clearly making rides available to and from major transportation hubs is very important, and Via should optimize pricing and strategy around this fact.
- Upper midtown remains important - due to the many office buildings but also perhaps due to the many tourist attractions in the area.
- Heatmaps can be a good indication of where to locate Via “hubs” or where to route drivers to when they do not currently have anyone to pick up.

Top Improvements / Next Steps

Algorithm

It would be better to have a more sophisticated trajectory-based algorithm as in real life passengers are added to a vehicle based on their distance to where the car currently is, not the original pickup location. For this approach to work, it would also be necessary to work with a rolling time frame as opposed to 5 minute intervals.

Car capacity

As mentioned before, in order to increase efficiency even further, it would be worth optimizing for a mix of different vehicles having different vehicle capacity in order to truly maximize efficiency and have lowest number of cars on row possible.

Quality of Code

The quality of the code could be cleaned up by putting code into functions and setting more variables.

Part 2 - Statistics

1) Assume Via has a demand prediction algorithm for predicting hourly demand, 24 hours in advance

a.1) How would you compare the performance of this algorithm across cities of varying size?

In order to try to understand if there is a statistically significant difference in the performance of the algorithm between cities of varying size, we could start by performing a 2-Sample t-test between 2 cities. We would compute the mean performance (for a given time frame) of each city as well as the variance, and judge whether or not there is a difference in performance based on the outcome of the test, or the p-value, is less than .05 in order to have 95% confidence that the difference is not due to chance. In case the sample size varies significantly between 2 cities, **Welch's Test** for Unequal Variances can be used instead of a simple t-test. We could then compare all combinations of cities and display the difference in a matrix.

a.2) Propose a way to predict performance of the demand algorithm in a city we're considering operating in.

Similar to this challenge with the yellow taxi cab data as indicator of demand, it would be useful to gather historical data on what the actual demand was, as well as the features that would be available to the demand algorithm. The algorithm would take these features as input, and output a certain demand estimate that we could compare to the actual demand based on the historical data of the city we are considering expanding to. We would then see how close the estimate matches the reality. (This is assuming the algorithm we are testing was not trained on the historical data of the new city we are considering operating in).

b) A data scientist builds a new demand prediction algorithm, but it is more resource intensive than the other one - we'll only switch if we're sure the new one is better. How would you determine if the new algorithm is worth it?

We can test the new algorithm on the upcoming week's data without rolling it out, and compare whether it did significantly better than the current one by checking for statistical significance using a 2 sample t-test. If we determine that the new algorithm would have performed better, we can quantify how that translates into increased revenue, by looking at the delta between the revenue the new algorithm would have brought in (for instance bringing in more riders) versus the revenue the current algorithm brought it. Conversely, we could also look at the delta between money saved by the new algorithm and the current one. Assuming the delta is positive (this would likely be the case since we determined through the t-test that the new algorithm is a "better" algorithm), we would then subtract the additional cost of resources needed to run the new algorithm from the projected money earned (or saved). If the result is positive (> 0), it would be worth implementing.

Revenue gain from running new algorithm - increase in cost to run new algorithm

2) A claim reports that between 22.4% and 43.9% of Via rides have excellent music. You can assume "excellent" is a binary decision at the ride level. What do you think the sample size was?

The formula for a one sample, dichotomous outcome is: $$n = p\left(1 - p\right)\left(\frac{Z}{E}\right)^2$$

- We will set Z = 1.96 as we would like a standard 95% confidence rate
- *p* is the population proportion, which we do not know, however we know that *p* ranges from 0 to 1, and in order to have the largest and therefore most conservative population proportion *p*(1 - *p*), we can set *p* at .5
- E is the desired margin of error, and in this case it would be ((.224 + .439) / 2) - .224, which is = .1075

We plug these numbers into the formula and get a sample size of **83.1 → 84 individuals**