CE 263: FINAL PROJECT

Group Members: Jules Pommies, Christopher Tong and Catalina Vanoli

Dataset: New York Taxi and Limousine Rides Records (As the volume of the data set is very large, we only used the rides from June 2016.)

Introduction:

Since 2009, the New York Taxi and Limousine Commission has kept a detailed record of all taxi rides in the region. The dataset provides an in-depth view of daily taxi operations as well as a profile of the people who use this form of public transportation. Due to the wide adoption of taxis as a mean of public transportation, the dataset can even be used as a proxy so study the city as a whole, from socio economic elements within the city's neighborhoods, to capacity of city streets.

Out project focuses specifically on the trips generated due to airport activities, specifically the trips to and from JFK Airport. We have several reasons to choose only these trips. First, focusing on a defined location allows our team to base our analysis on knowledge of a specific trip activity- in this case air travel. Secondly, the size of the data does not allow for a complete analysis of all trips; we are limited by computation power of our computers.

The goal of our project is to provide an interactive environment where the user can analyze trips. Secondary goals include fitting the data into simple models to provide users the context to analyze such data.

Within these trips, we explored certain features such as tips, travel times, and trip density by hour of day. In addition we characterized into the different taxi zones. These zones were downloaded from the TLC website and divide the city of New York into taxi serving zones.

Cleaning Data and Storage:

Due to the sheer size of the dataset and the limited capabilities of our computers we decided to focus on one month of data and specifically only on trips to and from JFK airport. We used Python GeoPandas to process individual sets of 1 million datapoints as any more would cause a memory storage error for our computers. After processing the 11million rows of data from June 2016 using the shapely library (next section) to only include dropoffs and pickups from JFK, we were left with only 384K points to analyze for the month.

The processed data was then stored in a PostGIS database (along with shapefiles of taxizones) for later spatial querying and to identify pickup and dropoff zones of individual trips. In order for Mapbox to visualize datasets on the browser however, the points need to be converted to a geojson format to be then stored in MongoDB and queried by Mapbox. At the moment we have had trouble setting up the backend server and so all datasets would be converted to a json flat file after processing. In future we would like to design a webapp with NodeJS and MongoDB as the backend.

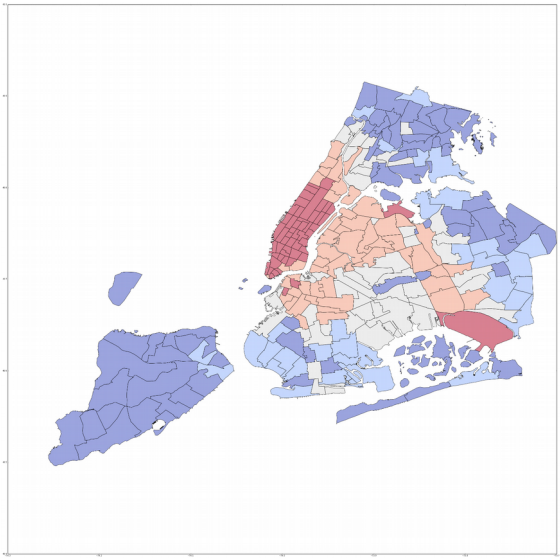
Spatial Querying

There is a boolean value in the dataset which taxi drivers can use to set price to and from Manhattan Island to the flat rate of $52. These do not account for taxi trips with negotiated values or those pickup or dropoff locations outside of Manhattan. In order to get these points, we will need to find all trips with start or end points contained within the JFK airport region. For the processing we used the Shapely contains function due to its compatibility with Pandas. Afterwards, to match taxi trips with zones, we used the postgis querying language to match pickup and dropoff zones for each individual trip.

Alternatively, the GDAL library was used to transform coordinates of the taxi zones shapely file and the new york city boroughs shapefile into latitude and longitude coordinates.

The shapely library was used with the contains and within functions to find the amount of rides made from each zone to the airport or the amount of rides generated in the airport to each taxi zone. With these information we created the following plots which shows the density of trips per zone.

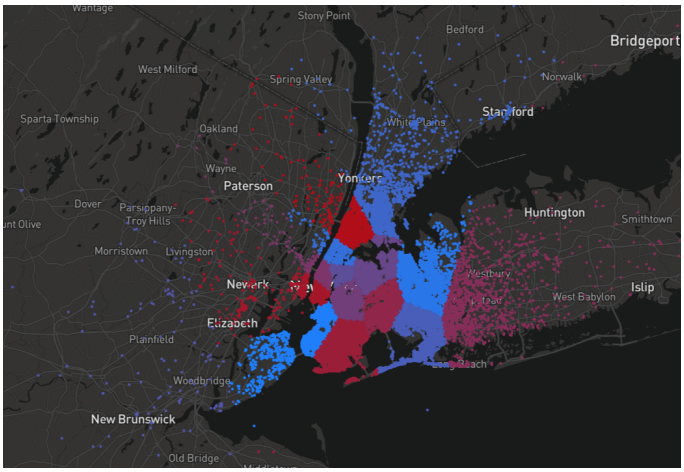
Graph 1 : Density of taxi rides from zones to the airport per Zone



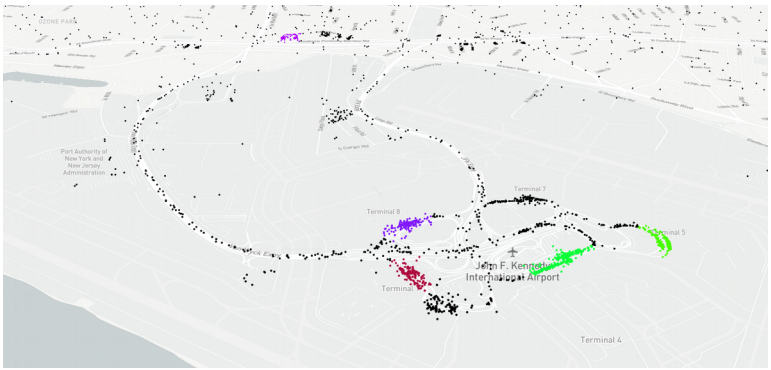
Clustering and Prediction Models:

To analyze the trips generated by the JFK airport activity we decided to cluster by location. We used two methods: Mini Batch KMeans and DBScan. We implemented both methods using the sklearn package.

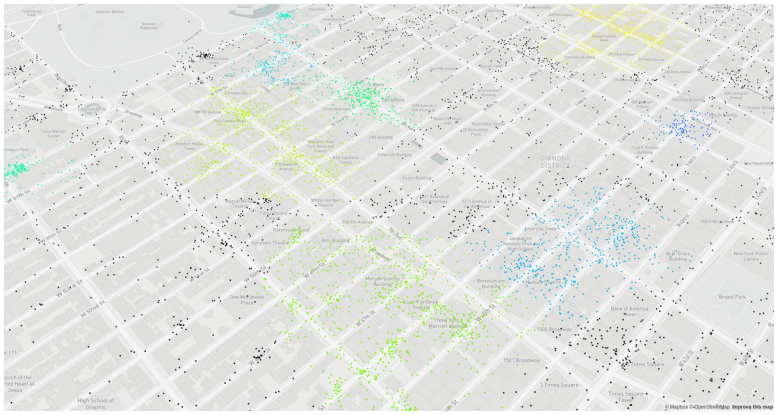
The MiniBatch KMeans clustering was faster than DBScan but its results were not useful. With a Batch size of 500 and 15 number of clusters, the following plot shows these results:



The DBScan method needed more time to process, so we used only the first 100,000 rows from the airport trips table. The epsilon value was set as .008 KM. With the results uploaded, some interesting places were highlighted in the clusters. These places are travellers destinations such as hotels, landmarks in New York City Like Union Square.



Graph: DBSCAN result of pickup and drop off locations in JFK airport. Clusters represent terminal locations. In the distance is a car rental area and airport hotel

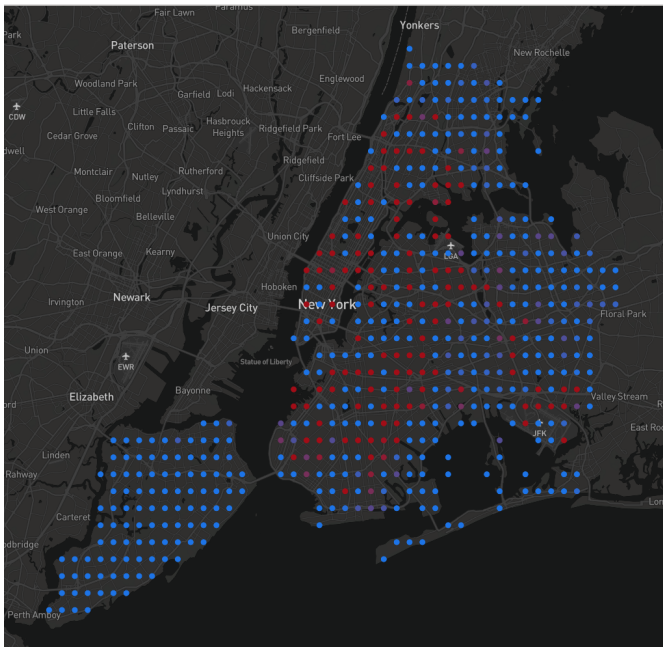


Graph : Clusters show prominent dropoff locations from airport such as Times Square (light green) , Waldroff Astoria Hotel (yellow) and Plaza Hotel (aqua)

The time prediction was based on travel time from JFK to each location of New York city during rush hour. We implemented the Gaussian Process Regressor from the sklearn package with a sigma value of 0.5. Only 10,000 rows were selected to be fitted in the model. The features introduced were longitude, latitude and pickup

hour. The hyperparameters of the kernel are optimized during the fitting by maximizing the log-marginal-likelihood (LML).

To test the fitting Regressor, we created a grid of pick up locations all over the area of the zones in study. Also to generate pick up hours, we randomly chose the points between 14:00 to 17:00, because the graphs of time vs travel time presented an increase during this period. The following plot shows the point of the generated grid with their colors as the travel time prediction. .



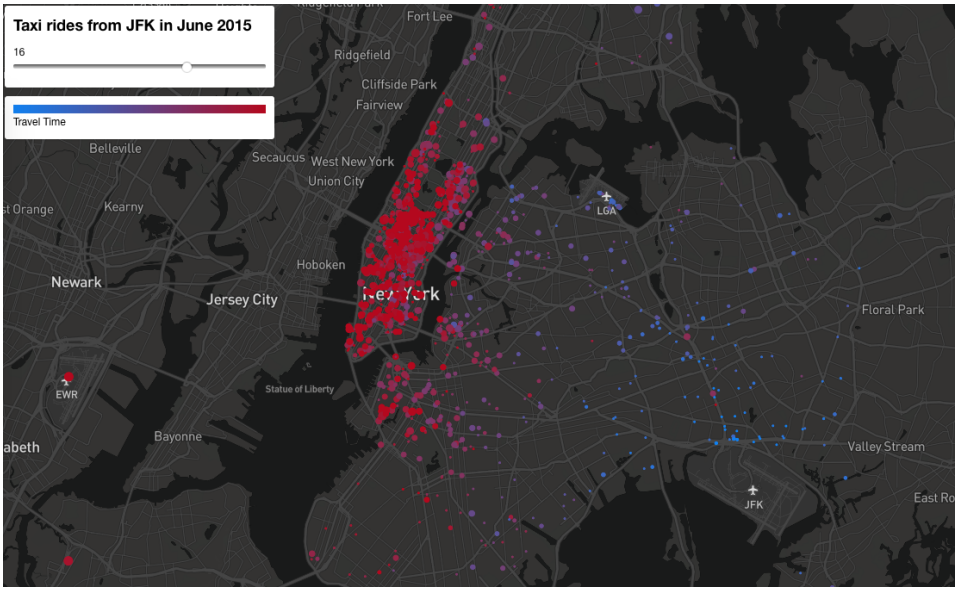
Graph: Gaussian Process to predict travel times from airport during rush hour.

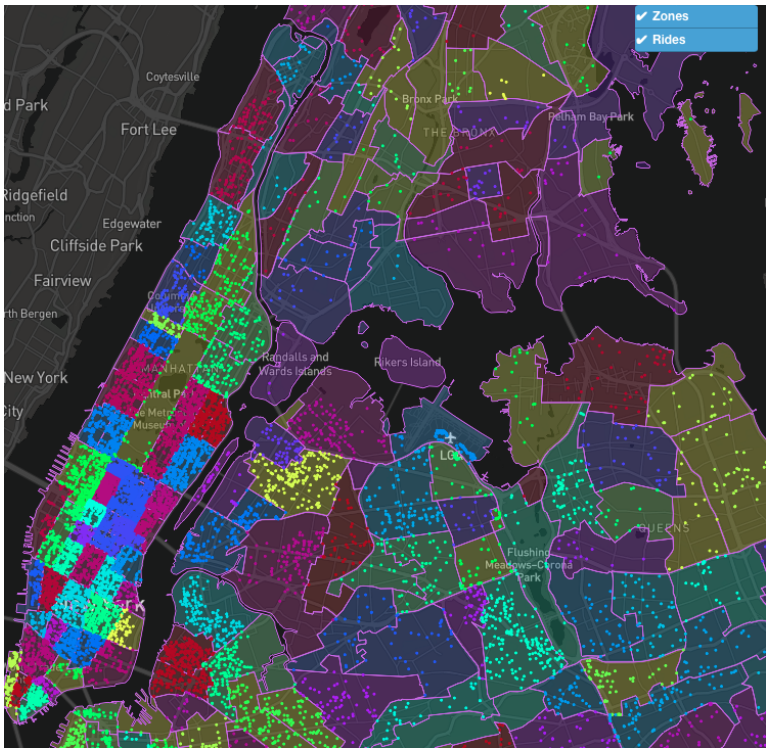
With more time and more computational power, the next step on this method would be to introduce other features such as weather, day of the week as well as incorporate more rows to train the model in a better way.

Mapbox Visualization:

We used the Mapbox API to display the results of our analysis. The goal to use the browser to be able to create an interface between the user and the data. We wanted to create a browser API where the user could select and sort the data he/she wants to display. We also took advantage of the data-driven styles available in Mapbox in order to add more information on one map.

Here is an example where the user can select the time of the day to display with a slide bar. Then the points are colored regarding the travel time from JFK and the size is representing the amount of tip of the ride.





The next example is a layer selection where the user is able to select which layer of data he/she wants to display on the map. In this specific case it is the different taxi zones, and/ or the rides colored regarding those very same zones.

Next Steps:

With more time, we plan to no longer use flat file to store visualization data and use MongoDB instead. The model parameters can still be optimized. In addition, we want to find a way to alternate seemlessly between each model in our visualization. We would also implement time series into our visualizations and establish a process to clean data, upload and fit into models.

It would also be interesting to continue investigating travel time and the fluctuations in time and space.

Appendix:

http://www.nyc.gov/html/tlc/html/about/trip_record_data.shtml.

The data available has the following features.

Field Name	Description
VendorID	A code indicating the TPEP provider that provided the record. **1= Creative Mobile Technologies, LLC; 2= VeriFone Inc.**
tpep_pickup_datetime	The date and time when the meter was engaged.
tpep_dropoff_datetime	The date and time when the meter was disengaged.
Passenger_count	The number of passengers in the vehicle. This is a driver-entered value.
Trip_distance	The elapsed trip distance in miles reported by the taximeter.
Pickup_longitude	Longitude where the meter was engaged.
Pickup_latitude	Latitude where the meter was engaged.
RateCodeID	The final rate code in effect at the end of the trip. **1= Standard rate** **2=JFK** **3=Newark** **4=Nassau or Westchester** **5=Negotiated fare** **6=Group ride**
Store_and_fwd_flag	This flag indicates whether the trip record was held in vehicle memory before sending to the vendor, aka "store and forward," because the vehicle did not have a connection to the server. **Y= store and forward trip** **N= not a store and forward trip**
Dropoff_longitude	Longitude where the meter was disengaged.
Dropoff_ latitude	Latitude where the meter was disengaged.
Payment_type	A numeric code signifying how the passenger paid for the trip. **1= Credit card** **2= Cash** **3= No charge** **4= Dispute** **5= Unknown** **6= Voided trip**
Fare_amount	The time-and-distance fare calculated by the meter.
Extra	Miscellaneous extras and surcharges. Currently, this only includes the $0.50 and $1 rush hour and overnight charges.
MTA_tax	$0.50 MTA tax that is automatically triggered based on the metered rate in use.
Improvement_surcharge	$0.30 improvement surcharge assessed trips at the flag drop. The improvement surcharge began being levied in 2015.
Tip_amount	Tip amount – This field is automatically populated for credit card tips. Cash tips are not included.
Tolls_amount	Total amount of all tolls paid in trip.
Total_amount	The total amount charged to passengers. Does not include cash tips.


Clifton
Passaic
Heights
Ridgefield Park
Fort Lee
Ridgefield
Montclair
Nutley
Rutherford
Cliffside Park
Fairview
Bloomfield
Belleville
North Arlington
Secaucus
Union City
Weehawken
East Orange
Kearny
Harrison
Newark
Hoboken
Jersey City
Hillside
Elizabeth
Bayonne
Carteret
Sands Point Seaplane Base
NY 25A
Mineola
Garden City
Hempstead
Valley Stream
Rockville Centre
East Rockaway
Long Beach
Leaflet | Data by OpenStreetMap, under ODbL.