Coursera Capstone project

Coursera IBM Applied Data Science

F. Brouwn

June 22th, 2020

Content

- Introduction
    - Background
    - Problem
    - Stakeholders
- Data
- Methodology
- Exploratory Data Analysis
- Data Cleaning
- Feature Engineering
- Data Visualization
- Machine Learning
- Results
- Conclusion

Introduction

- Background
  - New York City is the most populous city in the United States, home of many headquarters/global institutions, there are people from all over the world with different cultural backgrounds (800 Languages)
  - Therefor high demand for various cuisine in NYC neighborhoods
- Problem
  - Finding the right neighborhood for opening a new restaurant can challenging. A restaurant owner/investor needs a lot of data to make a substantiated decision for the choice of a certain location.
- Stakeholders
  - This analysis can be used for anyone who is interested in the distribution of different cuisines in NYC; Restaurant investors/owners, etc.

DATA

- **New York City Dataset**
  - Link: https://geo.nyu.edu/catalog/nyu_2451_34572
  - dataset will provide the addresses of neighborhood of NYC
  - JSON
- **Foursquare API**
  - Link: https://developer.foursquare.com/docs
  - Foursquare API, a location data provider, will be used to make API calls to retrieve data about venues in different neighborhoods

Methodology

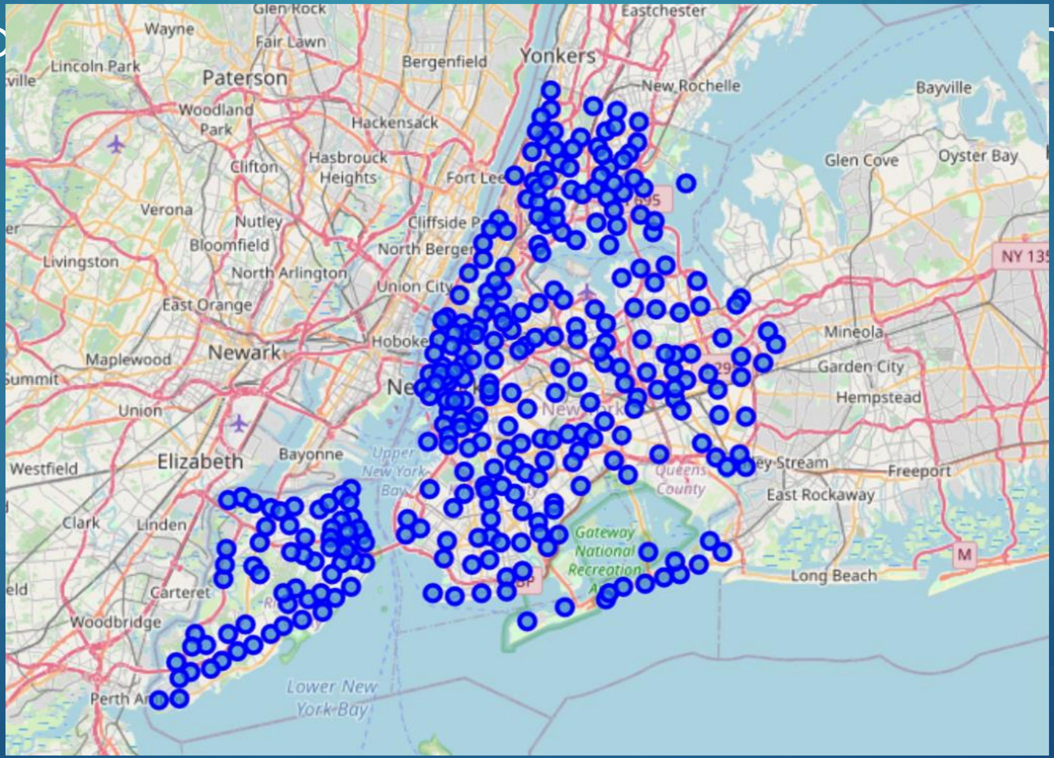
- Req. dataset that contains the 5 boroughs and the neighborhoods
  - JSON file --> Panda dataset

	Borough	Neighborhood	Latitude	Longitude
0	Bronx	Wakefield	40.894705	-73.847201
1	Bronx	Co-op City	40.874294	-73.829939
2	Bronx	Eastchester	40.887556	-73.827806
3	Bronx	Fieldston	40.895437	-73.905643
4	Bronx	Riverdale	40.890834	-73.912585

- 5 boroughs and 306 neighborhoods NYC

Methodology

- The curated dataframe is then used to visualize by creating a map of New York City with neighborhoods superimposed on top (using Folium Library)
- The Foursquare API is used to explo m



Mythologie

- There are many endpoints available on Foursquare for various GET requests. To explore the cuisines, it is required that all the venues extracted are from ‘Food’ category
  - found that there are 10 major categories (Foursquare API)

```
4d4b7104d754a06370d81259 Arts & Entertainment
4d4b7105d754a06372d81259 College & University
4d4b7105d754a06373d81259 Event
4d4b7105d754a06374d81259 Food
4d4b7105d754a06376d81259 Nightlife Spot
4d4b7105d754a06377d81259 Outdoors & Recreation
4d4b7105d754a06375d81259 Professional & Other Places
4e67e38e036454776db1fb3a Residence
4d4b7105d754a06378d81259 Shop & Service
4d4b7105d754a06379d81259 Travel & Transport
```

The ‘FOOD’ category (ID’ = ‘*4d4b7105d754a06374d81259*’)

Mythologie

- To overcome redundancy, a function ‘getNearbyFood’ is created. This functions loop through all the neighborhoods of New York City and creates an API request
  - Python list--> dataframe

```python
def getNearbyFood(names, latitudes, longitudes, radius=1000, LIMIT=500):
    not_found = 0
    print('***Start ', end='')
    venues_list=[]
    for name, lat, lng in zip(names, latitudes, longitudes):
        print(' .', end='')

        # create the API request URL
        url = 'https://api.foursquare.com/v2/venues/search?&client_id={}&client_secret={}&v={}&ll={},{}&radius={}&categoryId={}&lim
            CLIENT_ID,
            CLIENT_SECRET,
            VERSION,
            lat,
            lng,
            radius,
            "4d4b7105d754a06374d81259", # "Food" category id
            LIMIT)

        try:
            # make the GET request
            results = requests.get(url).json()['response']['venues']

            # return only relevant information for each nearby venue
            venues_list.append([(
                name,
                lat,
                lng,
                v['name'],
                v['location']['lat'],
                v['location']['lng'],
                v['categories'][0]['name']) for v in results])
        except:
            not_found += 1


    nearby_venues = pd.DataFrame([item for venue_list in venues_list for item in venue_list])
    nearby_venues.columns = ['Neighborhood',
                  'Neighborhood Latitude',
                  'Neighborhood Longitude',
                  'Venue',
                  'Venue Latitude',
                  'Venue Longitude',
                  'Venue Category']
    print("\nDone*** with {} venues with incompelete information.".format(not_found))
    return(nearby_venues)
```

Mythologie

- The returned 'nyc_venues' dataframe is as follows

	Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
0	Wakefield	40.894705	-73.847201	Central Deli	40.896728	-73.844387	Deli / Bodega
1	Wakefield	40.894705	-73.847201	Carvel Ice Cream	40.890487	-73.848568	Ice Cream Shop
2	Wakefield	40.894705	-73.847201	Cooler Runnings Jamaican Restaurant Inc	40.898083	-73.850259	Caribbean Restaurant
3	Wakefield	40.894705	-73.847201	Him Health Food Market	40.897665	-73.854638	Food
4	Wakefield	40.894705	-73.847201	Wakefield Deli	40.901998	-73.846910	Deli / Bodega

- There are now two python 'dataframe' are available:
  - 'neighborhoods' which contains the Borough, Neighborhood, Latitude and Longitude details of the New York City's neighborhood, and
  - 'nyc_venues' which is a merger between 'neighborhoods' dataframe and its 'Food' category venues searched with 'Radius' = 500 meters and 'Limit' = 100. Also, each venue has its own Latitude, Longitude and Category.

Exploratory Data Analysis

- The merged dataframe 'nyc_venues' has all the required information (13,724 venues total)

```
print(nyc_venues.shape)
nyc_venues.head()
```

```
(13724, 7)
```

	Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
0	Wakefield	40.894705	-73.847201	Central Deli	40.896728	-73.844387	Deli / Bodega
1	Wakefield	40.894705	-73.847201	Carvel Ice Cream	40.890487	-73.848568	Ice Cream Shop
2	Wakefield	40.894705	-73.847201	Cooler Runnings Jamaican Restaurant Inc	40.898083	-73.850259	Caribbean Restaurant
3	Wakefield	40.894705	-73.847201	Him Health Food Market	40.897665	-73.854638	Food
4	Wakefield	40.894705	-73.847201	Wakefield Deli	40.901998	-73.846910	Deli / Bodega

- There are 190 unique categories:

```
There are 190 uniques categories.
Venue Category
Deli / Bodega                 1266
Pizza Place                   1084
Coffee Shop                    936
Chinese Restaurant             684
Donut Shop                     644
Fast Food Restaurant           601
Bakery                         584
Italian Restaurant             447
Bagel Shop                     397
Café                           379
Mexican Restaurant             377
Ice Cream Shop                 332
Sandwich Place                 325
Caribbean Restaurant           322
Fried Chicken Joint            310
American Restaurant            304
```

Data Cleaning

- We are interested in the variation of cuisine types within a neighborhood, so we remove all the venues which have a general food category like f.e. coffee shop, café etc..
- First we collect al unique food categories, then manually we create a file with all 'general' general food categories

```
[ ] # manually create a list of generalized categories
    general_categories = ['Dessert Shop','Food','Ice Cream Shop','Donut Shop','Bakery','Sandwich Place','Comfort Food Restaurant',
                          'Deli / Bodega','Food Truck','Bagel Shop','Burger Joint','Restaurant','Frozen Yogurt Shop','Coffee Shop',
                          'Diner','Wings Joint','Café','Juice Bar','Breakfast Spot','Grocery Store','Bar','Cupcake Shop',
                          'Pub','Fish & Chips Shop','Cafeteria','Other Nightlife','Arcade','Hot Dog Joint','Food Court',
                          'Health Food Store','Convenience Store','Food & Drink Shop','Cocktail Bar','Cheese Shop',
                          'Snack Place','Sports Bar','Lounge','Theme Restaurant','Buffet','Bubble Tea Shop','Building',
                          'Irish Pub','College Cafeteria','Tea Room','Supermarket','Hotpot Restaurant','Gastropub','Beer Garden',
                          'Fish Market','Beer Bar','Clothing Store','Music Venue','Bistro','Salad Place','Wine Bar','Gourmet Shop',
                          'Indie Movie Theater','Art Gallery','Gift Shop','Pie Shop','Fruit & Vegetable Store',
                          'Street Food Gathering','Dive Bar','Factory','Farmers Market','Mac & Cheese Joint','Creperie',
                          'Candy Store','Event Space','Skating Rink','Miscellaneous Shop','Gas Station','Organic Grocery',
                          'Pastry Shop','Club House','Flea Market','Hotel','Furniture / Home Store','Bookstore','Pet Café',
                          'Gym / Fitness Center','Flower Shop','Financial or Legal Service','Hotel Bar','Hookah Bar','Poke Place',
                          'Market','Gluten-free Restaurant','Smoothie Shop','Butcher','Food Stand','Beach Bar','Beach',
                          'Soup Place','Rock Club','Residential Building (Apartment / Condo)','Laundry Service',
                          'Government Building','Bowling Alley','Nightclub','Park','Moving Target','Bike Shop','Beer Store',
                          'Hobby Shop','Chocolate Shop','Food Service','Indoor Play Area','Record Shop','Whisky Bar','Dosa Place']
```

- By subtraction of the 'Full' - and the ''General' food categories, we are able to create a dataframe with the required venues/food categories

Feature Engineering

- One hot encoding converts the categorical variables (which are 'Venue Category') into a
form that could be provided to ML algorithms to do a better job in prediction

```
[ ]  # one hot encoding
     nyc_onehot = pd.get_dummies(nyc_venues[['Venue Category']], prefix="", prefix_sep="")
     nyc_onehot.head()
```

	Afghan Restaurant	African Restaurant	American Restaurant	Arepa Restaurant	Argentinian Restaurant	Asian Restaurant	Australian Restaurant	Austrian Restaurant	BBQ Joint	Brazilian Restaurant	Burrito Place	Cajun / Creole Restaurant	Car Rest
0	0	0	0	0	0	0	0	0	0	0	0	0	
1	0	0	0	0	0	0	0	0	0	0	0	0	
2	0	0	0	0	0	0	0	0	0	0	0	0	
3	0	0	0	0	0	0	0	0	0	0	0	0	
4	0	0	0	0	0	0	0	0	0	0	0	0	

- the size is 6483 records

Feature Engineering

- The top 10 'Venue Categories' by concurrency

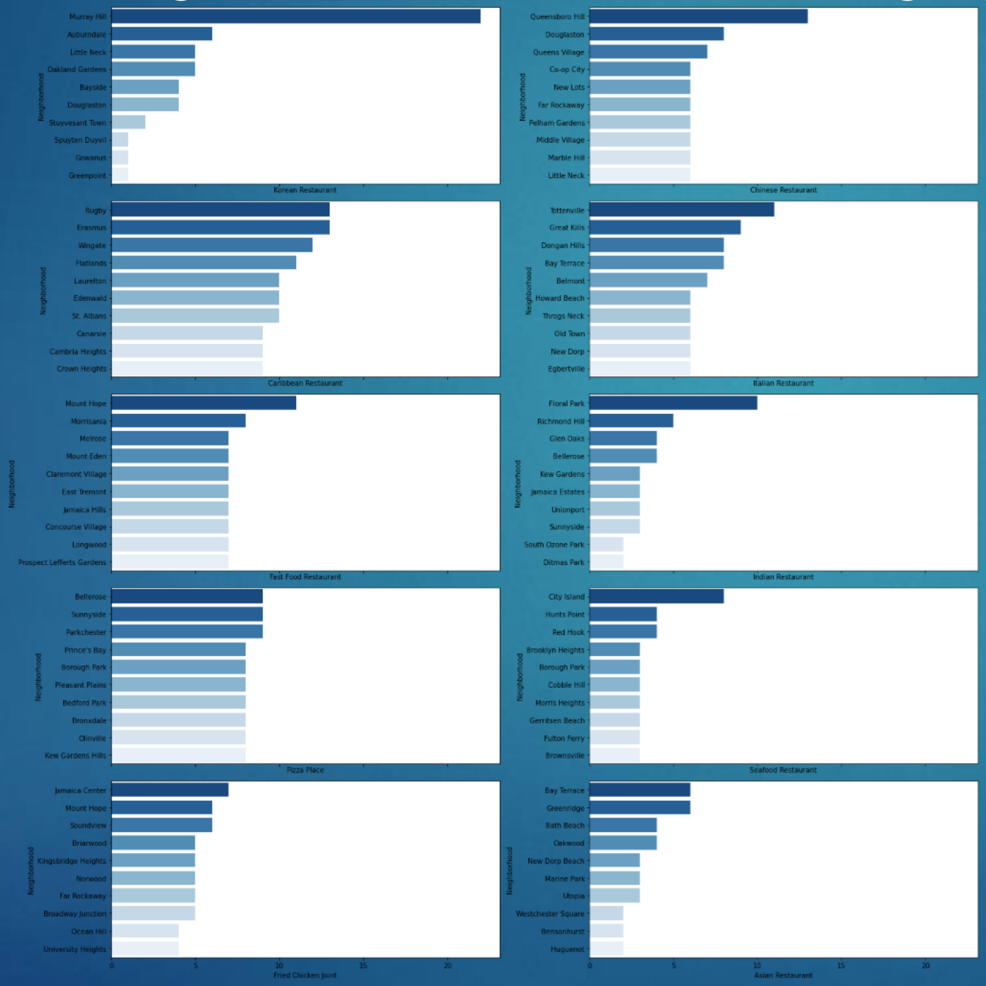
```
[ ] venue_counts_described = venue_counts.describe().transpose()
```

```
[ ] venue_top10 = venue_counts_described.sort_values('max', ascending=False)[0:10]
    venue_top10
```

	count	mean	std	min	25%	50%	75%	max
Korean Restaurant	302.0	0.238411	1.426883	0.0	0.0	0.0	0.0	22.0
Chinese Restaurant	302.0	2.264901	1.887895	0.0	1.0	2.0	3.0	13.0
Caribbean Restaurant	302.0	1.066225	2.393704	0.0	0.0	0.0	1.0	13.0
Italian Restaurant	302.0	1.480132	1.760947	0.0	0.0	1.0	2.0	11.0
Fast Food Restaurant	302.0	1.990066	1.941820	0.0	0.0	2.0	3.0	11.0
Indian Restaurant	302.0	0.311258	0.894319	0.0	0.0	0.0	0.0	10.0
Pizza Place	302.0	3.589404	1.995915	0.0	2.0	3.0	5.0	9.0
Seafood Restaurant	302.0	0.533113	0.906178	0.0	0.0	0.0	1.0	8.0
Fried Chicken Joint	302.0	1.026490	1.314105	0.0	0.0	1.0	2.0	7.0
Asian Restaurant	302.0	0.500000	0.857835	0.0	0.0	0.0	1.0	6.0

Data Visualization

- Top 10 neighborhoods for a food category:



Data Visualization

- A dataframe is created with the top 5 most common venues categories in the neighborhood

```
[ ] for ind in np.arange(nyc_grouped.shape[0]):
        neighborhoods_venues_sorted.iloc[ind, 1:] = return_most_common_venues(nyc_grouped.iloc[ind, :], num_top_venues)

    neighborhoods_venues_sorted.head()
```

	Neighborhood	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue
0	Allerton	Pizza Place	Chinese Restaurant	Mexican Restaurant	Fast Food Restaurant	Fried Chicken Joint
1	Annadale	Pizza Place	American Restaurant	Italian Restaurant	Sushi Restaurant	Japanese Restaurant
2	Arden Heights	Pizza Place	American Restaurant	Italian Restaurant	Mexican Restaurant	Chinese Restaurant
3	Arlington	Pizza Place	Fast Food Restaurant	American Restaurant	Spanish Restaurant	Latin American Restaurant
4	Arrochar	Italian Restaurant	Pizza Place	Latin American Restaurant	Japanese Restaurant	Mediterranean Restaurant

Machine Learning

- ‘k-means’ is an unsupervised machine learning algorithm which creates clusters of data points aggregated together because of certain similarities. This algorithm will be used to count neighborhoods for each cluster label for variable cluster size.
- To implement this algorithm, it is very important to determine the optimal number of clusters (i.e. k). There are 2 most popular methods for the same, namely ‘The Elbow Method’ and ‘The Silhouette Method’

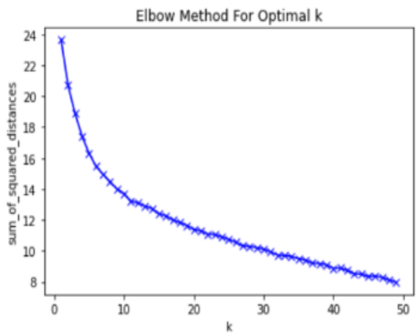
Machine Learning

- **The Elbow Method --> gradual decrease--> cannot be used for optimum**

```
[ ] sum_of_squared_distances = []
    K = range(1,50)
    for k in K:
        print(k, end=' ')
        kmeans = KMeans(n_clusters=k).fit(nyc_grouped_clustering)
        sum_of_squared_distances.append(kmeans.inertia_)
```

```
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49
```

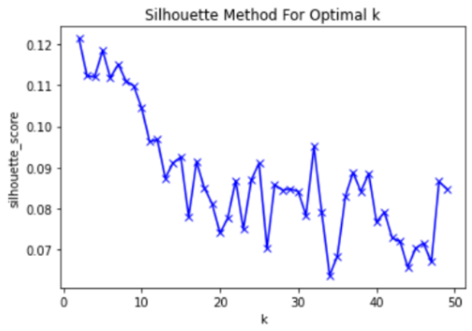
```
[ ] plt.plot(K, sum_of_squared_distances, 'bx-')
    plt.xlabel('k')
    plt.ylabel('sum_of_squared_distances')
    plt.title('Elbow Method For Optimal k');
```



Machine Learning

- **The Silhouette Method**
  - As quoted in Wikipedia — "The Silhouette Method measures how similar a point is to its own cluster (cohesion) compared to other clusters (separation)."

```
[ ] plt.plot(K_sil, sil, 'bx-')
    plt.xlabel('k')
    plt.ylabel('silhouette_score')
    plt.title('Silhouette Method For Optimal k')
    plt.show()
```



There is a peak at k = 2, k = 4 and k = 8. Two and four clusters will give a very broad classification of the venues.

  - There is a peak at k=2 and k=4.
    These would give a very broad classification of the venues, let's use K=8.

Machine Learning

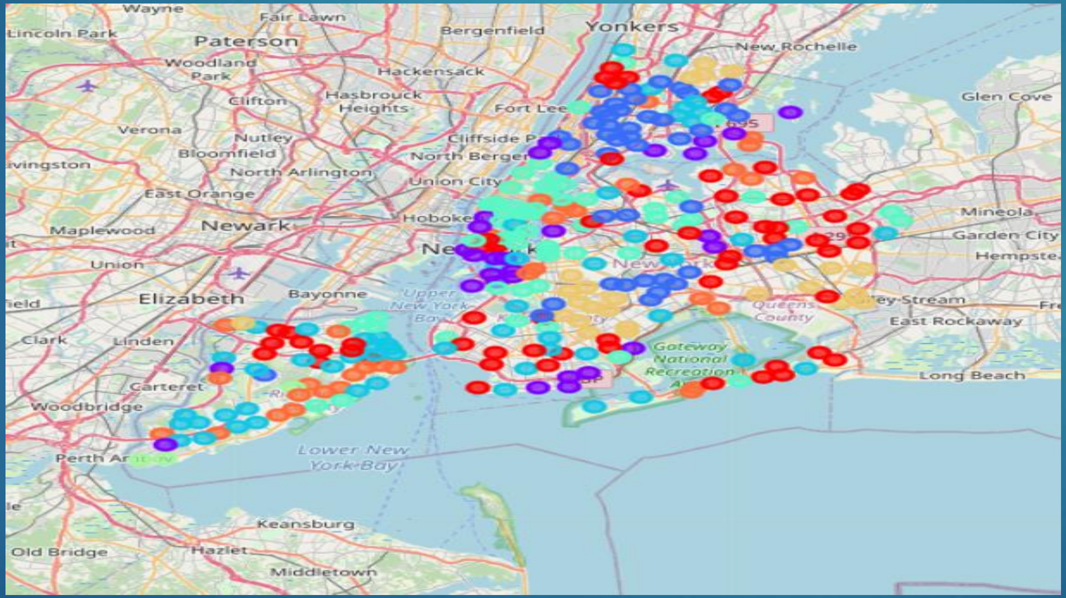
- The cluster labels curated are added to the dataframe to get the desired results of segmenting the neighborhood based upon the most common venues in its vicinity

```
# merge neighborhoods_venues_sorted with nyc_data to add latitude/longitude for each neighborhood
nyc_merged = neighborhoods_venues_sorted.join(neighborhoods.set_index('Neighborhood'), on='Neighborhood')
nyc_merged.head()
```

	Cluster Labels	Neighborhood	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	Borough	Latitude	Longitude
0	2	Allerton	Pizza Place	Chinese Restaurant	Mexican Restaurant	Fried Chicken Joint	Fast Food Restaurant	Bronx	40.865788	-73.859319
1	3	Annadale	Pizza Place	American Restaurant	Italian Restaurant	Sushi Restaurant	Japanese Restaurant	Staten Island	40.538114	-74.178549
2	3	Arden Heights	Pizza Place	American Restaurant	Italian Restaurant	Mexican Restaurant	Chinese Restaurant	Staten Island	40.549286	-74.185887
3	3	Arlington	Pizza Place	American Restaurant	Fast Food Restaurant	Spanish Restaurant	Caribbean Restaurant	Staten Island	40.635325	-74.165104
4	7	Arrochar	Italian Restaurant	Pizza Place	Japanese Restaurant	Polish Restaurant	Latin American Restaurant	Staten Island	40.596313	-74.067124

Machine Learning

- The New York City's neighborhoods are visualized utilizing the python 'folium' library, showing the cluster segmentation of the New York City's neighborhoods:



Results

- Following are the results of the Cluster analysis i.g.:

```
[ ] for col in required_column:
        print(cluster_0[col].value_counts(ascending = False))
        print("---------------------------------------------")
```

```
Chinese Restaurant       27
Pizza Place              26
Sushi Restaurant          2
Korean Restaurant         2
Greek Restaurant          1
American Restaurant       1
Fried Chicken Joint       1
Vietnamese Restaurant     1
Caribbean Restaurant      1
Name: 1st Most Common Venue, dtype: int64
---------------------------------------------
Chinese Restaurant       22
Pizza Place              12
Italian Restaurant        4
Fast Food Restaurant      4
Korean Restaurant         3
Cantonese Restaurant      2
Greek Restaurant          2
Fried Chicken Joint       2
Indian Restaurant         2
Japanese Restaurant       2
Sushi Restaurant          1
Vietnamese Restaurant     1
Russian Restaurant        1
Mexican Restaurant        1
Caribbean Restaurant      1
American Restaurant       1
Taco Place                1
Name: 2nd Most Common Venue, dtype: int64
---------------------------------------------
Queens           30
Brooklyn         11
Staten Island    10
Bronx             6
Manhattan         5
Name: Borough, dtype: int64
---------------------------------------------
```

Conclusion

- Tabulating the results of the k-Mean unsupervised machine learning algorithm

Cluster	1st most common venue	1st most common venue	Borough
0	Chinese restaurant	Pizza Place	Queens
1	Pizza Place, Italian restaurant	American restaurant, Italian restaurant	Brooklyn, Manhattan
2	American restaurant, Pizza Place	American restaurant, Pizza Place	Brooklyn, Bronx
3	Chinese restaurant	Chinese restaurant	Queens
4	Pizza Place	Chinese restaurant	Staten Island
5	Caribbean restaura nt	Chinese restaurant	Brooklyn
6	Caribbean restaurant	Chinese restaurant	Brooklyn, Queens
7	Italian restaurant	Pizza place	Staten island,

Conclusion

- Following could be the name
  of the clusters segmented and curated by k-
  Means unsupervised machine learning algorithm:
  - . Cluster 0 — *Chinese*
  - · Cluster 1 — *Italian*
  - · Cluster 2 — *American*
  - · Cluster 3 — *Chinese*
  - · Cluster 4 — *Pizza Place, Chinese*
  - · Cluster 5 — *Caribbean*
  - · Cluster 6 — *Caribbean*
  - · Cluster 7 — *Italian*