Coursera Capstone Project: Applied Data Science

Opening a Chinese Restaurant in Brooklyn, New York

Bingqing Ge
gebingqing@gmail.com

New York University

1 Introduction

New York is the largest city in the United States with a long history of international immigration. Over the last decade the city has been growing faster than the region. The New York region continues to be by far the leading metropolitan gateway for legal immigrants admitted into the United States. The New York City Metropolitan Area contains the largest ethnic Chinese population outside Asia. In the recent decades, American people especially show more and more interests in Chinese Food. However, when it comes to Chinese food, Americans have a stereotype of cheap, dirty, unhealthy, etc. It's hard to people to associate Chinese restaurants with high-end restaurants.

1.1 Business Problem

Our stakeholder is willing to open a high-end Chinese restaurant in New York City with middle-high level prices. Of course, choosing a location for business is one of the stressful and controversial tasks, since there are a lot of criteria that have to be satisfied in order to achieve the highest revenue. Here are some of them:

- the density of other restaurants
- the density of specifically Chinese restaurants
- population density around the location

- solvency of the population around the location
- Neighborhood HH Income, etc.

In this project, we will implement the basic analysis and try to find the most optimal Borough to open the high-end Chinese restaurant according to those criteria. It's obvious, that there are many additional factors, such as distance from parking places or distance from the main streets, but this analysis can be done after choosing the Borough, and thus will not be done within the scope of this project.

2 Data

2.1 Data Description

Based on criteria listed above the following data will be utilized in our analysis:

- the number of restaurants within the certain radius of each borough (Foresquare API)
- the net income per person in each borough. Since the restaurant will have middle-high prices, it is important to consider the solvency of population. Source: Kaggle (https://www.kaggle.com/goldenoakresearch/us-household-income-stats-geo-locations/version/8#)
- the population and the population density of the borough. Source: NYC Open Data (https://data.cityofnewyork.us/City-Government/New-York-City-Population-By-Neighborhood-Tabulatio/swpk-hqdp)
- the coordinates of the borough. Source: Kaggle (https://www.kaggle.com/goldenoakresearch/us-household-income-stats-geo-locations/version/8#)

2.2 Data Preparation

2.2.1 Get New York Borough Latitude and Longitude data and filter it to Brooklyn only.

```
with open('nyu-2451-34572-geojson.json') as json_data:
    newyork_data = json.load(json_data)
neighborhoods_data = newyork_data['features']
column_names = ['Borough', 'Neighborhood', 'Latitude', 'Longitude'] # define the dataframe columns
neighborhoods = pd.DataFrame(columns=column_names)# create the dataframe
for i in range(len(neighborhoods_data)): #Iterate through each entry in the JSON file
    borough = neighborhoods_data[i]['properties']['borough'] #Boroughs first
    neighborhood_name = neighborhoods_data[i]['properties']['name'] #Followed by Neighborhood name
    neighborhood_lat = neighborhoods_data[i]['properties']['bbox'][1] #Now latitude
    neighborhood_lon = neighborhoods_data[i]['properties']['bbox'][2] #And longitude

    neighborhoods = neighborhoods.append({'Borough': borough,
                                          'Neighborhood': neighborhood_name,
                                          'Latitude': neighborhood_lat,
                                          'Longitude': neighborhood_lon}, ignore_index=True)

neighborhoods.head()
```

```
mask = neighborhoods['Borough'] == 'Brooklyn' #Make a boolean mask for Brooklyn Borough
df = neighborhoods.copy() #Make a copy to leave our original DF intact
df = df[mask] #Now use the mask to eliminate entries not in Manhattan
df.reset_index(inplace = True) #Now reset the index so we can use it
df.drop('index', axis = 1, inplace = True)
df.head() #And take a look at our work so far
```

We also use geopy library to get the central latitude and longitude values of Brooklyn

```
address = 'Brooklyn, NY'

geolocator = Nominatim(user_agent="bk_explorer")
location = geolocator.geocode(address)
latitude = location.latitude
longitude = location.longitude
print('The geograpical coordinate of Brooklyn are {}, {}.'.format(latitude, longitude))
```

```
The geograpical coordinate of Brooklyn are 40.6501038, -73.9495823.
```

2.2.2 Use Foursquare to get all the venues in Brooklyn

```
def getNearbyVenues(names, latitudes, longitudes, radius=500):

    venues_list=[]
    for name, lat, lng in zip(names, latitudes, longitudes):
        print(name)

        # create the API request URL
        url = 'https://api.foursquare.com/v2/venues/explore?&client_id={}&client_secret={}&v={}&ll
={},{}&radius={}&limit={}'.format(
            CLIENT_ID,
            CLIENT_SECRET,
            VERSION,
            lat,
            lng,
            radius,
            LIMIT)

        # make the GET request
        results = requests.get(url).json()["response"]['groups'][0]['items']
        # return only relevant information for each nearby venue
        venues_list.append([(
            name,
            lat,
            lng,
            v['venue']['name'],
            v['venue']['location']['lat'],
            v['venue']['location']['lng'],
            v['venue']['categories'][0]['name'],
            v['venue']['id']) for v in results])

    nearby_venues = pd.DataFrame([item for venues_list in venues_list for item in venues_list])
    nearby_venues.columns = ['Neighborhood',
                  'Neighborhood Latitude',
                  'Neighborhood Longitude',
                  'Venue',
                  'Venue Latitude',
                  'Venue Longitude',
                  'Venue Category',
                  'Venue ID'
                            ]

    return(nearby_venues)
```

```
#Use the function to pull venues near each neighborhood
LIMIT = 20
brooklyn_venues = getNearbyVenues(names=df['Neighborhood'],
                                  latitudes=df['Latitude'],
                                  longitudes=df['Longitude']
                                 )
```

We pulled venues acrossing 233 unique categories from Foursquare. However, since our research is mainly focusing on Restaruant, we want to make sure the categories we pulled are limited to restaruants or food related categories.

```
matching = [s for s in list if "Restaurant" in s] + [s for s in list if "Bar" in s] + [s for s in
list if "Food" in s] + [s for s in list if "Dessert" in s]
matching
```

We converted the Venue category to dummy data and aggregated by neighborhood.

```
#Start the clustering analysis with one-hot encoding for pre-processing
# one hot encoding
brooklyn_onehot = pd.get_dummies(brooklyn_venues_new[['Venue Category']], prefix="", prefix_sep
="")

# add neighborhood column back to dataframe
brooklyn_onehot['Neighborhood'] = brooklyn_venues_new['Neighborhood']

# move neighborhood column to the first column
fixed_columns = [brooklyn_onehot.columns[-1]] + brooklyn_onehot.columns[:-1].tolist()
brooklyn_onehot = brooklyn_onehot[fixed_columns]

brooklyn_onehot.shape
```

```
brooklyn_grouped = brooklyn_onehot.groupby('Neighborhood').mean().reset_index()
brooklyn_grouped.head()
```

We can then further get more details of the venues such as likes and tips

```
def getDetails(name, VENUE_ID):

    restaurant_venues=[]
    for name, VENUE_ID in zip(name, VENUE_ID):
        print(name)

        # create the API request URL
        url = 'https://api.foursquare.com/v2/venues/{}?&client_id={}&client_secret={}&v={}'.forma
t(
            VENUE_ID,
            CLIENT_ID,
            CLIENT_SECRET,
            VERSION,
                )

        # make the GET request
        results = requests.get(url).json()

        # return only relevant information for each nearby venue
        restaurant_venues.append(results)

    return(restaurant_venues)

    #for i in range(len(details))
    #popularTimes = pd.DataFrame([item for popularHours in popularHours for item in popularHour
s])
    #popularTimes.columns = ['timeframes',

restaurant_venues = getDetails(name=brooklyn_venues_new['Venue'],
                                   VENUE_ID=brooklyn_venues_new['Venue ID'],
                                  )
```

2.2.3 Import the data from other sources we collected such as NYC Open data or Kaggle.

The data include Median Income, Population, Race, etc. We then merge those data with the Foursquare data set for our model. Below is an example of the final data set we will use:

Neighborhood	Median Income	Population	Asian	AA	Hispanic	White	Neighborhood Latitude	Neighborhood Longitude	Venue Latitude	Venue Longitude	Likes	Tips	American Restaurant
Bay Ridge	69989.0	123488	0.282	0.031	0.158	0.50900	40.625801	-74.030621	40.624251	-74.030346	40.625000	40.625000	0.000000
Bensonhurst	54513.0	205850	0.389	0.018	0.155	0.40400	40.611009	-73.995180	40.612326	-73.996229	11.750000	11.750000	0.000000
Sunset Park	57870.0	144332	0.327	0.024	0.399	0.22700	40.645103	-74.010316	40.646869	-74.009024	37.166667	37.166667	0.000000
Greenpoint	78069.0	152002	0.070	0.036	0.216	0.64500	40.730201	-73.954241	40.730615	-73.955152	176.666667	176.666667	0.000000
Sheepshead Bay	62443.0	171030	0.052	0.080	0.637	0.62443	40.586890	-73.943186	40.584168	-73.944172	62.909091	62.909091	0.090909
Flatbush	57678.0	150707	0.088	0.289	0.150	0.44200	40.636326	-73.958401	40.635423	-73.961846	9.142857	9.142857	0.000000
Crown Heights	61253.0	141725	0.056	0.557	0.117	0.24100	40.670829	-73.943291	40.670715	-73.944628	5.000000	5.000000	0.000000
East Flatbush	50290.0	140087	0.021	0.871	0.062	0.02700	40.641718	-73.936103	40.641618	-73.936250	10.000000	10.000000	0.000000
Prospect Heights	61253.0	141725	0.056	0.557	0.117	0.24100	40.676822	-73.964859	40.676587	-73.963800	177.900000	177.900000	0.000000
Brownsville	20640.0	111511	0.019	0.713	0.198	0.04300	40.663950	-73.910235	40.663760	-73.911446	1.600000	1.600000	0.000000
Williamsburg	78069.0	152002	0.070	0.036	0.216	0.64500	40.707144	-73.958115	40.709014	-73.956971	305.875000	305.875000	0.000000
Bushwick	51622.0	140474	0.056	0.170	0.539	0.21500	40.698116	-73.925258	40.699513	-73.926408	94.100000	94.100000	0.000000
Bedford Stuyvesant	52897.0	142027	0.027	0.488	0.194	0.26600	40.687232	-73.941785	40.684215	-73.943584	95.142857	95.142857	0.000000
Brooklyn Heights	94327.0	135444	0.090	0.258	0.149	0.47200	40.695864	-73.993782	40.695981	-73.993911	74.800000	74.800000	0.000000
Carroll Gardens	137375.0	116209	0.072	0.061	0.172	0.63900	40.680540	-73.994654	40.680919	-73.994313	192.142857	192.142857	0.000000
Fort Greene	94327.0	135444	0.090	0.258	0.149	0.47200	40.688527	-73.972906	40.689098	-73.971816	377.333333	377.333333	0.000000
Park Slope	137375.0	116209	0.072	0.061	0.172	0.63900	40.672321	-73.977050	40.670396	-73.978731	128.000000	128.000000	0.000000

```
nycdata_new.describe()
```

	Median Income	Population	Asian	AA	Hispanic	White	Neighborhood Latitude	Neighborhood Longitude	Venue Latitude	Venue Longitude	Likes	Tips
count	23.000000	23.000000	23.000000	23.000000	23.000000	23.000000	23.000000	23.000000	23.000000	23.000000	23.000000	23.000000
mean	67338.173913	148251.217391	0.098043	0.305435	0.207087	0.379845	40.657399	-73.958955	40.657465	-73.959119	80.495840	80.495840
std	28293.484119	30360.227639	0.099452	0.275905	0.145533	0.215230	0.038109	0.036380	0.038217	0.036390	103.319799	103.319799
min	20640.000000	111511.000000	0.018000	0.018000	0.062000	0.027000	40.574293	-74.030621	40.575665	-74.030346	0.500000	0.500000
25%	51834.000000	129466.000000	0.047000	0.048500	0.118500	0.225500	40.634347	-73.989591	40.635324	-73.988546	7.071429	7.071429
50%	61253.000000	141725.000000	0.070000	0.258000	0.158000	0.404000	40.663950	-73.958115	40.663760	-73.956971	37.166667	37.166667
75%	78089.000000	152002.000000	0.090000	0.557000	0.207000	0.589215	40.683886	-73.938944	40.682567	-73.939917	111.571429	111.571429
max	137375.000000	215637.000000	0.389000	0.871000	0.637000	0.689000	40.730201	-73.880699	40.730615	-73.879535	377.333333	377.333333

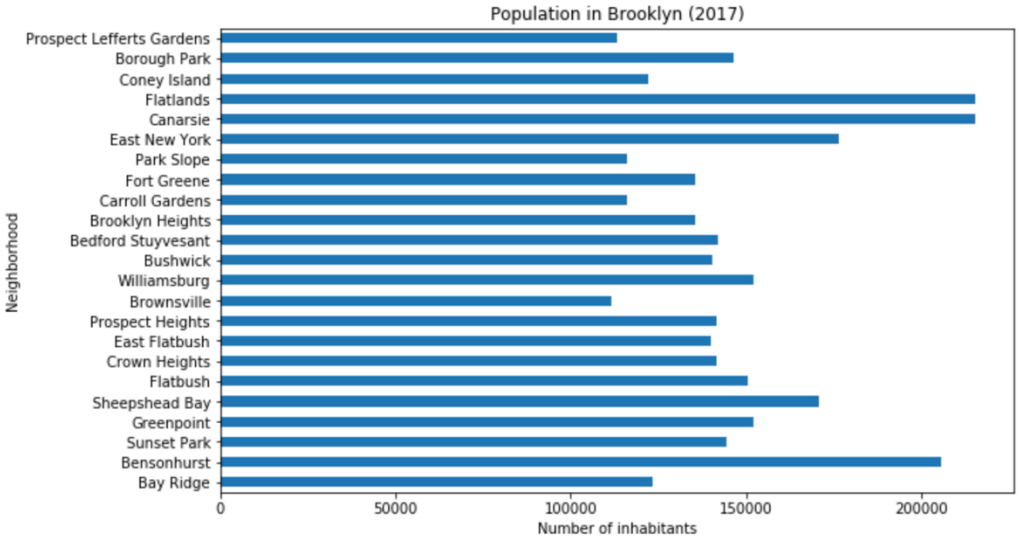
3 Methodology and Analysis

After cleaning and preparing the data, let us identify the steps, that have to be performed in order to find the most optimal neighborhood. Firstly, we will apply some basic exploratory analysis to our data. For that let's find the location of each neighborhood on the map. Then we can visually inspect some values in our data with the help of bar charts. Secondly, we have the possibility to reduce the number features in data frame by replacing them with more reasonable data. Finally,

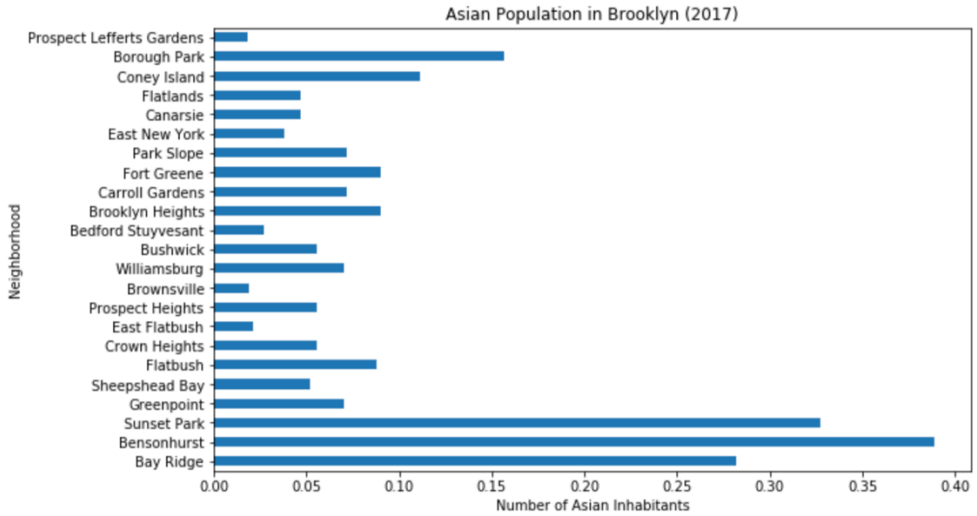
we will perform cluster analysis to find the best cluster of neighborhoods with meaningful features.

3.1 Exploratory Data Analysis

3.1.1 Understand the income and population.

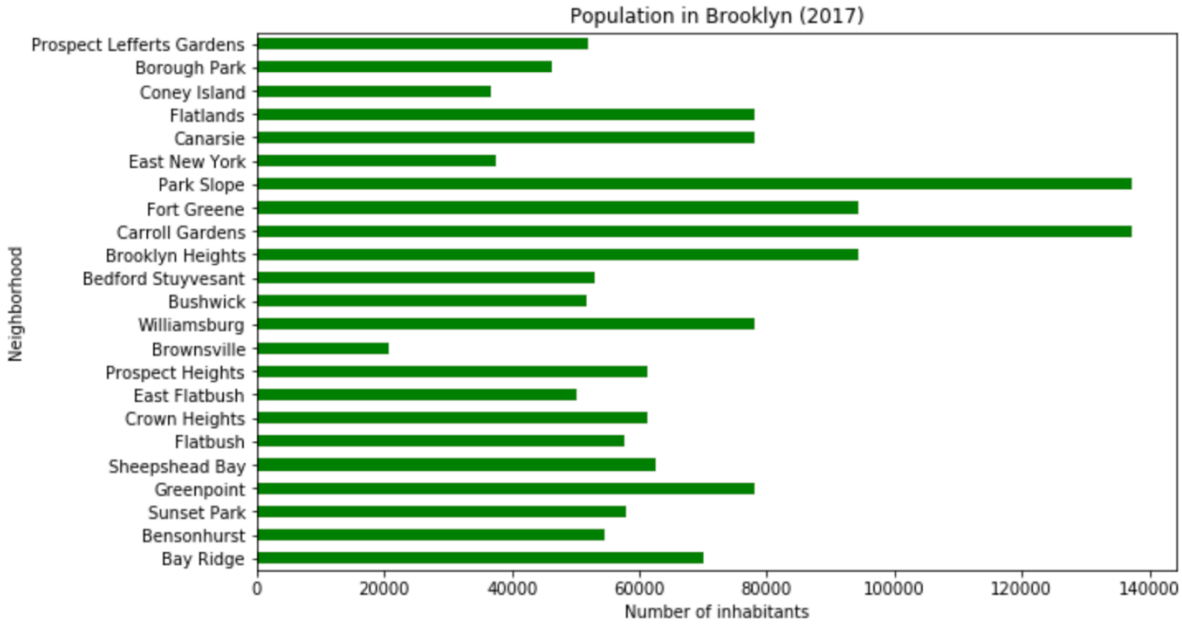


We can tell from the above plot that, the neighborhoods have most population would be Flatlands, Canarsie, Bensonhurst, etc. Now we can look at which Neighborhood would have most Asian Population



Seems like most of the Asian people live in Sunset Park, Bensonhurst and Bay Ridge which match our previous background research.

We also look at the incomes in the neighborhoods.



The neighborhoods have the higher median income are Park Slope and Carroll Garden.

3.1.2 Map out the Neighborhoods

3.2 Cluster Analysis

In order to identify groups (clusters) with similar characteristics, let's us apply the unsupervised learning method to our data, namely K-Means algorithm. But before that, we can reduce the number of features by looking at the variance of each variable. Low variance means there's no difference among different neighborhood, so we should not include those noise in our model.

3.2.1 Normalize the data before the model

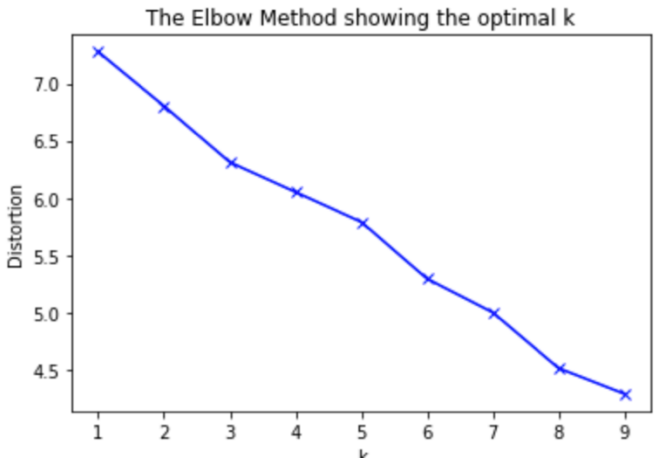
```
from sklearn.preprocessing import StandardScaler
nycdata_to_model = StandardScaler().fit_transform(nycdata_to_model)
nycdata_to_model
```

3.2.2 Identify the optimal K

```
from sklearn.cluster import KMeans
from sklearn import metrics
from scipy.spatial.distance import cdist
import numpy as np


distortions = []
K = range(1,10)
for k in K:
    kmeanModel = KMeans(n_clusters=k).fit(nycdata_to_model)
    kmeanModel.fit(nycdata_to_model)
    distortions.append(sum(np.min(cdist(nycdata_to_model, kmeanModel.cluster_centers_, 'euclidean'), axis=1)) / nycdata

# Plot the elbow
plt.plot(K, distortions, 'bx-')
plt.xlabel('k')
plt.ylabel('Distortion')
plt.title('The Elbow Method showing the optimal k')
plt.show()
```



Since like the optimal K is 3.

3.2.3 Run K-Mean clustering

```
# set number of clusters
kclusters = 3

# run k-means clustering
kmeans = KMeans(n_clusters=kclusters, random_state=0).fit(nycdata_to_model)

# check cluster labels generated for each row in the dataframe
kmeans.labels_[0:22]
```

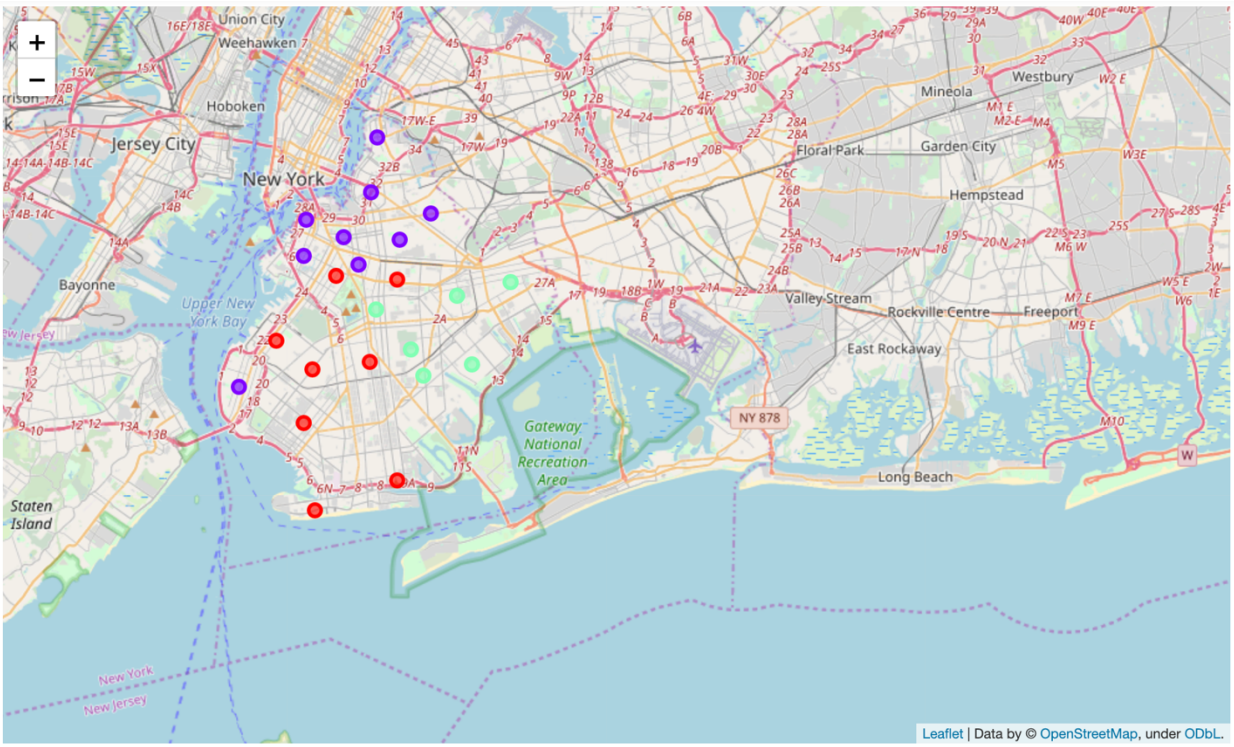
3.2.4 Map the cluster

```
# create map
map_clusters = folium.Map(location=[latitude, longitude], zoom_start=11)

# set color scheme for the clusters
x = np.arange(kclusters)
ys = [i + x + (i*x)**2 for i in range(kclusters)]
colors_array = cm.rainbow(np.linspace(0, 1, len(ys)))
rainbow = [colors.rgb2hex(i) for i in colors_array]

# add markers to the map
markers_colors = []
for lat, lon, poi, cluster in zip(nycdata_new['Neighborhood Latitude'], nycdata_new['Neighborhood Longitude'], nycdata_
    label = folium.Popup(str(poi) + ' Cluster ' + str(cluster), parse_html=True)
    folium.CircleMarker(
        [lat, lon],
        radius=5,
        popup=label,
        color=rainbow[cluster-1],
        fill=True,
        fill_color=rainbow[cluster-1],
        fill_opacity=0.7).add_to(map_clusters)

map_clusters
```



3.2.5 Understand the cluster

Now we can examine each cluster and determine the discriminating venue categories that distinguish each cluster. Based on the defining categories, we can then assign a name to each cluster.

```
nycdata_new[(nycdata_new['Cluster Labels'] == 0)][['Median Income','Population','Asian','Likes','Tips','Number of resta
```

```
Median Income             64271.000000
Population               149802.250000
Asian                         0.156500
Likes                        32.433577
Tips                         32.433577
Number of restaurants         4.857143
dtype: float64
```

```
cdata_new['Cluster Labels'] == 1)][['Median Income','Population','Asian','Likes','Tips','Number of restaurants']].mean(
```

```
Median Income             79769.777778
Population               137646.111111
Asian                         0.090333
Likes                       170.509524
Tips                        170.509524
Number of restaurants         5.111111
dtype: float64
```

```
a_new[(nycdata_new['Cluster Labels'] == 2)][['Median Income','Population','Asian','Likes','Tips','Number of restaurants
```

```
Median Income             52780.333333
Population               162090.833333
Asian                         0.031667
Likes                         9.558333
Tips                          9.558333
Number of restaurants         4.333333
dtype: float64
```

We can see from above:

- Cluster 0 have the 2nd most of people with the 2nd highest income and the most Asian population, and 2nd least restaurants.
- Cluster 1 have the least of people with the highest income and the 2nd most Asian population, most restaurants, and most likes and tips.
- Cluster 2 have the most of people with lowest income and the least Asian population, and least restaruants.

Based on the results above, we should aim for Cluster 1 since we want to target a high income neighborbood.

Now, lets take a look at the restaurants belong to cluster 1

Neighborhood	Cluster Labels	Neighborhood	Median Income	Population	Asian	AA	Hispanic	White	Neighborhood Latitude	Neighborhood Longitude	Venue Latitude	Venue Longitude	Likes	
Bay Ridge	1	Bay Ridge	69989.0	123488	0.282	0.031	0.158	0.509	40.625801	-74.030621	40.624251	-74.030346	40.625000	
Greenpoint	1	Greenpoint	78069.0	152002	0.070	0.036	0.216	0.645	40.730201	-73.954241	40.730615	-73.955152	176.666667	1
Prospect Heights	1	Prospect Heights	61253.0	141725	0.056	0.557	0.117	0.241	40.676822	-73.964859	40.676587	-73.963800	177.900000	1
Williamsburg	1	Williamsburg	78069.0	152002	0.070	0.036	0.216	0.645	40.707144	-73.958115	40.709014	-73.956971	305.875000	3
Bushwick	1	Bushwick	51622.0	140474	0.056	0.170	0.539	0.215	40.698116	-73.925258	40.699513	-73.926408	94.100000	
Bedford Stuyvesant	1	Bedford Stuyvesant	52897.0	142027	0.027	0.488	0.194	0.266	40.687232	-73.941785	40.684215	-73.943584	95.142857	
Brooklyn Heights	1	Brooklyn Heights	94327.0	135444	0.090	0.258	0.149	0.472	40.695864	-73.993782	40.695981	-73.993911	74.800000	
Carroll Gardens	1	Carroll Gardens	137375.0	116209	0.072	0.061	0.172	0.639	40.680540	-73.994654	40.680919	-73.994313	192.142857	1
Fort Greene	1	Fort Greene	94327.0	135444	0.090	0.258	0.149	0.472	40.688527	-73.972906	40.689098	-73.971816	377.333333	3

We have 9 Neighborhood in cluster 1. Among then, if we sort by Median Income, the top neighborhoods are:

- Carroll Gardens
- Brooklyn Heights
- Fort Greene
- Greenpoint
- Williamsburg

How about Asian Population

- Bay Ridge
- Brooklyn Heights
- Fort Greene
- Carroll Gardens
- Greenpoint

How about population

- Greenpoint
- Williamsburg

- Bedford Stuyvesant
- Prospect Heights
- Bushwich

If we find the neighborhood shows up in all above conditions, Greenpoint would be our choice!

4 Result and discussion

During the analysis, three clusters were defined. If we look at the locations on the map, neighborhoods in each cluster tend to clustering together.

When takeing a detailed look at the results, Cluster 0 have the 2nd most of people with the 2nd highest income and the most Asian population, and 2nd least restaurants. Cluster 1 have the least of people with the highest income and the 2nd most Asian population, most restaurants, and most likes and tips. Cluster 2 have the most of people with lowest income and the least Asian population, and least restaruants.

Since our stakeholders want to open a high-end restaurant, we want to find the cluster having the most income and that would be the cluster 1. Then when looking at cluster 1, we want to find the neighborhood that have the most income, most population (Asian) and most overall population. The neighborhood match all requirements is Greenpoint. However, one can perform further analysis of this particular cluster with additional features, such as distance to the center of city or to the center of cluster. After defining a Neighborhood one can perform deeper analysis to find the best exact location of the restaurant taking into account factors such as number of parking places in the vicinity of the spot or distances to the main streets.

What could be done better: Foursquare doesn't represent the full picture, since many venues are not on the list. For that reason, another maps could be utilized such as Google map or Openstreet

map. Neighborhoods have too complex geometry, thus defining the closest venues within the certain radius brings additional error to our analysis.

5 Conclusion

To conclude, the basic data analysis was performed to identify the most optimal Neighborhood in Brooklyn for the placement of the high-end Chinese restaurant. During the analysis, several important statistical features of the Neighborhoods were explored and visualized. Furthermore, clustering helped to highlight the group of optimal areas. Finally, Greenpoint was chosen as the most attractive options for the further analysis.

Reference

[1] Demographics of New York City

[2] American Chinese cuisine

[3] Forsquare API

[4] US household income stats by geolocations

[5] NYC population density