Capstone Project - The Battle of the Neighborhoods (Week 2)

Applied Data Science Capstone by IBM/Coursera

Harrison Osuorji

Table of contents

Introduction: Business Problem

New York City revolves around coffee. Considering there are 3,389 coffee shops in NYC. New York was one of the largest coffee roasting centers in the United States. More recently, mobile startup MassiveHealth found that New Yorkers drink 6.7 times the amount of coffee consumed by the average denizen of any other US city. And because New York loves Coffee so much, this makes it one of the best place to open a coffee shop.

We want to open a coffee shop in the New York City New York. i want to see what area has the most foot tracffic and to see if my coffee business will be profitable

Data

Based on definition of our problem, factors that will influence our decission are:

- number of existing coffee shot in the neighborhood (any type of Coffee shop)
- number of and distance to Coffee Shops in the neighborhood, if any
- distance of neighborhood from city center

We decided to use regularly spaced grid of locations, centered around city center, to define our neighborhoods.

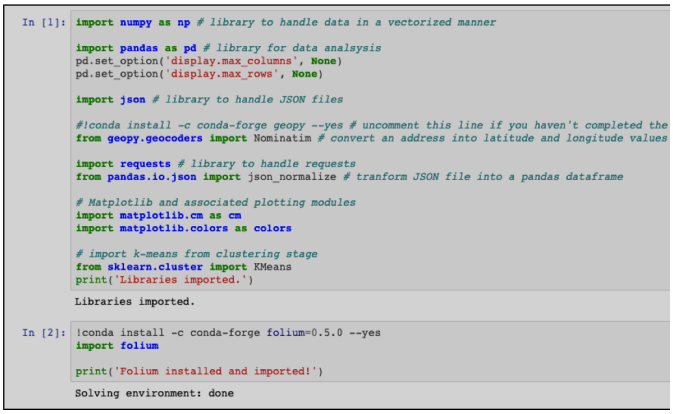
Following data sources will be needed to extract/generate the required information:

- centers of candidate areas will be generated algorithmically and approximate addresses of centers of those areas will be obtained using Google Maps API reverse geocoding
- number of Coffee Shop and their type and location in every neighborhood will be obtained using Foursquare API
- we will also going to Run k-means to cluster the neighborhood into 5 clusters

Neighborhood Candidates

Let's create latitude & longitude coordinates for centroids of our candidate neighborhoods. We will create a grid of cells covering our area of interest which is aprox. 12x12 killometers centered around New York City center.

Let's import all the Librarys we will need.



```
In [1]: import numpy as np # library to handle data in a vectorized manner

        import pandas as pd # library for data analsysis
        pd.set_option('display.max_columns', None)
        pd.set_option('display.max_rows', None)

        import json # library to handle JSON files

        #!conda install -c conda-forge geopy --yes # uncomment this line if you haven't completed the
        from geopy.geocoders import Nominatim # convert an address into latitude and longitude values

        import requests # library to handle requests
        from pandas.io.json import json_normalize # tranform JSON file into a pandas dataframe

        # Matplotlib and associated plotting modules
        import matplotlib.cm as cm
        import matplotlib.colors as colors

        # import k-means from clustering stage
        from sklearn.cluster import KMeans
        print('Libraries imported.')

        Libraries imported.

In [2]: !conda install -c conda-forge folium=0.5.0 --yes
        import folium

        print('Folium installed and imported!')

        Solving environment: done
```

Tranform the data into a pandas dataframe

The next task is essentially transforming this data of nested Python dictionaries into a pandas dataframe. So let's start by creating an empty dataframe.

In [8]:
```
# define the dataframe columns
column_names = ['Borough', 'Neighborhood', 'Latitude', 'Longitude']

# instantiate the dataframe
neighborhoods = pd.DataFrame(columns=column_names)
```

Take a look at the empty dataframe to confirm that the columns are as intended.

In [9]:
```
neighborhoods
```

Out[9]:

	Borough	Neighborhood	Latitude	Longitude

Then let's loop through the data and fill the dataframe one row at a time.

In [10]:
```
for data in neighborhoods_data:
    borough = neighborhood_name = data['properties']['borough']
    neighborhood_name = data['properties']['name']

    neighborhood_latlon = data['geometry']['coordinates']
    neighborhood_lat = neighborhood_latlon[1]
    neighborhood_lon = neighborhood_latlon[0]

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

Quickly examine the resulting dataframe.

In [11]:
```
neighborhoods.head()
```

Out[11]:

	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

And make sure that the dataset has all 5 boroughs and 306 neighborhoods.

In [12]:
```
print('The dataframe has {} boroughs and {} neighborhoods.'.format(
        len(neighborhoods['Borough'].unique()),
        neighborhoods.shape[0]
    )
)
```

```
The dataframe has 5 boroughs and 306 neighborhoods.
```

Use geopy library to get the latitude and longitude values of New York City

In order to define an instance of the geocoder, we need to define a user_agent. We will name our agent ny_explorer, as shown below

In [13]:
```
address = 'New York City, NY'

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

```
The geograpical coordinate of New York City are 40.7127281, -74.0060152.
```

Create a map of New York with neighborhoods superimposed on top.

In [14]:
```
# create map of New York using latitude and longitude values
map_newyork = folium.Map(location=[latitude, longitude], zoom_start=10)

# add markers to map
for lat, lng, borough, neighborhood in zip(neighborhoods['Latitude'], neighborhoods['Longitude'], neighborhoods['Borou
gh'], neighborhoods['Neighborhood']):
    label = '{}, {}'.format(neighborhood, borough)
    label = folium.Popup(label, parse_html=True)
    folium.CircleMarker(
        [lat, lng],
        radius=5,
        popup=label,
        color='blue',
        fill=True,
        fill_color='#3186cc',
        fill_opacity=0.7,
        parse_html=False).add_to(map_newyork)

map_newyork
```

ıt[14]:


Oakland
Wyckoff
Passaic County
Ridgewood
Glen Rock
Wayne
Fair Lawn
Lincoln Park
Montville
Paterson
Bergenfield
Yonkers
Dobbs Ferry
White Plains
Greenwich
Port Chester
Westwood
Eastchester
Harrison
New Rochelle
Bayville
Denville
Dover
Woodland Park
Hackensack
Parsippany-Troy Hills
Clifton
Hasbrouck Heights
Fort Lee
Glen Cove
Oyster Bay
Huntington
Verona
Nutley
Cliffside Park
Hanover
Bloomfield
North Bergen
Morristown
Livingston
North Arlington
Union City
NY 135
Madison
East Orange
Hoboken
Mineola
Maplewood
Newark
Garden City
Summit
Hempstead
West Babylon
Union
Berkeley Heights
Bayonne
Lindenhurst
Westfield
Elizabeth
Upper New York Bay
Queens County
Freeport
East Rockaway
Plainfield
Clark
Linden
Gateway National Recreation
Green Brook
South Plainfield
Long Beach
Carteret
Woodbridge
Manville
Piscataway
Lower New York Bay
New Brunswick
Sayreville
Keansburg
East Brunswick
Leaflet

However, for illustration purposes, let's simplify the above map and segment and cluster only the neighborhoods in Manhattan. So let's slice the original dataframe and create a new dataframe of the Manhattan data.

In [15]:
```
manhattan_data = neighborhoods[neighborhoods['Boroug
manhattan_data.head()
```

Out[15]:

	Borough	Neighborhood	Latitude	Longitude
0	Manhattan	Marble Hill	40.876551	-73.910660
1	Manhattan	Chinatown	40.715618	-73.994279
2	Manhattan	Washington Heights	40.851903	-73.936900
3	Manhattan	Inwood	40.867684	-73.921210
4	Manhattan	Hamilton Heights	40.823604	-73.949688

Let's get the geographical coordinates of Manhattan.

In [16]:
```
address = 'Manhattan, NY'

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

```
The geograpical coordinate of Manhattan are 40.7896239, -73.9598939.
```

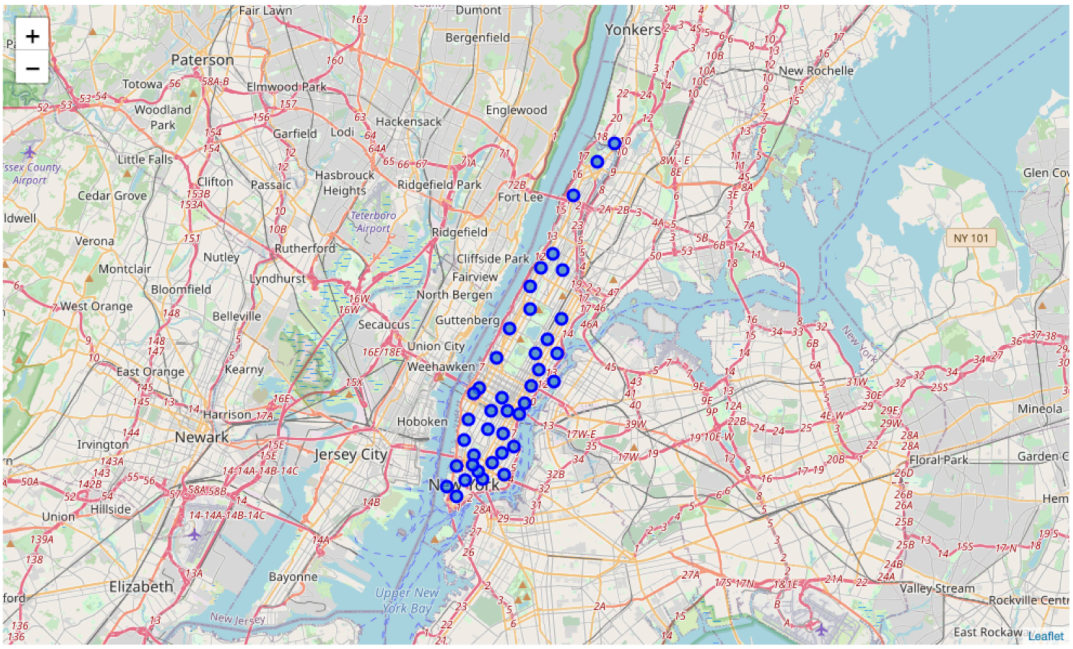
As we did with all of New York City, let's visualize Manhattan the neighborhoods in it.

In [17]:
```
# create map of Manhattan using latitude and longitude values
map_manhattan = folium.Map(location=[latitude, longitude], zoom_start=11)

# add markers to map
for lat, lng, label in zip(manhattan_data['Latitude'], manhattan_data['Longitude'], manhattan_data['Neighborhood']):
    label = folium.Popup(label, parse_html=True)
    folium.CircleMarker(
        [lat, lng],
        radius=5,
        popup=label,
        color='blue',
        fill=True,
        fill_color='#3186cc',
        fill_opacity=0.7,
        parse_html=False).add_to(map_manhattan)

map_manhattan
```

ut[17]:


Fair Lawn
Dumont
Bergenfield
Yonkers
Paterson
New Rochelle
Totowa
Elmwood Park
Woodland Park
Hackensack
Englewood
Garfield
Lodi
Little Falls
Clifton
Passaic
Hasbrouck Heights
Ridgefield Park
Fort Lee
Cedar Grove
Teterboro Airport
Ridgefield
Verona
Nutley
Rutherford
Cliffside Park
Montclair
Lyndhurst
Fairview
Bloomfield
North Bergen
West Orange
Belleville
Secaucus
Guttenberg
Union City
East Orange
Kearny
Weehawken
Harrison
Hoboken
Newark
Irvington
Jersey City
New York
Union
Hillside
Elizabeth
Bayonne
Upper New York Bay
New Jersey
Mineola
Floral Park
Valley Stream
Rockville Centre
East Rockaway
NY 101
Leaflet

Next, we are going to start utilizing the Foursquare API to explore the neighborhoods and segment them.

Define Foursquare Credentials and Version

Let's explore the first neighborhood in our dataframe

Get the neighborhood's name.

```
In [19]: manhattan_data.loc[0, 'Neighborhood']
```

```
Out[19]: 'Marble Hill'
```

Get the neighborhood's latitude and longitude values.

```
In [20]: neighborhood_latitude = manhattan_data.loc[0, 'Latitude'] # neighborhood latitude value
         neighborhood_longitude = manhattan_data.loc[0, 'Longitude'] # neighborhood longitude value

         neighborhood_name = manhattan_data.loc[0, 'Neighborhood'] # neighborhood name

         print('Latitude and longitude values of {} are {}, {}.'.format(neighborhood_name,
                                                                        neighborhood_latitude,
                                                                        neighborhood_longitude))
```

```
Latitude and longitude values of Marble Hill are 40.87655077879964, -73.91065965862981.
```

From the Foursquare lab in the previous module, we know that all the information is in the items key. Before we proceed, let's borrow the get_category_type function from the Foursquare lab.

In [23]:
```
# function that extracts the category of the venue
def get_category_type(row):
    try:
        categories_list = row['categories']
    except:
        categories_list = row['venue.categories']

    if len(categories_list) == 0:
        return None
    else:
        return categories_list[0]['name']
```

Now we are ready to clean the json and structure it into a pandas dataframe.

In [24]:
```
venues = results['response']['groups'][0]['items']

nearby_venues = json_normalize(venues) # flatten JSON

# filter columns
filtered_columns = ['venue.name', 'venue.categories', 'venue.location.lat', 'venue.location.lng']
nearby_venues =nearby_venues.loc[:, filtered_columns]

# filter the category for each row
nearby_venues['venue.categories'] = nearby_venues.apply(get_category_type, axis=1)

# clean columns
nearby_venues.columns = [col.split(".")[-1] for col in nearby_venues.columns]

nearby_venues.head()
```

Out[24]:

	name	categories	lat	lng
0	Arturo's	Pizza Place	40.874412	-73.910271
1	Bikram Yoga	Yoga Studio	40.876844	-73.906204
2	Tibbett Diner	Diner	40.880404	-73.908937
3	Starbucks	Coffee Shop	40.877531	-73.905582
4	Dunkin'	Donut Shop	40.877136	-73.906666

Explore Neighborhoods in Manhattan

Let's create a function to repeat the same process to all the neighborhoods in Manhattan

In [27]:
```
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={}&lim
it={}'.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']) for v in results])
```

```
    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']

    return(nearby_venues)
```

Now write the code to run the above function on each neighborhood and create a new dataframe called manhattan_venues

In [28]:
```
manhattan_venues = getNearbyVenues(names=manhattan_data['Neighborhood'],
                                   latitudes=manhattan_data['Latitude'],
                                   longitudes=manhattan_data['Longitude']
                                  )
```

In [28]:

```
Marble Hill
Chinatown
Washington Heights
Inwood
Hamilton Heights
Manhattanville
Central Harlem
East Harlem
Upper East Side
Yorkville
Lenox Hill
Roosevelt Island
Upper West Side
Lincoln Square
Clinton
Midtown
Murray Hill
Chelsea
Greenwich Village
East Village
Lower East Side
Tribeca
Little Italy
Soho
West Village
Manhattan Valley
Morningside Heights
Gramercy
Battery Park City
Financial District
Carnegie Hill
Noho
Civic Center
Midtown South
Sutton Place
Turtle Bay
Tudor City
Stuyvesant Town
Flatiron
Hudson Yards
```

Let's check the size of the resulting dataframe

In [29]:
```
print(manhattan_venues.shape)
manhattan_venues.head()
```

```
(3148, 7)
```

Out[29]:

	Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
0	Marble Hill	40.876551	-73.91066	Arturo's	40.874412	-73.910271	Pizza Place
1	Marble Hill	40.876551	-73.91066	Bikram Yoga	40.876844	-73.906204	Yoga Studio
2	Marble Hill	40.876551	-73.91066	Tibbett Diner	40.880404	-73.908937	Diner
3	Marble Hill	40.876551	-73.91066	Starbucks	40.877531	-73.905582	Coffee Shop
4	Marble Hill	40.876551	-73.91066	Dunkin'	40.877136	-73.906666	Donut Shop

Let's print each neighborhood along with the top 5 most common venues

In [35]:
```
num_top_venues = 5

for hood in manhattan_grouped['Neighborhood']:
    print("----"+hood+"----")
    temp = manhattan_grouped[manhattan_grouped['Neighborhood'] == hood].T.reset_index()
    temp.columns = ['venue','freq']
    temp = temp.iloc[1:]
    temp['freq'] = temp['freq'].astype(float)
    temp = temp.round({'freq': 2})
    print(temp.sort_values('freq', ascending=False).reset_index(drop=True).head(num_top_venues))
    print('\n')
```

```
----Battery Park City----
           venue  freq
0           Park  0.12
1          Hotel  0.06
2    Coffee Shop  0.06
3  Boat or Ferry  0.05
4  Memorial Site  0.05


----Carnegie Hill----
                venue  freq
0         Coffee Shop  0.09
1                Café  0.04
2  Italian Restaurant  0.04
3         Pizza Place  0.04
4                 Gym  0.03


----Central Harlem----
                  venue  freq
0    Chinese Restaurant  0.06
1  Gym / Fitness Center  0.04
```

Let's put that into a pandas dataframe

First, let's write a function to sort the venues in descending order.

In [36]:
```
def return_most_common_venues(row, num_top_venues):
    row_categories = row.iloc[1:]
    row_categories_sorted = row_categories.sort_values(ascending=False)

    return row_categories_sorted.index.values[0:num_top_venues]
```

Now let's create the new dataframe and display the top 10 venues for each neighborhood.

In [37]:
```
num_top_venues = 10

indicators = ['st', 'nd', 'rd']

# create columns according to number of top venues
columns = ['Neighborhood']
for ind in np.arange(num_top_venues):
    try:
        columns.append('{}{} Most Common Venue'.format(ind+1, indicators[ind]))
    except:
        columns.append('{}th Most Common Venue'.format(ind+1))

# create a new dataframe
neighborhoods_venues_sorted = pd.DataFrame(columns=columns)
neighborhoods_venues_sorted['Neighborhood'] = manhattan_grouped['Neighborhood']

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

Let's put that into a pandas dataframe

First, let's write a function to sort the venues in descending order.

In [36]:
```
def return_most_common_venues(row, num_top_venues):
    row_categories = row.iloc[1:]
    row_categories_sorted = row_categories.sort_values(ascending=False)

    return row_categories_sorted.index.values[0:num_top_venues]
```

Now let's create the new dataframe and display the top 10 venues for each neighborhood.

In [37]:
```
num_top_venues = 10

indicators = ['st', 'nd', 'rd']

# create columns according to number of top venues
columns = ['Neighborhood']
for ind in np.arange(num_top_venues):
    try:
        columns.append('{}{} Most Common Venue'.format(ind+1, indicators[ind]))
    except:
        columns.append('{}th Most Common Venue'.format(ind+1))

# create a new dataframe
neighborhoods_venues_sorted = pd.DataFrame(columns=columns)
neighborhoods_venues_sorted['Neighborhood'] = manhattan_grouped['Neighborhood']

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

Cluster Neighborhoods

Run k-means to cluster the neighborhood into 5 clusters.

In [38]:
```
# set number of clusters
kclusters = 5

manhattan_grouped_clustering = manhattan_grouped.drop('Neighborhood', 1)

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

# check cluster labels generated for each row in the dataframe
kmeans.labels_[0:10]
```

Out[38]:
```
array([1, 0, 2, 1, 0, 1, 1, 2, 0, 0], dtype=int32)
```

Let's create a new dataframe that includes the cluster as well as the top 10 venues for each neighborhood.

In [39]:
```
# add clustering labels
neighborhoods_venues_sorted.insert(0, 'Cluster Labels', kmeans.labels_)

manhattan_merged = manhattan_data

# merge toronto_grouped with toronto_data to add latitude/longitude for each neighborhood
manhattan_merged = manhattan_merged.join(neighborhoods_venues_sorted.set_index('Neighborhood'), on='Neigh

manhattan_merged.head() # check the last columns!
```

39]:

Borough	Neighborhood	Latitude	Longitude	Cluster Labels	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Mo Comm Ven
Manhattan	Marble Hill	40.876551	-73.910660	2	Sandwich Place	Coffee Shop	Gym	Pharmacy	Department Store	Diner
Manhattan	Chinatown	40.715618	-73.994279	0	Chinese Restaurant	Bakery	Dessert Shop	Vietnamese Restaurant	Cocktail Bar	Bar
Manhattan	Washington Heights	40.851903	-73.936900	2	Café	Bakery	Pizza Place	Grocery Store	Mobile Phone Shop	Donut Shop
Manhattan	Inwood	40.867684	-73.921210	2	Mexican Restaurant	Lounge	Café	Restaurant	Park	Frozen Yogurt Shop
Manhattan	Hamilton Heights	40.823604	-73.949688	2	Pizza Place	Deli / Bodega	Coffee Shop	Café	Mexican Restaurant	Bakery

Finally, let's visualize the resulting clusters

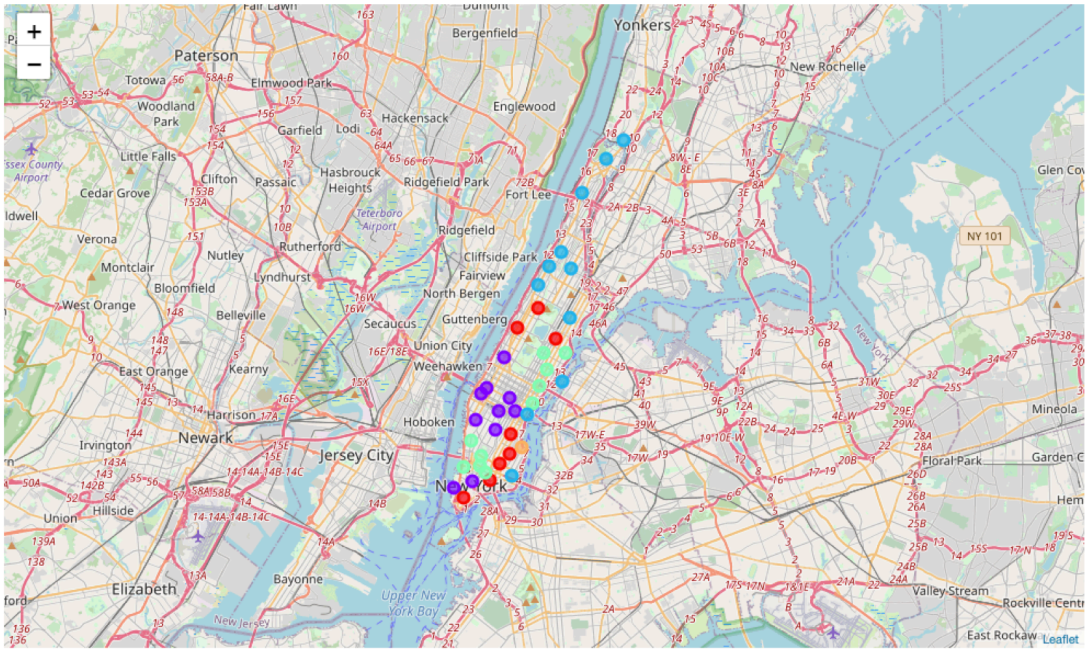
[40]:
```
# create map
map_clusters = folium.Map(location=[latitude, longitude], zoom_start=11)

# 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(manhattan_merged['Latitude'], manhattan_merged['Longitude'], manhattan_m
borhood'], manhattan_merged['Cluster Labels']):
    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
```

Out[40]:


Paterson
Totowa
Elmwood Park
Woodland Park
Garfield
Lodi
Hackensack
Englewood
Bergenfield
Yonkers
New Rochelle
Little Falls
Clifton
Passaic
Hasbrouck Heights
Ridgefield Park
Fort Lee
Cedar Grove
Teterboro Airport
Ridgefield
Verona
Nutley
Rutherford
Cliffside Park
Montclair
Bloomfield
Lyndhurst
Fairview
North Bergen
West Orange
Belleville
Secaucus
Guttenberg
Union City
Weehawken
East Orange
Kearny
Harrison
Hoboken
Irvington
Newark
Jersey City
New York
Mineola
Floral Park
Hillside
Union
Elizabeth
Bayonne
Upper New York Bay
New Jersey
Valley Stream
Rockville Centre
East Rockaway
NY 101
Leaflet

Examine Clusters

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

Cluster 1

In [41]:
```
manhattan_merged.loc[manhattan_merged['Cluster Labels'] == 0, manhattan_merged.columns[[1] + list(range(5, manhattan_merged.shape[1]))]]
```

Out[41]:

	Neighborhood	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue	7th Most Common Venue	8th Most Common Venue	9th Most Common Venue	10th Most Common Venue
1	Chinatown	Chinese Restaurant	Bakery	Dessert Shop	Vietnamese Restaurant	Cocktail Bar	Bar	Spa	Ice Cream Shop	Bubble Tea Shop	Hotpot Restaurant
12	Upper West Side	Italian Restaurant	Bar	Dessert Shop	Bakery	Wine Bar	Indian Restaurant	Coffee Shop	Thai Restaurant	Breakfast Spot	Ice Cream Shop
19	East Village	Bar	Mexican Restaurant	Cocktail Bar	Korean Restaurant	Pizza Place	Coffee Shop	Wine Bar	Ice Cream Shop	Japanese Restaurant	Speakeasy
25	Manhattan Valley	Coffee Shop	Bar	Mexican Restaurant	Pizza Place	Yoga Studio	Health Food Store	Peruvian Restaurant	Spa	Bubble Tea Shop	Café
27	Gramercy	Bar	Bagel Shop	Pizza Place	Coffee Shop	American Restaurant	Italian Restaurant	Playground	Grocery Store	Cocktail Bar	Mexican Restaurant
29	Financial District	Coffee Shop	Bar	American Restaurant	Cocktail Bar	Pizza Place	Italian Restaurant	Juice Bar	Steakhouse	Park	Event Space
30	Carnegie Hill	Coffee Shop	Café	Pizza Place	Italian Restaurant	Yoga Studio	Bookstore	Wine Shop	Gym / Fitness Center	Gym	French Restaurant
31	Noho	Pizza Place	Italian Restaurant	Grocery Store	Coffee Shop	Japanese Restaurant	Cocktail Bar	French Restaurant	Hotel	Mexican Restaurant	Wine Bar

Cluster 2

In [42]:
```
manhattan_merged.loc[manhattan_merged['Cluster Labels'] == 1, manhattan_merged.columns[[1] + list(range(5, manhattan_merged.shape[1]))]]
```

Out[42]:

	Neighborhood	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue	7th Most Common Venue	8th Most Common Venue	9th Most Common Venue	10th Most Common Venue
13	Lincoln Square	Café	Plaza	Italian Restaurant	Theater	Performing Arts Venue	Concert Hall	Gym / Fitness Center	Coffee Shop	Wine Shop	American Restaurant
14	Clinton	Theater	Italian Restaurant	Gym / Fitness Center	Coffee Shop	Wine Shop	Gym	Sandwich Place	Cocktail Bar	American Restaurant	Hotel
15	Midtown	Coffee Shop	Hotel	Bakery	Theater	Clothing Store	Pizza Place	Cuban Restaurant	Steakhouse	Sandwich Place	Bookstore
16	Murray Hill	Sandwich Place	Coffee Shop	Japanese Restaurant	Hotel	Pizza Place	American Restaurant	Gym / Fitness Center	Chinese Restaurant	Café	Juice Bar
17	Chelsea	Coffee Shop	Art Gallery	Ice Cream Shop	American Restaurant	French Restaurant	Bakery	Café	Cocktail Bar	Pizza Place	Market
28	Battery Park City	Park	Hotel	Coffee Shop	Memorial Site	Gym	Boat or Ferry	Food Court	Shopping Mall	Gourmet Shop	Mexican Restaurant
32	Civic Center	Coffee Shop	Gym / Fitness Center	Cocktail Bar	Spa	Hotel	French Restaurant	Yoga Studio	Café	Park	American Restaurant
33	Midtown South	Korean Restaurant	Hotel	Café	Japanese Restaurant	Lounge	Burger Joint	Cuban Restaurant	Coffee Shop	Cocktail Bar	Dessert Shop
38	Flatiron	Gym / Fitness Center	Café	Mediterranean Restaurant	New American Restaurant	Coffee Shop	Park	Italian Restaurant	Gym	Japanese Restaurant	Vegetarian / Vegan Restaurant
39	Hudson Yards	Hotel	Gym / Fitness Center	American Restaurant	Italian Restaurant	Gym	Bar	Park	Dog Run	Coffee Shop	Restaurant

Cluster 3

In [43]:
```
manhattan_merged.loc[manhattan_merged['Cluster Labels'] == 2, manhattan_merged.columns[[1] + list(range(5, manhattan_merged.shape[1]))]]
```

Out[43]:

	Neighborhood	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue	7th Most Common Venue	8th Most Common Venue	9th Most Common Venue	10th Most Common Venue
0	Marble Hill	Sandwich Place	Coffee Shop	Gym	Pharmacy	Department Store	Diner	Discount Store	Donut Shop	Kids Store	Deli / Bodega
2	Washington Heights	Café	Bakery	Pizza Place	Grocery Store	Mobile Phone Shop	Donut Shop	Mexican Restaurant	Sandwich Place	Bank	Coffee Shop
3	Inwood	Mexican Restaurant	Lounge	Café	Restaurant	Park	Frozen Yogurt Shop	Bakery	Spanish Restaurant	Chinese Restaurant	American Restaurant
4	Hamilton Heights	Pizza Place	Deli / Bodega	Coffee Shop	Café	Mexican Restaurant	Bakery	Sushi Restaurant	Cocktail Bar	Sandwich Place	Yoga Studio
5	Manhattanville	Deli / Bodega	Coffee Shop	Mexican Restaurant	Seafood Restaurant	Italian Restaurant	Chinese Restaurant	Dumpling Restaurant	Park	Sushi Restaurant	Supermarket
6	Central Harlem	Chinese Restaurant	Seafood Restaurant	Pizza Place	Bar	French Restaurant	Gym / Fitness Center	American Restaurant	African Restaurant	Deli / Bodega	Bookstore
7	East Harlem	Mexican Restaurant	Thai Restaurant	Bakery	Deli / Bodega	Sandwich Place	Latin American Restaurant	Gas Station	Liquor Store	Steakhouse	Grocery Store
11	Roosevelt Island	Park	Outdoors & Recreation	Residential Building (Apartment / Condo)	School	Liquor Store	Sandwich Place	Bus Line	Coffee Shop	Dry Cleaner	Greek Restaurant
20	Lower East Side	Chinese Restaurant	Pharmacy	Park	Bakery	Café	Cocktail Bar	Art Gallery	Ramen Restaurant	Coffee Shop	Performing Arts Venue
26	Morningside Heights	Bookstore	American Restaurant	Park	Coffee Shop	Sandwich Place	Deli / Bodega	Burger Joint	New American Restaurant	Mediterranean Restaurant	Mexican Restaurant
36	Tudor City	Café	Park	Mexican Restaurant	Diner	Deli / Bodega	Pizza Place	Dog Run	Coffee Shop	Garden	Seafood Restaurant

Cluster 4

In [44]:
```
manhattan_merged.loc[manhattan_merged['Cluster Labels'] == 3, manhattan_merged.columns[[1] + list(range(5, manhattan_merged.shape[1]))]]
```

Out[44]:

	Neighborhood	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue	7th Most Common Venue	8th Most Common Venue	9th Most Common Venue	10th Most Common Venue
8	Upper East Side	Italian Restaurant	Coffee Shop	Bakery	Gym / Fitness Center	Juice Bar	Yoga Studio	French Restaurant	Spa	Women's Store	Sushi Restaurant
9	Yorkville	Italian Restaurant	Gym	Coffee Shop	Sushi Restaurant	Bar	Deli / Bodega	Pizza Place	Japanese Restaurant	Mexican Restaurant	Wine Shop
10	Lenox Hill	Italian Restaurant	Coffee Shop	Pizza Place	Café	Cocktail Bar	Sushi Restaurant	Gym / Fitness Center	Gym	Burger Joint	Thai Restaurant
18	Greenwich Village	Italian Restaurant	Café	Sushi Restaurant	Clothing Store	French Restaurant	Ice Cream Shop	Chinese Restaurant	Sandwich Place	Pilates Studio	Bakery
21	Tribeca	Italian Restaurant	Park	American Restaurant	Spa	Wine Bar	Café	Greek Restaurant	Coffee Shop	Bakery	Hotel
22	Little Italy	Bakery	Chinese Restaurant	Spa	Bubble Tea Shop	Mediterranean Restaurant	Italian Restaurant	Café	Cosmetics Shop	Thai Restaurant	Sandwich Place
23	Soho	Italian Restaurant	Clothing Store	Mediterranean Restaurant	Sandwich Place	Coffee Shop	French Restaurant	Spa	Café	Hotel	Bakery
24	West Village	Italian Restaurant	New American Restaurant	Cocktail Bar	Park	Wine Bar	American Restaurant	Pizza Place	Coffee Shop	Bakery	Jazz Club
34	Sutton Place	Gym / Fitness Center	Italian Restaurant	Coffee Shop	Park	Furniture / Home Store	Bagel Shop	Gym	Pizza Place	Spa	Lingerie Store
35	Turtle Bay	Italian Restaurant	Coffee Shop	Park	Café	Sushi Restaurant	French Restaurant	Deli / Bodega	Seafood Restaurant	Bar	Japanese Restaurant

Cluster 5

In [45]:
```
manhattan_merged.loc[manhattan_merged['Cluster Labels'] == 4, manhattan_merged.columns[[1] + list(range(5, manhattan_merged.shape[1]))]]
```

Out[45]:

	Neighborhood	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue	7th Most Common Venue	8th Most Common Venue	9th Most Common Venue	10th Most Common Venue
37	Stuyvesant Town	Park	Baseball Field	Heliport	Bar	Cocktail Bar	Coffee Shop	Gym / Fitness Center	Harbor / Marina	Bistro	Gas Station

Results and Discussion

Beased on our reseach, we could see that New York City have lots of coffee shop. As a matter of fact their are over 3,389 coffee shops in NYC. After looking at all the factors, we beleive NYC is the perfect location for a new coffee shop. We took a look at all the Borough in NYC and figured out where is the best place for a new coffee shop. We look at foot tracffic, other kinds of food stores around the real. After all the research, we believe Manhattan is the best location,

We than broke Manhattan down to Neighborhood to see where is the best place for a new coffee shop. we found out that Marble Hill is the best location for a new coffee shop. accounding to foursqure data, coffee shop is the 2nd most popular venue in Marble Hill next to sandwhich store. We also ran k-means to cluster the neighborhood into 5 clusters and in the secound cluster, we could clearly see that coffee shop is very popular.

Conclusion

Why coffee shop

Coffee shops are undoubtedly a small piece of the much larger puzzle that is contemporary urban living; yet their importance should not be underestimated. From their influence on the enlightenment to their ever-evolving status. Openning a coffee shop is my own small way of contributing to my community. Coffee brings poeple together and i love bringing people together. If i could bring more people together, i believe i have done my part to help the human race along.