Battle of Neighborhood

New York City

Yadan Tang

1 Introduction

1.1 background

The Catalysis Society of Metropolitan New York (CSMNY), a local chapter of North American Catalysis Society (NACS), founded in 1958 to promote and encourage the growth and development of the science of catalysis in the New Jersey and Metro New York areas. They organize 7 monthly professional dinner seminar meetings of scientists - to report, discuss, and exchange information and viewpoints in the field of catalysis. They also organize an all-day Annual Symposium in March which features lectures from distinguished researchers and a poster session presented by university students working in the catalysis area.

1.2 Problem

This year, CSMNY is looking for recommendation of neighborhood in Manhattan to host their annual meeting. The attendees of the meetings are of various cultural background and the trip is for professional exchange as well as for tourism. The host seeks a good recommendation of neighborhood that can accommodate ~1000 people and restaurants that has a good variety of multicultural cuisines, plaza and fun activities. Some of the attendees who travel overseas might not have cars, so if the neighborhood has a variety of venues are within walking distance or easily accessible by public transportation would be preferred.

2 Data acquisition and cleaning

2.1 Data sources

The neighborhood data of New York is available online: https://cocl.us/new_york_dataset. The json data that contains all the information of NYC neighborhoods. The data is dictionary type, and under ‘features’ has all the information about the neighborhood, borough and the coordinates.

[3]:
```
import json
with open('newyork_data.json') as json_data:
    newyork_data = json.load(json_data)
```

[4]:
```
newyork_data
```

[4]:
```
{'type': 'FeatureCollection',
 'totalFeatures': 306,
 'features': [{'type': 'Feature',
   'id': 'nyu_2451_34572.1',
   'geometry': {'type': 'Point',
    'coordinates': [-73.84720052054902, 40.89470517661]},
   'geometry_name': 'geom',
   'properties': {'name': 'Wakefield',
    'stacked': 1,
    'annoline1': 'Wakefield',
    'annoline2': None,
    'annoline3': None,
    'annoangle': 0.0,
    'borough': 'Bronx',
    'bbox': [-73.84720052054902,
     40.89470517661,
     -73.84720052054902,
     40.89470517661]}},
```

Therefore, we will extract the information from “Features” into a Pandas dataframe, neighborhoods, for further processing. The dataframe has 5 unique boroughs and 306 neighborhoods.

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

[7]:
```
for data in neighbor_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)
```

2.2 Data cleaning

Now let’s look at the neighborhood data in Manhattan. Using groupby(‘Borough’).count(), we are able to obtain the information below:

[12]:
```
neighborhoods.groupby('Borough').count()
```

[12]:

Borough	Neighborhood	Latitude	Longitude
Bronx	52	52	52
Brooklyn	70	70	70
Manhattan	40	40	40
Queens	81	81	81
Staten Island	63	63	63

As we can see from the above query result, Manhattan is one of the Borough listed, and it contains 40 neighborhoods. We’ll create a dataframe that can narrow down only to the neighborhood and coordinates of Manhattan. The coordinate information will be necessary for visualization:

[13]:
```
mht_data=neighborhoods[neighborhoods['Borough']=='Manhattan'].reset_index(drop=True)
mht_data.head()
```

[13]:

	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

3 Visualization and connect to API

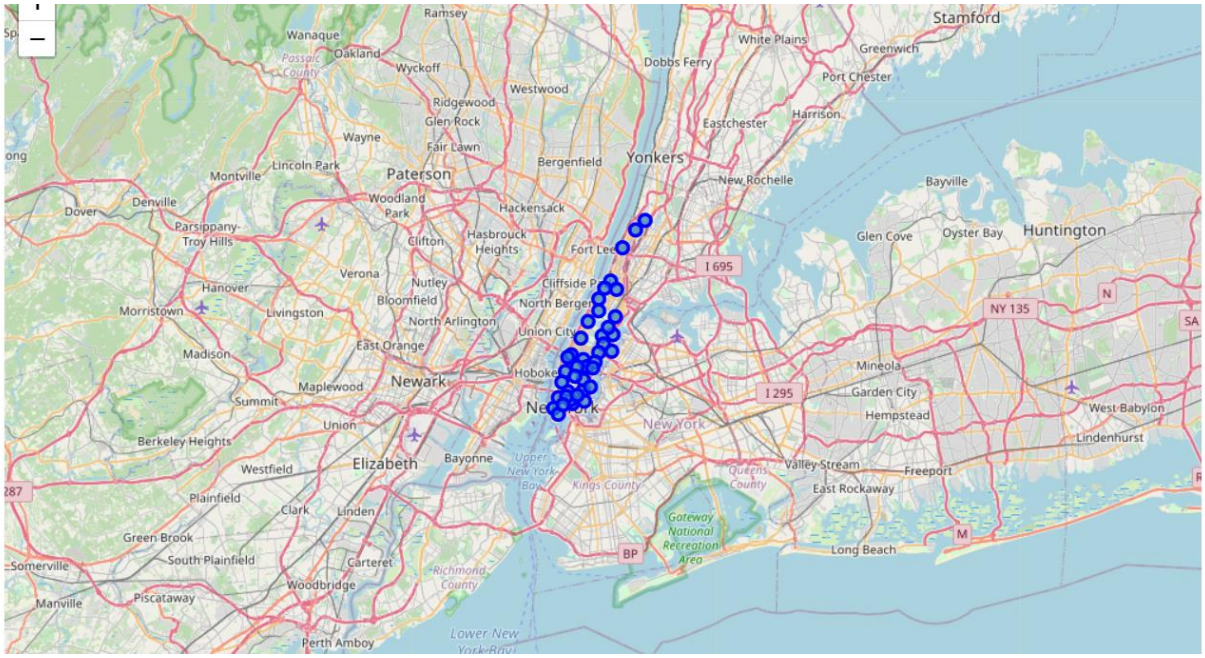
3.1 Visualize the neighborhood of Manhattan via Folium

Now let's visualize the neighborhood of Manhattan via folium package by the following commands:

```
[14]: address='Manhattan, New York City'

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

```
The geograpical coordinate of Manhattan, NYC are 40.7810178, -73.959299675.
```

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

# add markers to map
for lat, lng, borough, neighborhood in zip(mht_data['Latitude'], mht_data['Longitude'], mht_data['Borough'], mht_data['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_mht)

map_mht
```



Now we have successfully obtained the data that contains the neighborhoods and the coordinates of Manhattan. Next step is to explore all the nearby venues in Manhattan.

4 Connect to foursquare API to obtain all the venues in Manhattan

We can connect to Foursquare API using personal Foursquare ID and secrets. Define a function to get all the nearby venues in Manhattan, NYC. Return only the relevant information of each nearby venue and save it into a dataframe called "nearby_venues" which includes the name and the coordinates of the neighborhoods and its nearby venues.

[17]:
```
LIMIT=500
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']) for v in results])

    nearby_venues = pd.DataFrame([item for venue_list in venues_list for item in venue_list])
    nearby_venues.columns = ['Neighborhood',
```

The mht_venues dataframe contains 7 columns. We would like to get the top 10 venues in each neighborhood of Manhattan. and what we are interested in is the category of venues and the frequency of each venues in the neighborhoods. Group the dataframe by venue category, the count show there are 331 unique venue categories.

[18]:

	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

We now can explore the top 10 venues in each neighborhood in Manhattan by using the following function:

[34]:
```
import numpy as np
top_venue_number = 10

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

# create columns according to number of top venues
columns = ['Neighborhood']
for ind in np.arange(top_venue_number):
    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'] = mht_grouped['Neighborhood']

for ind in np.arange(mht_grouped.shape[0]):
    neighborhoods_venues_sorted.iloc[ind, 1:] = topVenues(mht_grouped.iloc[ind, :], top_venue_number)

neighborhoods_venues_sorted
```

[34]:

	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	Battery Park City	Park	Hotel	Coffee Shop	Gym	Memorial Site	Playground	Gourmet Shop	Food Court	Mexican Restaurant	Shopping Mall

Convert descriptive values in the ”venue category” into numeric values by using one hot coding, this is the preparation step for the K-means clustering in order to segment neighborhoods in Manhattan.

[20]:
```
# one hot encoding
mht_onehot = pd.get_dummies(mht_venues[['Venue Category']], prefix="", prefix_sep="")

# add neighborhood column back to dataframe
mht_onehot['Neighborhood'] = mht_venues['Neighborhood']

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

print(mht_onehot.shape)
mht_onehot.head()
```

•••

[21]:
```
mht_grouped=mht_onehot.groupby('Neighborhood').mean().reset_index()
mht_grouped.head()
```

[21]:

	Neighborhood	Accessories Store	Adult Boutique	Afghan Restaurant	African Restaurant	American Restaurant	Antique Shop	Arcade	Arepa Restaurant	Argentinian Restaurant	...	Video Store	Vietnamese Restaurant	Volleyball Court	Waterfront
0	Battery Park City	0.0	0.0	0.0	0.000000	0.015385	0.0	0.0	0.0	0.000000	...	0.0	0.000000	0.0	0.0
1	Carnegie Hill	0.0	0.0	0.0	0.000000	0.011494	0.0	0.0	0.0	0.011494	...	0.0	0.022989	0.0	0.0
2	Central Harlem	0.0	0.0	0.0	0.066667	0.044444	0.0	0.0	0.0	0.000000	...	0.0	0.000000	0.0	0.0
3	Chelsea	0.0	0.0	0.0	0.000000	0.030000	0.0	0.0	0.0	0.000000	...	0.0	0.000000	0.0	0.0

5 K-means clustering

K-means clustering helps to segment the neighborhoods based on the features in the venue category. We will start 5 as the initial Kcluster size, and fit the mht_grouped dataframe work the KMeans clustering. It generates labels of the neighborhood in array. We add the labels back to the mht_merged dataframe so that we can visualize the clustering results.

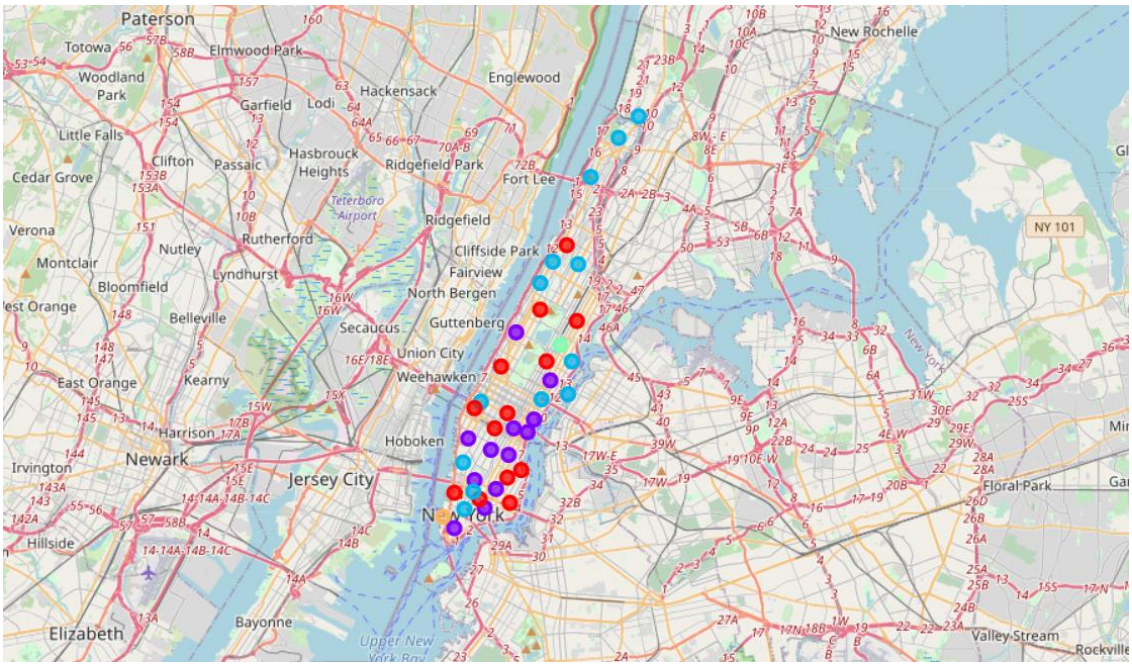
[25]:
```
# import k-means from clustering stage
from sklearn.cluster import KMeans
```

[26]:
```
kclusters=5

mht_cluster=mht_grouped.drop('Neighborhood',1)

kMeans=KMeans(n_clusters=kclusters, random_state=3).fit(mht_cluster)

kMeans.labels_
```

[26]:
```
array([1, 2, 1, 2, 2, 1, 1, 2, 2, 1, 2, 2, 2, 2, 1, 2, 2, 2, 2, 2, 2,
       4, 1, 1, 1, 1, 2, 3, 2, 0, 2, 1, 2, 2, 2, 2, 2, 2, 2], dtype=int32)
```

[27]:
```
mht_merged=mht_data
mht_merged['Labels']=kMeans.labels_
mht_merged=mht_merged.join(neighborhoods_venues_sorted.set_index('Neighborhood'), on='Neighborhood')
mht_merged.head()
```

Visualize the clustering results by using folium:



We further explore the top 10 venues in each cluster using following code:

[44]:
```
cluster_0=mht_merged.loc[mht_merged['Labels']==0, mht_merged.columns[[1] + list(range(4, mht_merged.shape[1]))]]
cluster_0.head()
```

[44]:

	Neighborhood	Labels	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
4	Hamilton Heights	0	Pizza Place	Deli / Bodega	Coffee Shop	Mexican Restaurant	Café	Yoga Studio	Sandwich Place	Sushi Restaurant	Bakery	Caribbean Restaurant
7	East Harlem	0	Bakery	Mexican Restaurant	Thai Restaurant	Sandwich Place	Latin American Restaurant	Deli / Bodega	Gas Station	Liquor Store	Steakhouse	Seafood Restaurant
8	Upper East Side	0	Italian Restaurant	Coffee Shop	Gym / Fitness Center	Bakery	French Restaurant	Spa	Yoga Studio	Juice Bar	American Restaurant	Hotel
13	Lincoln Square	0	Plaza	Café	Italian Restaurant	Gym / Fitness Center	Concert Hall	Theater	Performing Arts Venue	Gym	French Restaurant	Coffee Shop
15	Midtown	0	Hotel	Coffee Shop	Bakery	Theater	Pizza Place	Sushi Restaurant	Japanese Restaurant	Cuban Restaurant	Clothing Store	Cosmetics Shop

And by looking at the cluster results, it is obvious to see that cluster_0 and cluster_1 have good options of hotels and various options of international cuisine and activities available. Especially Midtown and Murray Hill are good options as hotel is their top venue, followed by a variety of food options and recreation activities.

6 Conclusions:

Based on the exercise, we have successfully explore the neighborhood of Manhattan, NYC and the nearby venues of each neighborhood. K-mean clustering helps to identify the neighborhoods in Cluster_0 and 1 are excellent options for the event and especially Midtown and Murray Hill will have Hotels as the 1st common venue and still have a good combination of multicultural restaurants, deli/café, gyms and plaza.