Capstone Project - The Battle of Neighborhoods

Introduction

The City of New York, usually called either New York City (NYC) or simply New York (NY), is the most populous city in the United States. With an estimated 2018 population of 8,398,748 distributed over a land area of about 302.6 square miles (784 km2), New York is also the most densely populated major city in the United States.

New York City's is the largest city in the United States with a long history of international immigration. New York City was home to nearly 8.5 million people in 2014, accounting for over 40% of the population of New York State and a slightly lower percentage of the New York metropolitan area, home to approximately 23.6 million. 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.

Throughout its history, New York City has been a major point of entry for immigrants; the term "melting pot" was coined to describe densely populated immigrant neighborhoods on the Lower East Side. As many as 800 languages are spoken in New York, making it the most linguistically diverse city in the world. English remains the most widely spoken language, although there are areas in the outer boroughs in which up to 25% of people speak English as an alternate language, and/or have limited or no English language fluency. English is least spoken in neighborhoods such as Flushing, Sunset Park, and Corona.

With its diverse culture, comes diverse food items. There are many restaurants in New York City, each belonging to different categories like Chinese, Indian, French etc. So as part of this project, we will list and visualize all major parts of New York City that has great Chinese Restaurants.

Data

We will different set of data related to the New York city. Below are the data and their source that we will need for this project:

New York City data that contains list Boroughs, Neighborhoods along with their latitude and longitude.

Data source: https://cocl.us/new_york_dataset

Description: This data set contains the required information. And we will use this data set to explore various neighborhoods of New York city.

Indian restaurants in each neighborhood of New York city.

Data source: Foursquare API

Description: By using this API we will get all the venues in each neighborhood. We can filter these

venues to get only Chinese restaurants or type of Venus we want.

GeoSpace data

Data source: https://data.cityofnewyork.us/City-Government/Borough-Boundaries/tqmj-j8zm

Description: By using this geo space data we will get the New York Borough boundaries that will help us visualize choropleth map.

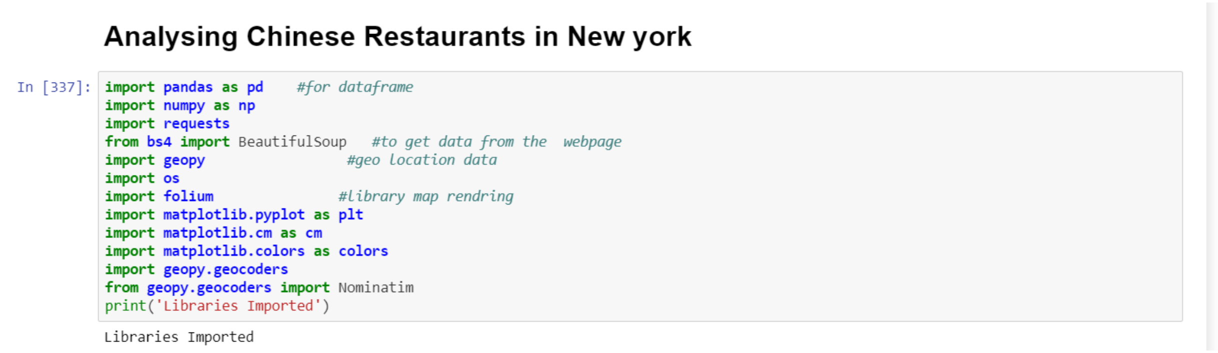
Approach

- We have collected the whole data about the neighborhood in New York city.
- Then we will search for the top venues in each neighborhood using Foursquare API.
- Then we will search for Chinese restaurants from the venue in each neighborhood.
- List ratings, Likes, and tips for each restaurant.
- Sort Each restaurant according to the rating and neighborhood.
- Visualize the data using the folium data.

Analysis

We have used python and pandas to analyze the data and below is my analysis.

Python library used:



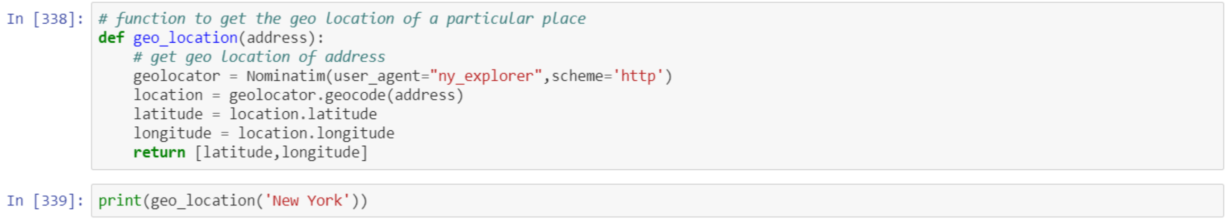
Analysing Chinese Restaurants in New york

In [337]:

```
import pandas as pd    #for dataframe
import numpy as np
import requests
from bs4 import BeautifulSoup   #to get data from the  webpage
import geopy                 #geo location data
import os
import folium               #library map rendring
import matplotlib.pyplot as plt
import matplotlib.cm as cm
import matplotlib.colors as colors
import geopy.geocoders
from geopy.geocoders import Nominatim
print('Libraries Imported')
```

```
Libraries Imported
```

Function to get New York coordinates:



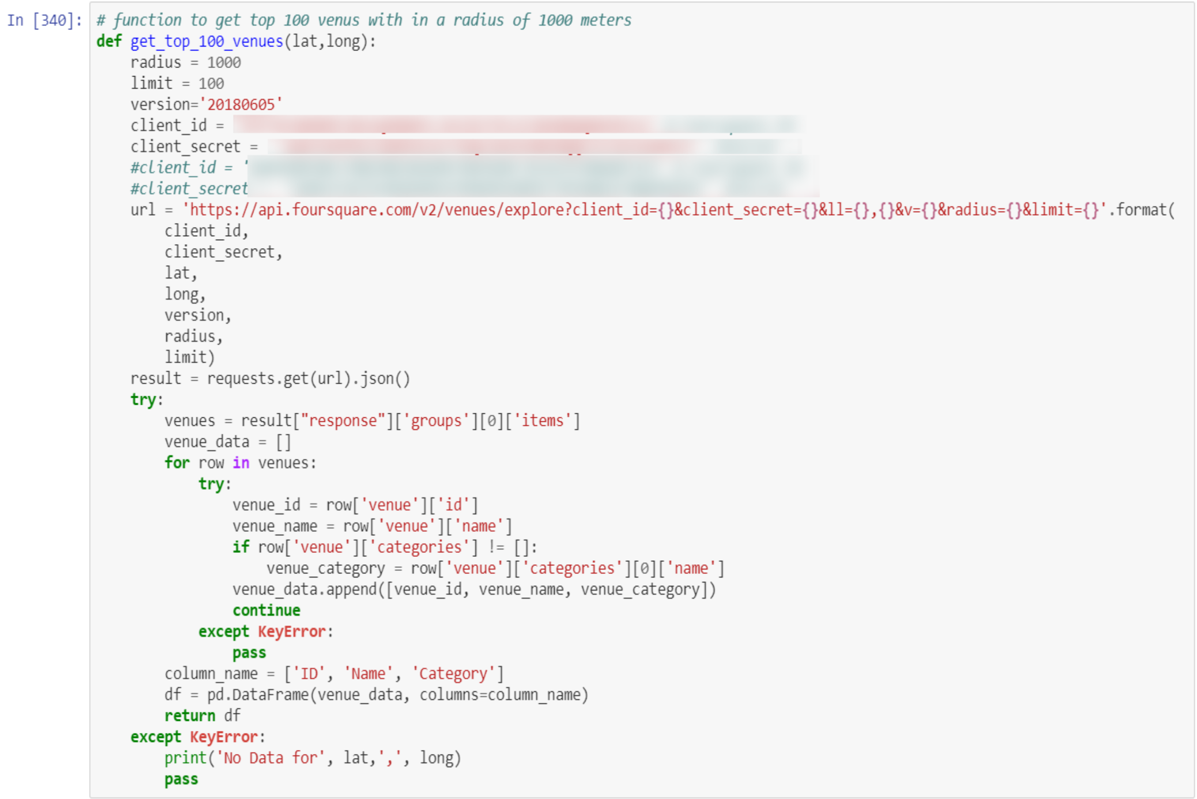
In [338]:

```
# function to get the geo location of a particular place
def geo_location(address):
    # get geo location of address
    geolocator = Nominatim(user_agent="ny_explorer",scheme='http')
    location = geolocator.geocode(address)
    latitude = location.latitude
    longitude = location.longitude
    return [latitude,longitude]
```

In [339]:

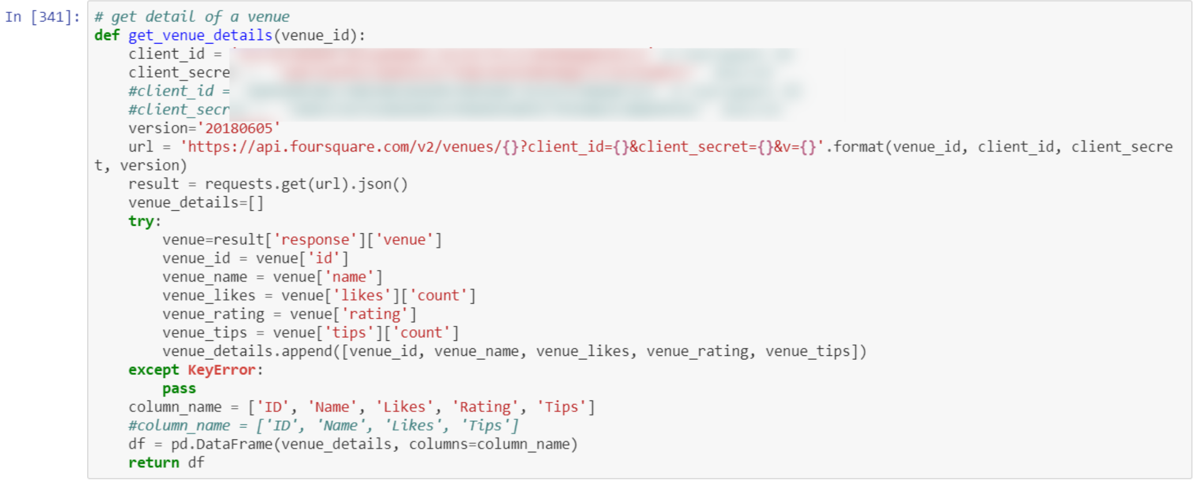
```
print(geo_location('New York'))
```

Get Top 100 values in any neighboring:



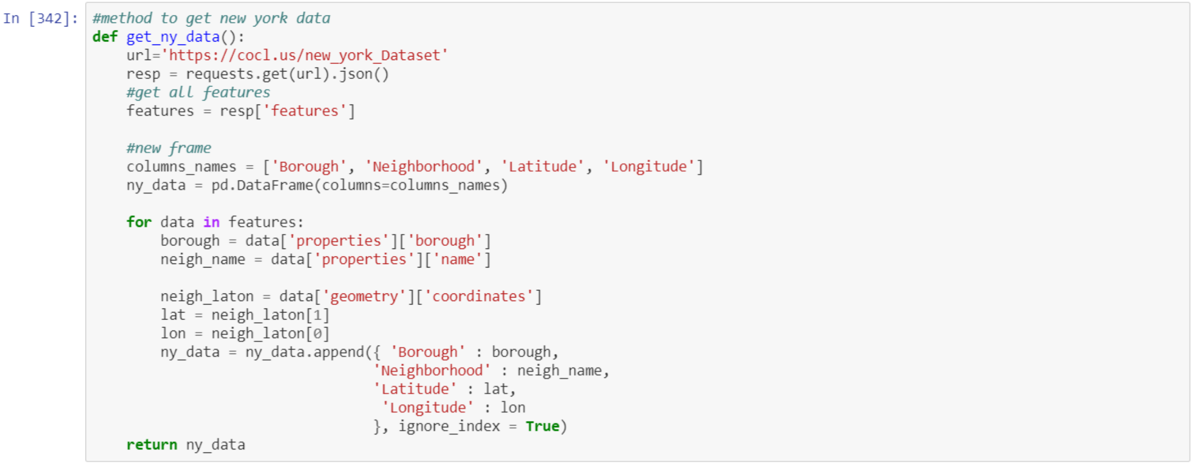
```
In [340]: # function to get top 100 venus with in a radius of 1000 meters
          def get_top_100_venues(lat,long):
              radius = 1000
              limit = 100
              version='20180605'
              client_id = [redacted]
              client_secret = [redacted]
              #client_id = '[redacted]
              #client_secret [redacted]
              url = 'https://api.foursquare.com/v2/venues/explore?client_id={}&client_secret={}&ll={},{}&v={}&radius={}&limit={}'.format(
                  client_id,
                  client_secret,
                  lat,
                  long,
                  version,
                  radius,
                  limit)
              result = requests.get(url).json()
              try:
                  venues = result["response"]['groups'][0]['items']
                  venue_data = []
                  for row in venues:
                      try:
                          venue_id = row['venue']['id']
                          venue_name = row['venue']['name']
                          if row['venue']['categories'] != []:
                              venue_category = row['venue']['categories'][0]['name']
                          venue_data.append([venue_id, venue_name, venue_category])
                          continue
                      except KeyError:
                          pass
                  column_name = ['ID', 'Name', 'Category']
                  df = pd.DataFrame(venue_data, columns=column_name)
                  return df
              except KeyError:
                  print('No Data for', lat,',', long)
                  pass
```

Get detail of a venue:



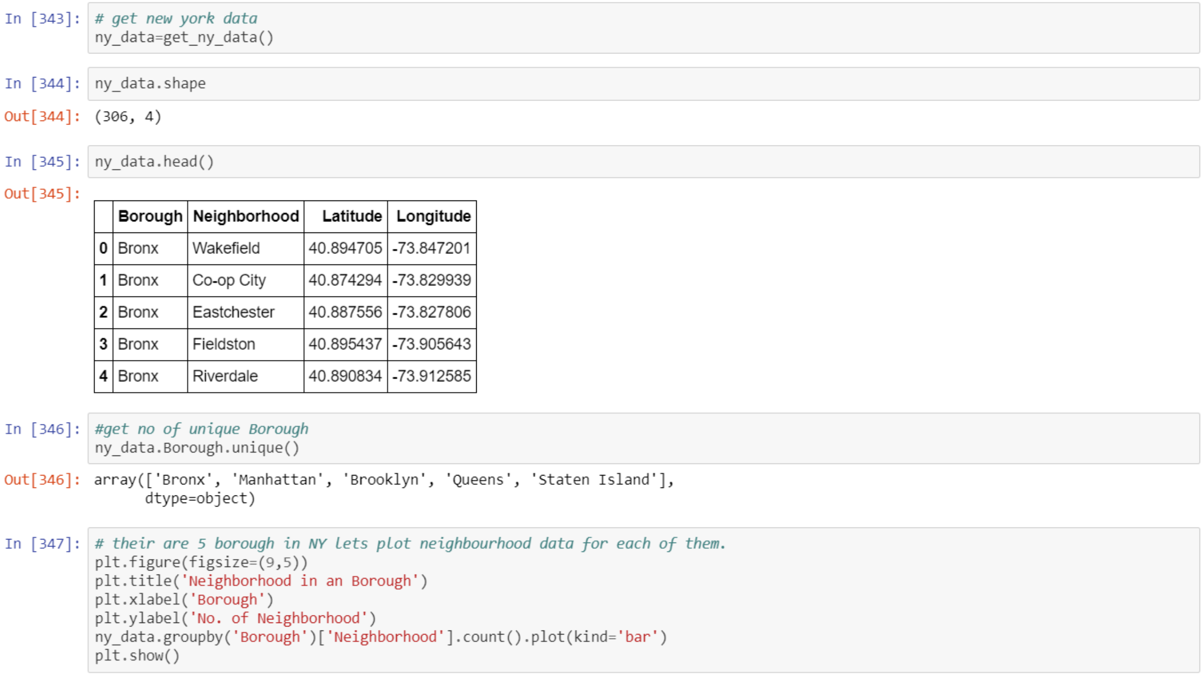
```
In [341]: # get detail of a venue
          def get_venue_details(venue_id):
              client_id = '[redacted]
              client_secre[redacted]
              #client_id = [redacted]
              #client_secr[redacted]
              version='20180605'
              url = 'https://api.foursquare.com/v2/venues/{}?client_id={}&client_secret={}&v={}'.format(venue_id, client_id, client_secre
          t, version)
              result = requests.get(url).json()
              venue_details=[]
              try:
                  venue=result['response']['venue']
                  venue_id = venue['id']
                  venue_name = venue['name']
                  venue_likes = venue['likes']['count']
                  venue_rating = venue['rating']
                  venue_tips = venue['tips']['count']
                  venue_details.append([venue_id, venue_name, venue_likes, venue_rating, venue_tips])
              except KeyError:
                  pass
              column_name = ['ID', 'Name', 'Likes', 'Rating', 'Tips']
              #column_name = ['ID', 'Name', 'Likes', 'Tips']
              df = pd.DataFrame(venue_details, columns=column_name)
              return df
```

Get new your city coordinates:



```
In [342]: #method to get new york data
          def get_ny_data():
              url='https://cocl.us/new_york_Dataset'
              resp = requests.get(url).json()
              #get all features
              features = resp['features']

              #new frame
              columns_names = ['Borough', 'Neighborhood', 'Latitude', 'Longitude']
              ny_data = pd.DataFrame(columns=columns_names)

              for data in features:
                  borough = data['properties']['borough']
                  neigh_name = data['properties']['name']

                  neigh_laton = data['geometry']['coordinates']
                  lat = neigh_laton[1]
                  lon = neigh_laton[0]
                  ny_data = ny_data.append({ 'Borough' : borough,
                                             'Neighborhood' : neigh_name,
                                             'Latitude' : lat,
                                              'Longitude' : lon
                                             }, ignore_index = True)
              return ny_data
```

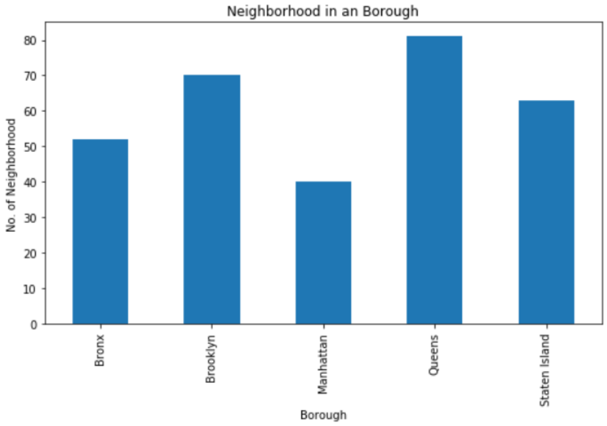
Analyzing New York data:



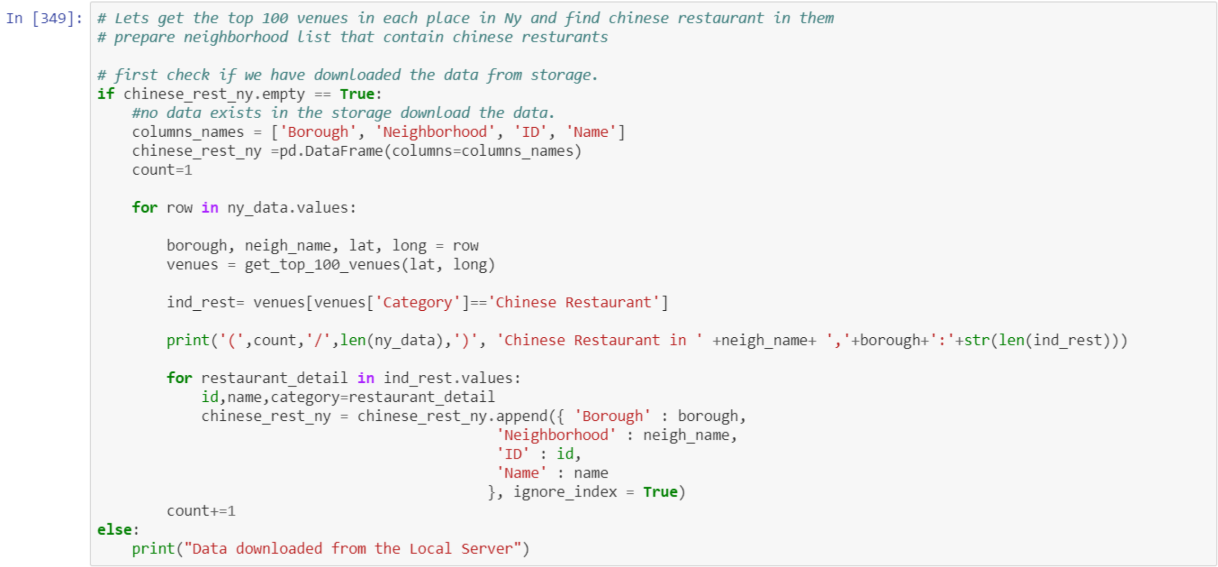
```
In [343]: # get new york data
          ny_data=get_ny_data()

In [344]: ny_data.shape

Out[344]: (306, 4)

In [345]: ny_data.head()

Out[345]:
```

	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

```
In [346]: #get no of unique Borough
          ny_data.Borough.unique()

Out[346]: array(['Bronx', 'Manhattan', 'Brooklyn', 'Queens', 'Staten Island'],
                dtype=object)

In [347]: # their are 5 borough in NY lets plot neighbourhood data for each of them.
          plt.figure(figsize=(9,5))
          plt.title('Neighborhood in an Borough')
          plt.xlabel('Borough')
          plt.ylabel('No. of Neighborhood')
          ny_data.groupby('Borough')['Neighborhood'].count().plot(kind='bar')
          plt.show()
```



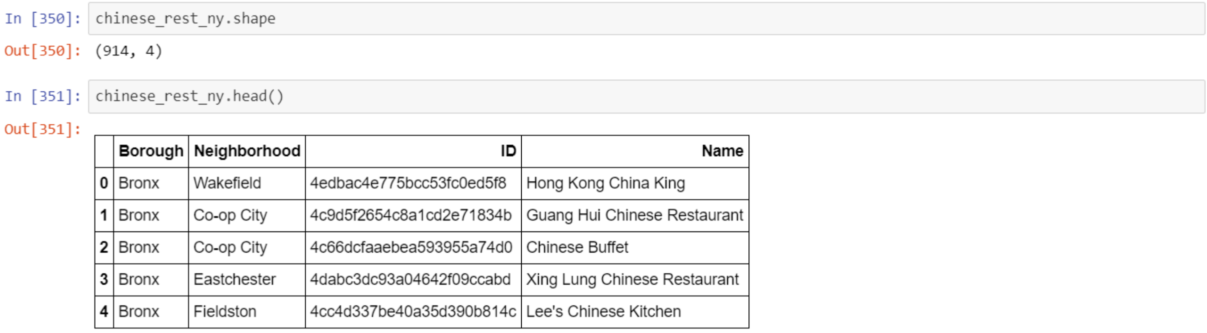
Get top 100 venue in each neighborhood:



In [349]:

```
# Lets get the top 100 venues in each place in Ny and find chinese restaurant in them
# prepare neighborhood list that contain chinese resturants

# first check if we have downloaded the data from storage.
if chinese_rest_ny.empty == True:
    #no data exists in the storage download the data.
    columns_names = ['Borough', 'Neighborhood', 'ID', 'Name']
    chinese_rest_ny =pd.DataFrame(columns=columns_names)
    count=1

    for row in ny_data.values:

        borough, neigh_name, lat, long = row
        venues = get_top_100_venues(lat, long)

        ind_rest= venues[venues['Category']=='Chinese Restaurant']

        print('(',count,'/',len(ny_data),')', 'Chinese Restaurant in ' +neigh_name+ ','+borough+':'+str(len(ind_rest)))

        for restaurant_detail in ind_rest.values:
            id,name,category=restaurant_detail
            chinese_rest_ny = chinese_rest_ny.append({ 'Borough' : borough,
                                                'Neighborhood' : neigh_name,
                                                'ID' : id,
                                                'Name' : name
                                               }, ignore_index = True)
        count+=1
else:
    print("Data downloaded from the Local Server")
```

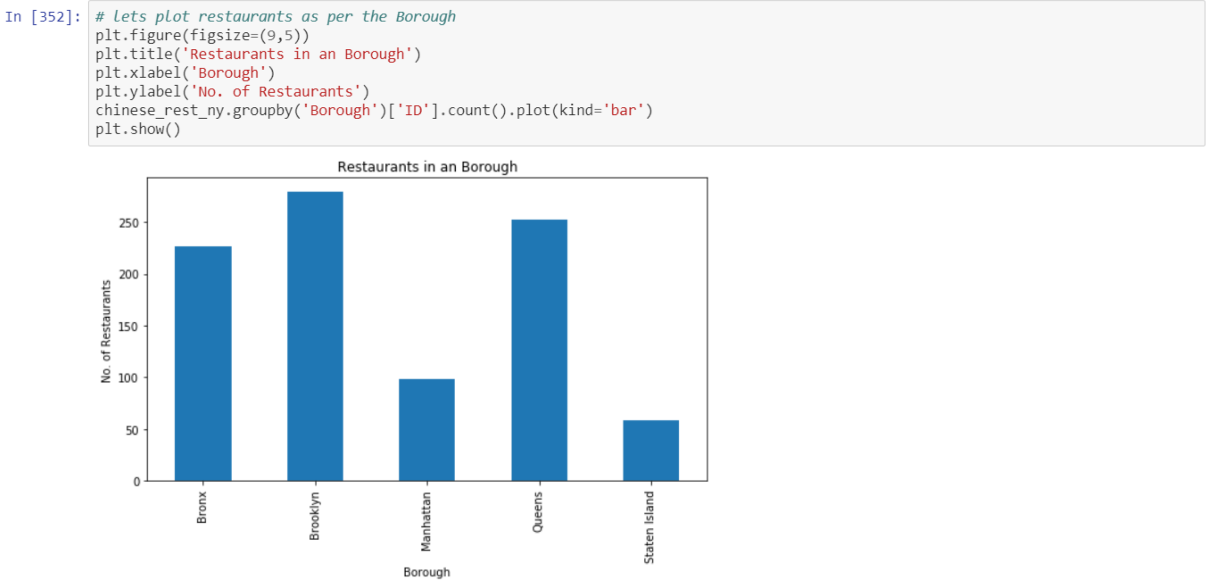
Analyze Restaurants in each neighborhood:



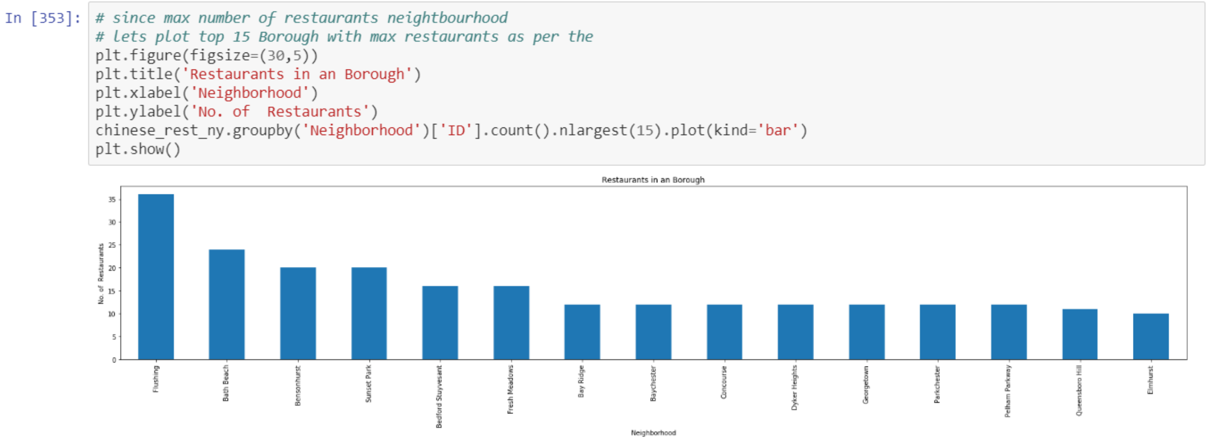
```
In [350]: chinese_rest_ny.shape
Out[350]: (914, 4)
```

```
In [351]: chinese_rest_ny.head()
Out[351]:
```

	Borough	Neighborhood	ID	Name
0	Bronx	Wakefield	4edbac4e775bcc53fc0ed5f8	Hong Kong China King
1	Bronx	Co-op City	4c9d5f2654c8a1cd2e71834b	Guang Hui Chinese Restaurant
2	Bronx	Co-op City	4c66dcfaaebea593955a74d0	Chinese Buffet
3	Bronx	Eastchester	4dabc3dc93a04642f09ccabd	Xing Lung Chinese Restaurant
4	Bronx	Fieldston	4cc4d337be40a35d390b814c	Lee's Chinese Kitchen



```
In [352]: # lets plot restaurants as per the Borough
          plt.figure(figsize=(9,5))
          plt.title('Restaurants in an Borough')
          plt.xlabel('Borough')
          plt.ylabel('No. of Restaurants')
          chinese_rest_ny.groupby('Borough')['ID'].count().plot(kind='bar')
          plt.show()
```



```
In [353]: # since max number of restaurants neightbourhood
          # lets plot top 15 Borough with max restaurants as per the
          plt.figure(figsize=(30,5))
          plt.title('Restaurants in an Borough')
          plt.xlabel('Neighborhood')
          plt.ylabel('No. of  Restaurants')
          chinese_rest_ny.groupby('Neighborhood')['ID'].count().nlargest(15).plot(kind='bar')
          plt.show()
```

Get data related to each restaurant:

In [355]:

```
# lets get rating for each restaurant

### get file from the saved values
# read file from storage.
# Since call to foursquare are limited we save the data for later use.
my_file = project.get_file('chinese_restaurant_Rating.csv')
my_file.seek(0)

chinese_rest_rating_csv = pd.read_csv(my_file)

chinese_rest_rating_csv.shape
chinese_rest_rating = chinese_rest_rating_csv

if chinese_rest_rating.empty == True:

    # prepare neighborhood list that contain resturants
    columns_names = ['Borough', 'Neighborhood', 'ID', 'Name', 'Likes', 'Rating', 'Tips']
    chinese_rest_rating=pd.DataFrame(columns=columns_names)
    count = 1

    for row in chinese_rest_ny.values:
        Borough, Neighborhood, id, name = row
        try:
            venue_details=get_venue_details(id)
            print(venue_details)
            id, name, likes, rating, tips = venue_details.iloc[0]
        except IndexError:
            print('No data Availabe for ID', id)
            id, name, likes, rating, tips = [0]*5

        print('(',count,'/',len(chinese_rest_ny),')','processed')
        chinese_rest_rating = chinese_rest_rating.append({'Borough': Borough,
                                                    'Neighborhood': Neighborhood,
                                                    'ID': id,
                                                    'Name' : name,
                                                    'Likes' : likes,
                                                    'Rating' : rating,
                                                    'Tips' : tips
                                                   }, ignore_index=True)
        count+=1
else:
    print("Data loaded from Local server")
```

Analyzing rating data:

In [366]:

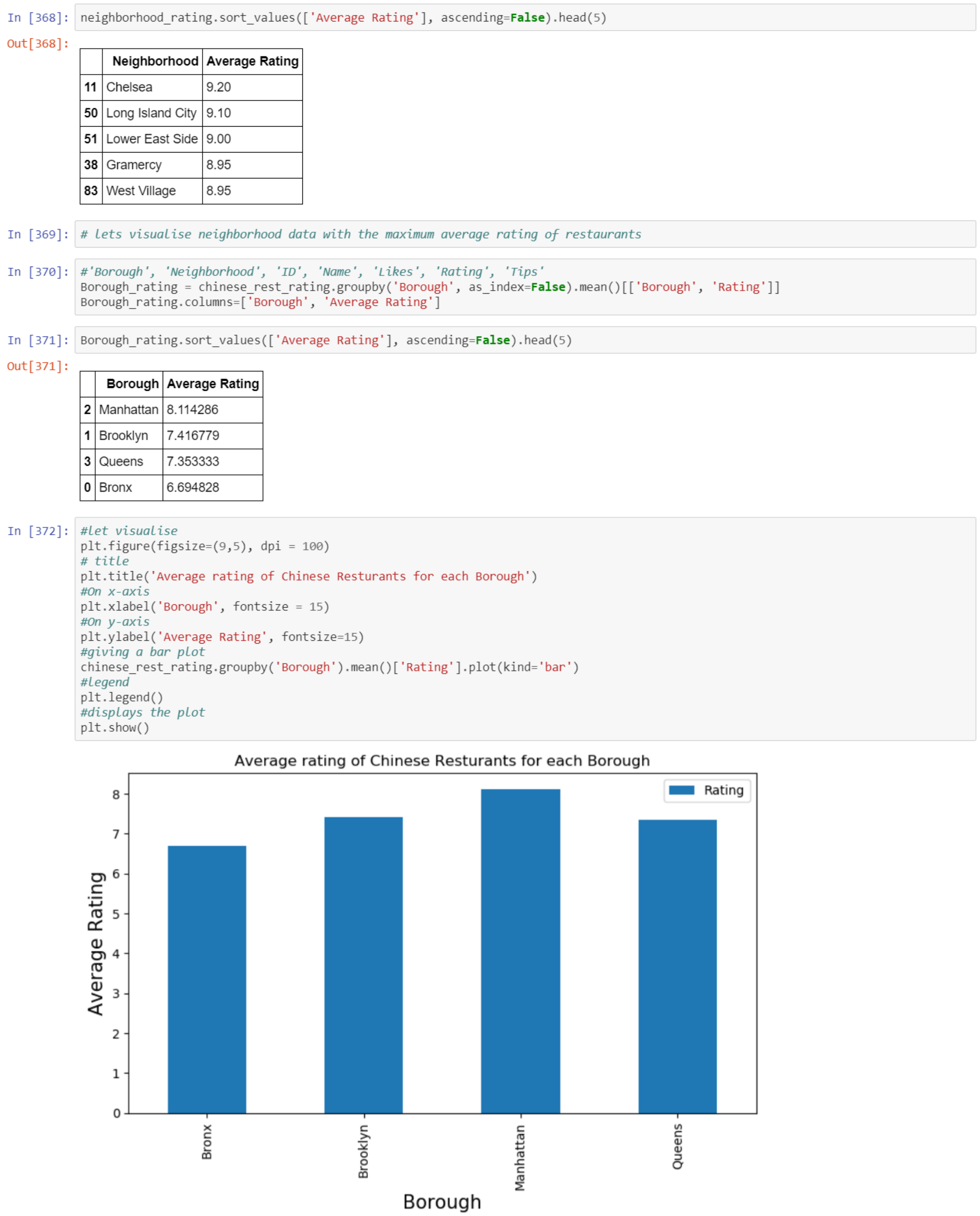
```
# lets visualise neighborhood data with the maximum average rating of restaurants
```

In [367]:

```
#'Borough', 'Neighborhood', 'ID', 'Name', 'Likes', 'Rating', 'Tips'
neighborhood_rating = chinese_rest_rating.groupby('Neighborhood', as_index=False).mean()[['Neighborhood', 'Rating']]
neighborhood_rating.columns = ['Neighborhood', 'Average Rating']
neighborhood_rating
```

Out[367]:

	Neighborhood	Average Rating
0	Bath Beach	6.725000
1	Bay Ridge	7.633333
2	Baychester	6.800000
3	Bedford Park	6.550000
4	Bedford Stuyvesant	7.025000
5	Belmont	7.400000
6	Bensonhurst	7.100000
7	Boerum Hill	8.750000
8	Briarwood	6.900000
9	Cambria Heights	5.700000
10	Central Harlem	7.550000
11	Chelsea	9.200000
12	Chinatown	8.800000
13	City Island	6.800000



```
In [368]: neighborhood_rating.sort_values(['Average Rating'], ascending=False).head(5)
Out[368]:
```

	Neighborhood	Average Rating
11	Chelsea	9.20
50	Long Island City	9.10
51	Lower East Side	9.00
38	Gramercy	8.95
83	West Village	8.95

```
In [369]: # lets visualise neighborhood data with the maximum average rating of restaurants
```

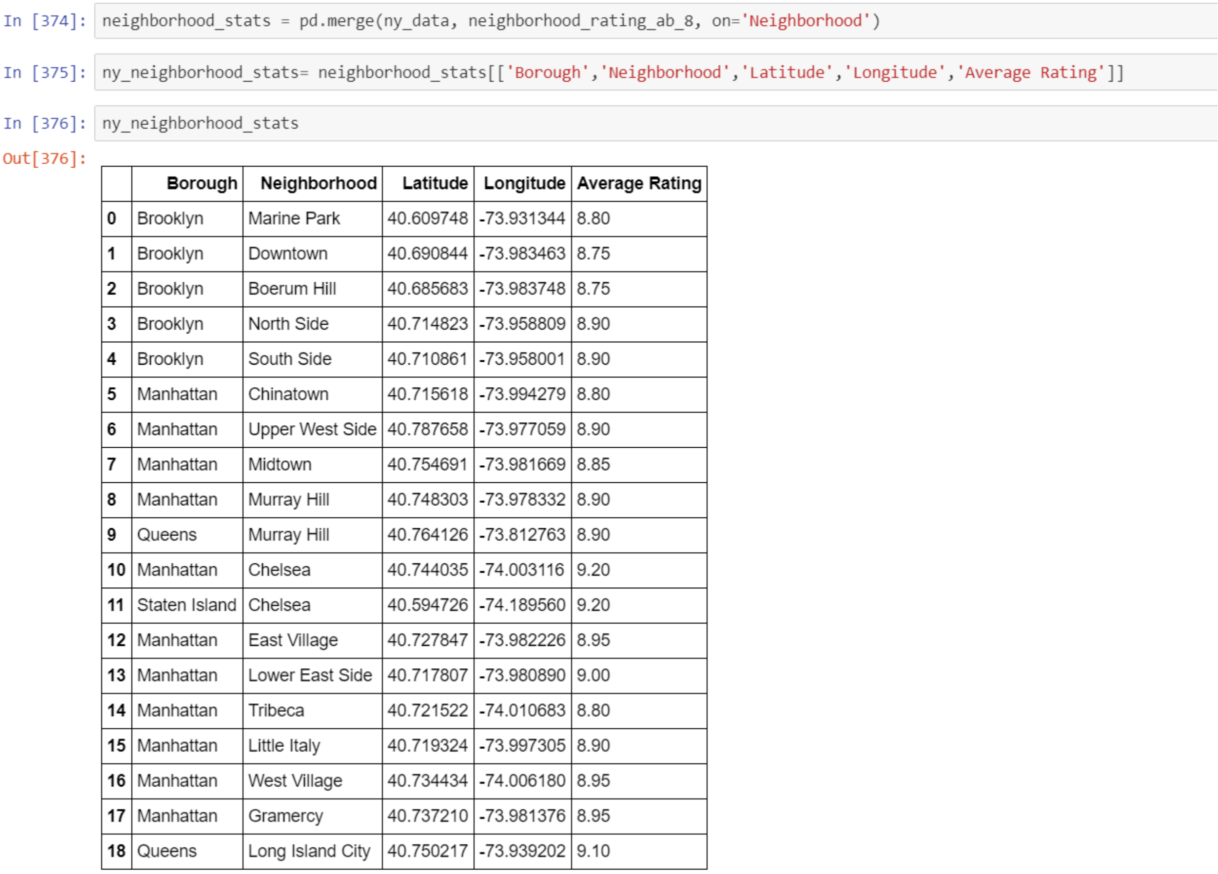
```
In [370]: #'Borough', 'Neighborhood', 'ID', 'Name', 'Likes', 'Rating', 'Tips'
          Borough_rating = chinese_rest_rating.groupby('Borough', as_index=False).mean()[['Borough', 'Rating']]
          Borough_rating.columns=['Borough', 'Average Rating']
```

```
In [371]: Borough_rating.sort_values(['Average Rating'], ascending=False).head(5)
Out[371]:
```

	Borough	Average Rating
2	Manhattan	8.114286
1	Brooklyn	7.416779
3	Queens	7.353333
0	Bronx	6.694828

```
In [372]: #let visualise
          plt.figure(figsize=(9,5), dpi = 100)
          # title
          plt.title('Average rating of Chinese Resturants for each Borough')
          #On x-axis
          plt.xlabel('Borough', fontsize = 15)
          #On y-axis
          plt.ylabel('Average Rating', fontsize=15)
          #giving a bar plot
          chinese_rest_rating.groupby('Borough').mean()['Rating'].plot(kind='bar')
          #legend
          plt.legend()
          #displays the plot
          plt.show()
```

Neighborhood Stats:



```
In [374]: neighborhood_stats = pd.merge(ny_data, neighborhood_rating_ab_8, on='Neighborhood')
```

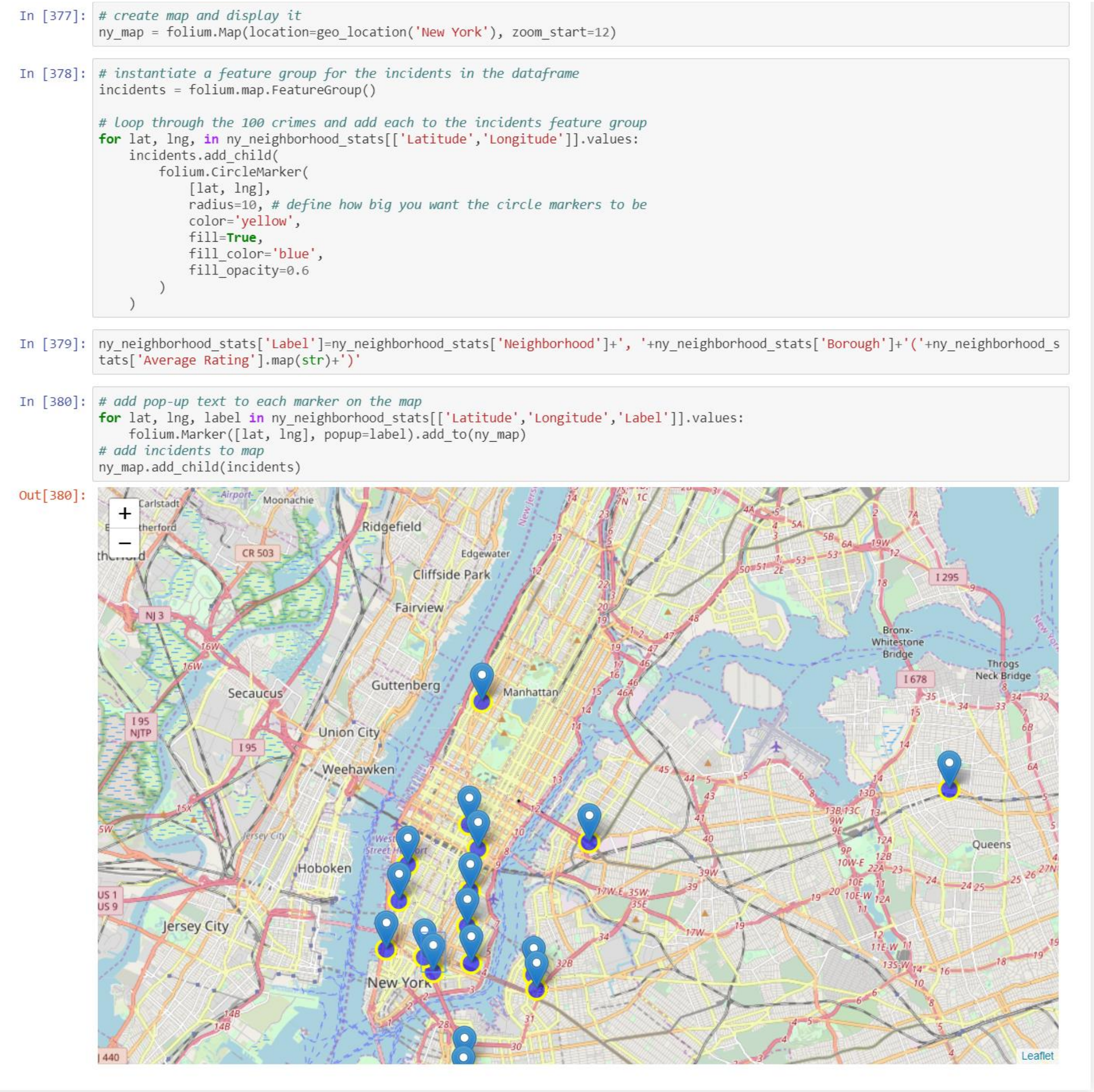
```
In [375]: ny_neighborhood_stats= neighborhood_stats[['Borough','Neighborhood','Latitude','Longitude','Average Rating']]
```

```
In [376]: ny_neighborhood_stats
```

Out[376]:

	Borough	Neighborhood	Latitude	Longitude	Average Rating
0	Brooklyn	Marine Park	40.609748	-73.931344	8.80
1	Brooklyn	Downtown	40.690844	-73.983463	8.75
2	Brooklyn	Boerum Hill	40.685683	-73.983748	8.75
3	Brooklyn	North Side	40.714823	-73.958809	8.90
4	Brooklyn	South Side	40.710861	-73.958001	8.90
5	Manhattan	Chinatown	40.715618	-73.994279	8.80
6	Manhattan	Upper West Side	40.787658	-73.977059	8.90
7	Manhattan	Midtown	40.754691	-73.981669	8.85
8	Manhattan	Murray Hill	40.748303	-73.978332	8.90
9	Queens	Murray Hill	40.764126	-73.812763	8.90
10	Manhattan	Chelsea	40.744035	-74.003116	9.20
11	Staten Island	Chelsea	40.594726	-74.189560	9.20
12	Manhattan	East Village	40.727847	-73.982226	8.95
13	Manhattan	Lower East Side	40.717807	-73.980890	9.00
14	Manhattan	Tribeca	40.721522	-74.010683	8.80
15	Manhattan	Little Italy	40.719324	-73.997305	8.90
16	Manhattan	West Village	40.734434	-74.006180	8.95
17	Manhattan	Gramercy	40.737210	-73.981376	8.95
18	Queens	Long Island City	40.750217	-73.939202	9.10

Let’s plot a map for the same:



In [377]:
```
# create map and display it
ny_map = folium.Map(location=geo_location('New York'), zoom_start=12)
```

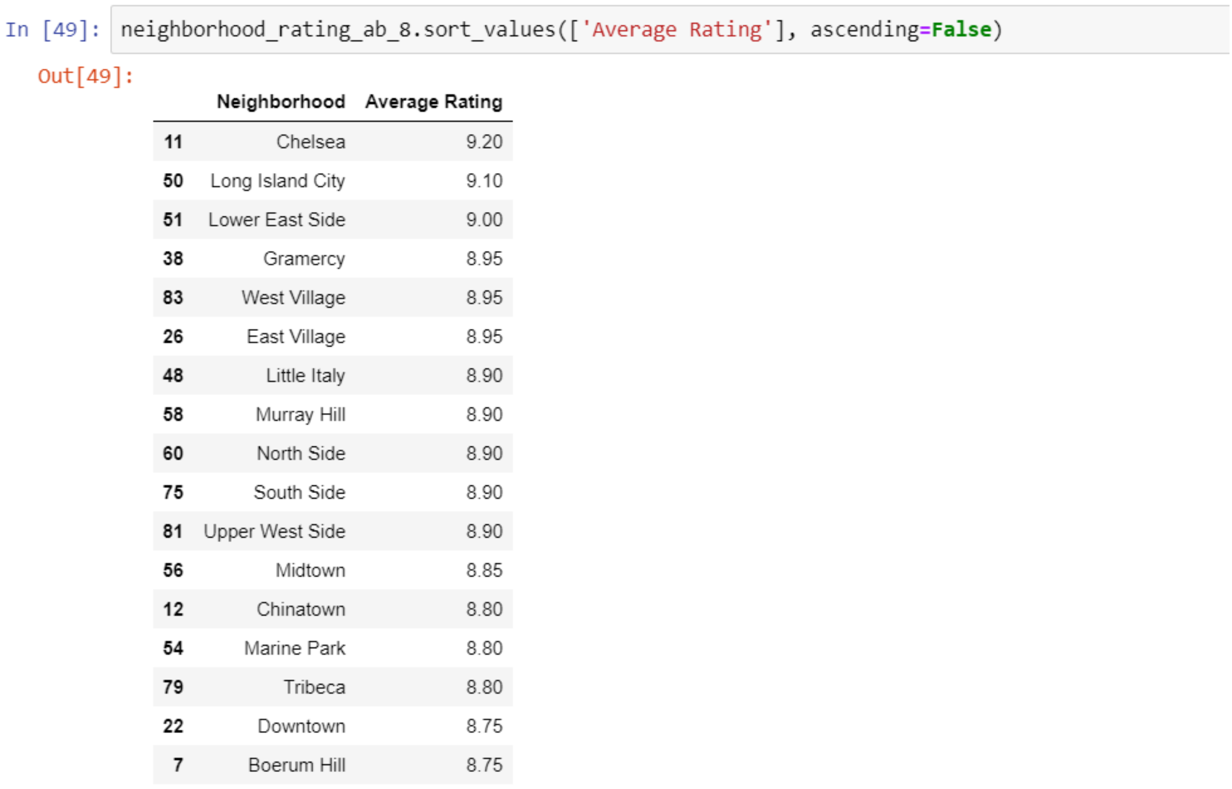
In [378]:
```
# instantiate a feature group for the incidents in the dataframe
incidents = folium.map.FeatureGroup()

# loop through the 100 crimes and add each to the incidents feature group
for lat, lng, in ny_neighborhood_stats[['Latitude','Longitude']].values:
    incidents.add_child(
        folium.CircleMarker(
            [lat, lng],
            radius=10, # define how big you want the circle markers to be
            color='yellow',
            fill=True,
            fill_color='blue',
            fill_opacity=0.6
        )
    )
```

In [379]:
```
ny_neighborhood_stats['Label']=ny_neighborhood_stats['Neighborhood']+', '+ny_neighborhood_stats['Borough']+'('+ny_neighborhood_s
tats['Average Rating'].map(str)+')'
```

In [380]:
```
# add pop-up text to each marker on the map
for lat, lng, label in ny_neighborhood_stats[['Latitude','Longitude','Label']].values:
    folium.Marker([lat, lng], popup=label).add_to(ny_map)
# add incidents to map
ny_map.add_child(incidents)
```

Out[380]:

Neighborhood with the highest Rating:



```
In [49]: neighborhood_rating_ab_8.sort_values(['Average Rating'], ascending=False)
```

Out[49]:

	Neighborhood	Average Rating
11	Chelsea	9.20
50	Long Island City	9.10
51	Lower East Side	9.00
38	Gramercy	8.95
83	West Village	8.95
26	East Village	8.95
48	Little Italy	8.90
58	Murray Hill	8.90
60	North Side	8.90
75	South Side	8.90
81	Upper West Side	8.90
56	Midtown	8.85
12	Chinatown	8.80
54	Marine Park	8.80
79	Tribeca	8.80
22	Downtown	8.75
7	Boerum Hill	8.75

Conclusion:

- Chelsea (Manhattan), Long Island City (Queens) and Lower East side (Manhattan) are the neighborhood with the highest rating Chinese restaurants.
- Manhattan have the potential market for the Chinese restaurants.
- Manhattan is the best place if you want to open a restaurant.