Identifying potential Business opportunities in Bronx Borough, New York City

Introduction to the Business idea

The business proposition I would like to take on will be in the realm of providing clients with information on potential business opportunities available in New York in general and Bronx Burough in particular. The information will be based on foursquare data on venues by category that and location by neighbourhood in food and tourism industry. The potential clients to be targeted will be those that are planning to establish ventures that deal in food and tourism related services in Bronx Burough of New York. To that end, my service will be tailored toward delivering background and general informatio to clients about potential neighbourhoods which potential clients will use as initial input into the decision process that will inform their feasibility studies related to establising food and tourism related businesses in Bronx Burough of New York, USA.

Description of the dataset to use

To obtain information that will provide insights to potential clients, the project will use data to be obtained from foursquare application programming interface(API). The data is based on personal information that visitors of venues make that include what the venue has to offer, category in which the venues falls, personal impressions about the venue, whether the venue was liked or otherwise, tips for others interested in getting the best experience , among others. My main focus was on venues tha fell into the food and tourism attraction categories.

Methodology

The approach used to glean insights from the data included but not limited to: importing packages and dependencies such as numpy, pandas (necessary to manipulate data for easier presentation in arrays and dataframes; json(since fpourswuare data is in json file format);geocoder application that was used to generate latitudes and longitudes (coordinates) based on addresses, request application which was used to scrape html data and transform it into text for analysis; matplolib, which was used in plotting the insights gleaned from extracted data; and folium, which was used to draw maps of the locations using geographical coordinates. Insights were clustered into four potential segments using K means machine learning application which is part of sklearn library. Results were presented which later formed the basis of recommendations for potential clients.

Presentation of the report

The first step was to import all the libraries deemed import in the downloading, processing, tabulating, mapping and clustering the data; presenting the results, followed by discussion of the results and drawing conclusions and recommendations.

### Importing the packages

```
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

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

!conda install -c conda-forge folium=0.5.0 --yes # uncomment this line if you haven't completed the
Foursquare API lab
import folium # map rendering library

print('Libraries imported.')
```

```
## Getting the data and taking a sneakpeek at  it
```

In [2]:

```
#Get data
York_data = open('newyork_data.json')
```

In [3]:

```
with open('newyork_data.json') as nyk_city_data:
    nwyk_data = json.load(nyk_city_data)
```

```
In [4]: nwyk_data
```

```
Out[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]}},
          {'type': 'Feature',
           'id': 'nyu_2451_34572.2',
```

In [7]:
```
neighborhoods_nwyk = nwyk_data['features']
neighborhoods_nwyk
```

Out[7]:
```
[{'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]}},
 {'type': 'Feature',
  'id': 'nyu_2451_34572.2',
  'geometry': {'type': 'Point',
   'coordinates': [-73.82993910812398, 40.87429419303012]}
```

In [8]:

```
neighborhoods_nwyk[0:20] # Taking a snapshot at New York neighborhoods
```

Out[8]:

```
[{'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]}},
 {'type': 'Feature',
  'id': 'nyu_2451_34572.2',
  'geometry': {'type': 'Point',
   'coordinates': [-73.82993910812398, 40.87429419303012]},
```

Creating dataframe for New York data

In [9]:

```python
# create an empty dataframw to  use in  populating  neighborhood data

neighborhoods = pd.DataFrame(columns=['Borough', 'Neighborhood', 'Latitude', 'Longitude'])


for data in neighborhoods_nwyk:
    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)
```

In [10]:

```python
neighborhoods.head()
```

Out[10]:

	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 [52]:

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

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

In [11]:

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

```
C:\Users\Zubairmap\Anaconda3\Anaconda31\lib\site-packages\ipykernel_launcher.py:2: DeprecationWarni
ng: Using Nominatim with the default "geopy/1.18.1" `user_agent` is strongly discouraged, as it vio
lates Nominatim's ToS https://operations.osmfoundation.org/policies/nominatim/ (https://operations.
osmfoundation.org/policies/nominatim/) and may possibly cause 403 and 429 HTTP errors. Please speci
fy a custom `user_agent` with `Nominatim(user_agent="my-application")` or by overriding the default
`user_agent`: `geopy.geocoders.options.default_user_agent = "my-application"`. In geopy 2.0 this wi
ll become an exception.


The geograpical coordinate of New York City are 40.7308619, -73.9871558.
```

Generating a map of New york using folium package

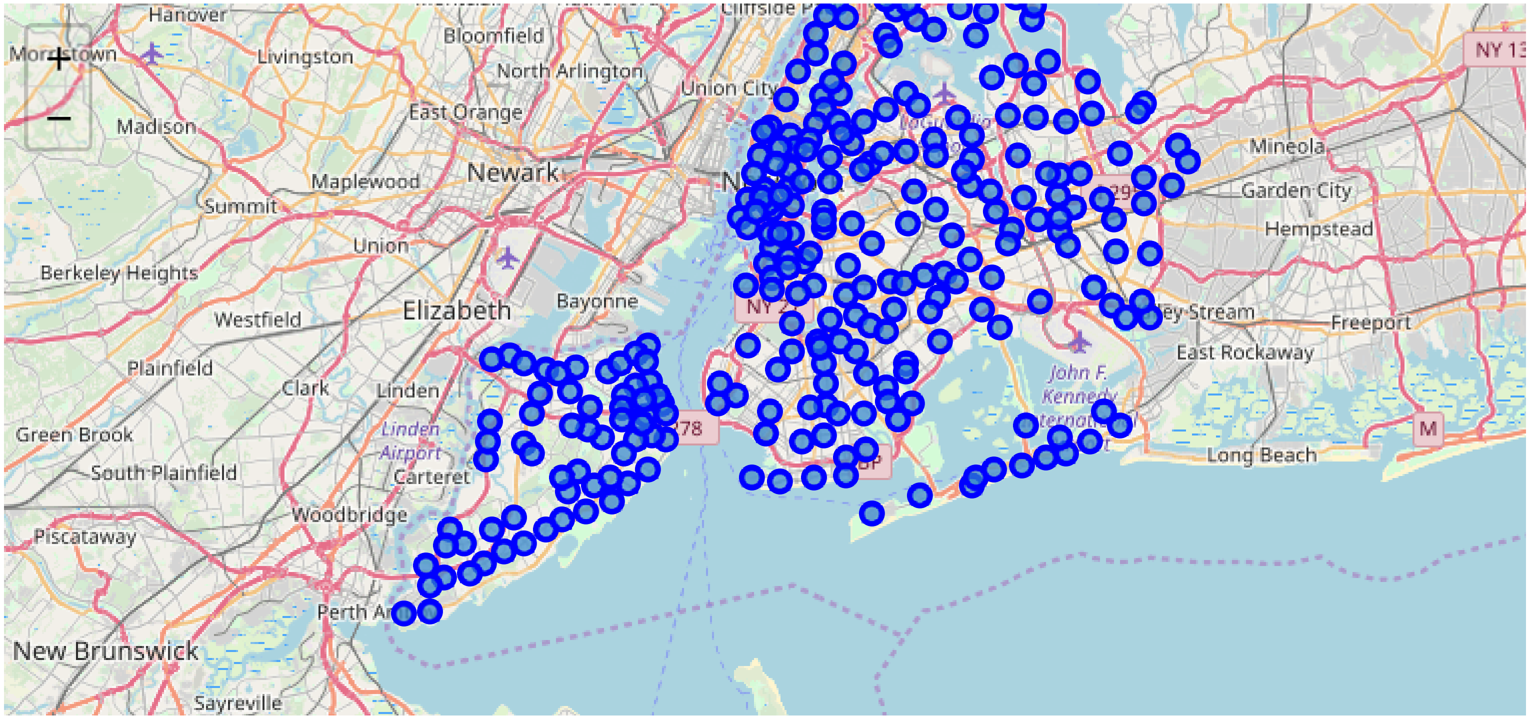
In [12]:

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

# add markers to map
for lat, lng, borough, neighborhood in zip(neighborhoods['Latitude'], neighborhoods['Longitude'], neigh
    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(newyork_map)

newyork_map
```

Out[12]:



In [13]:

```
CLIENT_ID = 'PISLAD2G24MAO0MDXVTSKFIXVIJXWFDLAQYPABPTXPYBITQ5' # your Foursquare ID
CLIENT_SECRET = '01OX3V13KQA3XBWW0OGOMEI0MJ3MKUY22BQEHY1NQE1000I3' # your Foursquare Secret
VERSION = '20180605' # Foursquare API version

print('Muyanja Ssenyonga:')
print('CLIENT_ID: ' + CLIENT_ID)
print('CLIENT_SECRET:' + CLIENT_SECRET)
```

```
Muyanja Ssenyonga:
CLIENT_ID: PISLAD2G24MAO0MDXVTSKFIXVIJXWFDLAQYPABPTXPYBITQ5
CLIENT_SECRET:01OX3V13KQA3XBWW0OGOMEI0MJ3MKUY22BQEHY1NQE1000I3
```

In [14]:

```
manhattan = neighborhoods[neighborhoods['Borough'] == 'Manhattan'].reset_index(drop=True)
manhattan.head(20)
```

Out[14]:

	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
5	Manhattan	Manhattanville	40.816934	-73.957385
6	Manhattan	Central Harlem	40.815976	-73.943211
7	Manhattan	East Harlem	40.792249	-73.944182
8	Manhattan	Upper East Side	40.775639	-73.960508
9	Manhattan	Yorkville	40.775930	-73.947118
10	Manhattan	Lenox Hill	40.768113	-73.958860
11	Manhattan	Roosevelt Island	40.762160	-73.949168
12	Manhattan	Upper West Side	40.787658	-73.977059
13	Manhattan	Lincoln Square	40.773529	-73.985338
14	Manhattan	Clinton	40.759101	-73.996119
15	Manhattan	Midtown	40.754691	-73.981669
16	Manhattan	Murray Hill	40.748303	-73.978332
17	Manhattan	Chelsea	40.744035	-74.003116
18	Manhattan	Greenwich Village	40.726933	-73.999914
19	Manhattan	East Village	40.727847	-73.982226

In [15]:

```
neighborhood_latitude = manhattan.loc[0, 'Latitude'] # neighborhood latitude value
neighborhood_longitude = manhattan.loc[0, 'Longitude'] # neighborhood longitude value

neighborhood_name = manhattan.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.
```

In [16]:

```
neighborhood_latitude = manhattan.loc[10, 'Latitude'] # neighborhood latitude value
neighborhood_longitude = manhattan.loc[10, 'Longitude'] # neighborhood longitude value

neighborhood_name = manhattan.loc[10, 'Neighborhood'] # neighborhood name

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

```
Latitude and longitude values of Lenox Hill are 40.76811265828733, -73.9588596881376.
```

In [17]:

```
# htttps://type your answer here
LIMIT = 50

radius = 500
url = 'https://api.foursquare.com/v2/venues/explore?&client_id={}&client_secret={}&v={}&ll={},{}&radiu
    CLIENT_ID,
    CLIENT_SECRET,
    VERSION,
    neighborhood_latitude,
    neighborhood_longitude,
    radius,
    LIMIT)
```

## Parsing json data into easily readible and understandable text

```
In [18]: results = requests.get(url).json()
         results
```

```
Out[18]: {'meta': {'code': 200, 'requestId': '5c32bbd5351e3d262428d05e'},
          'response': {'suggestedFilters': {'header': 'Tap to show:',
            'filters': [{'name': 'Open now', 'key': 'openNow'}]},
           'headerLocation': 'Upper East Side',
           'headerFullLocation': 'Upper East Side, New York',
           'headerLocationGranularity': 'neighborhood',
           'totalResults': 165,
           'suggestedBounds': {'ne': {'lat': 40.77261266278733,
             'lng': -73.95292907281076},
            'sw': {'lat': 40.76361265378733, 'lng': -73.96479030346443}},
           'groups': [{'type': 'Recommended Places',
             'name': 'recommended',
             'items': [{'reasons': {'count': 0,
                'items': [{'summary': 'This spot is popular',
                  'type': 'general',
                  'reasonName': 'globalInteractionReason'}]},
               'venue': {'id': '5230a2787e48c8a77ab092d3',
                'name': 'Whiskey & Wine Off 69',
                'location': {'address': '1321 2nd Ave',
                 'crossStreet': 'btwn 69th and 70th St',
```

Delving into Bronx Burough for the insights

```
In [19]: CLIENT_ID = 'PISLAD2G24MAO0MDXVTSKFIXVIJXWFDLAQYPABPTXPYBITQ5' # your Foursquare ID
         CLIENT_SECRET = '01OX3V13KQA3XBWW0OGOMEI0MJ3MKUY22BQEHY1NQE1000I3' # your Foursquare Secret
         VERSION = '20180605' # Foursquare API version

         print('Muyanja Ssenyonga:')
         print('CLIENT_ID: ' + CLIENT_ID)
         print('CLIENT_SECRET:' + CLIENT_SECRET)
```

```
Muyanja Ssenyonga:
CLIENT_ID: PISLAD2G24MAO0MDXVTSKFIXVIJXWFDLAQYPABPTXPYBITQ5
CLIENT_SECRET:01OX3V13KQA3XBWW0OGOMEI0MJ3MKUY22BQEHY1NQE1000I3
```

```
In [20]: Bronx = neighborhoods[neighborhoods['Borough'] == 'Bronx'].reset_index(drop=True)
         Bronx.head(20)
```

Out[20]:

	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	Bronx	Kingsbridge	40.881687	-73.902818
6	Bronx	Woodlawn	40.898273	-73.867315
7	Bronx	Norwood	40.877224	-73.879391
8	Bronx	Williamsbridge	40.881039	-73.857446
9	Bronx	Baychester	40.866858	-73.835798
10	Bronx	Pelham Parkway	40.857413	-73.854756
11	Bronx	City Island	40.847247	-73.786488
12	Bronx	Bedford Park	40.870185	-73.885512
13	Bronx	University Heights	40.855727	-73.910416
14	Bronx	Morris Heights	40.847898	-73.919672
15	Bronx	Fordham	40.860997	-73.896427
16	Bronx	East Tremont	40.842696	-73.887356
17	Bronx	West Farms	40.839475	-73.877745
18	Bronx	High Bridge	40.836623	-73.926102
19	Bronx	Melrose	40.819754	-73.909422

```
In [21]: neighborhood_latitude = Bronx.loc[10, 'Latitude'] # neighborhood latitude value
         neighborhood_longitude = Bronx.loc[10, 'Longitude'] # neighborhood longitude value

         neighborhood_name = Bronx.loc[10, 'Neighborhood'] # neighborhood name

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

```
Latitude and longitude values of Pelham Parkway are 40.85741349808865, -73.85475564017999.
```

```
In [22]: results = requests.get(url).json()
         results
```

```
Out[22]: {'meta': {'code': 200, 'requestId': '5c32bbe5db04f5569239fca3'},
          'response': {'suggestedFilters': {'header': 'Tap to show:',
            'filters': [{'name': '$-$$$$', 'key': 'price'},
             {'name': 'Open now', 'key': 'openNow'}]},
           'headerLocation': 'Upper East Side',
           'headerFullLocation': 'Upper East Side, New York',
           'headerLocationGranularity': 'neighborhood',
           'totalResults': 165,
           'suggestedBounds': {'ne': {'lat': 40.77261266278733,
             'lng': -73.95292907281076},
            'sw': {'lat': 40.76361265378733, 'lng': -73.96479030346443}},
           'groups': [{'type': 'Recommended Places',
             'name': 'recommended',
             'items': [{'reasons': {'count': 0,
                'items': [{'summary': 'This spot is popular',
                  'type': 'general',
                  'reasonName': 'globalInteractionReason'}]},
               'venue': {'id': '5230a2787e48c8a77ab092d3',
                'name': 'Whiskey & Wine Off 69',
                'location': {'address': '1321 2nd Ave',
```

Results

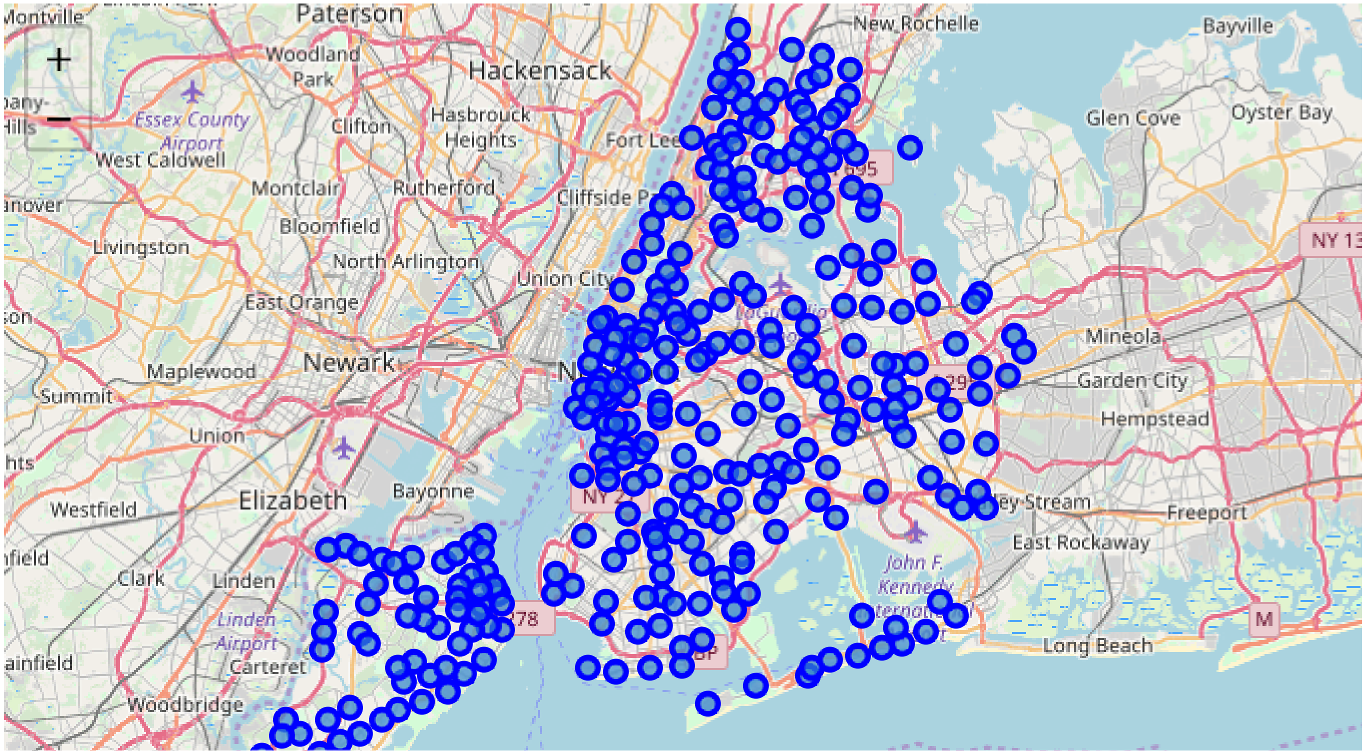
In [23]:

```
# creating map of Bronx Burough
# create map of New York using latitude and longitude values
Bronx_map = folium.Map(location=[latitude, longitude], zoom_start=10)

# add markers to map
for lat, lng, borough, neighborhood in zip(neighborhoods['Latitude'], neighborhoods['Longitude'], neigh
    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(Bronx_map)

Bronx_map
```

Out[23]:



### A sneakpeek at venue categories in Bronx by name, category, and geographical location.

Foursquare data returned 2375 venues in all that ranged from dessert joints, restaurants, wine shops, fast food stalls, places serving Tacos, among others.

In [24]:

```
# a sneakpeek at venue categories in Bronx
# 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']
```

In [29]:

```python
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(15)
```

Out[29]:

	name	categories	lat	lng
0	Whiskey & Wine Off 69	Liquor Store	40.767272	-73.959544
1	Up Thai	Thai Restaurant	40.769898	-73.957598
2	sweetgreen	Salad Place	40.767128	-73.956846
3	Cigar Inn	Smoke Shop	40.768776	-73.956222
4	Anthropologie	Women's Store	40.769296	-73.961085
5	A Matter of Health	Health Food Store	40.768028	-73.955933
6	Marymount Manhattan College	College Academic Building	40.769232	-73.959658
7	La Esquina	Taco Place	40.769451	-73.957782
8	Donna Margherita	Pizza Place	40.766452	-73.959905
9	Bohemian National Hall	Art Gallery	40.769003	-73.956666
10	Equinox East 74th Street	Gym	40.770460	-73.957387
11	Bohemian Spirit Restaurant	Czech Restaurant	40.768989	-73.956662
12	Refine Method	Gym	40.767315	-73.956887
13	Dr. Wine	Wine Shop	40.766071	-73.957119
14	Tone House UES	Gym / Fitness Center	40.768988	-73.960732

In [26]:

```
print('{} venues were returned by Foursquare.'.format(nearby_venues.shape[0]))
```

```
50 venues were returned by Foursquare.
```

### Nearby venues with the radius of 1000 meters

In [33]:

```
def getNearbyVenues(names, latitudes, longitudes, radius=1000):

    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={},
            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)
```

In [34]:

```
# creating dataframe to contain nearby venues within a radius of 1000 meters
Bronx_venues = getNearbyVenues(names=Bronx['Neighborhood'],
                                   latitudes=Bronx['Latitude'],
                                   longitudes=Bronx['Longitude']
                                  )
```

```
Wakefield
Co-op City
Eastchester
Fieldston
Riverdale
Kingsbridge
Woodlawn
Norwood
Williamsbridge
Baychester
Pelham Parkway
City Island
Bedford Park
University Heights
Morris Heights
Fordham
East Tremont
West Farms
High  Bridge
Melrose
Mott Haven
Port Morris
Longwood
Hunts Point
Morrisania
Soundview
Clason Point
Throgs Neck
Country Club
Parkchester
Westchester Square
Van Nest
Morris Park
Belmont
Spuyten Duyvil
North Riverdale
```

```
In [38]: print(Bronx_venues.shape)
         Bronx_venues.head()
```

```
(2375, 7)
```

Out[38]:

	Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
0	Wakefield	40.894705	-73.847201	Lollipops Gelato	40.894123	-73.845892	Dessert Shop
1	Wakefield	40.894705	-73.847201	Ripe Kitchen & Bar	40.898152	-73.838875	Caribbean Restaurant
2	Wakefield	40.894705	-73.847201	Jackie's West Indian Bakery	40.889283	-73.843310	Caribbean Restaurant
3	Wakefield	40.894705	-73.847201	Rite Aid	40.896521	-73.844680	Pharmacy
4	Wakefield	40.894705	-73.847201	Ali's Roti Shop	40.894036	-73.856935	Caribbean Restaurant

```
There are 2375 neaby venues categorized by neighborhood, venue category and venue coordinates
```

In [39]:

```
print(Bronx_venues.shape)
Bronx_venues.tail(10)
```

```
(2375, 7)
```

Out[39]:

	Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
2365	Kingsbridge Heights	40.870392	-73.901523	Blink Fitness Riverdale	40.877147	-73.905837	Gym
2366	Kingsbridge Heights	40.870392	-73.901523	Lehman Center For The Performing Arts	40.872638	-73.893407	Theater
2367	Kingsbridge Heights	40.870392	-73.901523	Panda Express	40.863001	-73.900894	Chinese Restaurant
2368	Kingsbridge Heights	40.870392	-73.901523	Poe Park	40.865502	-73.892300	Park
2369	Kingsbridge Heights	40.870392	-73.901523	Lot Less Closeouts	40.878270	-73.905265	Discount Store
2370	Kingsbridge Heights	40.870392	-73.901523	Broadway Pizza & Pasta	40.878822	-73.904494	Pizza Place
2371	Kingsbridge Heights	40.870392	-73.901523	Fordham Restaurant	40.862734	-73.896778	Diner
2372	Kingsbridge Heights	40.870392	-73.901523	Lucille Roberts	40.862049	-73.899981	Gym
2373	Kingsbridge Heights	40.870392	-73.901523	Diamante Poblano 1 Bar & Restaurant	40.861872	-73.901596	Restaurant
2374	Kingsbridge Heights	40.870392	-73.901523	Celia's Restaurant Bar & Grill	40.862590	-73.907201	Latin American Restaurant

In [40]:
```
Bronx_venues.groupby('Neighborhood').count()
```

Out[40]:

	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
Neighborhood						
Allerton	50	50	50	50	50	50
Baychester	50	50	50	50	50	50
Bedford Park	50	50	50	50	50	50
Belmont	50	50	50	50	50	50
Bronxdale	50	50	50	50	50	50
Castle Hill	41	41	41	41	41	41
City Island	39	39	39	39	39	39
Claremont Village	15	15	15	15	15	15
Clason Point	8	8	8	8	8	8
Co-op City	50	50	50	50	50	50
Concourse	50	50	50	50	50	50
Concourse Village	50	50	50	50	50	50
Country Club	50	50	50	50	50	50
East Tremont	50	50	50	50	50	50
Eastchester	50	50	50	50	50	50
Edenwald	41	41	41	41	41	41
Edgewater Park	41	41	41	41	41	41
Fieldston	47	47	47	47	47	47
Fordham	50	50	50	50	50	50
High Bridge	38	38	38	38	38	38
Hunts Point	21	21	21	21	21	21
Kingsbridge	50	50	50	50	50	50
Kingsbridge Heights	31	31	31	31	31	31
Longwood	50	50	50	50	50	50

The number of venues in Bronx based on neighborhood categorization are 8 in Clason point (the smallest number), 15 in claremont, 21 in Hnts point, 31 in Throgs Neck and Knightsbridge heights, 39 in City Island, 45 each in Wakefield and Riverdale, 46 each in Pelham Gardens and North Riverdale, 47 in West Frams and Fieldston, 49 in Soundview, and 50 venues each in the remaining neighborhoods (the largest number).

In [41]:
```
print('There are {} uniques categories.'.format(len(Bronx_venues['Venue Category'].unique())))
```

```
There are 213 uniques categories.
```

In [42]:
```
# Analyzing the Uniques venues
Bronx_hotties = pd.get_dummies(Bronx_venues[['Venue Category']], prefix="", prefix_sep="")

# add neighborhood column back to dataframe
Bronx_hotties['Neighborhood'] = Bronx_venues['Neighborhood']

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

Bronx_hotties.head()
```

Out[42]:

	Zoo Exhibit	Accessories Store	African Restaurant	Airport Tram	American Restaurant	Arcade	Arepa Restaurant	Art Gallery	Art Museum	Arts & Crafts Store	Asian Restaurant	Athletics & Sports	Aut Dealershi
0	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	0
2	0	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	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0

Discussion

The list of unique venue categories included : Zoo Exhibit Accessories Store African Restaurant Airport Tram American Restaurant Arcade Arepa Restaurant Art Gallery Art Museum Arts & Crafts Store Asian Restaurant Athletics & Sports Auto Dealership Auto Workshop BBQ Joint Bagel Shop Bakery Bank Bar Baseball Field Baseball Stadium Basketball Court Beach Beer Bar Beer Garden Boat or Ferry Bookstore Border Crossing Botanical Garden Bowling Alley Breakfast Spot Brewery Bridge Buffet Burger Joint Burrito Place Bus Line Bus Station Bus Stop Business Service Café Candy Store Caribbean Restaurant Check Cashing Service Cheese Shop Chinese Restaurant Clothing Store Cocktail Bar Coffee Shop Comfort Food Restaurant Construction & Landscaping Convenience Store Cosmetics Shop Cuban Restaurant Cupcake Shop Dance Studio Deli / Bodega Dentist's Office Department Store Dessert Shop Diner Discount Store Distillery Dive Bar Dog Run Donut Shop Dumpling Restaurant Eastern European Restaurant Electronics Store Empanada Restaurant Farmers Market Fast Food Restaurant Fish & Chips Shop Fish Market Food Food & Drink Shop Food Stand Food Truck French Restaurant Fried Chicken Joint Frozen Yogurt Shop Fruit & Vegetable Store Furniture / Home Store Garden Gas Station Gift Shop Golf Course Gourmet Shop Greek Restaurant Grocery Store Gym Gym / Fitness Center Harbor / Marina Hardware Store Health & Beauty Service Health Food Store Historic Site History Museum Hobby Shop Home Service Hookah Bar Hot Dog Joint Hotel Ice Cream Shop Indian Restaurant Indie Theater Intersection Italian Restaurant Japanese Restaurant Jewelry Store Juice Bar Kids Store Kitchen Supply Store Latin American Restaurant Laundromat Lingerie Store Liquor Store Lounge Mac & Cheese Joint Market Martial Arts Dojo Mattress Store Medical Center Men's Store Metro Station Mexican Restaurant Middle Eastern Restaurant Miscellaneous Shop Mobile Phone Shop Motel Movie Theater Moving Target Music Store Music Venue Nature Preserve Neighborhood New American Restaurant Nightclub Nightlife Spot Optical Shop Other Great Outdoors Other Nightlife Paella Restaurant Paper / Office Supplies Store Park Performing Arts Venue Peruvian Restaurant Pet Store Pharmacy Photography Studio Pizza Place Playground Plaza Pool Pool Hall Pub Public Art Rental Car Location Restaurant River Salon / Barbershop Sandwich Place Scenic Lookout School Sculpture Garden Seafood Restaurant Shipping Store Shoe Store Shopping Mall Shopping Plaza Smoke Shop Snack Place Soccer Field Soup Place Southern / Soul Food Restaurant Spa Spanish Restaurant Sporting Goods Shop Sports Bar State / Provincial Park Steakhouse Supermarket Supplement Shop Sushi Restaurant Tapas Restaurant Tattoo Parlor Tennis Court Tennis Stadium Thai Restaurant Theater Theme Park Theme Park Ride / Attraction Theme Restaurant Thrift / Vintage Store Toll Plaza Track Track Stadium Trail Train Station Travel & Transport Tunnel Vegetarian / Vegan Restaurant Video Game Store Video Store Vietnamese Restaurant Volleyball Court Warehouse Store Waste Facility Wine Shop Wings Joint Women's Store Yoga Studio and Zoo. Meanwhile , based on the count of venues in Bronx by neighborhood, in Clason point has the smallest number (8), while claremont has 15 there were 21 in Hunts point, 31 in Throgs Neck and Knightsbridge heights, 39 in City Island, 45 each in Wakefield and Riverdale, 46 each in Pelham Gardens and North Riverdale, 47 in West Frams and Fieldston, 49 in Soundview, and 50 venues each in the remaining neighborhoods (the largest number).

```
Meanwhile based on the mean of the spread of 213 unique venues is concentrated in 52 neighbourhoods in
Bronx is presented below
```

In [43]:
```
Bronx_grouped = Bronx_hotties.groupby('Neighborhood').mean().reset_index()
Bronx_grouped
```

Out[43]:

	Neighborhood	Zoo Exhibit	Accessories Store	African Restaurant	Airport Tram	American Restaurant	Arcade	Arepa Restaurant	Art Gallery	Art Museum	Arts & Crafts Store	Asian Restaurant
0	Allerton	0.000000	0.00	0.00	0.000000	0.000000	0.000000	0.00	0.000000	0.000000	0.00000	0.000000
1	Baychester	0.000000	0.02	0.00	0.000000	0.000000	0.000000	0.00	0.000000	0.000000	0.00000	0.000000
2	Bedford Park	0.000000	0.00	0.00	0.000000	0.000000	0.000000	0.00	0.000000	0.000000	0.00000	0.000000
3	Belmont	0.000000	0.00	0.00	0.000000	0.000000	0.000000	0.00	0.000000	0.000000	0.00000	0.000000
4	Bronxdale	0.000000	0.00	0.00	0.000000	0.000000	0.000000	0.02	0.000000	0.000000	0.00000	0.000000
5	Castle Hill	0.000000	0.00	0.00	0.000000	0.000000	0.000000	0.00	0.000000	0.000000	0.02439	0.000000
6	City Island	0.000000	0.00	0.00	0.000000	0.051282	0.000000	0.00	0.000000	0.000000	0.00000	0.000000
7	Claremont Village	0.000000	0.00	0.00	0.000000	0.000000	0.000000	0.00	0.000000	0.000000	0.00000	0.000000
8	Clason Point	0.000000	0.00	0.00	0.000000	0.000000	0.000000	0.00	0.000000	0.000000	0.00000	0.000000

In [44]:
```
Bronx_grouped.shape
```

Out[44]:
```
(52, 213)
```

In [45]:

```
num_top_venues = 5

for hood in Bronx_grouped['Neighborhood']:
    print("----"+hood+"----")
    temp = Bronx_grouped[Bronx_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')
```

```
----Allerton----
                  venue  freq
0  Fast Food Restaurant  0.10
1            Donut Shop  0.08
2           Pizza Place  0.08
3        Sandwich Place  0.06
4           Bus Station  0.06


----Baychester----
              venue  freq
0  Department Store  0.08
1    Clothing Store  0.06
2     Shopping Mall  0.06
3    Discount Store  0.06
4       Pizza Place  0.04


----Bedford Park----
              venue  freq
```

In [46]:

```
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]
```

### Presentation of ten most common venues in Bronx by neighborhood

In [47]:

```
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'] = Bronx_grouped['Neighborhood']

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

neighborhoods_venues_sorted
```

Out[47]:

	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
0	Allerton	Fast Food Restaurant	Pizza Place	Donut Shop	Sandwich Place	Bus Station	Pharmacy	Bank	Dessert Shop	Caribbean Restaurant
1	Baychester	Department Store	Clothing Store	Discount Store	Shopping Mall	Supermarket	Kids Store	Bakery	Bagel Shop	Italian Restaurant
2	Bedford Park	Park	Diner	Botanical Garden	Supermarket	Garden	Pizza Place	Mexican Restaurant	Donut Shop	Chinese Restaurant
3	Belmont	Italian Restaurant	Bakery	Deli / Bodega	Pizza Place	Dessert Shop	Bar	Food & Drink Shop	Café	Fast Food Restaurant
4	Bronxdale	Pizza Place	Italian Restaurant	Coffee Shop	Chinese Restaurant	Park	Deli / Bodega	Bank	Donut Shop	Food Truck
5	Castle Hill	Bus Stop	Pizza Place	Bus Station	Baseball Field	Deli / Bodega	Wine Shop	Spanish Restaurant	Fried Chicken Joint	Liquor Store
		Harbor /	Seafood		Italian		Thrift /	American	Chinese	

```
Based on ten most unique venues in the 52 neighborhoods, while most are good for food related business
some are not that much. This includes Coop city, Clarement vilage, Morris heights, Morrisania, high
bridge, University heights, and Baychester. Meanwhile as relates to tourism development,  all the 52
neighborhoods have the potential to become very good food , shopping, exercising, walking and jogging,
and travelling tourist attractions.
```

### Picking through the haystack: Clustering Bronx neighborhoods

In [48]:

```
# setting  number of clusters
kclusters = 4

Bronx_clustering = Bronx_grouped.drop('Neighborhood', 1)

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

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

Out[48]:

```
array([0, 3, 1, 3, 3, 1, 1, 2, 2, 0])
```

In [49]:

```
Bronx_merged = Bronx

# add clustering labels
Bronx_merged['Cluster Labels'] = kmeans.labels_

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

Bronx_merged.head(10) # check the last columns!
```

Out[49]:

	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 Most Common Venue	7th M Comm Ve
0	Bronx	Wakefield	40.894705	-73.847201	0	Pharmacy	Caribbean Restaurant	Fast Food Restaurant	Donut Shop	Supermarket	Fried Chicken Joint	Busin Serv
1	Bronx	Co-op City	40.874294	-73.829939	3	Shoe Store	Shopping Mall	Mobile Phone Shop	Mattress Store	Bakery	Department Store	Pi Pl
2	Bronx	Eastchester	40.887556	-73.827806	1	Caribbean Restaurant	Diner	Fast Food Restaurant	Shopping Mall	Donut Shop	Supplement Shop	Wome St
3	Bronx	Fieldston	40.895437	-73.905643	3	Pizza Place	Chinese Restaurant	Deli / Bodega	Bar	Plaza	Mexican Restaurant	I Sta
4	Bronx	Riverdale	40.890834	-73.912585	3	Pizza Place	Japanese Restaurant	Playground	Plaza	Park	Mexican Restaurant	Co Sl
5	Bronx	Kingsbridge	40.881687	-73.902818	1	Pizza Place	Mexican Restaurant	Bakery	Japanese Restaurant	Coffee Shop	Diner	Spar Restau

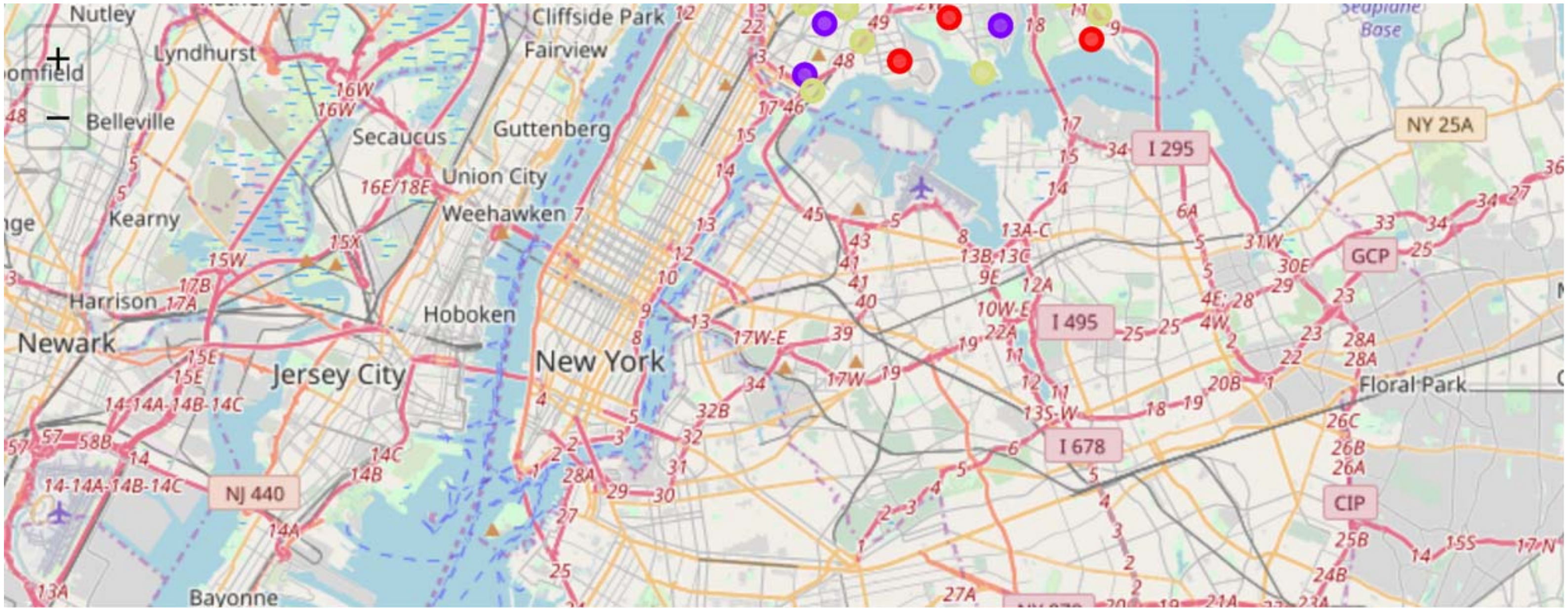
```
In [50]: # creating a map to identify clusters
         cluster_map = folium.Map(location=[latitude, longitude], zoom_start=11)

         # 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(Bronx_merged['Latitude'], Bronx_merged['Longitude'], Bronx_merged['N
             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(cluster_map)

         cluster_map
```

Out[50]:



Getting insights from clusters(if any)

Cluster 1

In [51]:

```
Bronx_merged.loc[Bronx_merged['Cluster Labels'] == 0, Bronx_merged.columns[[1] + list(range(4, Bronx_m
```

Out[51]:

	Neighborhood	Cluster 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 Mo Comm Ven
0	Wakefield	0	Pharmacy	Caribbean Restaurant	Fast Food Restaurant	Donut Shop	Supermarket	Fried Chicken Joint	Business Service	Mobile Phone Shop	Span Restaura
9	Baychester	0	Department Store	Clothing Store	Discount Store	Shopping Mall	Supermarket	Kids Store	Bakery	Bagel Shop	Ital Restaura
10	Pelham Parkway	0	Pizza Place	Donut Shop	Sandwich Place	Chinese Restaurant	Italian Restaurant	Supermarket	Bakery	Pharmacy	Cosmet Sh
13	University Heights	0	Grocery Store	Pizza Place	Shoe Store	Supermarket	Spanish Restaurant	Donut Shop	Sandwich Place	Latin American Restaurant	Ice Crea Sh
14	Morris Heights	0	Pharmacy	Food Truck	Deli / Bodega	Pizza Place	Fast Food Restaurant	Gym / Fitness Center	Ice Cream Shop	Grocery Store	Span Restaura
				Italian		Latin	Deli /	Chinese			

Cluser 2

In [52]:

```
Bronx_merged.loc[Bronx_merged['Cluster Labels'] == 1, Bronx_merged.columns[[1] + list(range(4, Bronx_m
```

Out[52]:

	Neighborhood	Cluster 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	10 C
2	Eastchester	1	Caribbean Restaurant	Diner	Fast Food Restaurant	Shopping Mall	Donut Shop	Supplement Shop	Women's Store	Burger Joint	Discount Store	Re
5	Kingsbridge	1	Pizza Place	Mexican Restaurant	Bakery	Japanese Restaurant	Coffee Shop	Diner	Spanish Restaurant	Burger Joint	Bar	Dor
6	Woodlawn	1	Deli / Bodega	Bar	Pub	Discount Store	Italian Restaurant	Rental Car Location	Liquor Store	Donut Shop	Pizza Place	P
19	Melrose	1	Mexican Restaurant	Pizza Place	Gym	Italian Restaurant	Diner	Discount Store	Donut Shop	Mobile Phone Shop	Fried Chicken Joint	
20	Mott Haven	1	Pizza Place	Donut Shop	Mexican Restaurant	Discount Store	Supermarket	Grocery Store	Gym	Spanish Restaurant	Brewery	Ste
39	Castle Hill	1	Bus Stop	Pizza Place	Bus Station	Baseball Field	Deli / Bodega	Wine Shop	Spanish Restaurant	Fried Chicken Joint	Liquor Store	
42	Concourse	1	Donut Shop	Pizza Place	Gym	Park	Sandwich Place	Fast Food Restaurant	Pharmacy	Diner	Discount Store	

Clster 3

In [53]:

```
Bronx_merged.loc[Bronx_merged['Cluster Labels'] == 2, Bronx_merged.columns[[1] + list(range(4, Bronx_m
```

Out[53]:

	Neighborhood	Cluster 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 Mos Commor Venue
7	Norwood	2	Pizza Place	Pharmacy	Park	Donut Shop	Bank	Sandwich Place	Caribbean Restaurant	Diner	Mobile Phone Shop	Spanisl Restauran
8	Williamsbridge	2	Pizza Place	Caribbean Restaurant	Fast Food Restaurant	Discount Store	Bakery	Spa	Sandwich Place	Pharmacy	Supermarket	Seafooc Restauran

Clster 4

```
In [54]: Bronx_merged.loc[Bronx_merged['Cluster Labels'] == 3, Bronx_merged.columns[[1] + list(range(4, Bronx_m
```

```
Out[54]:
```

	Neighborhood	Cluster 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	9t Co
1	Co-op City	3	Shoe Store	Shopping Mall	Mobile Phone Shop	Mattress Store	Bakery	Department Store	Pizza Place	Discount Store	Coffe
3	Fieldston	3	Pizza Place	Chinese Restaurant	Deli / Bodega	Bar	Plaza	Mexican Restaurant	Bus Station	Sandwich Place	Donu
4	Riverdale	3	Pizza Place	Japanese Restaurant	Playground	Plaza	Park	Mexican Restaurant	Coffee Shop	Bar	
11	City Island	3	Harbor / Marina	Seafood Restaurant	Boat or Ferry	Italian Restaurant	Bar	Thrift / Vintage Store	American Restaurant	Chinese Restaurant	
12	Bedford Park	3	Park	Diner	Botanical Garden	Supermarket	Garden	Pizza Place	Mexican Restaurant	Donut Shop	C Res
16	East Tremont	3	Donut Shop	Pizza Place	Zoo Exhibit	Bank	Park	Sandwich Place	Shoe Store	Breakfast Spot	Di
17	West Farms	3	Zoo	Donut Shop	Fried Chicken Joint	Park	Pizza Place	Sandwich Place	Discount Store	Fast Food Restaurant	L
18	High Bridge	3	Baseball Stadium	Lounge	Plaza	Sandwich Place	Smoke Shop	History Museum	Burger Joint	Liquor Store	Be
21	Port Morris	3	Baseball Field	Pizza Place	Donut Shop	Restaurant	Optical Shop	Brewery	Mobile Phone Shop	Sporting Goods Shop	S Res
22	Longwood	3	Fast Food Restaurant	Sandwich Place	Discount Store	Donut Shop	Pizza Place	Pharmacy	Grocery Store	Bakery	An Res
24	Morrisania	3	Discount Store	Pharmacy	Donut Shop	Fast Food Restaurant	Mobile Phone Shop	Gym	Pizza Place	Grocery Store	Super
26	Clason Point	3	Park	Bus Stop	Trail	Gym / Fitness Center	Discount Store	Bus Station	Deli / Bodega	Wings Joint	Fish &
28	Country Club	3	Italian Restaurant	Pizza Place	Sandwich Place	Bakery	Pharmacy	Deli / Bodega	Dive Bar	Donut Shop	

Conclusion

Bronx Burough is bustling location for food and tourism industry. Its 213 unique venues are spread in 52 neighbourhoods with most of which have the potential to become vibrant locations for food and tourism related businesses. Based on clusters generated, it is apparent that cluster one can be described as 'tribal' or exotic cousine food cluster; cluster two, the shopping and food cluster, cluster three has the credentials of being a hefty food and entertainment segment, while cluster four deserves to become Asian and Mexican food and tourism attraction segment. There is need to identify key factors that explain the relartively small number of unique venues in Clason point (8), Claremont(15) and Huntspoint(21) compared to the average of 50 venues in most neighborhoods. This could be potential entry point for new food and tourism attraction related business if the problem is lack of facilities that support the two related and interdepedent sectors. Nonethless, it could also shed light on obstacles that have prevented the establishment of vibrant and popular food and tourism attraction venues in such neighborhoods which could save potential investors from investing in neighborhood that have the potential to generate low return in investment.

Work is as good as done: What an illuminating assignment!