Requirements:

For this week, you will required to submit the following:

1. A description of the problem and a discussion of the background. (15 marks)
2. A description of the data and how it will be used to solve the problem. (15 marks)

Project Description

In the city of New York, I want to open a new grocery shop. I want to find the best place to open it.

We will use the data from Foursquare about venues in New York City and use KNN cluster to decide location which has high concentration of good Shop Venue. That will be the interesting point to open a new Shop.

Another option is to find a location have in the same cluster with high concentration cluster but locate in further neighbor with less competition.

Import Data for venues in NY

In [1]:

```
# Import Libraries

import numpy as np # library to handle data in a vectorized manner
import wget

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 com
from geopy.geocoders import Nominatim # convert an address into latitude and longit

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

# 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 have
import folium # map rendering library

print('Libraries imported.')
```

```
Libraries imported.
```

Download and Explore Dataset

Neighborhood has a total of 5 boroughs and 306 neighborhoods. In order to segement the neighborhoods and explore them, we will essentially need a dataset that contains the 5 boroughs and the neighborhoods that exist in each borough as well as the the latitude and logitude coordinates of each neighborhood.

In [2]:

```
#!wget -q -o'newyork_data.json' https://cocl.us/new_york_dataset
#url  = 'https://cocl.us/new_york_dataset'
#file = wget.download(url)
print('Data downloaded!')
```

```
Data downloaded!
```

In [3]:

```
with open('new_york_dataset') as json_data:
    newyork_data = json.load(json_data)
```

In [42]:

```
#newyork_data
```

In [43]:

```
# Take the list of neighbor from features key from Json file
neighborhoods_data = newyork_data['features']
```

In [6]:

```
# Transform the data into pandas df

# define the dataframe columns
column_names = ['Borough', 'Neighborhood', 'Latitude', 'Longitude']

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

Out[6]:

Borough	Neighborhood	Latitude	Longitude

In [7]:

```
# loop through data and fill dataframe one row at a time

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_in
```

In [8]:

```
neighborhoods.head()
```

Out[8]:

	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 [9]:

```
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 lat and long values on NYC

In [10]:

```
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, lo
```

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

Create map of NY and its neighborhoods

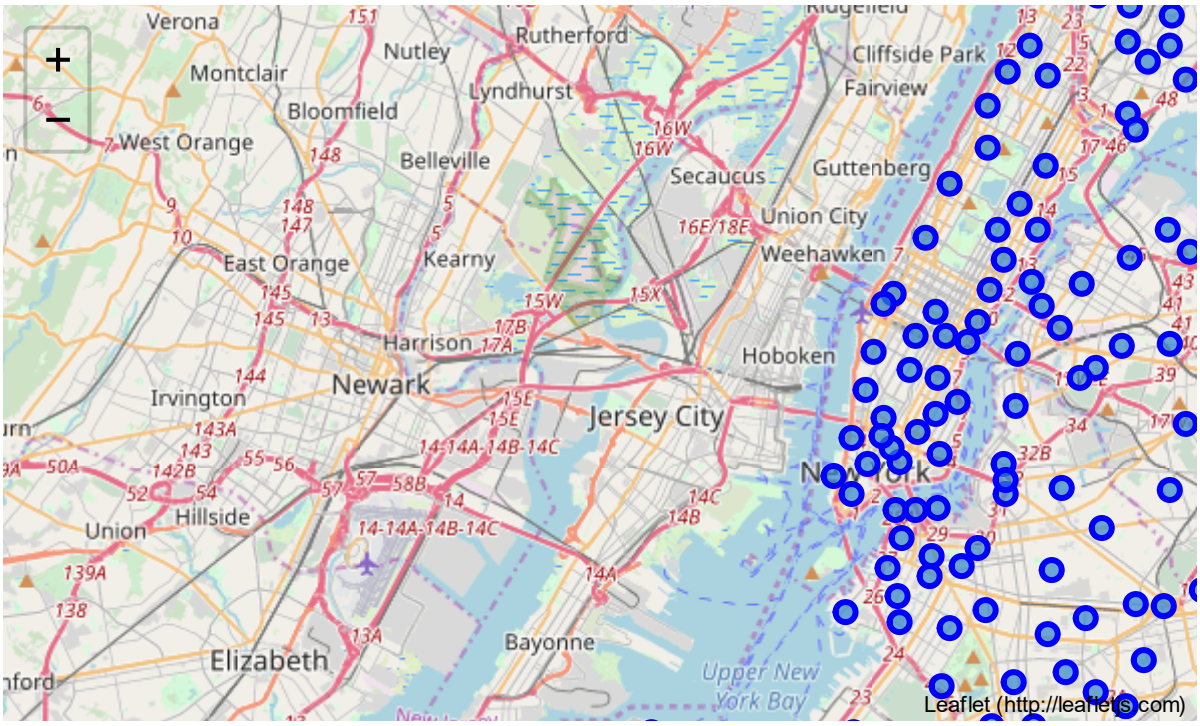
In [11]:

```
# 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
    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
```

Out[11]:



Define Foursquare Credentials and Version

In [12]:

```
CLIENT_ID = 'L5ZVQIFSZXJWKXR3131RTBYXLULHLMZB0M1QXDEMUNTMJWKD' # your Foursquare ID
CLIENT_SECRET = '1IAZIRFGEEACMPJ2XX0VIEQX4VNQWLTGSXQANIZCVL4UNFVW' # your Foursquar
VERSION = '20191212' # Foursquare API version
LIMIT = 100

print('Your credentails:')
print('CLIENT_ID: ' + CLIENT_ID)
print('CLIENT_SECRET:' + CLIENT_SECRET)
```

```
Your credentails:
CLIENT_ID: L5ZVQIFSZXJWKXR3131RTBYXLULHLMZB0M1QXDEMUNTMJWKD
CLIENT_SECRET:1IAZIRFGEEACMPJ2XX0VIEQX4VNQWLTGSXQANIZCVL4UNFVW
```

In [17]:

```
def getNearbyVenues(names, latitudes, longitudes, radius=100):

    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_se
            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 ve
    nearby_venues.columns = ['Neighborhood',
                  'Neighborhood Latitude',
                  'Neighborhood Longitude',
                  'Venue',
                  'Venue Latitude',
                  'Venue Longitude',
                  'Venue Category']

    return(nearby_venues)
```

request all venues with 100m radius from all neighborhoods in NY

In [ ]:

```
newyork_venues = getNearbyVenues(names=neighborhoods['Neighborhood'],
                                 latitudes=neighborhoods['Latitude'],
                                 longitudes=neighborhoods['Longitude']
                                )
```

In [19]:

```
1 newyork_venues.head()
```

Out[19]:

	Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
0	Fieldston	40.895437	-73.905643	8l06	40.895668	-73.904750	Wine Shop
1	Fieldston	40.895437	-73.905643	nicksemlerSPA	40.894942	-73.905475	Spa
2	Kingsbridge	40.881687	-73.902818	Garden Gourmet Market	40.881350	-73.903389	Gourmet Shop
3	Kingsbridge	40.881687	-73.902818	MyUnique	40.881966	-73.903584	Thrift / Vintage Store
4	Kingsbridge	40.881687	-73.902818	Mattress Firm	40.881580	-73.903277	Mattress Store

In [20]:

```
1 newyork_venues.shape
```

Out[20]:

```
(873, 7)
```

In [45]:

```
1 newyork_venues.groupby('Neighborhood').count().head()
```

Out[45]:

Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
Allerton	4	4	4	4	4	4
Arlington	2	2	2	2	2	2
Bath Beach	3	3	3	3	3	3
Battery Park City	3	3	3	3	3	3
Bay Ridge	11	11	11	11	11	11

In [22]:

```
1 # Unique categories
2 print('There are {} uniques categories.'.format(len(newyork_venues['Venue Category'
```

```
There are 218 uniques categories.
```

Analyze

What is the categories of venue in NY?

In [23]:

```
1 newyork_venues['Venue Category'].unique()
```

Out[23]:

```
array(['Wine Shop', 'Spa', 'Gourmet Shop', 'Thrift / Vintage Store',
       'Mattress Store', 'Discount Store', 'Pizza Place', 'Pub',
       'Indian Restaurant', 'Deli / Bodega', 'Bar', 'Supermarket',
       'Juice Bar', 'Diner', 'Ice Cream Shop', 'French Restaurant',
       'American Restaurant', 'Park', 'Grocery Store', 'Pharmacy',
       'Jewelry Store', 'Music Venue', 'Fried Chicken Joint',
       'Sandwich Place', 'Café', 'Food', 'Bus Station',
       'Asian Restaurant', 'Gym / Fitness Center', 'Gym', 'Liquor Stor
e',
       'Fish & Chips Shop', 'Chinese Restaurant', 'Convenience Store',
       'Latin American Restaurant', 'Fast Food Restaurant',
       'Check Cashing Service', 'Italian Restaurant', 'Bus Line', 'Ban
k',
       'Dog Run', 'Caribbean Restaurant', 'Bus Stop',
       'Caucasian Restaurant', 'Lounge', 'Coffee Shop', 'Hookah Bar',
       'Sushi Restaurant', 'Pool Hall', 'Spanish Restaurant',
       'Greek Restaurant', 'Mexican Restaurant', 'Mobile Phone Shop',
       'Bagel Shop', 'Gift Shop', 'Pet Store', 'Polish Restaurant',
```

Find all venue which is Store

In [24]:

```
1 ny_allstore = newyork_venues[newyork_venues['Venue Category'].str.contains('Store')
2 ny_allstore.head()
```

Out[24]:

	Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
0	Kingsbridge	40.881687	-73.902818	MyUnique	40.881966	-73.903584	Thrift / Vintage Store
1	Kingsbridge	40.881687	-73.902818	Mattress Firm	40.881580	-73.903277	Mattress Store
2	Kingsbridge	40.881687	-73.902818	Dollar Tree	40.881715	-73.903187	Discount Store
3	City Island	40.847247	-73.786488	Connies New Way Supermarket	40.847146	-73.786546	Grocery Store
4	City Island	40.847247	-73.786488	Kaleidoscope Gallery	40.846466	-73.786226	Jewelry Store

In [25]:

```
1 ny_allstore['Venue Category'].unique()
```

Out[25]:

```
array(['Thrift / Vintage Store', 'Mattress Store', 'Discount Store',
       'Grocery Store', 'Jewelry Store', 'Liquor Store',
       'Convenience Store', 'Pet Store', 'Arts & Crafts Store',
       'Fruit & Vegetable Store', 'Hardware Store', 'Shoe Store',
       'Furniture / Home Store', 'Electronics Store', 'Accessories Store',
       'Toy / Game Store', 'Department Store', 'Big Box Store',
       'Lingerie Store', 'Camera Store', "Women's Store", "Men's Store",
       'Shipping Store', 'Paper / Office Supplies Store', 'Video Store',
       'Kids Store', 'Health Food Store', 'Clothing Store'], dtype=object)
```

My store will sell similar goods as many stores so it's good to take only competitive stores in the consideration

In [26]:

```
1 #store_list = ['Grocery Store', 'Convenience Store', 'Liquor Store',
2 #              'Fruit & Vegetable Store', 'Paper / Office Supplies Store',
3 #            'Kitchen Supply Store',  'Outdoor Supply Store']
```

In [27]:

```
1 #newyork_venues[newyork_venues['Venue Category']== store_list]
```

In [44]:

```
1 ny_allstore.groupby('Neighborhood').count().head()
```

Out[44]:

Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
Bushwick	1	1	1	1	1	1
Cambria Heights	1	1	1	1	1	1
Carroll Gardens	2	2	2	2	2	2
Charleston	2	2	2	2	2	2
City Island	2	2	2	2	2	2

Analyze Neighborhood

In [29]:

```
# one hot encoding
ny_allstore_onehot = pd.get_dummies(ny_allstore[['Venue Category']], prefix="", pre

# add neighborhood column back to dataframe
ny_allstore_onehot['Neighborhood'] = ny_allstore['Neighborhood']

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

ny_allstore_onehot.head()
```

Out[29]:

	Neighborhood	Accessories Store	Arts & Crafts Store	Big Box Store	Camera Store	Clothing Store	Convenience Store	Department Store	Disco St
0	Kingsbridge	0	0	0	0	0	0	0	
1	Kingsbridge	0	0	0	0	0	0	0	
2	Kingsbridge	0	0	0	0	0	0	0	
3	City Island	0	0	0	0	0	0	0	
4	City Island	0	0	0	0	0	0	0	

In [30]:

```
ny_allstore_onehot.shape
```

Out[30]:

```
(80, 29)
```

In [ ]:

```
ny_allstore_grouped = ny_allstore_onehot.groupby('Neighborhood').mean().reset_index
ny_allstore_grouped
```

In [32]:

```
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]
```

In [33]:

```
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'] = ny_allstore_grouped['Neighborhood']

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

neighborhoods_venues_sorted.head()
```

Out[33]:

	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	
0	Bushwick	Thrift / Vintage Store	Women's Store	Hardware Store	Arts & Crafts Store	Big Box Store	Camera Store	Clothing Store	C
1	Cambria Heights	Health Food Store	Video Store	Arts & Crafts Store	Big Box Store	Camera Store	Clothing Store	Convenience Store	
2	Carroll Gardens	Arts & Crafts Store	Shoe Store	Women's Store	Hardware Store	Big Box Store	Camera Store	Clothing Store	C
3	Charleston	Arts & Crafts Store	Department Store	Women's Store	Video Store	Big Box Store	Camera Store	Clothing Store	C
4	City Island	Grocery Store	Jewelry Store	Women's Store	Hardware Store	Arts & Crafts Store	Big Box Store	Camera Store	

Cluster Neighborhoods

In [37]:

```
# set number of clusters
kclusters = 10

ny_allstore_grouped_clustering = ny_allstore_grouped.drop('Neighborhood', 1)

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

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

Out[37]:

```
array([5, 8, 0, 0, 4, 3, 2, 0, 1, 2])
```

In [ ]:

```
# add clustering labels
neighborhoods_venues_sorted.insert(0, 'Cluster Labels', kmeans.labels_)

ny_allstore_merged = ny_allstore

# merge toronto_grouped with toronto_data to add latitude/longitude for each neighb
ny_allstore_merged = ny_allstore_merged.join(neighborhoods_venues_sorted.set_index(

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

In [39]:

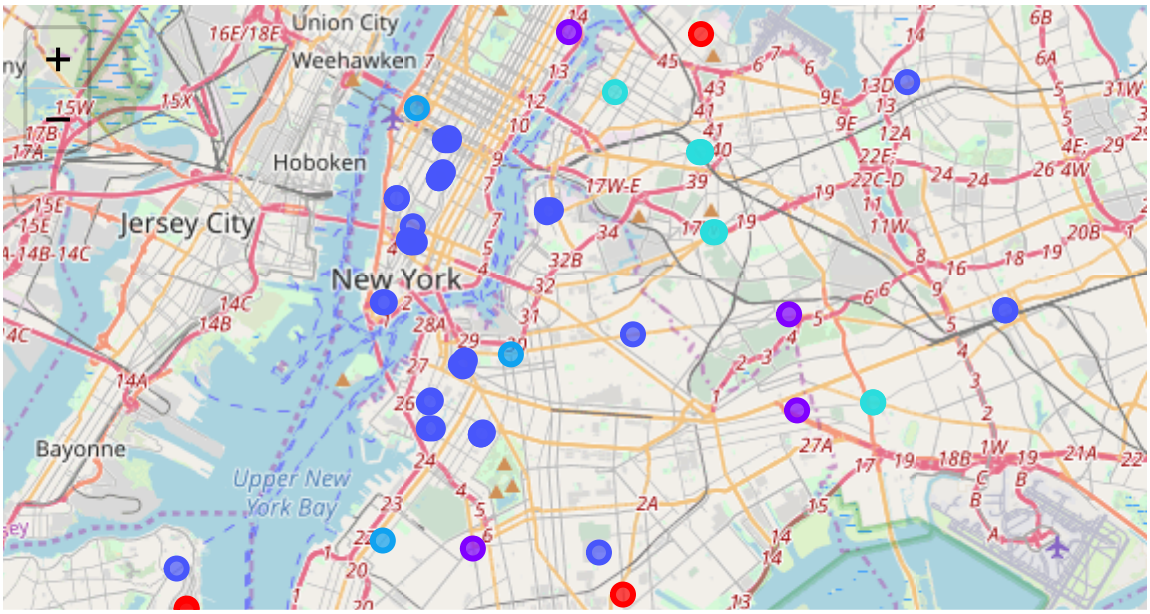
```
# create map
map_clusters = folium.Map(location=[latitude, longitude], zoom_start=11)

# 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(ny_allstore_merged['Venue Latitude'], ny_allstore
    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[39]:



From the map we see the Cluster 2 has most store venue and most of them located in Mahatan and Brooklyn. We are more interest in the outer region at Queen such as Cambria Heights, Jamaica, Flushing which in the same cluster but less competition.

In [41]:

```
#ny_allstore_merged.loc[ny_allstore_merged['Cluster Labels'] == 1, ny_allstore_merg
```

In [ ]:

```
1
```

In [ ]:

```
1
```