

Coursera Capstone Project: Segmenting and Clustering Neighborhoods in Toronto

Author: Kapil Kumar Nagwanshi


Assignmen Question 1

- For this assignment, we have to explore and cluster the neighborhoods in Toronto.


Part 1 of Question 1

1. Start by creating a new Notebook for this assignment. --This is the notebook

Before we get the data and start exploring it, let's download all the dependencies that we will need.

```
In [2]:  #import necessary libraries
import numpy as np # Library to handle data in a vectorized manner
import pandas as pd # Library for data analysis
import requests # Library for web scraping
import requests
from urllib.request import urlopen
from bs4 import BeautifulSoup
import ssl
import csv
print('Library import done...')
```

Library import done...

```
In [3]:  # Ignore SSL certificate errors
ctx = ssl.create_default_context()
ctx.check_hostname = False
ctx.verify_mode = ssl.CERT_NONE
print('SSL certificate errors ignored...')
```

SSL certificate errors ignored...

Part 2 of Question 1

2. Use the Notebook to build the code to scrape the following Wikipedia page, https://en.wikipedia.org/wiki/List_of_postal_codes_of_Canada:_M (https://en.wikipedia.org/wiki/List_of_postal_codes_of_Canada:_M), in order to obtain the data that is in the table of postal codes and to transform the data into a pandas dataframe like the one shown in assignment.

This part shows raw data frame obtained from https://en.wikipedia.org/wiki/List_of_postal_codes_of_Canada:_M (https://en.wikipedia.org/wiki/List_of_postal_codes_of_Canada:_M) withput cleaning

```
In [4]:  #beautifulSoup instances
res_site = requests.get("https://en.wikipedia.org/wiki/List_of_postal_codes_of_Canada:_M")
soup = BeautifulSoup(res_site.content, 'lxml')
table = soup.find_all('table')[0]
#toronto_data = pd.read_html(str(table))[0]
#####
table_rows = table.tbody.find_all("tr")


res = []
for tr in table_rows:
    td = tr.find_all("td")
    row = [tr.text for tr in td]

    # Only process the cells that have an assigned borough. Ignore cells with a borough that is Not assigned.
    if row != [] and row[1] != "Not assigned":
        # If a cell has a borough but a "Not assigned" neighborhood, then the neighborhood will be the same as the borough.
        if "Not assigned" in row[2]:
            row[2] = row[1]
        res.append(row)

# Dataframe with 3 columns
df_toronto = pd.DataFrame(res, columns = ["PostalCode", "Borough", "Neighborhood"])
df_toronto.head()
```


Out[4]:

	PostalCode	Borough	Neighborhood
0	M3A	North York	Parkwoods\n
1	M4A	North York	Victoria Village\n
2	M5A	Downtown Toronto	Harbourfront\n
3	M6A	North York	Lawrence Heights\n
4	M6A	North York	Lawrence Manor\n

```
In [5]:  # Remove '\n' from Neighborhood
df_toronto["Neighborhood"] = df_toronto["Neighborhood"].str.replace("\n", "")
df_toronto.head()
```

Out[5]:

	PostalCode	Borough	Neighborhood
0	M3A	North York	Parkwoods
1	M4A	North York	Victoria Village
2	M5A	Downtown Toronto	Harbourfront
3	M6A	North York	Lawrence Heights
4	M6A	North York	Lawrence Manor

```
In [6]:  df_toronto.shape
```

Out[6]: (210, 3)

Part 3 of Question 1

3. To create the above dataframe:
4. The dataframe will consist of three columns: PostalCode, Borough, and Neighborhood
5. Only process the cells that have an assigned borough. Ignore cells with a borough that is Not assigned.
6. More than one neighborhood can exist in one postal code area. For example, in the table on the Wikipedia page, you will notice that M5A is listed twice and has two neighborhoods: Harbourfront and Regent Park. These two rows will be combined into one row with the neighborhoods separated with a comma as shown in row 11 in the above table.
7. If a cell has a borough but a Not assigned neighborhood, then the neighborhood will be the same as the borough.
8. Clean your Notebook and add Markdown cells to explain your work and any assumptions you are making.
9. In the last cell of your notebook, use the .shape method to print the number of rows of your dataframe.

This part shows cleaned obtained dataframe https://en.wikipedia.org/wiki/List_of_postal_codes_of_Canada:_M (https://en.wikipedia.org/wiki/List_of_postal_codes_of_Canada:_M) with grouping of same postal codes

```
In [7]: df_toronto = df_toronto.groupby(["PostalCode", "Borough"])["Neighborhood"].apply(", ".join).reset_index()  
df_toronto.head()
```

Out[7]:

	PostalCode	Borough	Neighborhood
0	M1B	Scarborough	Rouge, Malvern
1	M1C	Scarborough	Highland Creek, Rouge Hill, Port Union
2	M1E	Scarborough	Guildwood, Morningside, West Hill
3	M1G	Scarborough	Woburn
4	M1H	Scarborough	Cedarbrae

```
In [8]: df_toronto.shape
```

Out[8]: (103, 3)

```
In [9]: df_toronto.head(15)
```

```
Out[9]:
```

	PostalCode	Borough	Neighborhood
0	M1B	Scarborough	Rouge, Malvern
1	M1C	Scarborough	Highland Creek, Rouge Hill, Port Union
2	M1E	Scarborough	Guildwood, Morningside, West Hill
3	M1G	Scarborough	Woburn
4	M1H	Scarborough	Cedarbrae
5	M1J	Scarborough	Scarborough Village
6	M1K	Scarborough	East Birchmount Park, Ionview, Kennedy Park
7	M1L	Scarborough	Clairlea, Golden Mile, Oakridge
8	M1M	Scarborough	Cliffcrest, Cliffside, Scarborough Village West
9	M1N	Scarborough	Birch Cliff, Cliffside West
10	M1P	Scarborough	Dorset Park, Scarborough Town Centre, Wexford ...
11	M1R	Scarborough	Maryvale, Wexford
12	M1S	Scarborough	Agincourt
13	M1T	Scarborough	Clarks Corners, Sullivan, Tam O'Shanter
14	M1V	Scarborough	Agincourt North, L'Amoreaux East, Milliken, St...

Part 4 of the Question 1

- Submit a link to your Notebook on your Github repository. (10 marks)

End of Question 1

Assignment Question 2

Now that you have built a dataframe of the postal code of each neighborhood along with the borough name and neighborhood name, in order to utilize the Foursquare location data, we need to get the latitude and the longitude coordinates of each neighborhood.

In an older version of this course, we were leveraging the Google Maps Geocoding API to get the latitude and the longitude coordinates of each neighborhood. However, recently Google started charging for their API: <http://geoawesomeness.com/developers-up-in-arms-over-google-maps-api-insane-price-hike/> (<http://geoawesomeness.com/developers-up-in-arms-over-google-maps-api-insane-price-hike/>), so we will use the Geocoder Python package instead: <https://geocoder.readthedocs.io/index.html> (<https://geocoder.readthedocs.io/index.html>).

The problem with this Package is you have to be persistent sometimes in order to get the geographical coordinates of a given postal code. So you can make a call to get the latitude and longitude coordinates of a given postal code and the result would be None, and then make the call again and you would get the coordinates. So, in order to make sure that you get the coordinates for all of our neighborhoods, you can run a while loop for each postal code.

- Check for geopy and geocoder packages

```
In [10]: ▶ import geopy
from geopy.geocoders import Nominatim
nominatim_service = Nominatim(user_agent='X@yy.com') # Important Line
geopy.geocoders.options.default_user_agent = "X@yy.com" # Important Line
geolocator = Nominatim()
```

```
In [11]: ▶ city = "Toronto"
country = "Canada"
loc = geolocator.geocode(city+', '+ country)
print("latitude is :-" ,loc.latitude,"\nlongtitude is:-" ,loc.longitude)

latitude is :- 43.653963
longitude is:- -79.387207
```

```
In [12]: ▶ location = geolocator.geocode("Toronto, North York, Parkwoods")
print(location.address)
print('')
print((location.latitude, location.longitude))
print('')
print(location.raw)
```

Parkwoods Village Drive, Parkway East, Don Valley East, North York, Toronto, Golden Horseshoe, Ontario, M3A 2X2, Canada

(43.7587999, -79.3201966)

```
{'place_id': 124974741, 'licence': 'Data © OpenStreetMap contributors, ODbL 1.0. https://osm.org/copyright', (https://osm.org/copyright',) 'osm_type': 'way', 'osm_id': 160406961, 'boundingbox': ['43.7576231', '43.761106', '-79.3239088', '-79.316215'], 'lat': '43.7587999', 'lon': '-79.3201966', 'display_name': 'Parkwoods Village Drive, Parkway East, Don Valley East, North York, Toronto, Golden Horseshoe, Ontario, M3A 2X2, Canada', 'class': 'highway', 'type': 'secondary', 'importance': 0.51}
```

Get the latitude and the longitude coordinates of each neighborhood

```
In [13]: ▶ import geopy
from geopy.geocoders import Nominatim
import pandas as pd
locator = Nominatim(user_agent="KapilsGeocoder")
location = locator.geocode("Toronto, Canada")
from geopy.extra.rate_limiter import RateLimiter
# PostalCode    Borough Neighborhood
df_temp=df_toronto
# 1 - convneint function to delay between geocoding calls
geocode = RateLimiter(locator.geocode, min_delay_seconds=1)
# 2- - create location column
df_temp['Address'] = df_temp['PostalCode'].astype(str) + ', ' + ' Toronto'
df_temp['Location'] = df_temp['Address'].apply(geocode)
# 3 - create longitude, laatitude and altitude from location column (returns tuple)
df_temp['Point'] = df_temp['Location'].apply(lambda loc: tuple(loc.point) if loc else None)
# 4 - split point column into latitude, longitude and altitude columns
# df_temp[['latitude', 'longitude', 'altitude']] = pd.DataFrame(df_temp['Point'].tolist(), index=df_temp.index)
```

```
In [14]: df_temp
```

Out[14]:

	PostalCode	Borough	Neighborhood	Address	Location	Point
0	M1B	Scarborough	Rouge, Malvern	M1B, Toronto	(Toronto, Punta Gorda, Montevideo, 11403, Urug...	(-34.8899421, -56.0790982, 0.0)
1	M1C	Scarborough	Highland Creek, Rouge Hill, Port Union	M1C, Toronto	(Toronto, Punta Gorda, Montevideo, 11403, Urug...	(-34.8899421, -56.0790982, 0.0)
2	M1E	Scarborough	Guildwood, Morningside, West Hill	M1E, Toronto	None	None
3	M1G	Scarborough	Woburn	M1G, Toronto	(Scarborough—Guildwood, Scarborough, Toronto, ...	(43.76571676956549, -79.22189842824983, 0.0)
4	M1H	Scarborough	Cedarbrae	M1H, Toronto	None	None
...
98	M9N	York	Weston	M9N, Toronto	None	None
99	M9P	Etobicoke	Westmount	M9P, Toronto	None	None
100	M9R	Etobicoke	Kingsview Village, Martin Grove Gardens, Richv...	M9R, Toronto	(Etobicoke Centre, Etobicoke, Toronto, Golden ...	(43.69516618990701, -79.55088985426742, 0.0)
101	M9V	Etobicoke	Albion Gardens, Beaumont Heights, Humbergate, ...	M9V, Toronto	None	None
102	M9W	Etobicoke	Northwest	M9W, Toronto	None	None

103 rows × 6 columns

From above simple solution, we are not able to get the geohraphical coordinates of the neighborhoods using the Geocoder package, we use the given csv file instead.

```
In [15]: df_geo_coor = pd.read_csv("./Geospatial_Coordinates.csv")
df_geo_coor.head()
```

Out[15]:

	Postal Code	Latitude	Longitude
0	M1B	43.806686	-79.194353
1	M1C	43.784535	-79.160497
2	M1E	43.763573	-79.188711
3	M1G	43.770992	-79.216917
4	M1H	43.773136	-79.239476

```
In [16]: df_toronto.head()
# df_toronto dataframe played through geocoder
# beacause of call limits it won't work for all see 'none' in point columns
```

Out[16]:

	PostalCode	Borough	Neighborhood	Address	Location	Point
0	M1B	Scarborough	Rouge, Malvern	M1B, Toronto	(Toronto, Punta Gorda, Montevideo, 11403, Urug...	(-34.8899421, -56.0790982, 0.0)
1	M1C	Scarborough	Highland Creek, Rouge Hill, Port Union	M1C, Toronto	(Toronto, Punta Gorda, Montevideo, 11403, Urug...	(-34.8899421, -56.0790982, 0.0)
2	M1E	Scarborough	Guildwood, Morningside, West Hill	M1E, Toronto	None	None
3	M1G	Scarborough	Woburn	M1G, Toronto	(Scarborough—Guildwood, Scarborough, Toronto, ...	(43.76571676956549, -79.22189842824983, 0.0)
4	M1H	Scarborough	Cedarbrae	M1H, Toronto	None	None

```
In [17]: # drop address location and point columns to get original dataframe
df_toronto.drop(['Address', 'Location', 'Point'], axis=1, inplace=True)
df_toronto.head()
```

Out[17]:

	PostalCode	Borough	Neighborhood
0	M1B	Scarborough	Rouge, Malvern
1	M1C	Scarborough	Highland Creek, Rouge Hill, Port Union
2	M1E	Scarborough	Guildwood, Morningside, West Hill
3	M1G	Scarborough	Woburn
4	M1H	Scarborough	Cedarbrae

Now We need to couple 2 dataframes "df_toronto" and "df_geo_coor" into one dataframe.

```
In [18]: df_toronto2 = pd.merge(df_toronto, df_geo_coor, how='left', left_on = 'PostalCode', right_on = 'Postal Code')
# remove the "Postal Code" column
df_toronto2.drop("Postal Code", axis=1, inplace=True)
df_toronto2.head()
```

Out[18]:

	PostalCode	Borough	Neighborhood	Latitude	Longitude
0	M1B	Scarborough	Rouge, Malvern	43.806686	-79.194353
1	M1C	Scarborough	Highland Creek, Rouge Hill, Port Union	43.784535	-79.160497
2	M1E	Scarborough	Guildwood, Morningside, West Hill	43.763573	-79.188711
3	M1G	Scarborough	Woburn	43.770992	-79.216917
4	M1H	Scarborough	Cedarbrae	43.773136	-79.239476

Assignment Question 3 Explore and cluster the neighborhoods in Toronto

Explore and cluster the neighborhoods in Toronto. You can decide to work with only boroughs that contain the word Toronto and then replicate the same analysis we did to the New York City data. It is up to you.

Just make sure:

1. to add enough Markdown cells to explain what you decided to do and to report any observations you make.
2. to generate maps to visualize your neighborhoods and how they cluster together.

```
In [19]: ▶ address = "Toronto, ON"
geolocator = Nominatim(user_agent="toronto_explorer") # Changed user agent due to collision
location = geolocator.geocode(address)
latitude = location.latitude
longitude = location.longitude
print('The geograpical coordinate of Toronto city are {}, {}'.format(latitude, longitude))
```

The geograpical coordinate of Toronto city are 43.653963, -79.387207.

Create a map of the whole Toronto City with neighborhoods superimposed on top

```
In [20]: ▶ import folium # map rendering library

# import k-means from clustering stage
from sklearn.cluster import KMeans

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


Out[21]:

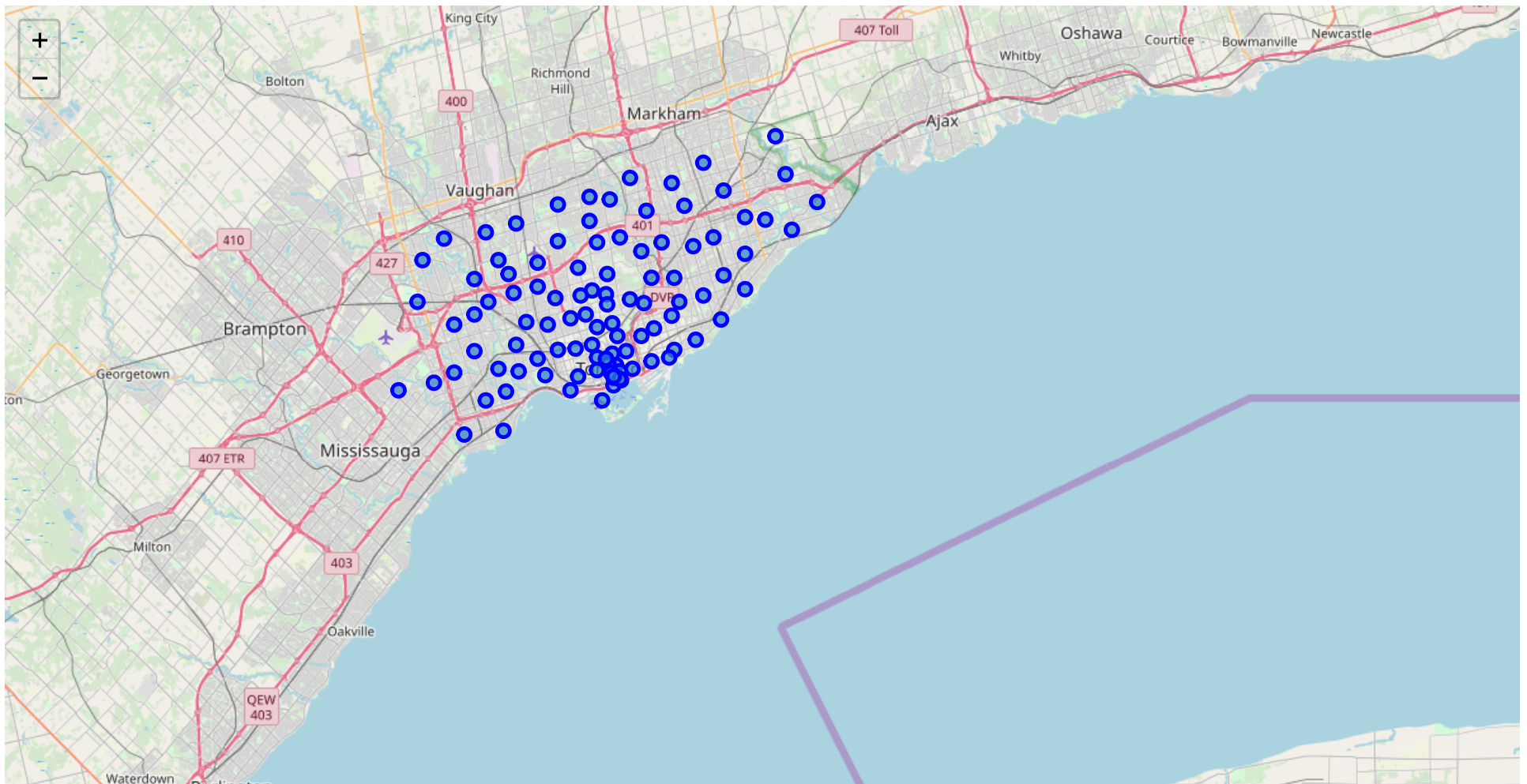


Add markers to the map.

```
In [22]: for lat, lng, borough, neighborhood in zip(
    df_toronto2['Latitude'],
    df_toronto2['Longitude'],
    df_toronto2['Borough'],
    df_toronto2['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_toronto)
```

map_toronto

Out[22]:



Map of a part of Toronto City

We are going to work with only the boroughs that contain the word "Toronto".

```
In [23]: # "denc" = [D]owntown Toronto, [E]ast Toronto, [N]orth Toronto, [C]entral Toronto
df_toronto_denc = df_toronto2[df_toronto['Borough'].str.contains("Toronto")].reset_index(drop=True)
df_toronto_denc.head()
```

Out[23]:

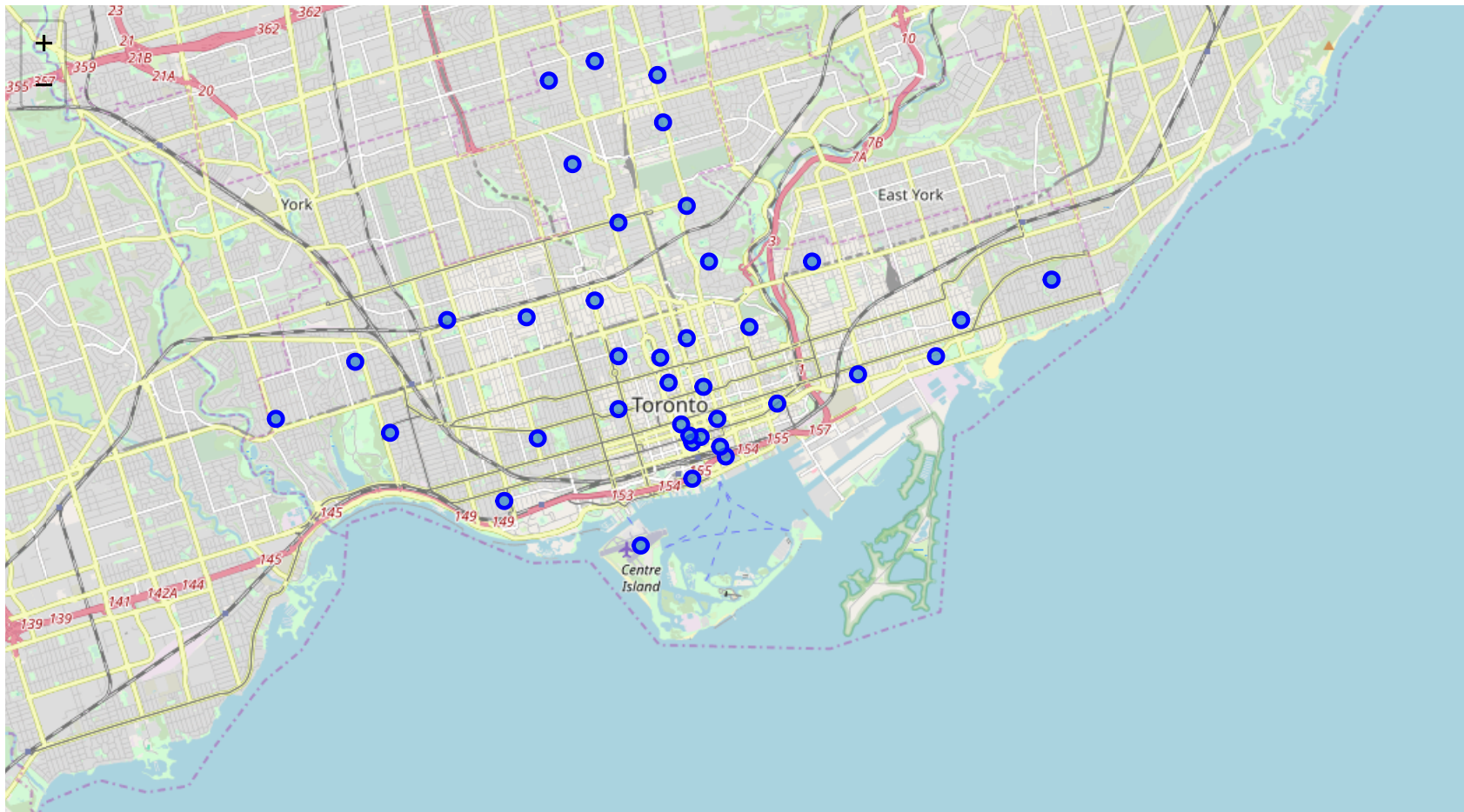
	PostalCode	Borough	Neighborhood	Latitude	Longitude
0	M4E	East Toronto	The Beaches	43.676357	-79.293031
1	M4K	East Toronto	The Danforth West, Riverdale	43.679557	-79.352188
2	M4L	East Toronto	The Beaches West, India Bazaar	43.668999	-79.315572
3	M4M	East Toronto	Studio District	43.659526	-79.340923
4	M4N	Central Toronto	Lawrence Park	43.728020	-79.388790

New marked map


```
In [24]: map_toronto_denc = folium.Map(location=[latitude, longitude], zoom_start=12)
for lat, lng, borough, neighborhood in zip(
    df_toronto_denc['Latitude'],
    df_toronto_denc['Longitude'],
    df_toronto_denc['Borough'],
    df_toronto_denc['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_toronto_denc)
```

map_toronto_denc

Out[24]:



Define Foursquare Credentials and Version

On the public repository on Github, I has removed this field for the privacy!

```
In [25]: ► CLIENT_ID = 'DDUUMMYDDUUMMYDDUUMMYDDUUMMYDDUUMMY' # your Foursquare ID
CLIENT_SECRET = 'DDUUMMYDDUUMMYDDUUMMYDDUUMMYDDUUMMY' # your Foursquare Secret
VERSION = '12345678'
LIMIT = 30
print('Your credentails:')
print('CLIENT_ID: ' + CLIENT_ID)
print('CLIENT_SECRET:' + CLIENT_SECRET)
```

```
Your credentails:
CLIENT_ID: DDUUMMYDDUUMMYDDUUMMYDDUUMMYDDUUMMY
CLIENT_SECRET:DDUUMMYDDUUMMYDDUUMMYDDUUMMYDDUUMMY
```

Explore the first neighborhood in our data frame "df_toronto_denc"

```
In [28]: ► neighborhood_name = df_toronto_denc.loc[0, 'Neighborhood']
print(f"The first neighborhood's name is '{neighborhood_name}'.")
```

```
The first neighborhood's name is 'The Beaches'.
```

Get the neighborhood's latitude and longitude values.

```
In [29]: ► neighborhood_latitude = df_toronto_denc.loc[0, 'Latitude'] # neighborhood latitude value
neighborhood_longitude = df_toronto_denc.loc[0, 'Longitude'] # neighborhood Longitude value

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

```
Latitude and longitude values of The Beaches are 43.67635739999999, -79.2930312.
```

Now, let's get the top 100 venues that are in The Beaches within a radius of 500 meters.

```
In [31]: ▶ #CLIENT_ID = 'DDUUMMYDDUUMMYDDUUMMYDDUUMMYDDUUMMY' # your Foursquare ID
#CLIENT_SECRET = 'DDUUMMYDDUUMMYDDUUMMYDDUUMMY' # your Foursquare Secret
#VERSION = '12345678'
url = 'https://api.foursquare.com/v2/venues/explore?&client_id={}&client_secret={}&v={}&ll={},{}&radius={}&limit={}'.format(
    CLIENT_ID,
    CLIENT_SECRET,
    VERSION,
    neighborhood_latitude,
    neighborhood_longitude,
    radius,
    LIMIT)
# get the result to a json file
results = requests.get(url).json()
```

Function that extracts the category of the venue

```
In [32]: ▶ def get_category_type(row):
    try:
        categories_list = row['categories']
    except:
        categories_list = row['venue.categories']

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

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

```
In [33]: from pandas.io.json import json_normalize # tranform JSON file into a pandas dataframe
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(".")[0] for col in nearby_venues.columns]

nearby_venues
```

C:\ProgramData\Anaconda3\lib\site-packages\ipykernel_launcher.py:3: FutureWarning: pandas.io.json.json_normalize is deprecated, use pandas.json_normalize instead
This is separate from the ipykernel package so we can avoid doing imports until

Out[33]:

	name	categories	lat	lng
0	Glen Manor Ravine	Trail	43.676821	-79.293942
1	The Big Carrot Natural Food Market	Health Food Store	43.678879	-79.297734
2	Grover Pub and Grub	Pub	43.679181	-79.297215
3	Domino's Pizza	Pizza Place	43.679058	-79.297382
4	Upper Beaches	Neighborhood	43.680563	-79.292869

Explore neighborhoods in a part of Toronto City

We are working on the data frame df_toronto_denc. Recall that, this region contain DENC of Toronto where,

"DENC" = [D]owntown Toronto, [E]ast Toronto, [N]orth Toronto, [C]entral Toronto

First, let's create a function to repeat the same process to all the neighborhoods in DENC of Toronto.

```

In [36]: ▶ 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',
                            'Neighborhood Latitude',
                            'Neighborhood Longitude',
                            'Venue',
                            'Venue Latitude',
                            'Venue Longitude',
                            'Venue Category']

    return(nearby_venues)

```

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


```
In [35]: ► toronto_denc_venues = getNearbyVenues(names=df_toronto_denc['Neighborhood'],
                                                latitudes=df_toronto_denc['Latitude'],
                                                longitudes=df_toronto_denc['Longitude']
                                                )

toronto_denc_venues.head()
```

Out[35]:

	Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
0	The Beaches	43.676357	-79.293031	Glen Manor Ravine	43.676821	-79.293942	Trail
1	The Beaches	43.676357	-79.293031	The Big Carrot Natural Food Market	43.678879	-79.297734	Health Food Store
2	The Beaches	43.676357	-79.293031	Grover Pub and Grub	43.679181	-79.297215	Pub
3	The Beaches	43.676357	-79.293031	Domino's Pizza	43.679058	-79.297382	Pizza Place
4	The Beaches	43.676357	-79.293031	Upper Beaches	43.680563	-79.292869	Neighborhood

Let's check how many venues were returned for each neighborhood.

In [37]:

toronto_denc_venues.groupby('Neighborhood').count()

Out[37]:

	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
Neighborhood						
Adelaide, King, Richmond	100	100	100	100	100	100
Berczy Park	57	57	57	57	57	57
Brockton, Exhibition Place, Parkdale Village	22	22	22	22	22	22
Business Reply Mail Processing Centre 969 Eastern	18	18	18	18	18	18
CN Tower, Bathurst Quay, Island airport, Harbourfront West, King and Spadina, Railway Lands, South Niagara	16	16	16	16	16	16
Cabbagetown, St. James Town	44	44	44	44	44	44
Central Bay Street	79	79	79	79	79	79
Chinatown, Grange Park, Kensington Market	87	87	87	87	87	87
Christie	17	17	17	17	17	17
Church and Wellesley	85	85	85	85	85	85
Commerce Court, Victoria Hotel	100	100	100	100	100	100
Davisville	38	38	38	38	38	38
Davisville North	9	9	9	9	9	9
Deer Park, Forest Hill SE, Rathnelly, South Hill, Summerhill West	15	15	15	15	15	15
Design Exchange, Toronto Dominion Centre	100	100	100	100	100	100
Dovercourt Village, Dufferin	17	17	17	17	17	17
First Canadian Place, Underground city	100	100	100	100	100	100
Forest Hill North, Forest Hill West	4	4	4	4	4	4
Harbord, University of Toronto	39	39	39	39	39	39
Harbourfront	50	50	50	50	50	50
Harbourfront East, Toronto Islands, Union Station	100	100	100	100	100	100
High Park, The Junction South	23	23	23	23	23	23
Lawrence Park	3	3	3	3	3	3
Little Portugal, Trinity	57	57	57	57	57	57
Moore Park, Summerhill East	2	2	2	2	2	2
North Toronto West	20	20	20	20	20	20
Parkdale, Roncesvalles	15	15	15	15	15	15
Queen's Park	41	41	41	41	41	41
Rosedale	4	4	4	4	4	4
Roselawn	3	3	3	3	3	3
Runnymede, Swansea	38	38	38	38	38	38
Ryerson, Garden District	100	100	100	100	100	100

	Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
	St. James Town	100	100	100	100	100	100
	Stn A PO Boxes 25 The Esplanade	95	95	95	95	95	95
	Studio District	43	43	43	43	43	43
	The Annex, North Midtown, Yorkville	22	22	22	22	22	22
	The Beaches	5	5	5	5	5	5
	The Beaches West, India Bazaar	19	19	19	19	19	19
	The Danforth West, Riverdale	41	41	41	41	41	41

Let's find out how many unique categories can be curated from all the returned venues

```
In [38]: print('There are {} uniques categories.'.format(len(toronto_denc_venues['Venue Category'].unique())))
```

There are 236 uniques categories.

Analyze Each Neighborhood

```
In [39]: # one hot encoding
toronto_denc_onehot = pd.get_dummies(toronto_denc_venues[['Venue Category']], prefix="", prefix_sep="")

# add neighborhood column back to dataframe
toronto_denc_onehot['Neighborhood'] = toronto_denc_venues['Neighborhood']

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

toronto_denc_onehot.head()
```

Out[39]:

	Yoga Studio	Afghan Restaurant	Airport	Airport Food Court	Airport Gate	Airport Lounge	Airport Service	Airport Terminal	American Restaurant	Antique Shop	...	Toy / Game Store	Trail	Train Station	Vegetarian / Vegan Restaurant	Video Game Store	Vietnamese Restaurant	Wine Bar	Wine Shop	Wings Joint	Women's Store
0	0	0	0	0	0	0	0	0	0	0	...	0	1	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	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	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	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	0	0	0	0	0	0	0

5 rows × 236 columns

Now, let's group rows by neighborhood and by taking the mean of the frequency of occurrence of each category

In [40]:

```
toronto_denc_grouped = toronto_denc_onehot.groupby('Neighborhood').mean().reset_index()
toronto_denc_grouped.head()
```

Out[40]:

	Neighborhood	Yoga Studio	Afghan Restaurant	Airport	Airport Food Court	Airport Gate	Airport Lounge	Airport Service	Airport Terminal	American Restaurant	...	Toy / Game Store	Trail	Train Station	Vegetarian / Vegan Restaurant	Video Game Store	Vietnamese Restaurant	Wine Bar	Wine Shop	Wings Joint	Women's Store
0	Adelaide, King, Richmond	0.000000	0.0	0.0000	0.0000	0.0000	0.000	0.0000	0.000	0.02	...	0.0	0.0	0.0	0.020000	0.0	0.0	0.01	0.0	0.0	0.01
1	Berczy Park	0.000000	0.0	0.0000	0.0000	0.0000	0.000	0.0000	0.000	0.00	...	0.0	0.0	0.0	0.017544	0.0	0.0	0.00	0.0	0.0	0.00
2	Brockton, Exhibition Place, Parkdale Village	0.000000	0.0	0.0000	0.0000	0.0000	0.000	0.0000	0.000	0.00	...	0.0	0.0	0.0	0.000000	0.0	0.0	0.00	0.0	0.0	0.00
3	Business Reply Mail Processing Centre 969 Eastern	0.055556	0.0	0.0000	0.0000	0.0000	0.000	0.0000	0.000	0.00	...	0.0	0.0	0.0	0.000000	0.0	0.0	0.00	0.0	0.0	0.00
4	CN Tower, Bathurst Quay, Island airport, Harbo...	0.000000	0.0	0.0625	0.0625	0.0625	0.125	0.1875	0.125	0.00	...	0.0	0.0	0.0	0.000000	0.0	0.0	0.00	0.0	0.0	0.00

5 rows × 236 columns

Check the 10 most common venues in each neighborhood.

```
In [41]: 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]

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'] = toronto_denc_grouped['Neighborhood']

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

neighborhoods_venues_sorted.head()
```

Out[41]:

	Neighborhood	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue	7th Most Common Venue	8th Most Common Venue	9th Most Common Venue	10th Most Common Venue
0	Adelaide, King, Richmond	Coffee Shop	Thai Restaurant	Restaurant	Bar	Café	Steakhouse	Sushi Restaurant	Seafood Restaurant	Gastropub	Cosmetics Shop
1	Berczy Park	Coffee Shop	Seafood Restaurant	French Restaurant	Farmers Market	Bakery	Restaurant	Cheese Shop	Café	Cocktail Bar	Beer Bar
2	Brockton, Exhibition Place, Parkdale Village	Breakfast Spot	Café	Coffee Shop	Gym	Bakery	Stadium	Burrito Place	Restaurant	Climbing Gym	Pet Store
3	Business Reply Mail Processing Centre 969 Eastern	Yoga Studio	Auto Workshop	Park	Pizza Place	Restaurant	Butcher	Burrito Place	Brewery	Skate Park	Smoke Shop
4	CN Tower, Bathurst Quay, Island airport, Harbo...	Airport Service	Airport Terminal	Airport Lounge	Boutique	Rental Car Location	Boat or Ferry	Harbor / Marina	Sculpture Garden	Bar	Airport Gate

Cluster neighborhoods

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

```
In [42]: # set number of clusters to 5
kclusters = 5

toronto_denc_grouped_clustering = toronto_denc_grouped.drop('Neighborhood', 1)

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

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

```
Out[42]: array([1, 1, 1, 1, 1, 1, 1, 1, 1, 1])
```

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

```
In [43]: # add clustering labels
neighborhoods_venues_sorted.insert(0, 'Cluster Labels', kmeans.labels_)

toronto_denc_merged = df_toronto_denc

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

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

```
Out[43]:
```

	PostalCode	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 Most Common Venue	8th Most Common Venue	9th Most Common Venue	10th Most Common Venue
0	M4E	East Toronto	The Beaches	43.676357	-79.293031	2	Trail	Pizza Place	Health Food Store	Pub	Dog Run	Dim Sum Restaurant	Diner	Discount Store	Distribution Center	Women's Store
1	M4K	East Toronto	The Danforth West, Riverdale	43.679557	-79.352188	1	Greek Restaurant	Coffee Shop	Italian Restaurant	Ice Cream Shop	Bookstore	Furniture / Home Store	Lounge	Spa	Brewery	Bubble Tea Shop
2	M4L	East Toronto	The Beaches West, India Bazaar	43.668999	-79.315572	2	Sandwich Place	Park	Gym	Pub	Burrito Place	Fast Food Restaurant	Italian Restaurant	Fish & Chips Shop	Steakhouse	Sushi Restaurant
3	M4M	East Toronto	Studio District	43.659526	-79.340923	1	Café	Coffee Shop	Gastropub	Bakery	Brewery	Italian Restaurant	American Restaurant	Yoga Studio	Comfort Food Restaurant	Seafood Restaurant
4	M4N	Central Toronto	Lawrence Park	43.728020	-79.388790	2	Park	Bus Line	Swim School	Women's Store	Dim Sum Restaurant	Ethiopian Restaurant	Electronics Store	Eastern European Restaurant	Dumpling Restaurant	Donut Shop

```

In [44]: # create map
map_clusters = folium.Map(location=[latitude, longitude], zoom_start=11)

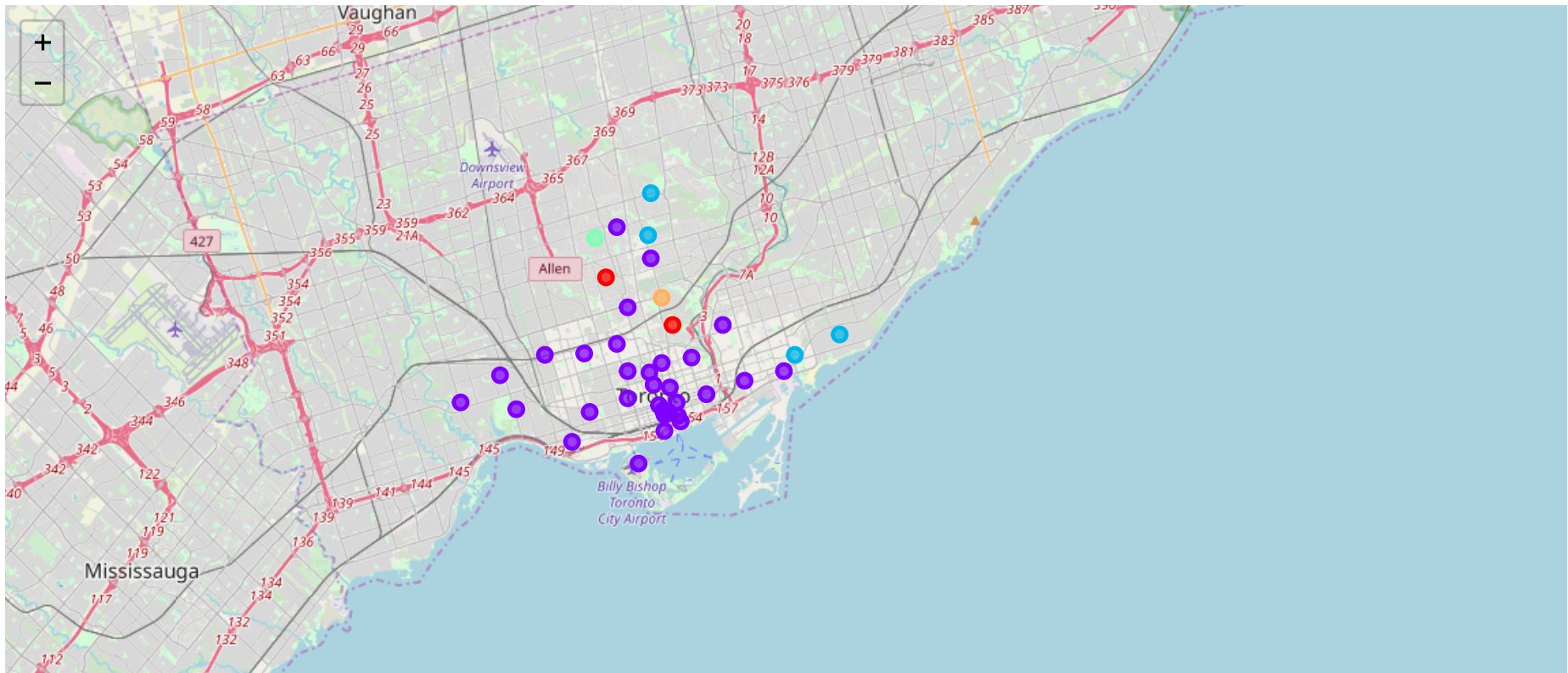
# set color scheme for the clusters
x = np.arange(kclusters)
ys = [i + x + (i*x)**2 for i in range(kclusters)]
colors_array = cm.rainbow(np.linspace(0, 1, len(ys)))
rainbow = [colors.rgb2hex(i) for i in colors_array]

# add markers to the map
markers_colors = []
for lat, lon, poi, cluster in zip(
    toronto_denc_merged['Latitude'],
    toronto_denc_merged['Longitude'],
    toronto_denc_merged['Neighborhood'],
    toronto_denc_merged['Cluster Labels']):
    label = folium.Popup(str(poi) + ' Cluster ' + str(cluster), parse_html=True)
    folium.CircleMarker(
        [lat, lon],
        radius=5,
        popup=label,
        color=rainbow[cluster-1],
        fill=True,
        fill_color=rainbow[cluster-1],
        fill_opacity=0.7).add_to(map_clusters)

map_clusters

```

Out[44]:





Leaflet (<http://leafletjs.com>)

Examine Clusters

Now, you can examine each cluster and determine the discriminating venue categories that distinguish each cluster.

Cluster 0

```
In [45]: ► toronto_denc_merged.loc[toronto_denc_merged['Cluster Labels'] == 0, toronto_denc_merged.columns[[1] + list(range(5, toronto_denc_merged.shape[1]))]]
```

Out[45]:

	Borough	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 Most Common Venue
10	Downtown Toronto	0	Park	Playground	Trail	Department Store	Ethiopian Restaurant	Electronics Store	Eastern European Restaurant	Dumpling Restaurant	Donut Shop	Doner Restaurant
23	Central Toronto	0	Park	Jewelry Store	Trail	Sushi Restaurant	Dessert Shop	Ethiopian Restaurant	Electronics Store	Eastern European Restaurant	Dumpling Restaurant	Donut Shop

Cluster 1


```
In [46]: toronto_denc_merged.loc[toronto_denc_merged['Cluster Labels'] == 1, toronto_denc_merged.columns[[1] + list(range(5, toronto_denc_merged.shape[1]))]]
```

Out[46]:

	Borough	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 Most Common Venue
1	East Toronto	1	Greek Restaurant	Coffee Shop	Italian Restaurant	Ice Cream Shop	Bookstore	Furniture / Home Store	Lounge	Spa	Brewery	Bubble Tea Shop
3	East Toronto	1	Café	Coffee Shop	Gastropub	Bakery	Brewery	Italian Restaurant	American Restaurant	Yoga Studio	Comfort Food Restaurant	Seafood Restaurant
6	Central Toronto	1	Clothing Store	Coffee Shop	Café	Restaurant	Dessert Shop	Miscellaneous Shop	Salon / Barbershop	Chinese Restaurant	Fast Food Restaurant	Diner
7	Central Toronto	1	Dessert Shop	Sandwich Place	Pizza Place	Italian Restaurant	Sushi Restaurant	Café	Coffee Shop	Gym	Indoor Play Area	Japanese Restaurant
9	Central Toronto	1	Coffee Shop	Pub	Pizza Place	Bagel Shop	Restaurant	Fried Chicken Joint	Sports Bar	Supermarket	American Restaurant	Liquor Store
11	Downtown Toronto	1	Coffee Shop	Café	Market	Italian Restaurant	Pizza Place	Bakery	Pub	Restaurant	Gastropub	Indian Restaurant
12	Downtown Toronto	1	Coffee Shop	Japanese Restaurant	Gay Bar	Restaurant	Sushi Restaurant	Pub	Men's Store	Mediterranean Restaurant	Hotel	Gastropub
13	Downtown Toronto	1	Coffee Shop	Park	Bakery	Pub	Mexican Restaurant	Breakfast Spot	Café	Restaurant	Theater	Shoe Store
14	Downtown Toronto	1	Coffee Shop	Clothing Store	Bubble Tea Shop	Middle Eastern Restaurant	Café	Japanese Restaurant	Ramen Restaurant	Italian Restaurant	Bookstore	Electronics Store
15	Downtown Toronto	1	Coffee Shop	Café	Restaurant	Clothing Store	Hotel	Breakfast Spot	American Restaurant	Cosmetics Shop	Italian Restaurant	Bakery
16	Downtown Toronto	1	Coffee Shop	Seafood Restaurant	French Restaurant	Farmers Market	Bakery	Restaurant	Cheese Shop	Café	Cocktail Bar	Beer Bar
17	Downtown Toronto	1	Coffee Shop	Italian Restaurant	Japanese Restaurant	Juice Bar	Sandwich Place	Burger Joint	Chinese Restaurant	Bar	Department Store	Salad Place
18	Downtown Toronto	1	Coffee Shop	Thai Restaurant	Restaurant	Bar	Café	Steakhouse	Sushi Restaurant	Seafood Restaurant	Gastropub	Cosmetics Shop
19	Downtown Toronto	1	Coffee Shop	Aquarium	Hotel	Café	Italian Restaurant	Scenic Lookout	Brewery	Fried Chicken Joint	Restaurant	Bakery
20	Downtown Toronto	1	Coffee Shop	Café	Restaurant	Hotel	Bakery	Seafood Restaurant	American Restaurant	Italian Restaurant	Japanese Restaurant	Bar
21	Downtown Toronto	1	Coffee Shop	Café	Restaurant	Hotel	Gym	American Restaurant	Seafood Restaurant	Japanese Restaurant	Deli / Bodega	Italian Restaurant
24	Central Toronto	1	Café	Sandwich Place	Coffee Shop	American Restaurant	Middle Eastern Restaurant	Pub	BBQ Joint	History Museum	Metro Station	Pizza Place
25	Downtown Toronto	1	Café	Bookstore	Restaurant	Bakery	Bar	Japanese Restaurant	Italian Restaurant	Flower Shop	Pub	Poutine Place
26	Downtown Toronto	1	Bar	Café	Vietnamese Restaurant	Vegetarian / Vegan Restaurant	Bakery	Coffee Shop	Chinese Restaurant	Mexican Restaurant	Dumpling Restaurant	Grocery Store
27	Downtown Toronto	1	Airport Service	Airport Terminal	Airport Lounge	Boutique	Rental Car Location	Boat or Ferry	Harbor / Marina	Sculpture Garden	Bar	Airport Gate
28	Downtown Toronto	1	Coffee Shop	Restaurant	Café	Japanese Restaurant	Seafood Restaurant	Beer Bar	Hotel	Italian Restaurant	Pub	Cheese Shop

	Borough	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 Most Common Venue
29	Downtown Toronto	1	Coffee Shop	Café	Restaurant	American Restaurant	Hotel	Seafood Restaurant	Bar	Steakhouse	Japanese Restaurant	Gym
30	Downtown Toronto	1	Grocery Store	Café	Park	Restaurant	Baby Store	Candy Store	Diner	Italian Restaurant	Coffee Shop	Gas Station
31	West Toronto	1	Bakery	Pharmacy	Music Venue	Bank	Brewery	Café	Art Gallery	Middle Eastern Restaurant	Pool	Gym / Fitness Center
32	West Toronto	1	Bar	Coffee Shop	Restaurant	Asian Restaurant	Men's Store	Café	Pizza Place	Bakery	Vietnamese Restaurant	Wine Bar
33	West Toronto	1	Breakfast Spot	Café	Coffee Shop	Gym	Bakery	Stadium	Burrito Place	Restaurant	Climbing Gym	Pet Store
34	West Toronto	1	Mexican Restaurant	Thai Restaurant	Bar	Café	Diner	Italian Restaurant	Bakery	Flea Market	Speakeasy	Fried Chicken Joint
35	West Toronto	1	Breakfast Spot	Gift Shop	Movie Theater	Eastern European Restaurant	Italian Restaurant	Dog Run	Bar	Bank	Dessert Shop	Bookstore
36	West Toronto	1	Coffee Shop	Café	Sushi Restaurant	Italian Restaurant	Pizza Place	Gym	Bookstore	Scenic Lookout	Sandwich Place	Restaurant
37	Downtown Toronto	1	Coffee Shop	Gym	Park	Burger Joint	Fast Food Restaurant	Portuguese Restaurant	Nightclub	Music Venue	Mexican Restaurant	Juice Bar
38	East Toronto	1	Yoga Studio	Auto Workshop	Park	Pizza Place	Restaurant	Butcher	Burrito Place	Brewery	Skate Park	Smoke Shop

Cluster 2

In [47]:

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

Out[47]:

	Borough	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 Most Common Venue
0	East Toronto	2	Trail	Pizza Place	Health Food Store	Pub	Dog Run	Dim Sum Restaurant	Diner	Discount Store	Distribution Center	Women's Store
2	East Toronto	2	Sandwich Place	Park	Gym	Pub	Burrito Place	Fast Food Restaurant	Italian Restaurant	Fish & Chips Shop	Steakhouse	Sushi Restaurant
4	Central Toronto	2	Park	Bus Line	Swim School	Women's Store	Dim Sum Restaurant	Ethiopian Restaurant	Electronics Store	Eastern European Restaurant	Dumpling Restaurant	Donut Shop
5	Central Toronto	2	Gym	Hotel	Convenience Store	Department Store	Sandwich Place	Dog Run	Breakfast Spot	Food & Drink Shop	Park	Gas Station

Cluster 3

```
In [48]: ► toronto_denc_merged.loc[toronto_denc_merged['Cluster Labels'] == 3, toronto_denc_merged.columns[[1] + list(range(5, toronto_denc_merged.shape[1]))]]
```

Out[48]:

	Borough	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 Most Common Venue
22	Central Toronto	3	Pool	Garden	Ice Cream Shop	Women's Store	Dessert Shop	Ethiopian Restaurant	Electronics Store	Eastern European Restaurant	Dumpling Restaurant	Donut Shop

Cluster 4

```
In [49]: ► toronto_denc_merged.loc[toronto_denc_merged['Cluster Labels'] == 4, toronto_denc_merged.columns[[1] + list(range(5, toronto_denc_merged.shape[1]))]]
```

Out[49]:

	Borough	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 Most Common Venue
8	Central Toronto	4	Playground	Tennis Court	Women's Store	Department Store	Ethiopian Restaurant	Electronics Store	Eastern European Restaurant	Dumpling Restaurant	Donut Shop	Doner Restaurant

End of Coursera Capstone Project: Segmenting and Clustering Neighborhoods in Toronto

Thank you ! Evaluator

Kapil Kumar Nagwanshi