Capstone Project Week 3

In this notebook all the threee parts of the assignment are included.

Include and setup of globals

In [1]:
```
import numpy as np

import pandas as pd
pd.set_option('display.max_columns', None)
pd.set_option('display.max_rows', None)

# !conda install -c conda-forge folium=0.5.0 --yes
import folium # map rendering library

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

# for webscraping import Beautiful Soup
from bs4 import BeautifulSoup

# library to process xml
import xml

# library to handle JSON files
import json

# library to handle requests
import requests

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

# tranform JSON file into a pandas dataframe
from pandas.io.json import json_normalize

# !conda install -c conda-forge geocoder --yes
import geocoder

# !conda install -c conda-forge geopy --yes
import geopy
from geopy.geocoders import Nominatim

print('Libraries imported.')


# your Foursquare ID
CLIENT_ID = '...'
# your Foursquare Secret
CLIENT_SECRET = '...'
# Foursquare API version
VERSION = '20180605'

FOURSCARE_FULL_ONLINE = False

TORONTO_LATITUDE  = 43.6529
TORONTO_LONGITUDE = -79.3849

print('Globals setup.')
```

```
Libraries imported.
Globals setup.
```

Capstone Project Week 3 - Part 1

Assignment

For this assignment, you will be required to explore and cluster the neighborhoods in Toronto.

1. Start by creating a new Notebook for this assignment.
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)](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 below:
3. To create the above dataframe:
   - The dataframe will consist of three columns: PostalCode, Borough, and Neighborhood
   - Only process the cells that have an assigned borough. Ignore cells with a borough that is Not assigned.
   - 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.
   - If a cell has a borough but a Not assigned neighborhood, then the neighborhood will be the same as the borough. So for the 9th cell in the table on the Wikipedia page, the value of the Borough and the Neighborhood columns will be Queen's Park.
   - Clean your Notebook and add Markdown cells to explain your work and any assumptions you are making.
   - In the last cell of your notebook, use the .shape method to print the number of rows of your dataframe.
4. Submit a link to your Notebook on your Github repository. (10 marks)

Get Data about Toronto

Initialize Web Scraper and pull data from Wikipedia page

In [2]:
```
url = requests.get('https://en.wikipedia.org/wiki/List_of_postal_codes_of_Canada:_M').te
xt
soup = BeautifulSoup(url,'lxml')
```

Extract html tags for table \< table> and rows \< tr>

In [3]:
```
#Find table
table = soup.find('table')

#Find ands extract rows
rows = []
table_rows = table.select('tr')
for row in table_rows:
    rows.append(row.get_text())

print("Rows loaded: ",len(rows))
```

```
Rows loaded:  288
```

Build a pandas dataframe from loaded rows, split entries into columns and name columns

Note: *dataframe has two extra columns*

In [4]:
```
#Build dataframe, split columns and update columns
df_str = pd.DataFrame(rows)
df_raw = df_str[0].str.split('\n', expand=True)

#Assign names to columns and remove header row
df_raw.rename(columns=df_raw.iloc[0], inplace=True)
df_raw.drop(df_raw.index[0], inplace=True)

print('Shape:', df_raw.shape)
df_raw.head(8)
```

```
Shape: (287, 5)
```

Out[4]:

	Postcode	Borough	Neighbourhood
1	M1A	Not assigned	Not assigned
2	M2A	Not assigned	Not assigned
3	M3A	North York	Parkwoods
4	M4A	North York	Victoria Village
5	M5A	Downtown Toronto	Harbourfront
6	M6A	North York	Lawrence Heights
7	M6A	North York	Lawrence Manor
8	M7A	Queen's Park	Not assigned

Extract columns required

Note: *two extracted columns removed*

In [5]:
```
#Extract relevant columns
df_pure = df_raw[['Postcode', 'Borough', 'Neighbourhood']]

#Adjust column naming
df_pure.columns = ['Postcode', 'Borough', 'Neighborhood']

print('Shape:', df_pure.shape)
df_pure.head(8)
```

```
Shape: (287, 3)
```

Out[5]:

	Postcode	Borough	Neighborhood
1	M1A	Not assigned	Not assigned
2	M2A	Not assigned	Not assigned
3	M3A	North York	Parkwoods
4	M4A	North York	Victoria Village
5	M5A	Downtown Toronto	Harbourfront
6	M6A	North York	Lawrence Heights
7	M6A	North York	Lawrence Manor
8	M7A	Queen's Park	Not assigned

Drop rows which have "Not Assigned" in Borough

In [6]:
```
# Filter entries which have 'Not assigned' in Borough column
borough_notassigned = df_pure[df_pure['Borough'] == 'Not assigned'].index

# Delete these row indexes from dataFrame
df_clean = df_pure.drop(borough_notassigned)

print('Shape:', df_clean.shape)
df_clean.head(8)
```

```
Shape: (210, 3)
```

Out[6]:

	Postcode	Borough	Neighborhood
3	M3A	North York	Parkwoods
4	M4A	North York	Victoria Village
5	M5A	Downtown Toronto	Harbourfront
6	M6A	North York	Lawrence Heights
7	M6A	North York	Lawrence Manor
8	M7A	Queen's Park	Not assigned
10	M9A	Downtown Toronto	Queen's Park
11	M1B	Scarborough	Rouge

Set Neighborhoods with value "Not Assigned" to the value of the Borough

In [7]:
```
#Replace Not assigned in Neighborhood to value of Borough
df_clean.loc[(df_clean.Neighborhood == 'Not assigned'),'Neighborhood'] = df_clean.Boroug
h

print('Shape:', df_clean.shape)
df_clean.head(8)
```

```
Shape: (210, 3)
```

Out[7]:

	Postcode	Borough	Neighborhood
3	M3A	North York	Parkwoods
4	M4A	North York	Victoria Village
5	M5A	Downtown Toronto	Harbourfront
6	M6A	North York	Lawrence Heights
7	M6A	North York	Lawrence Manor
8	M7A	Queen's Park	Queen's Park
10	M9A	Downtown Toronto	Queen's Park
11	M1B	Scarborough	Rouge

Group data on Postalcode and Borough

In [8]:
```
#Merge same PostCodes
df_grouped = df_clean.groupby(['Postcode', 'Borough'])['Neighborhood'].apply(', '.join).
reset_index()

print('Shape:', df_grouped.shape)
df_grouped.head(8)
```

```
Shape: (103, 3)
```

Out[8]:

	Postcode	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

Save dataframe in CSV

In [9]:
```
df_grouped.to_csv('capstone-data-package-part-1.csv', index = False)
print("Saved.")
```

```
Saved.
```

Capstone Project Week 3 - Part 2

Assignment

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.

Given that the geocoder package has been experienced as unreliable, data were comlpemented by the csv file here: http://cocl.us/Geospatial_data (http://cocl.us/Geospatial_data) to create a dataframe as follows:

Once you are able to create the above dataframe, submit a link to the new Notebook on your Github repository. (2 marks)

Prepare Exploration of Toronto

Load cleansed dataset from csv file

In [10]:
```
df_cleansed = pd.read_csv('capstone-data-package-part-1.csv')
print('Shape: ',df_cleansed.shape)
df_cleansed.head(10)
```

```
Shape:  (103, 3)
```

Out[10]:

	Postcode	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

Load geospatial data

Note: *Alternative approach instead of first download file is direct download (code commented out)*

In [ ]:
```
# Geocoder is unstable, therefore I disabled this part and I will use the CSV file provi
ded by the instructor
'''
# Install and import geocoder
!conda install -c conda-forge geocoder --yes
import geocoder # import geocoder

# initialize your variable to None
lat_lng_coords = None

for postal_code in toronto_grouped.Postcode:
    # loop until you get the coordinates
    while(lat_lng_coords is None):
          g = geocoder.google('{}, Toronto, Ontario'.format(postal_code))
          lat_lng_coords = g.latlng
    latitude = lat_lng_coords[0]
    longitude = lat_lng_coords[1]
'''
```

In [11]:
```
!wget -q -O 'geospatial_data.csv' http://cocl.us/Geospatial_data
print('Data downloaded...')

df_geo = pd.read_csv('geospatial_data.csv')
df_geo.rename(columns={'Postal Code': 'Postcode'}, inplace=True)
print('Shape: ', df_geo.shape)
df_geo.head()
```

```
Data downloaded...
Shape:  (103, 3)
```

Out[11]:

	Postcode	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

Merge geospatial data with Boroughs along the Postcode

In [12]:
```
df_extended = pd.merge(df_cleansed, df_geo, on='Postcode')
print('Data extended...')
print('Shape:', df_extended.shape)
df_extended.head(8)
```

```
Data extended...
Shape: (103, 5)
```

Out[12]:

	Postcode	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
5	M1J	Scarborough	Scarborough Village	43.744734	-79.239476
6	M1K	Scarborough	East Birchmount Park, Ionview, Kennedy Park	43.727929	-79.262029
7	M1L	Scarborough	Clairlea, Golden Mile, Oakridge	43.711112	-79.284577

Save dataframe into CSV

In [13]:
```
df_extended.to_csv('capstone-data-package-part-2.csv', index = False)
print("Saved.")
```

```
Saved.
```

Capstone Project Week 3 - Part 3

Assignment

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.

Once you are happy with your analysis, submit a link to the new Notebook on your Github repository. (3 marks)

Explore Toronto's Bouroughs and Neighborhoods

Load cleansed dataset from csv file

In [14]:
```
neighborhoods = pd.read_csv('capstone-data-package-part-2.csv')
neighborhoods.head()
```

Out[14]:

	Postcode	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

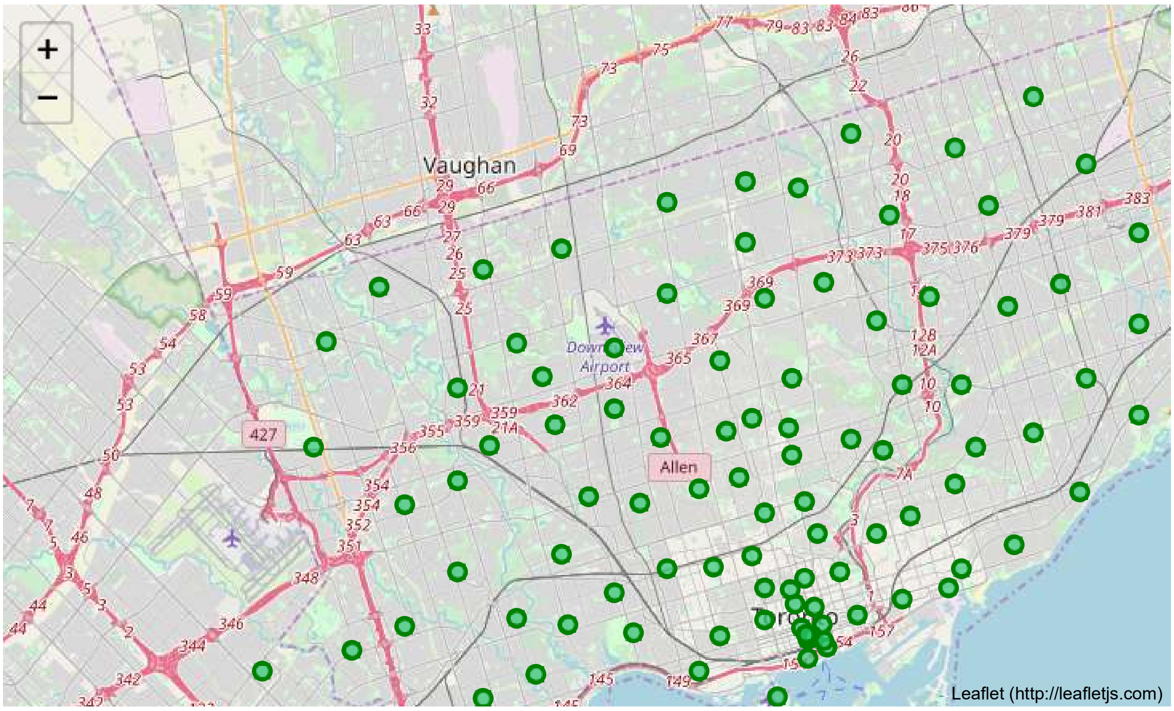
Preprare helper function to build a map using folium

In [15]:
```
def draw_map_and_neighborhood(df,latitude, longitude):

        map = folium.Map(location=[latitude, longitude], zoom_start=11)

        # draw markers on map
        for lat, lng, borough, neighborhood in zip(df['Latitude'], df['Longitude'], df['
Borough'], df['Neighborhood']):

            label = '{}, {}'.format(neighborhood, borough)

            label = folium.Popup(label, parse_html=True)

            folium.CircleMarker(
                [lat, lng],
                radius=5,
                popup=label,
                color='green',
                fill=True,
                fill_color='#31cc77',
                fill_opacity=0.7,
                parse_html=False).add_to(map)

        return map
```

Build Toronto Map with Neighborhoods

In [16]:
```
# evaluate a center of venues
latitude = neighborhoods['Latitude'].median()
longitude = neighborhoods['Longitude'].median()

draw_map_and_neighborhood(neighborhoods, latitude, longitude)
```

Out[16]:



Select Neigborhood in Boroughs with keyword 'Toronto'

In [17]:
```
selected_neighborhood = neighborhoods[neighborhoods['Borough'].str.contains('Toronto')]
selected_neighborhood.reset_index(drop=True, inplace=True)
print('Shape:',selected_neighborhood.shape)
selected_neighborhood.head()
```

```
Shape: (39, 5)
```

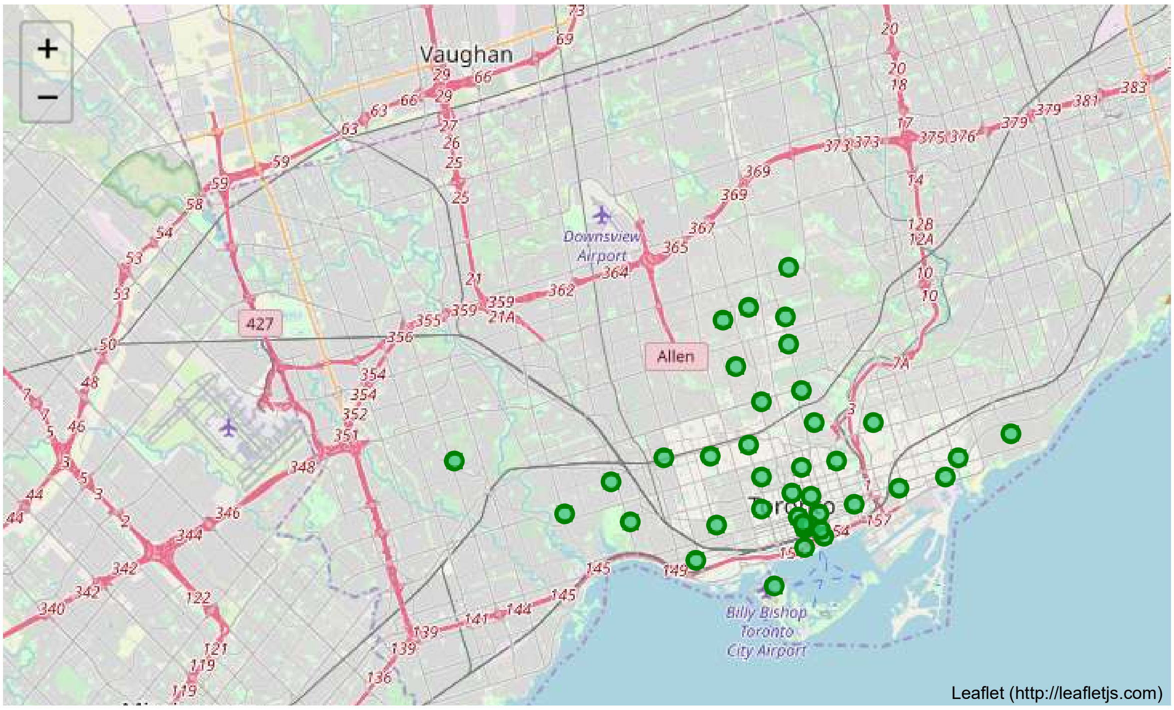
Out[17]:

	Postcode	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

Display selected neighborhood on the map

In [18]:
```
# evaluate a center of venues
latitude = selected_neighborhood['Latitude'].median()
longitude = selected_neighborhood['Longitude'].median()

draw_map_and_neighborhood(selected_neighborhood, latitude, longitude)
```

Out[18]:



Initial exploration

Select first neighborhood

In [19]:
```
#get center of bourgouh
select_id = 0
selected_neighborhood.loc[select_id]
```

Out[19]:
```
Postcode                 M4E
Borough         East Toronto
Neighborhood     The Beaches
Latitude             43.6764
Longitude            -79.293
Name: 0, dtype: object
```

Pull data from foursquare

In [ ]:
```
LIMIT = 200
radius = 999

url = 'https://api.foursquare.com/v2/venues/explore?&client_id={}&client_secret={}&v={}&
ll={},{}&radius={}&limit={}'.format(
        CLIENT_ID,
        CLIENT_SECRET,
        VERSION,
        selected_neighborhood.loc[select_id, 'Latitude'],
        selected_neighborhood.loc[select_id, 'Longitude'],
        radius,
        LIMIT)

print(url)

results = requests.get(url).json()
results
```

Build function to clean up specific result rows

In [21]:
```
# 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']
```

Process result received from foursquare

In [22]:
```
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.locat
ion.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.columns = ['Venue', 'Category', 'Latitude', 'Longitude']

print('Shape:',nearby_venues.shape)
nearby_venues.head()
```

```
Shape: (82, 4)
```

Out[22]:

	Venue	Category	Latitude	Longitude
0	Glen Manor Ravine	Trail	43.676821	-79.293942
1	Tori's Bakeshop	Vegetarian / Vegan Restaurant	43.672114	-79.290331
2	The Fox Theatre	Indie Movie Theater	43.672801	-79.287272
3	Ed's Real Scoop	Ice Cream Shop	43.672630	-79.287993
4	The Beech Tree	Gastropub	43.680493	-79.288846

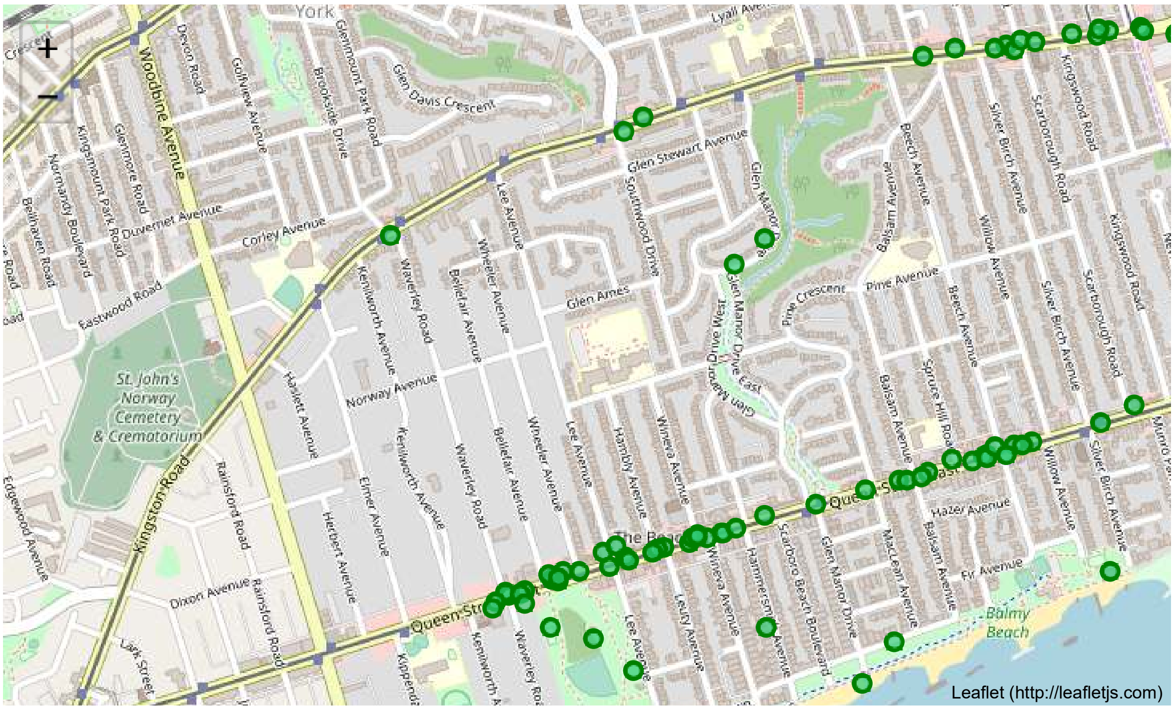
Build function to display dataframe of venues on a map

In [23]:
```python
def draw_map_and_venues(df_venues, start_lat, start_long):

        map = folium.Map(location=[start_lat, start_long], zoom_start=15)

        # draw markers on map
        for lat, lng, categories, name in zip(df_venues['Latitude'],
                                              df_venues['Longitude'],
                                              df_venues['Category'],
                                              df_venues['Venue']):

            label = '{}, {}'.format(categories, name)

            label = folium.Popup(label, parse_html=True)

            folium.CircleMarker(
                [lat, lng],
                radius=5,
                popup=label,
                color='green',
                fill=True,
                fill_color='#31cc77',
                fill_opacity=0.7,
                parse_html=False).add_to(map)

        return map
```

Display map with venues within the Bourough

In [24]:
```python
# evaluate a center of venues
latitude = nearby_venues['Latitude'].median()
longitude = nearby_venues['Longitude'].median()

#draw map with venues
draw_map_and_venues(nearby_venues, latitude, longitude)
```

Out[24]:



Explore all venues across all neighborhoods

Build function, which automates all the steps performed before along a list of coordinates

In [25]:
```
def getNearbyVenues(names, latitudes, longitudes, radius=500):

    LIMIT = 200

    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_secre
t={}&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_l
ist])
    nearby_venues.columns = ['Neighborhood',
                  'Neighborhood Latitude',
                  'Neighborhood Longitude',
                  'Venue',
                  'Latitude',
                  'Longitude',
                  'Category']

    return(nearby_venues)
```

Get all venues from foursquare and process response

_**Note:** The FOURSCARE_FULL*ONLINE is used to reduce the number of API calls during development*

In [26]:

```
if FOURSCARE_FULL_ONLINE:
    # Pull data from foursquare
    selected_neighborhood_venues = getNearbyVenues(names=selected_neighborhood['Neighbor
hood'],
                                                    latitudes=selected_neighborhood['Lati
tude'],
                                                    longitudes=selected_neighborhood['Lon
gitude'],
                                                    radius=999
                                                    )
    # Store result in file
    selected_neighborhood_venues.to_csv('capstone-data-package-part-3-neighborhood_venue
s.csv', index = False)
    print("Result pulled, processed and saved.")
else:
    # Load data from file instead from foursquare
    selected_neighborhood_venues = pd.read_csv('capstone-data-package-part-3-neighborhoo
d_venues.csv')
    print("Result Loaded.")

print('Shape:',selected_neighborhood_venues.shape)
selected_neighborhood_venues.head()
```

```
Result Loaded.
Shape: (3092, 7)
```

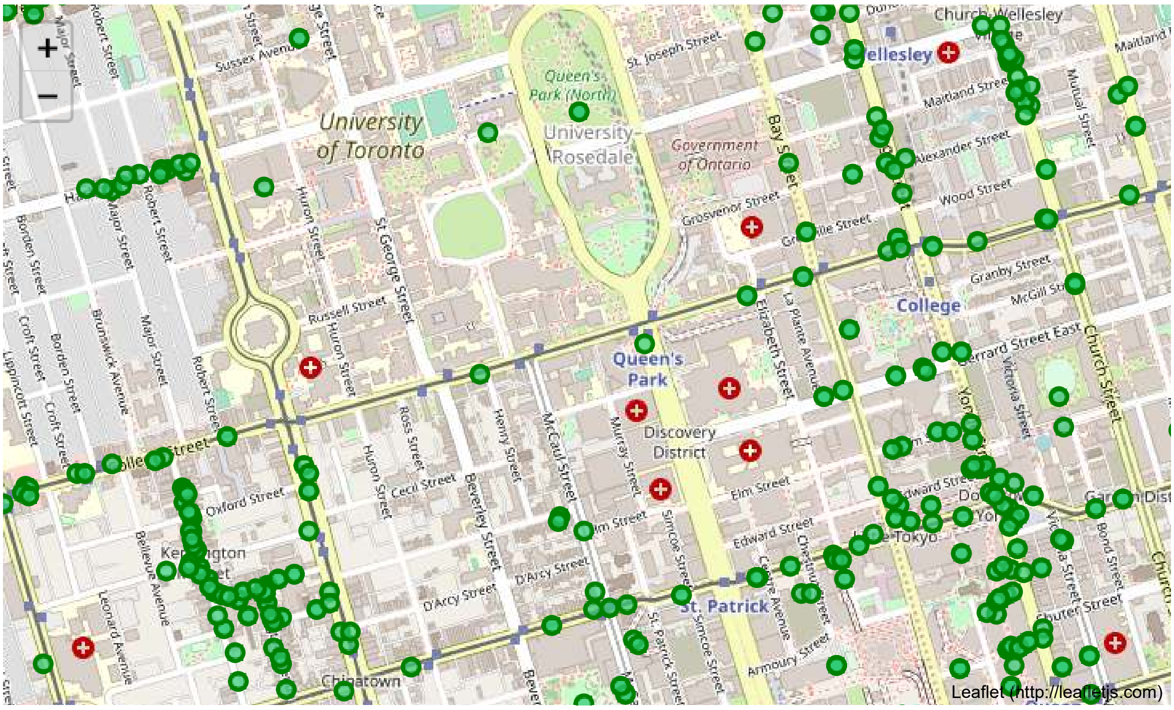
Out[26]:

	Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Latitude	Longitude	Category
0	The Beaches	43.676357	-79.293031	Glen Manor Ravine	43.676821	-79.293942	Trail
1	The Beaches	43.676357	-79.293031	Tori's Bakeshop	43.672114	-79.290331	Vegetarian / Vegan Restaurant
2	The Beaches	43.676357	-79.293031	The Fox Theatre	43.672801	-79.287272	Indie Movie Theater
3	The Beaches	43.676357	-79.293031	Ed's Real Scoop	43.672630	-79.287993	Ice Cream Shop
4	The Beaches	43.676357	-79.293031	The Beech Tree	43.680493	-79.288846	Gastropub

Build map with all venues in the selected neighborhoods

In [27]:
```
# evaluate a center of venues
latitude = selected_neighborhood_venues['Latitude'].median()
longitude = selected_neighborhood_venues['Longitude'].median()

# draw map with venues
draw_map_and_venues(selected_neighborhood_venues, latitude, longitude)
```

Out[27]:



Check how many venues were returned for per neighborhood

In [28]:
```
selected_neighborhood_venues.groupby('Neighborhood').count()
```

Out[28]:

Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Latitude	Longitude	Category
Adelaide, King, Richmond	100	100	100	100	100	100
Berczy Park	100	100	100	100	100	100
Brockton, Exhibition Place, Parkdale Village	100	100	100	100	100	100
Business Reply Mail Processing Centre 969 Eastern	48	48	48	48	48	48
CN Tower, Bathurst Quay, Island airport, Harbourfront West, King and Spadina, Railway Lands, South Niagara	15	15	15	15	15	15
Cabbagetown, St. James Town	36	36	36	36	36	36
Central Bay Street	100	100	100	100	100	100
Chinatown, Grange Park, Kensington Market	100	100	100	100	100	100
Christie	100	100	100	100	100	100
Church and Wellesley	100	100	100	100	100	100
Commerce Court, Victoria Hotel	100	100	100	100	100	100
Davisville	100	100	100	100	100	100
Davisville North	100	100	100	100	100	100
Deer Park, Forest Hill SE, Rathnelly, South Hill, Summerhill West	77	77	77	77	77	77
Design Exchange, Toronto Dominion Centre	100	100	100	100	100	100
Dovercourt Village, Dufferin	69	69	69	69	69	69
First Canadian Place, Underground city	100	100	100	100	100	100
Forest Hill North, Forest Hill West	46	46	46	46	46	46
Harbord, University of Toronto	100	100	100	100	100	100
Harbourfront	100	100	100	100	100	100
Harbourfront East, Toronto Islands, Union Station	100	100	100	100	100	100
High Park, The Junction South	100	100	100	100	100	100
Lawrence Park	8	8	8	8	8	8
Little Portugal, Trinity	100	100	100	100	100	100
Moore Park, Summerhill East	59	59	59	59	59	59
North Toronto West	43	43	43	43	43	43
Parkdale, Roncesvalles	100	100	100	100	100	100
Queen's Park	10	10	10	10	10	10
Rosedale	22	22	22	22	22	22
Roselawn	24	24	24	24	24	24
Runnymede, Swansea	73	73	73	73	73	73
Ryerson, Garden District	100	100	100	100	100	100
St. James Town	100	100	100	100	100	100
Stn A PO Boxes 25 The Esplanade	100	100	100	100	100	100
Studio District	100	100	100	100	100	100
The Annex, North Midtown, Yorkville	100	100	100	100	100	100
The Beaches	82	82	82	82	82	82
The Beaches West, India Bazaar	80	80	80	80	80	80
The Danforth West, Riverdale	100	100	100	100	100	100

Evaluate the number of venues categories listed

In [29]:
```
print('There are {} uniques categories.'.format(len(selected_neighborhood_venues['Catego
ry'].unique())))
```

```
There are 278 uniques categories.
```

Analyze the Neighborhoods

In [30]:
```
# one hot encoding
venues_onehot = pd.get_dummies(selected_neighborhood_venues[['Category']], prefix="", pr
efix_sep="")

# add neighborhood column back to dataframe
venues_onehot['Neighborhood'] = selected_neighborhood_venues['Neighborhood']

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

print("Shape: ",venues_onehot.shape)
venues_onehot.head()
```

```
Shape:  (3092, 278)
```

Out[30]:

	Zoo	Accessories Store	Afghan Restaurant	Airport	Airport Lounge	American Restaurant	Amphitheater	Animal Shelter	Antique Shop	Aquarium	Argentinian Restaurant
0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0

Group rows by neighborhood and by taking the mean of the frequency of occurrence of each category

In [31]:
```
venues_grouped = venues_onehot.groupby('Neighborhood').mean().reset_index()
print("Shape: ", venues_grouped.shape)
venues_grouped.head()
```

```
Shape:  (39, 278)
```

Out[31]:

	Neighborhood	Zoo	Accessories Store	Afghan Restaurant	Airport	Airport Lounge	American Restaurant	Amphitheater	Animal Shelter	Antique Shop	Aqua
0	Adelaide, King, Richmond	0.0	0.00	0.0	0.000000	0.000000	0.020000	0.0	0.0	0.0	
1	Berczy Park	0.0	0.00	0.0	0.000000	0.000000	0.010000	0.0	0.0	0.0	
2	Brockton, Exhibition Place, Parkdale Village	0.0	0.01	0.0	0.000000	0.000000	0.010000	0.0	0.0	0.0	
3	Business Reply Mail Processing Centre 969 Eastern	0.0	0.00	0.0	0.000000	0.000000	0.020833	0.0	0.0	0.0	
4	CN Tower, Bathurst Quay, Island airport, Harbo...	0.0	0.00	0.0	0.066667	0.066667	0.000000	0.0	0.0	0.0	

Create a function to sort the venues in descending order.

In [32]:
```
def return_most_common_venues(row, num_top_venues):

    #remove first row
    row_categories = row.iloc[1:]

    #sort rows
    row_categories_sorted = row_categories.sort_values(ascending=False)

    #sort return only the defined number of entries
    return row_categories_sorted.index.values[0:num_top_venues]
```

Put the top 5 common venuwa into *pandas* dataframe

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 with the new columns
neighborhoods_venues_sorted = pd.DataFrame(columns=columns)
neighborhoods_venues_sorted['Neighborhood'] = venues_grouped['Neighborhood']

# process all neighborhoods
for ind in np.arange(venues_grouped.shape[0]):
    neighborhoods_venues_sorted.iloc[ind, 1:] = return_most_common_venues(venues_groupe
d.iloc[ind, :], num_top_venues)

print('Shape:', neighborhoods_venues_sorted.shape)
neighborhoods_venues_sorted.head()
```

```
Shape: (39, 11)
```

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	8th Most Common Venue	9th Most Common Venue
0	Adelaide, King, Richmond	Café	Hotel	Coffee Shop	Theater	Sushi Restaurant	Ramen Restaurant	Restaurant	Bakery	Steakhouse
1	Berczy Park	Coffee Shop	Café	Hotel	Beer Bar	Restaurant	Japanese Restaurant	Seafood Restaurant	Steakhouse	Italian Restaurant
2	Brockton, Exhibition Place, Parkdale Village	Café	Coffee Shop	Restaurant	Bakery	Bar	Furniture / Home Store	Vegetarian / Vegan Restaurant	Tibetan Restaurant	Lounge
3	Business Reply Mail Processing Centre 969 Eastern	Park	Coffee Shop	Pizza Place	Brewery	Pet Store	Sushi Restaurant	Italian Restaurant	Flea Market	French Restaurant
4	CN Tower, Bathurst Quay, Island airport, Harbo...	Harbor / Marina	Coffee Shop	Garden	Café	Airport	Airport Lounge	Sculpture Garden	Dog Run	Tunnel

Cluster Neighborhoods

Run k-means to cluster the neighborhood along venues categories

In [34]:
```
# set number of clusters
kclusters = 5

venues_grouped_clustering = venues_grouped.drop('Neighborhood', 1)

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

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

Out[34]:
```
array([0, 0, 4, 0, 1, 4, 0, 4, 4, 0], dtype=int32)
```

Create a new dataframe which includes the cluster as well as the top 10 venues for each neighborhood.

In [35]:
```
# remove clustering labels in case the column is already there
# neighborhoods_venues_sorted.drop('Cluster', axis=1, inplace=True)

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

# Prepare dataframe to merge with cordinates
neighborhoods_merged = selected_neighborhood

# merge neighborhoods_venues_sorted with df_explore to add latitude/longitude for each n
eighborhood
neighborhoods_merged = neighborhoods_merged.join(neighborhoods_venues_sorted.set_index('
Neighborhood'), on='Neighborhood')

neighborhoods_merged.head()
```

Out[35]:

	Postcode	Borough	Neighborhood	Latitude	Longitude	Cluster	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5 C
0	M4E	East Toronto	The Beaches	43.676357	-79.293031	0	Pub	Coffee Shop	Pizza Place	Japanese Restaurant	
1	M4K	East Toronto	The Danforth West, Riverdale	43.679557	-79.352188	0	Greek Restaurant	Coffee Shop	Café	Pub	Ic
2	M4L	East Toronto	The Beaches West, India Bazaar	43.668999	-79.315572	4	Indian Restaurant	Coffee Shop	Park	Café	
3	M4M	East Toronto	Studio District	43.659526	-79.340923	4	Coffee Shop	Bar	Vietnamese Restaurant	Bakery	A Re
4	M4N	Central Toronto	Lawrence Park	43.728020	-79.388790	2	Bookstore	College Quad	Gym / Fitness Center	College Gym	

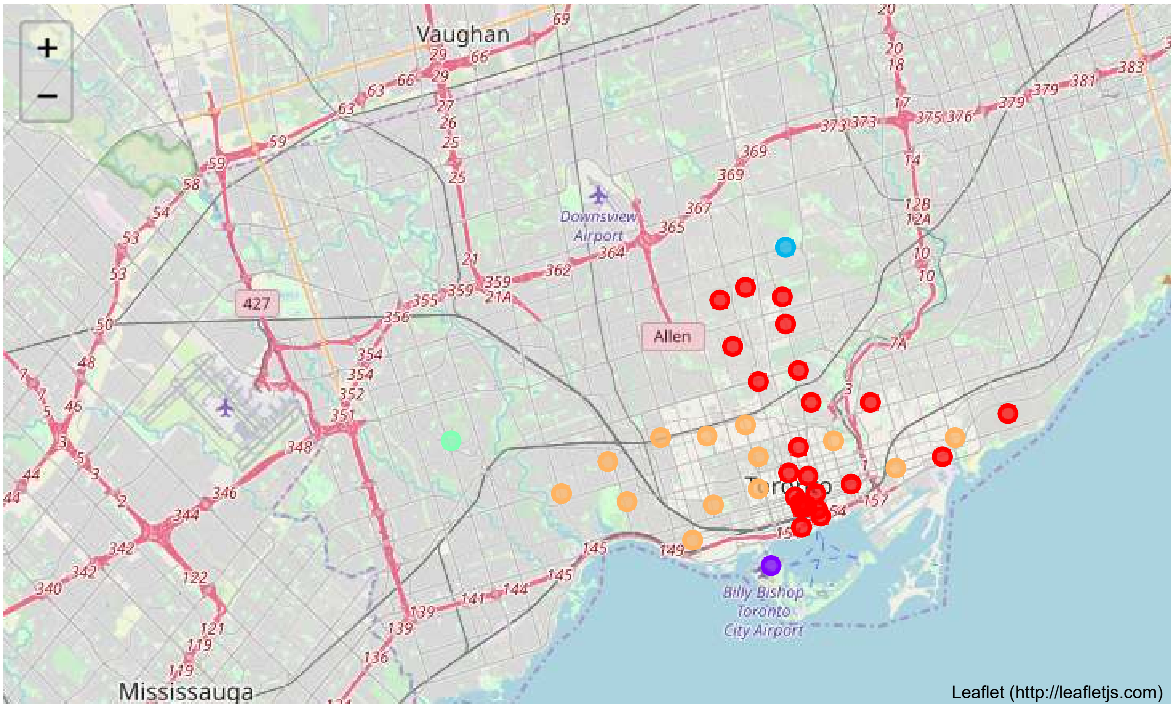
Apply some data cleansing

In [36]:
```
# drop rows undifined clusters
#neighborhoods_merged = neighborhoods_merged[~neighborhoods_merged['Cluster Label'].isnu
ll()]
neighborhoods_merged = neighborhoods_merged.dropna()

# ensure cluster labels are int (can get changed due to NaN entries)
neighborhoods_merged['Cluster'] = neighborhoods_merged['Cluster'].astype(int)
```

Visualize the clustering

In [37]:
```
# 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(neighborhoods_merged['Latitude'],
                                  neighborhoods_merged['Longitude'],
                                  neighborhoods_merged['Neighborhood'],
                                  neighborhoods_merged['Cluster']):

    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[37]:



Analyze Clusters

Cluster 0: Leisure and well-being area with many restaurants, pubs, gyms and some well-being offerings

In [38]:
```
print('Shape:', neighborhoods_merged.shape)

def showCluster(cluster):
    return neighborhoods_merged.loc[neighborhoods_merged['Cluster'] == cluster, neighbor
hoods_merged.columns[[2] + list(range(6, neighborhoods_merged.shape[1]))]]
```

```
Shape: (39, 16)
```

In [39]:
```
showCluster(0)
```

Out[39]:

	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	
0	The Beaches	Pub	Coffee Shop	Pizza Place	Japanese Restaurant	Park	Bakery	Bar	Beach	
1	The Danforth West, Riverdale	Greek Restaurant	Coffee Shop	Café	Pub	Ice Cream Shop	Italian Restaurant	Fast Food Restaurant	Restaurant	P
5	Davisville North	Coffee Shop	Italian Restaurant	Fast Food Restaurant	Café	Sushi Restaurant	Pharmacy	Pizza Place	Dessert Shop	
6	North Toronto West	Park	Skating Rink	Sporting Goods Shop	Italian Restaurant	Coffee Shop	Diner	Café	Restaurant	Del
7	Davisville	Coffee Shop	Italian Restaurant	Sushi Restaurant	Pizza Place	Pub	Gym	Indian Restaurant	Café	Des
8	Moore Park, Summerhill East	Italian Restaurant	Coffee Shop	Grocery Store	Gym	Park	Bagel Shop	Pizza Place	Pub	F
9	Deer Park, Forest Hill SE, Rathnelly, South Hi...	Coffee Shop	Sushi Restaurant	Park	Italian Restaurant	Thai Restaurant	Gym / Fitness Center	Sandwich Place	Pub	B
10	Rosedale	Coffee Shop	Park	Grocery Store	Metro Station	BBQ Joint	Playground	Convenience Store	Sandwich Place	Ca
12	Church and Wellesley	Coffee Shop	Japanese Restaurant	Sushi Restaurant	Park	Gay Bar	Men's Store	Café	Italian Restaurant	Med F
13	Harbourfront	Coffee Shop	Theater	Café	Restaurant	Park	Pub	Italian Restaurant	Diner	
14	Ryerson, Garden District	Coffee Shop	Clothing Store	Middle Eastern Restaurant	Tea Room	Diner	Italian Restaurant	Cosmetics Shop	Fast Food Restaurant	F
15	St. James Town	Coffee Shop	Café	Hotel	Restaurant	Cosmetics Shop	Italian Restaurant	Seafood Restaurant	Bakery	
16	Berczy Park	Coffee Shop	Café	Hotel	Beer Bar	Restaurant	Japanese Restaurant	Seafood Restaurant	Steakhouse	F
17	Central Bay Street	Coffee Shop	Italian Restaurant	Café	Japanese Restaurant	Ramen Restaurant	Park	Mexican Restaurant	Gastropub	F
18	Adelaide, King, Richmond	Café	Hotel	Coffee Shop	Theater	Sushi Restaurant	Ramen Restaurant	Restaurant	Bakery	S
19	Harbourfront East, Toronto Islands, Union Station	Coffee Shop	Café	Hotel	Restaurant	Aquarium	Italian Restaurant	Bar	Japanese Restaurant	
20	Design Exchange, Toronto Dominion Centre	Coffee Shop	Hotel	Café	Italian Restaurant	Steakhouse	Gastropub	Restaurant	Bakery	F
21	Commerce Court, Victoria Hotel	Coffee Shop	Café	Hotel	Japanese Restaurant	Steakhouse	Beer Bar	Restaurant	Concert Hall	F
22	Roselawn	Sushi Restaurant	Pharmacy	Coffee Shop	Bank	Café	Italian Restaurant	Bakery	Japanese Restaurant	B
23	Forest Hill North, Forest Hill West	Park	Café	Coffee Shop	Trail	Burger Joint	Liquor Store	Sushi Restaurant	Deli / Bodega	F
28	Stn A PO Boxes 25 The Esplanade	Coffee Shop	Café	Restaurant	Hotel	Japanese Restaurant	Beer Bar	Gastropub	Art Gallery	C
29	First Canadian Place, Underground city	Hotel	Café	Coffee Shop	Italian Restaurant	Steakhouse	Restaurant	Theater	Concert Hall	
37	Business Reply Mail Processing Centre 969 Eastern	Park	Coffee Shop	Pizza Place	Brewery	Pet Store	Sushi Restaurant	Italian Restaurant	Flea Market	F

Cluster 1: Nature and Scenery

In [40]:
```
showCluster(1)
```

Out[40]:

	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	1 M Comn Ver
27	CN Tower, Bathurst Quay, Island airport, Harbo...	Harbor / Marina	Coffee Shop	Garden	Café	Airport	Airport Lounge	Sculpture Garden	Dog Run	Tunnel	Sce Look

Cluster 2: A little bit of everything with focus on education

In [41]:
```
showCluster(2)
```

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 M Comn Ve
4	Lawrence Park	Bookstore	College Quad	Gym / Fitness Center	College Gym	Coffee Shop	Café	Park	Trail	Yoga Studio	Eas Europ Restau

Cluster 3: A little bit of everything with focus on recreation and some outdoor activities

In [42]:
```
showCluster(3)
```

Out[42]:

	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 Co
38	Queen's Park	Pharmacy	Playground	Grocery Store	Shopping Mall	Skating Rink	Park	Café	Golf Course	Bank	E Eur Rest

Cluster 4: Eating and Drinking around the world

In [43]:
```
showCluster(4)
```

Out[43]:

	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	9 C
2	The Beaches West, India Bazaar	Indian Restaurant	Coffee Shop	Park	Café	Beach	Italian Restaurant	Burger Joint	Burrito Place	S
3	Studio District	Coffee Shop	Bar	Vietnamese Restaurant	Bakery	American Restaurant	Brewery	Diner	Italian Restaurant	
11	Cabbagetown, St. James Town	Gastropub	Park	Japanese Restaurant	Diner	Café	Caribbean Restaurant	Taiwanese Restaurant	Jewelry Store	Stea
24	The Annex, North Midtown, Yorkville	Café	Pub	Coffee Shop	Restaurant	Gym	Vegetarian / Vegan Restaurant	Italian Restaurant	Museum	
25	Harbord, University of Toronto	Café	Bar	Bakery	Vegetarian / Vegan Restaurant	Coffee Shop	Restaurant	Bookstore	Mexican Restaurant	Te
26	Chinatown, Grange Park, Kensington Market	Café	Bar	Vegetarian / Vegan Restaurant	Art Gallery	Vietnamese Restaurant	Coffee Shop	Bakery	Mexican Restaurant	
30	Christie	Korean Restaurant	Café	Coffee Shop	Grocery Store	Cocktail Bar	Ice Cream Shop	Ethiopian Restaurant	Pizza Place	Re
31	Dovercourt Village, Dufferin	Café	Coffee Shop	Park	Bar	Sushi Restaurant	Brewery	Gourmet Shop	Convenience Store	
32	Little Portugal, Trinity	Café	Bar	Bakery	Restaurant	Italian Restaurant	Coffee Shop	Asian Restaurant	Pizza Place	
33	Brockton, Exhibition Place, Parkdale Village	Café	Coffee Shop	Restaurant	Bakery	Bar	Furniture / Home Store	Vegetarian / Vegan Restaurant	Tibetan Restaurant	
34	High Park, The Junction South	Café	Bar	Coffee Shop	Thai Restaurant	Italian Restaurant	Fast Food Restaurant	Convenience Store	Park	Re
35	Parkdale, Roncesvalles	Coffee Shop	Bar	Sushi Restaurant	Pizza Place	Restaurant	Pub	Bakery	Breakfast Spot	
36	Runnymede, Swansea	Coffee Shop	Café	Bakery	Pizza Place	Italian Restaurant	Gastropub	Sushi Restaurant	Park	Re

Save result to CSV File

In [44]:
```
# Save to CSV File
neighborhoods_merged.to_csv("capstone-data-package-part-3-final.csv")
```

Summary and closing thoughts

With clustering, urban areas could be characterised as follows:

Cluster 0: Leisure and well-being

Central area of Toronto can be characterized as leisure and well-being area with many restaurants, pubs, gyms and some well-being offerings

Cluster 1: Nature and Scenery

Solely on Toronto Island recreation are, dominated by nature and scenery offering

Cluster 2: A little bit of everything with focus on education

Quite external area with a broad but lean offering

Cluster 3: A little bit of everything with focus on recreation and some outdoor activities

External area with offering outdoor activities

Cluster 4: Eating and Drinking around the world

Central area of Toronto for eating and drinking around the world

In [ ]:

```
```