Explore, segment, and cluster the neighborhoods in the city of Toronto

In this assignment, you will be required to explore, segment, and cluster the neighborhoods in the city of Toronto. However, unlike New York, the neighborhood data is not readily available on the internet. What is interesting about the field of data science is that each project can be challenging in its unique way, so you need to learn to be agile and refine the skill to learn new libraries and tools quickly depending on the project.

For the Toronto neighborhood data, a Wikipedia page exists that has all the information we need to explore and cluster the neighborhoods in Toronto. You will be required to scrape the Wikipedia page and wrangle the data, clean it, and then read it into a pandas dataframe so that it is in a structured format like the New York dataset.

Once the data is in a structured format, you can replicate the analysis that we did to the New York City dataset to explore and cluster the neighborhoods in the city of Toronto.

Your submission will be a link to your Jupyter Notebook on your Github repository.

Get Toronto neighborhood data

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), 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:
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. 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.
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.
10. Submit a link to your Notebook on your Github repository.

Pull the data

Let's load the necessary libraries first...

In [1]:

```
### libraries
import requests
import pandas as pd
pd.set_option('display.max_columns', 100)
pd.set_option('display.max_rows', 100)

#!conda install -c conda-forge beautifulsoup4 lxml html5lib --yes
from bs4 import BeautifulSoup
```

Now let's pull down the data from that Wiki page...

In [2]:

```
### data
wiki_url = "https://en.wikipedia.org/wiki/List_of_postal_codes_of_Canada:_M"

# request
myres = requests.get(wiki_url)
mysoup = BeautifulSoup(myres.content, 'html5lib')

# read table into df
mytable = mysoup.find_all('table')[0]
mydf = pd.read_html(str(mytable))[0]

# check it
mydf.head()
```

Out[2]:

	Postcode	Borough	Neighbourhood
0	M1A	Not assigned	Not assigned
1	M2A	Not assigned	Not assigned
2	M3A	North York	Parkwoods
3	M4A	North York	Victoria Village
4	M5A	Downtown Toronto	Harbourfront

Process the data

Time for some filtering and postprocessing...

In [3]:

```
# get rid of the "Not assigned" boroughs
mydf_bor = mydf[~mydf.Borough.isin(['Not assigned'])]
mydf_bor.head()
```

Out[3]:

	Postcode	Borough	Neighbourhood
2	M3A	North York	Parkwoods
3	M4A	North York	Victoria Village
4	M5A	Downtown Toronto	Harbourfront
5	M5A	Downtown Toronto	Regent Park
6	M6A	North York	Lawrence Heights

Postcodes appearing multiple times should be combined into one line with the neighbourhoods concatenated...

In [4]:

```
# make those postcodes unique
mydf_combined = mydf_bor.groupby(['Postcode','Borough'])['Neighbourhood'].apply(', '.join).
mydf_combined.head()
```

Out[4]:

	Postcode	Borough	Neighbourhood
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

Finally, let's fix those rows where the Neighbourhood is "not assigned" by filling it with the content of the Borough column...

In [5]:

```
# fix the neighbourhood column where it contains "not assigned"
mydf_combined.loc[mydf_combined['Neighbourhood']=="Not assigned", 'Neighbourhood'] = mydf_c
mydf_combined.head()
```

Out[5]:

	Postcode	Borough	Neighbourhood
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 [6]:

```
# save the Toronto data to CSV for later use
mydf_combined.to_csv('toronto_postcodes.csv', index=False)
```

In [7]:

```
mydf_combined.shape
```

Out[7]:

```
(103, 3)
```

Enrich the Data

Getting geospatial coordinates

Since the routine for obtaining geo coordinates described in the task outline did indeed prove to be rather unreliable, I opted for proceeding with the provided CSV file.

In [8]:

```
# read geo coordinates from CSV
mydf_geo = pd.read_csv('Geospatial_Coordinates.csv')
mydf_geo.head()
```

Out[8]:

	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

Let's add the 2 new Latitude and Longitude columns to the existing data frame.

In [9]:

```
# add geo columns to the existing data frame, remove redundant Postal Code column
mydf_post = pd.merge(mydf_combined, mydf_geo, how='left', left_on='Postcode', right_on='Pos
mydf_post.drop('Postal Code', axis=1, inplace=True)
mydf_post.head()
```

Out[9]:

	Postcode	Borough	Neighbourhood	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

In [10]:

```
mydf_post.shape
```

Out[10]:

```
(103, 5)
```

Explore and cluster the neighborhoods in Toronto

In the following, we want to separate Toronto's Neighbourhoods into different clusters as a guide to people thinking about moving to or within Toronto. To do so, we use the previously prepared data and enrich it with the venues present in Toronto using Foursquare as an external data source. With this data, we proceed to determine the top 10 venues found in each neighbourhood and use this data to actually find clusters. In the process, we will also make an informed decision about how many clusters make sense and proceed to visualize them using Folium, after which the clusters are inspected and a final conclusion will be drawn.

Let's download all the dependencies that we will need.

In [11]:

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

import json # library to handle JSON files

#!conda install -c conda-forge geopy --yes # uncomment this line if you haven't completed t
from geopy.geocoders import Nominatim # convert an address into latitude and longitude valu

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

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

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

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

print('Libraries imported.')
```

```
Libraries imported.
```

Create a map of Toronto with neighborhoods superimposed on top.

Since our analysis will focus on Toronto, let's determine the central coordinates using Nominatim.

In [12]:

```
address = 'Toronto, ON'

geolocator = Nominatim(user_agent="toronto_explorer")
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.
```

To verify that we're on the right track, let's quickly plot a map centered on those coordinates.

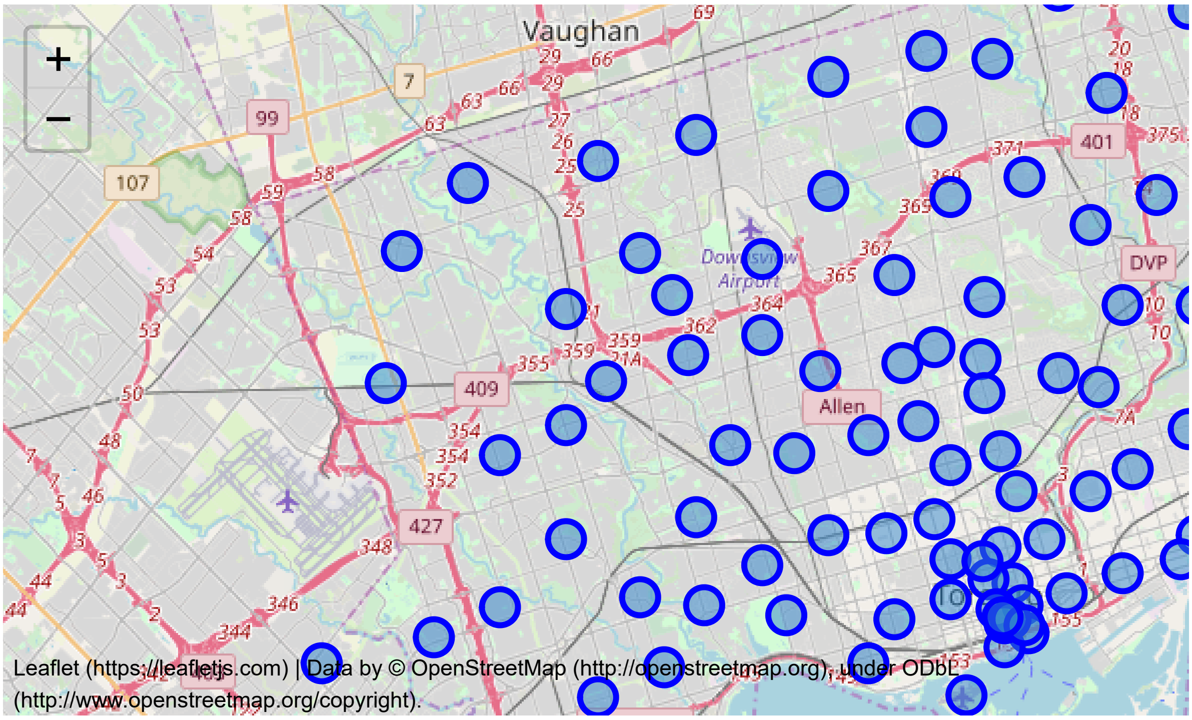
In [13]:

```
# create map of Toronto using latitude and longitude values
map_toronto = folium.Map(location=[latitude, longitude], zoom_start=11)

# add markers to map
for lat, lng, borough, neighborhood in zip(mydf_post['Latitude'], mydf_post['Longitude'], m
    label = '{}, {}'.format(neighborhood, borough)
    label = folium.Popup(label, parse_html=True)
    folium.CircleMarker(
        [lat, lng],
        radius=9,
        popup=label,
        color='blue',
        fill=True,
        fill_color='#3287cd',
        fill_opacity=0.5,
        parse_html=False).add_to(map_toronto)

map_toronto
```

Out[13]:



Get Foursquare data

First off, we need to enter our credentials in order to access the Foursquare API.

In [14]:

```
CLIENT_ID = 'EOGS2ZA3IH1DZAGOT0G0MLFHMRLQSAV1TMGAIW4N2EJEGPFG' # your Foursquare ID
CLIENT_SECRET = 'M2QXMSBBAOTLRW5RCOSNJUGYM0SRZB3WMT3QMR4SB1HVOYIT' # your Foursquare Secret
VERSION = '20180605' # Foursquare API version

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

Next, let's set some sensible defaults to limit the amount of data we are pulling from Foursquare.

In [15]:

```
LIMIT = 100 # limit of number of venues returned by Foursquare API
radius = 500 # define radius
```

To make gathering the nearby venues more efficient, let's create a function to pull the relevant data for all the Boroughs that returns the data in a neat data frame.

In [16]:

```
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={}&
            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 = ['Neighbourhood',
                  'Neighbourhood Latitude',
                  'Neighbourhood Longitude',
                  'Venue',
                  'Venue Latitude',
                  'Venue Longitude',
                  'Venue Category']

    return(nearby_venues)
```

Now let's call the function with our data as parameter.

In [17]:

```
toronto_venues = getNearbyVenues(names=mydf_post['Neighbourhood'],
                                   latitudes=mydf_post['Latitude'],
                                   longitudes=mydf_post['Longitude']
                                  )
```

```
Rouge, Malvern
Highland Creek, Rouge Hill, Port Union
Guildwood, Morningside, West Hill
Woburn
Cedarbrae
Scarborough Village
East Birchmount Park, Ionview, Kennedy Park
Clairlea, Golden Mile, Oakridge
Cliffcrest, Cliffside, Scarborough Village West
Birch Cliff, Cliffside West
Dorset Park, Scarborough Town Centre, Wexford Heights
Maryvale, Wexford
Agincourt
Clarks Corners, Sullivan, Tam O'Shanter
Agincourt North, L'Amoreaux East, Milliken, Steeles East
L'Amoreaux West
Upper Rouge
Hillcrest Village
Fairview, Henry Farm, Oriole
Bayview Village
Silver Hills, York Mills
Newtonbrook, Willowdale
Willowdale South
York Mills West
Willowdale West
Parkwoods
Don Mills North
Flemingdon Park, Don Mills South
Bathurst Manor, Downsview North, Wilson Heights
Northwood Park, York University
CFB Toronto, Downsview East
Downsview West
Downsview Central
Downsview Northwest
Victoria Village
Woodbine Gardens, Parkview Hill
Woodbine Heights
The Beaches
Leaside
Thorncliffe Park
East Toronto
The Danforth West, Riverdale
The Beaches West, India Bazaar
Studio District
Lawrence Park
Davisville North
North Toronto West
Davisville
Moore Park, Summerhill East
Deer Park, Forest Hill SE, Rathnelly, South Hill, Summerhill West
Rosedale
Cabbagetown, St. James Town
Church and Wellesley
Harbourfront, Regent Park
```

```
Ryerson, Garden District
St. James Town
Berczy Park
Central Bay Street
Adelaide, King, Richmond
Harbourfront East, Toronto Islands, Union Station
Design Exchange, Toronto Dominion Centre
Commerce Court, Victoria Hotel
Bedford Park, Lawrence Manor East
Roselawn
Forest Hill North, Forest Hill West
The Annex, North Midtown, Yorkville
Harbord, University of Toronto
Chinatown, Grange Park, Kensington Market
CN Tower, Bathurst Quay, Island airport, Harbourfront West, King and Spadin
a, Railway Lands, South Niagara
Stn A PO Boxes 25 The Esplanade
First Canadian Place, Underground city
Lawrence Heights, Lawrence Manor
Glencairn
Humewood-Cedarvale
Caledonia-Fairbanks
Christie
Dovercourt Village, Dufferin
Little Portugal, Trinity
Brockton, Exhibition Place, Parkdale Village
Downsview, North Park, Upwood Park
Del Ray, Keelesdale, Mount Dennis, Silverthorn
The Junction North, Runnymede
High Park, The Junction South
Parkdale, Roncesvalles
Runnymede, Swansea
Queen's Park
Canada Post Gateway Processing Centre
Business Reply Mail Processing Centre 969 Eastern
Humber Bay Shores, Mimico South, New Toronto
Alderwood, Long Branch
The Kingsway, Montgomery Road, Old Mill North
Humber Bay, King's Mill Park, Kingsway Park South East, Mimico NE, Old Mill
South, The Queensway East, Royal York South East, Sunnylea
Kingsway Park South West, Mimico NW, The Queensway West, Royal York South We
st, South of Bloor
Islington Avenue
Cloverdale, Islington, Martin Grove, Princess Gardens, West Deane Park
Bloordale Gardens, Eringate, Markland Wood, Old Burnhamthorpe
Humber Summit
Emery, Humberlea
Weston
Westmount
Kingsview Village, Martin Grove Gardens, Richview Gardens, St. Phillips
Albion Gardens, Beaumond Heights, Humbergate, Jamestown, Mount Olive, Silver
stone, South Steeles, Thistletown
Northwest
```

Let's inspect how much data we got back.

In [18]:

```
print(toronto_venues.shape)
toronto_venues.head()
```

```
(2244, 7)
```

Out[18]:

	Neighbourhood	Neighbourhood Latitude	Neighbourhood Longitude	Venue	Venue Latitude	Venue Longitude	Venu Categor
0	Rouge, Malvern	43.806686	-79.194353	Wendy's	43.807448	-79.199056	Fast Foo Restaurar
1	Highland Creek, Rouge Hill, Port Union	43.784535	-79.160497	Royal Canadian Legion	43.782533	-79.163085	Ba
2	Guildwood, Morningside, West Hill	43.763573	-79.188711	Swiss Chalet Rotisserie & Grill	43.767697	-79.189914	Pizz Plac
3	Guildwood, Morningside, West Hill	43.763573	-79.188711	G & G Electronics	43.765309	-79.191537	Electronic Stor
4	Guildwood, Morningside, West Hill	43.763573	-79.188711	Big Bite Burrito	43.766299	-79.190720	Mexica Restaurar

Analyze Neighbourhoods

With the venue data from Foursquare, we are now ready to analyze the neighborhoods. Since we are interested in the frequency of the venue categories in each neighbourhood, we'll first one-hot encode those features and proceed with calculating the relative frequencies.

In [19]:

```
toronto_venues['Neighbourhood'].unique
```

Out[19]:

```
<bound method Series.unique of 0                                                   Ro
uge, Malvern
1                   Highland Creek, Rouge Hill, Port Union
2                        Guildwood, Morningside, West Hill
3                        Guildwood, Morningside, West Hill
4                        Guildwood, Morningside, West Hill
5                        Guildwood, Morningside, West Hill
6                        Guildwood, Morningside, West Hill
7                        Guildwood, Morningside, West Hill
8                        Guildwood, Morningside, West Hill
9                                                   Woburn
10                                                  Woburn
11                                                  Woburn
12                                               Cedarbrae
13                                               Cedarbrae
14                                               Cedarbrae
15                                               Cedarbrae
16                                               Cedarbrae
17                                               Cedarbrae
18                                               Cedarbrae
19                                     Scarborough Village
20                                     Scarborough Village
21                                     Scarborough Village
22              East Birchmount Park, Ionview, Kennedy Park
23              East Birchmount Park, Ionview, Kennedy Park
24              East Birchmount Park, Ionview, Kennedy Park
25              East Birchmount Park, Ionview, Kennedy Park
26              East Birchmount Park, Ionview, Kennedy Park
27              East Birchmount Park, Ionview, Kennedy Park
28                            Clairlea, Golden Mile, Oakridge
29                            Clairlea, Golden Mile, Oakridge
30                            Clairlea, Golden Mile, Oakridge
31                            Clairlea, Golden Mile, Oakridge
32                            Clairlea, Golden Mile, Oakridge
33                            Clairlea, Golden Mile, Oakridge
34                            Clairlea, Golden Mile, Oakridge
35                            Clairlea, Golden Mile, Oakridge
36                            Clairlea, Golden Mile, Oakridge
37          Cliffcrest, Cliffside, Scarborough Village West
38          Cliffcrest, Cliffside, Scarborough Village West
39          Cliffcrest, Cliffside, Scarborough Village West
40                                Birch Cliff, Cliffside West
41                                Birch Cliff, Cliffside West
42                                Birch Cliff, Cliffside West
43                                Birch Cliff, Cliffside West
44                                Birch Cliff, Cliffside West
45        Dorset Park, Scarborough Town Centre, Wexford ...
46        Dorset Park, Scarborough Town Centre, Wexford ...
47        Dorset Park, Scarborough Town Centre, Wexford ...
48        Dorset Park, Scarborough Town Centre, Wexford ...
49        Dorset Park, Scarborough Town Centre, Wexford ...
                                  ...
2194      Kingsway Park South West, Mimico NW, The Queen...
2195      Kingsway Park South West, Mimico NW, The Queen...
2196      Kingsway Park South West, Mimico NW, The Queen...
```

```
2197    Kingsway Park South West, Mimico NW, The Queen...
2198    Kingsway Park South West, Mimico NW, The Queen...
2199    Kingsway Park South West, Mimico NW, The Queen...
2200    Kingsway Park South West, Mimico NW, The Queen...
2201    Kingsway Park South West, Mimico NW, The Queen...
2202    Kingsway Park South West, Mimico NW, The Queen...
2203    Kingsway Park South West, Mimico NW, The Queen...
2204    Kingsway Park South West, Mimico NW, The Queen...
2205    Kingsway Park South West, Mimico NW, The Queen...
2206    Kingsway Park South West, Mimico NW, The Queen...
2207    Cloverdale, Islington, Martin Grove, Princess ...
2208    Bloordale Gardens, Eringate, Markland Wood, Ol...
2209    Bloordale Gardens, Eringate, Markland Wood, Ol...
2210    Bloordale Gardens, Eringate, Markland Wood, Ol...
2211    Bloordale Gardens, Eringate, Markland Wood, Ol...
2212    Bloordale Gardens, Eringate, Markland Wood, Ol...
2213    Bloordale Gardens, Eringate, Markland Wood, Ol...
2214    Bloordale Gardens, Eringate, Markland Wood, Ol...
2215    Bloordale Gardens, Eringate, Markland Wood, Ol...
2216    Bloordale Gardens, Eringate, Markland Wood, Ol...
2217                                        Humber Summit
2218                                        Humber Summit
2219                                     Emery, Humberlea
2220                                               Weston
2221                                               Weston
2222                                            Westmount
2223                                            Westmount
2224                                            Westmount
2225                                            Westmount
2226                                            Westmount
2227                                            Westmount
2228                                            Westmount
2229                                            Westmount
2230    Kingsview Village, Martin Grove Gardens, Richv...
2231    Kingsview Village, Martin Grove Gardens, Richv...
2232    Albion Gardens, Beaumond Heights, Humbergate, ...
2233    Albion Gardens, Beaumond Heights, Humbergate, ...
2234    Albion Gardens, Beaumond Heights, Humbergate, ...
2235    Albion Gardens, Beaumond Heights, Humbergate, ...
2236    Albion Gardens, Beaumond Heights, Humbergate, ...
2237    Albion Gardens, Beaumond Heights, Humbergate, ...
2238    Albion Gardens, Beaumond Heights, Humbergate, ...
2239    Albion Gardens, Beaumond Heights, Humbergate, ...
2240    Albion Gardens, Beaumond Heights, Humbergate, ...
2241                                            Northwest
2242                                            Northwest
2243                                            Northwest
Name: Neighbourhood, Length: 2244, dtype: object>
```

In [20]:

```
# one hot encoding
toronto_onehot = pd.get_dummies(toronto_venues[['Venue Category']], prefix="", prefix_sep="
print(toronto_onehot.shape)

# add neighborhood column back to dataframe
toronto_onehot['Neighbourhood'] = toronto_venues['Neighbourhood']
print(toronto_onehot.shape)

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

toronto_onehot.head()
```

```
(2244, 280)
(2244, 281)
```

Out[20]:

	Neighbourhood	Accessories Store	Afghan Restaurant	Airport	Airport Food Court	Airport Gate	Airport Lounge	Airport Service	Airpor Termina
0	Rouge, Malvern	0	0	0	0	0	0	0	(
1	Highland Creek, Rouge Hill, Port Union	0	0	0	0	0	0	0	(
2	Guildwood, Morningside, West Hill	0	0	0	0	0	0	0	(
3	Guildwood, Morningside, West Hill	0	0	0	0	0	0	0	(
4	Guildwood, Morningside, West Hill	0	0	0	0	0	0	0	(

5 rows × 281 columns

Now let's group by Neighbourhood to determine the frequencies of the venue types.

In [21]:

```
toronto_grouped = toronto_onehot.groupby('Neighbourhood').mean().reset_index()
print(toronto_grouped.shape)
toronto_grouped.head()
```

(101, 281)

Out[21]:

	Neighbourhood	Accessories Store	Afghan Restaurant	Airport	Airport Food Court	Airport Gate	Airport Lounge	Airport Service	Airpor Termina
0	Adelaide, King, Richmond	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.(
1	Agincourt	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.(
2	Agincourt North, L'Amoreaux East, Milliken, St...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.(
3	Albion Gardens, Beaumond Heights, Humbergate, ...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.(
4	Alderwood, Long Branch	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.(

5 rows × 281 columns

Obviously, we're dealing with a very sparse data set here, so let's determine the top 10 most frequent venue types per neighbourhood. First, let's create a function that sorts the venue frequencies in descending order.

In [22]:

```
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]
```

Now let's create the new dataframe containing the top 10 venues for each neighborhood.

In [23]:

```
num_top_venues = 10

indicators = ['st', 'nd', 'rd']

# create columns according to number of top venues
columns = ['Neighbourhood']
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['Neighbourhood'] = toronto_grouped['Neighbourhood']

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

neighborhoods_venues_sorted.head()
```

Out[23]:

	Neighbourhood	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 Mc Comm Ven
0	Adelaide, King, Richmond	Coffee Shop	Café	Thai Restaurant	Bar	Steakhouse	Gym	Restaura
1	Agincourt	Lounge	Breakfast Spot	Skating Rink	Chinese Restaurant	Sandwich Place	Eastern European Restaurant	Dor Restaura
2	Agincourt North, L'Amoreaux East, Milliken, St...	Park	Playground	Yoga Studio	Eastern European Restaurant	Dive Bar	Dog Run	Dor Restaura
3	Albion Gardens, Beaumond Heights, Humbergate, ...	Grocery Store	Fast Food Restaurant	Pizza Place	Sandwich Place	Coffee Shop	Beer Store	Pharma
4	Alderwood, Long Branch	Pizza Place	Gym	Pool	Skating Rink	Pharmacy	Pub	Coffi Sh

Neighbourhood clustering

We're now ready to cluster the neighbourhoods based on the prevalence of venue types.

Finding k

One question that always comes up is how to choose k for the clustering. We solve this by looking at the inertia of the clusters and applying the elbox rule.

In [24]:

```
import matplotlib.pyplot as plt

# remove Neighbourhood column
toronto_grouped_clustering = toronto_grouped.drop('Neighbourhood', 1)


# set number of clusters to test
maxk = 8
cost = np.zeros((maxk-1))

for n in range(1, maxk):
    # run k-means clustering
    kmeans = KMeans(n_clusters=n, random_state=0).fit(toronto_grouped_clustering)
    cost[n-1] = kmeans.inertia_
    #print(cost[n-1])

# plot inertia to find best value for K
plt.plot(range(2, maxk), cost[1:maxk], 'g')
plt.show()
```

```
<Figure size 640x480 with 1 Axes>
```

Performing the clustering

Judging from the plot above, it looks like setting k to 7 will be the best balance, so let's run the final clustering like this.

In [25]:

```
kclusters = 7
kmeans = KMeans(n_clusters=kclusters, random_state=0).fit(toronto_grouped_clustering)
# check cluster labels generated for each row in the dataframe
kmeans.labels_[0:10]
```

Out[25]:

```
array([0, 0, 6, 5, 5, 0, 3, 0, 0, 0])
```

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

In [26]:

```
# add clustering labels
#neighborhoods_venues_sorted.drop('Cluster Labels', axis=1, inplace=True)
neighborhoods_venues_sorted.insert(0, 'Cluster Labels', kmeans.labels_)

toronto_merged = mydf_post

# inner join toronto_merged with neighborhood_venues_sorted to add latitude/longitude for e
toronto_labeled = pd.merge(toronto_merged, neighborhoods_venues_sorted, how='inner', on='Ne

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

Out[26]:

	Postcode	Borough	Neighbourhood	Latitude	Longitude	Cluster Labels	1st Most Common Venue	2nd Mos Commor Venue
0	M1B	Scarborough	Rouge, Malvern	43.806686	-79.194353	5	Fast Food Restaurant	Yoga Studic
1	M1C	Scarborough	Highland Creek, Rouge Hill, Port Union	43.784535	-79.160497	0	Bar	Yoga Studic
2	M1E	Scarborough	Guildwood, Morningside, West Hill	43.763573	-79.188711	0	Breakfast Spot	Rental Ca Locatior
3	M1G	Scarborough	Woburn	43.770992	-79.216917	0	Coffee Shop	Korear Restauran
4	M1H	Scarborough	Cedarbrae	43.773136	-79.239476	0	Hakka Restaurant	Tha Restauran

Clustering results

Finally, let's visualize the resulting clusters of similar neighbourhoods. Let's start by counting the number of neighbourhoods in each cluster.

In [27]:

```
toronto_labeled.groupby('Cluster Labels').count()['Neighbourhood']
```

Out[27]:

```
Cluster Labels
0    72
1     1
2     1
3     2
4     1
5    10
6    14
Name: Neighbourhood, dtype: int64
```

This is very interesting, as it appears that there are 3 main clusters accompanied by 4 outliers.

Mapping the clusters

Let's see how this looks on the map using Folium.

In [28]:

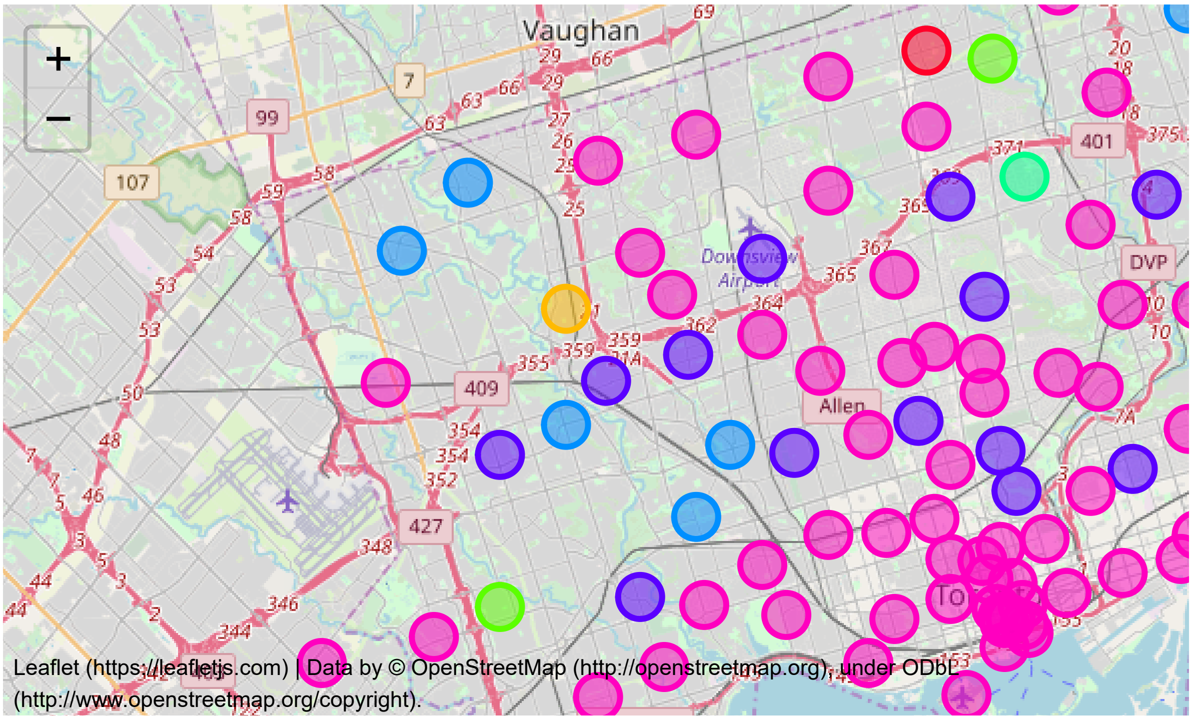
```python
# create map
map_clusters = folium.Map(location=[latitude, longitude], zoom_start=11)

# 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)))
colors_array = cm.gist_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_labeled['Latitude'], toronto_labeled['Longitude']
    label = folium.Popup(str(poi) + ' Cluster ' + str(cluster), parse_html=True)
    folium.CircleMarker(
        [lat, lon],
        radius=11,
        popup=label,
        color=rainbow[cluster-1],
        fill=True,
        fill_color=rainbow[cluster-1],
        fill_opacity=0.5).add_to(map_clusters)

map_clusters
```

Out[28]:



Analysis of the clusters

Now, we are ready to examine each cluster and determine the discriminating venue categories that distinguish each cluster. Based on the defining categories, we can assign a name to each cluster.

Cluster 1: Coffee to go

In [29]:

```
toronto_labeled.loc[toronto_labeled['Cluster Labels'] == 0, toronto_labeled.columns[[1] + l
```

Out[29]:

	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
64	Central Toronto	0	Sandwich Place	Café	Coffee Shop	Pizza Place	Gym	Pharmacy
57	Downtown Toronto	0	Coffee Shop	Café	Thai Restaurant	Bar	Steakhouse	Gym
46	Central Toronto	0	Pizza Place	Sandwich Place	Dessert Shop	Italian Restaurant	Restaurant	Thai Restaurant
12	Scarborough	0	Lounge	Breakfast Spot	Skating Rink	Chinese Restaurant	Sandwich Place	Eastern European Restaurant
9	Scarborough	0	College Stadium	Farm	Café	Skating Rink	General Entertainment	Yoga Studio

Cluster 2: Mixed bag

In [30]:

```
toronto_labeled.loc[toronto_labeled['Cluster Labels'] == 1, toronto_labeled.columns[[1] + l
```

Out[30]:

	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
20	North York	1	Park	Yoga Studio	Eastern European Restaurant	Dive Bar	Dog Run	Doner Restaurant	Donut Shop

Cluster 3: Sports

In [31]:

```
toronto_labeled.loc[toronto_labeled['Cluster Labels'] == 2, toronto_labeled.columns[[1] + l
```

Out[31]:

	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 Mos Commo Venu
95	North York	2	Baseball Field	Yoga Studio	Electronics Store	Doner Restaurant	Donut Shop	Drugstore	Dumplin Restaurai

Cluster 4: Banking

In [32]:

```
toronto_labeled.loc[toronto_labeled['Cluster Labels'] == 3, toronto_labeled.columns[[1] + l
```

Out[32]:

	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 M Comn Ver
18	North York	3	Café	Chinese Restaurant	Japanese Restaurant	Bank	Yoga Studio	Dog Run	Dc Sl
92	Etobicoke	3	Bank	Yoga Studio	Electronics Store	Doner Restaurant	Donut Shop	Drugstore	Dump Restaui

Cluster 5: Oddball

In [33]:

```
toronto_labeled.loc[toronto_labeled['Cluster Labels'] == 4, toronto_labeled.columns[[1] + l
```

Out[33]:

	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 Mos Commo Venu
19	North York	4	Cafeteria	Yoga Studio	Electronics Store	Doner Restaurant	Donut Shop	Drugstore	Dumplin Restaurai

Cluster 6: Food & Shopping

In [34]:

```
toronto_labeled.loc[toronto_labeled['Cluster Labels'] == 5, toronto_labeled.columns[[1] + l
```

Out[34]:

	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	
88	Etobicoke	5	Pizza Place	Gym	Pool	Skating Rink	Pharmacy	Pub	
15	Scarborough	5	Fast Food Restaurant	Chinese Restaurant	Coffee Shop	Nail Salon	Grocery Store	Pharmacy	F
97	Etobicoke	5	Pizza Place	Coffee Shop	Discount Store	Chinese Restaurant	Middle Eastern Restaurant	Sandwich Place	I
0	Scarborough	5	Fast Food Restaurant	Yoga Studio	Electronics Store	Dog Run	Doner Restaurant	Donut Shop	
34	East York	5	Fast Food Restaurant	Pizza Place	Gastropub	Bank	Intersection	Athletics & Sports	

Cluster 7: Rest & Relaxation

In [35]:

```
toronto_labeled.loc[toronto_labeled['Cluster Labels'] == 6, toronto_labeled.columns[[1] + l
```

Out[35]:

	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
63	Central Toronto	6	Park	Trail	Sushi Restaurant	Jewelry Store	Event Space	Ethiopian Restaurant
49	Downtown Toronto	6	Park	Playground	Building	Trail	Dumpling Restaurant	Discount Store
73	York	6	Park	Women's Store	Market	Fast Food Restaurant	Convenience Store	Dog Run
89	Etobicoke	6	Park	River	Yoga Studio	Dumpling Restaurant	Discount Store	Dive Bar
22	North York	6	Park	Convenience Store	Bank	Yoga Studio	Electronics Store	Doner Restaurant

Conclusion

As we already suspected, the map is dominated by the majority cluster 0, while the other two big clusters (5 & 6) only start appearing once you leave the center of Toronto city. As for the remaining 4 outlier clusters, they appear on the outskirts of the Toronto city area and only have one or two members in them.

Looking at the clusters more closely, there seems to be an abundance of fast food & beverage venues found in the biggest cluster. The 2nd and 3rd biggest clusters seem to address the needs of people in search of places to have lunch and/or dinner, or places to get outdoors and/or relax respectively. The rest of the clusters seem to be in neighbourhoods that cater to more specific needs, like electronics shopping or banking, which explains the low number of member neighbourhoods in them.

Overall, the visualization on the map accompanied by the inspection of the data reveals that there are mainly to types of areas in Toronto. The first is the downtown city life while the other appears to be more work related on the outskirts of the city. This is important information for people considering moving to or within Toronto.