Segmenting and Clustering Toronto Neighbourhoods

Objective: Segment and cluster the Toronto neighbourhoods based on post codes

Download the table of post codes for neighbourhoods in Toronto from Wikipedia. https://en.wikipedia.org/wiki/List_of_postal_codes_of_Canada:_M (https://en.wikipedia.org/wiki/List_of_postal_codes_of_Canada:_M)

I have Excel 2016, which can hold 1,048,576 rows. The post code table has only 287 rows. I tried using BeautifulSoup first. But when I was half-way through and dealing with all those issues of cleaning the data, I realized that for such a small table, directlying downloading it into Excel, then uploading it into Jupyter Notebook will be much easier and faster as well. So, I scraped all my BeautifulSoup codes.

In [1]:
```
import pandas as pd
postcode_df=pd.read_excel('Toronto Post Codes.xlsx')
postcode_df.head()
```

Out[1]:

	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

In [2]:
```
# size of the table
postcode_df.shape
```

Out[2]:
```
(287, 3)
```

In [3]:
```
# number of unique Boroughs
import numpy as np
postcode_df["Borough"].value_counts()
```

Out[3]:
```
Not assigned        77
Etobicoke           45
North York          38
Downtown Toronto    37
Scarborough         37
Central Toronto     17
West Toronto        13
York                 9
East Toronto         7
East York            6
Mississauga          1
Name: Borough, dtype: int64
```

Okay. So there are 77 "Not assigned" in the Borough column. I could use drop.duplicates() to drop those "Not assigned". However, that will drop the other boroughs as well, since there are boroughs that have more than one neighbourhoods assigned to it. So, drop by "Borough" won't work.

Let's check the Neighbourhood column. The neighbourhoods should be unique, except for the "Not assigned".

In [4]:
```
postcode_df["Neighbourhood"].value_counts()
```

Out[4]:
```
Not assigned               77
St. James Town              2
Runnymede                   2
Yorkville                   1
Glencairn                   1
                           ..
Exhibition Place            1
Morningside                 1
Parkdale                    1
Bathurst Manor              1
Scarborough Village West    1
Name: Neighbourhood, Length: 209, dtype: int64
```

All the neighbourhoods are unique, except for 1) the "Not assigned" and 2) "Runnymede" and "St. James Town" where there are two copies.

In [5]:
```
#Remove the "Not assigned" from the Borough column.
postcode_df.drop_duplicates(subset="Neighbourhood",keep=False, inplace=True)
postcode_df
```

Out[5]:

	Postcode	Borough	Neighbourhood
2	M3A	North York	Parkwoods
3	M4A	North York	Victoria Village
4	M5A	Downtown Toronto	Harbourfront
5	M6A	North York	Lawrence Heights
6	M6A	North York	Lawrence Manor
...	...	...	...
281	M8Z	Etobicoke	Kingsway Park South West
282	M8Z	Etobicoke	Mimico NW
283	M8Z	Etobicoke	The Queensway West
284	M8Z	Etobicoke	Royal York South West
285	M8Z	Etobicoke	South of Bloor

206 rows × 3 columns

All the "Not assigned" in the Neighbourhood column has been dropped. However, since the parameter "keep" was set to False. The process above dropped the two neighbourhoods "Runnemede" and "St. James Town" as well. But I would like to keep those two neighbourhoods, since each copy belong to two different boroughs.

Put the two neighbourhoods back.

In [6]:
```
append1=pd.DataFrame({"Postcode":["M5C","M6N","M6S","M4X"],"Borough":["Downtown Toronto","York","West Toronto","Downtown Tor
postcode_df.append(append1,ignore_index=False)
```

Out[6]:

	Postcode	Borough	Neighbourhood
2	M3A	North York	Parkwoods
3	M4A	North York	Victoria Village
4	M5A	Downtown Toronto	Harbourfront
5	M6A	North York	Lawrence Heights
6	M6A	North York	Lawrence Manor
...	...	...	...
285	M8Z	Etobicoke	South of Bloor
0	M5C	Downtown Toronto	St. James Town
1	M6N	York	Runnymede
2	M6S	West Toronto	Runnymede
3	M4X	Downtown Toronto	St. James Town

210 rows × 3 columns

In [12]:
```
postcode_df=postcode_df.append(append1,ignore_index=False)
```

In [13]:
```
print("The cleaned table has", len(postcode_df["Postcode"]), "row")
```

```
The cleaned table has 210 row
```

As shown above,the original table has 287 rows in total and 77 "Not assigned" in the Borough column. After the 77 "Not assigned" rows are dropped, the cleaned table should have 287-77=210 rows. The result of the above code block shows 210 rows.

Let's confirm that there is no more "Not assigned" in Borough column.

In [14]:
```
postcode_df["Borough"].value_counts()
```

Out[14]:
```
Etobicoke           45
North York          38
Downtown Toronto    37
Scarborough         37
Central Toronto     17
West Toronto        13
York                 9
East Toronto         7
East York            6
Mississauga          1
Name: Borough, dtype: int64
```

Cool. There is no more "Not assigned" in the Borough column.

The table is cleaned now. I can start working with it. The next thing I am going to do is to put the neighbourhoods that have the same postcode in the same row. Let's see how many unique postcodes are in the cleaned table.

In [16]:
```
postcode_df["Postcode"].value_counts()
```

Out[16]:
```
M8Y    8
M9V    8
M5V    7
M8Z    5
M9B    5
      ..
M4A    1
M4N    1
M5G    1
M5N    1
M6G    1
Name: Postcode, Length: 103, dtype: int64
```

There are 103 unique postcodes. Most of the postcodes has only 1 neighbourhoods associated with it. However, postcodeS "M8Y" and "M9V" have 8 neighbourhoods associated with it, "M5V" has 7, "M4V" and "M8Z" have 5. I need to concatenate all those neighbourhoods under the postcode they are associated with, in one row.

In [17]:
```
postcode_df=pd.DataFrame(postcode_df.groupby(["Postcode","Borough"])["Neighbourhood"].apply(lambda x: ','.join(x)))
postcode_df
```

Out[17]:

Postcode	Borough	Neighbourhood
M1B	Scarborough	Rouge,Malvern
M1C	Scarborough	Highland Creek,Rouge Hill,Port Union
M1E	Scarborough	Guildwood,Morningside,West Hill
M1G	Scarborough	Woburn
M1H	Scarborough	Cedarbrae
...	...	...
M9N	York	Weston
M9P	Etobicoke	Westmount
M9R	Etobicoke	Kingsview Village,Martin Grove Gardens,Richvie...
M9V	Etobicoke	Albion Gardens,Beaumond Heights,Humbergate,Jam...
M9W	Etobicoke	Northwest

103 rows × 1 columns

All the neighbourhoods are now concatenated under the postcode they are associated with. However, instead of regular index, the table has "Postcode" and "Borough" as its multi-column index. I am going to fix it below.

In [18]:
```
postcode_df.reset_index(inplace=True)
postcode_df
```

Out[18]:

	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
...	...	...	...
98	M9N	York	Weston
99	M9P	Etobicoke	Westmount
100	M9R	Etobicoke	Kingsview Village,Martin Grove Gardens,Richvie...
101	M9V	Etobicoke	Albion Gardens,Beaumond Heights,Humbergate,Jam...
102	M9W	Etobicoke	Northwest

103 rows × 3 columns

Get the size of the cleaned table.

In [19]:
```
print(postcode_df.shape)
```

```
(103, 3)
```

Next, find the latitude and longitude coordinates for each postcode.

In [20]:
```
# Load the "Geospatial_Coordinates.csv" dataset
Geo_df=pd.read_csv("Geospatial_Coordinates.csv")
Geo_df.head()
```

Out[20]:

	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

In [21]:
```
# Merge "postcode_df" and "Geo_df".
# Use "left" join so that all records from "postcode_df" and matched records from "Geo_df" are returned.
Geomerge_df=pd.merge(postcode_df,Geo_df, how="left")
Geomerge_df
```

Out[21]:

	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
...	...	...	...	...	...
98	M9N	York	Weston	43.706876	-79.518188
99	M9P	Etobicoke	Westmount	43.696319	-79.532242
100	M9R	Etobicoke	Kingsview Village,Martin Grove Gardens,Richvie...	43.688905	-79.554724
101	M9V	Etobicoke	Albion Gardens,Beaumond Heights,Humbergate,Jam...	43.739416	-79.588437
102	M9W	Etobicoke	Northwest	43.706748	-79.594054

103 rows × 5 columns

Cluster the neighbourhoods and create visualizations.

Import the necessary libraries first.

In [22]:
```
import pandas as pd
pd.set_option('display.max_columns', None)
pd.set_option('display.max_rows', None)

import json # library to handle JSON files

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

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

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

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

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

Get the latitude and longitude for Toronto City.

In [23]:
```
address = 'Toronto, Canada'

geolocator = Nominatim(user_agent="ny_explorer")
location = geolocator.geocode(address)
latitude = location.latitude
longitude = location.longitude
print('The geograpical coordinate of Toronto, Canada are {}, {}.'.format(latitude, longitude))
```

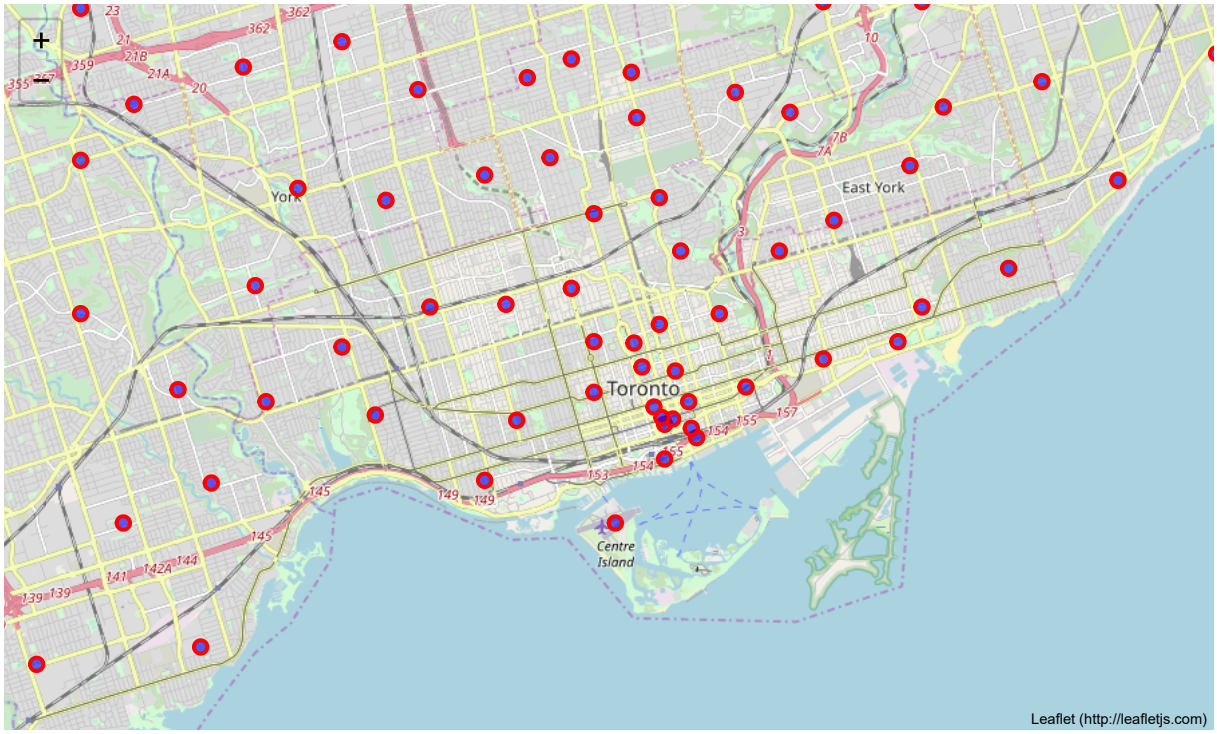
```
The geograpical coordinate of Toronto, Canada are 43.653963, -79.387207.
```

Create a map of Toronto City with the neighbourhoods superimposed on it.

In [24]:

```
# Toronto latitude and longitude.
latitude= 43.653963
longitude=-79.387207

# create map and display it
toronto_map = folium.Map(location=[latitude, longitude], zoom_start=12)

# instantiate a feature group for the incidents in the dataframe
incidents = folium.map.FeatureGroup()

# loop through the 100 crimes and add each to the incidents feature group
for lat, lng, in zip(Geomerge_df.Latitude, Geomerge_df.Longitude):
    incidents.add_child(
        folium.features.CircleMarker(
            [lat, lng],
            radius=5, # define how big you want the circle markers to be
            color='red',
            fill=True,
            fill_color='blue',
            fill_opacity=0.6
        )
    )

# add incidents to map
toronto_map.add_child(incidents)
```

Out[24]:



Simplify the above map and segment and cluster only the neighborhoods with "Toronto" in their borough names. These are the neighbourhoods associated with the following boroughs: Central Toronto, Downtown Toronto, East Toronto and West Toronta. So let's slice the original dataframe and create a new dataframe for the Toronto boroughs.

In [25]:

```
toronto_data1 = Geomerge_df[Geomerge_df['Borough'] == 'Central Toronto'].reset_index(drop=True)
toronto_data2 = Geomerge_df[Geomerge_df['Borough'] == 'Downtown Toronto'].reset_index(drop=True)
toronto_data3 = Geomerge_df[Geomerge_df['Borough'] == 'East Toronto'].reset_index(drop=True)
toronto_data4 = Geomerge_df[Geomerge_df['Borough'] == 'West Toronto'].reset_index(drop=True)
frames=[toronto_data1,toronto_data2,toronto_data3,toronto_data4]
toronto_data=pd.concat(frames)
toronto_data.head()
```

Out[25]:

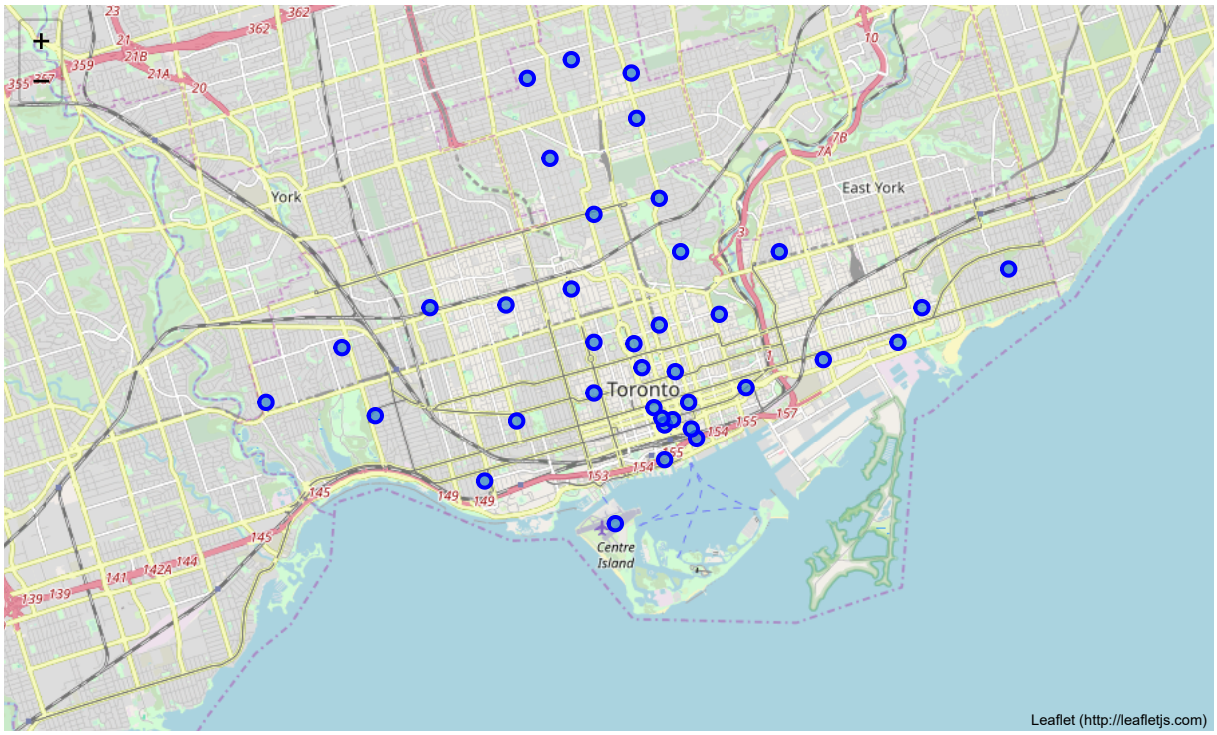
	Postcode	Borough	Neighbourhood	Latitude	Longitude
0	M4N	Central Toronto	Lawrence Park	43.728020	-79.388790
1	M4P	Central Toronto	Davisville North	43.712751	-79.390197
2	M4R	Central Toronto	North Toronto West	43.715383	-79.405678
3	M4S	Central Toronto	Davisville	43.704324	-79.388790
4	M4T	Central Toronto	Moore Park,Summerhill East	43.689574	-79.383160

Let's get the latitude and longitude for Downtown Toronto.

Visualize the Toronto boroughs with their neighbourhoods.

In [26]:
```
# create map of Manhattan using latitude and longitude values
toronto_map = folium.Map(location=[latitude, longitude], zoom_start=12)

# add markers to map
for lat, lng, label in zip(toronto_data['Latitude'], toronto_data['Longitude'], toronto_data['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(toronto_map)

toronto_map
```

Out[26]:



Let's cluster the neighbourhoods in Downtown Toronto.

In [27]:
```
# Onehot coding
toronto_data_onehot = pd.get_dummies(toronto_data[['Neighbourhood']], prefix="", prefix_sep="")

# add Borough column back to dataframe
toronto_data_onehot['Borough'] = toronto_data['Borough']

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

toronto_data_onehot.head(10)
```

Out[27]:

	Borough	Adelaide,King,Richmond	Berczy Park	Brockton,Exhibition Place,Parkdale Village	Business Reply Mail Processing Centre 969 Eastern	CN Tower,Bathurst Quay,Island airport,Harbourfront West,King and Spadina,Railway Lands,South Niagara	Cabbagetown,St. James Town	Central Bay Street	Chinatown,Grange Park,Kensington Market
0	Central Toronto	0	0	0	0	0	0	0	0
1	Central Toronto	0	0	0	0	0	0	0	0
2	Central Toronto	0	0	0	0	0	0	0	0
3	Central Toronto	0	0	0	0	0	0	0	0
4	Central Toronto	0	0	0	0	0	0	0	0
5	Central Toronto	0	0	0	0	0	0	0	0
6	Central Toronto	0	0	0	0	0	0	0	0
7	Central Toronto	0	0	0	0	0	0	0	0
8	Central Toronto	0	0	0	0	0	0	0	0
0	Downtown Toronto	0	0	0	0	0	0	0	0

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

In [28]:
```
toronto_grouped =toronto_data_onehot.groupby('Borough').mean().reset_index()
toronto_grouped
```

Out[28]:

	Borough	Adelaide,King,Richmond	Berczy Park	Brockton,Exhibition Place,Parkdale Village	Business Reply Mail Processing Centre 969 Eastern	CN Tower,Bathurst Quay,Island airport,Harbourfront West,King and Spadina,Railway Lands,South Niagara	Cabbagetown,St. James Town	Central Bay Street	Chinatown,Grang Park,Kensingtc Mark
0	Central Toronto	0.000000	0.000000	0.000000	0.0	0.000000	0.000000	0.000000	0.00000
1	Downtown Toronto	0.052632	0.052632	0.000000	0.0	0.052632	0.052632	0.052632	0.05263
2	East Toronto	0.000000	0.000000	0.000000	0.2	0.000000	0.000000	0.000000	0.00000
3	West Toronto	0.000000	0.000000	0.166667	0.0	0.000000	0.000000	0.000000	0.00000

Cluster the neighbourhoods in Central Toronto, Downtown Toronto, East Toronto and West Toronto.

In [29]:
```
from sklearn.cluster import KMeans
# set number of clusters
kclusters = 4

toronto_clustering = toronto_grouped.drop('Borough', 1)

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

# check cluster labels generated for each row in the dataframe
kmeans.labels_[0:3]
```

Out[29]:
```
array([0, 3, 1])
```

Visualize the clusters of the neighbourhoods in Central Toronto, Downtown Toronto, East Toronto and West Toronto on the map.

In [30]:
```
# Add cluster labels
toronto_grouped.insert(0,"Cluster label",kmeans.labels_)
```

In [31]:
```
toronto_data=toronto_data.join(toronto_grouped.set_index("Borough"),on="Borough")
toronto_data.head()
```

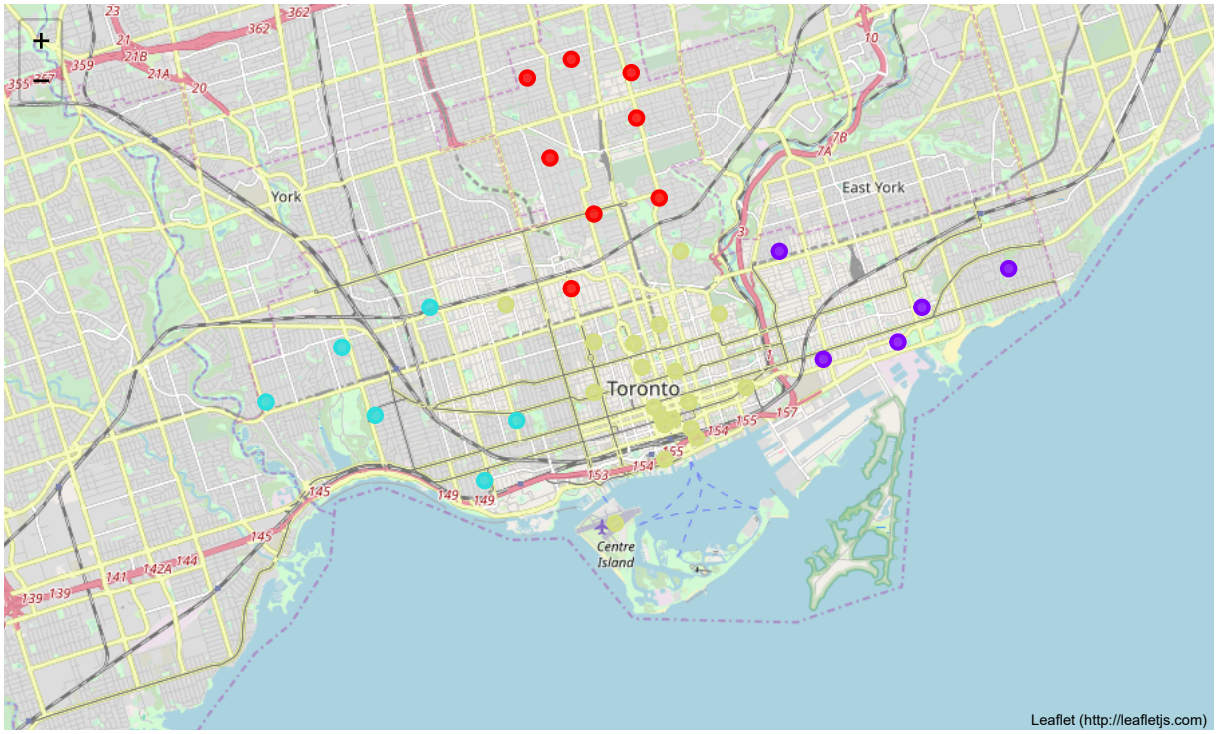
Out[31]:

	Postcode	Borough	Neighbourhood	Latitude	Longitude	Cluster label	Adelaide,King,Richmond	Berczy Park	Brockton,Exhibition Place,Parkdale Village	Business Reply Mail Processing Centre 969 Eastern	CN Tow (airport,H We Spad L
0	M4N	Central Toronto	Lawrence Park	43.728020	-79.388790	0	0.0	0.0	0.0	0.0	
1	M4P	Central Toronto	Davisville North	43.712751	-79.390197	0	0.0	0.0	0.0	0.0	
2	M4R	Central Toronto	North Toronto West	43.715383	-79.405678	0	0.0	0.0	0.0	0.0	
3	M4S	Central Toronto	Davisville	43.704324	-79.388790	0	0.0	0.0	0.0	0.0	
4	M4T	Central Toronto	Moore Park,Summerhill East	43.689574	-79.383160	0	0.0	0.0	0.0	0.0	

In [32]:

```
# create map
map_clusters = folium.Map(location=[latitude, longitude], zoom_start=11.5)

# 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_data['Latitude'], toronto_data['Longitude'], toronto_data['Borough'],toronto_data[
    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=0.8).add_to(map_clusters)

map_clusters
```

Out[32]:



There we have the map of Toronto City with the four boroughs with Toronton in their names (Central Toronto, Downtown Toronto, East Toronto and West Toronto) superimposed on it. Each color represents a different borough, while each dot represents a neighbbourhood. To see which borough and cluster a dot represents, hover the cursor over the dot and click on it.

Thanks for reading!

In [ ]: