Segmenting and Clustering Neighborhoods in Toronto

Part 1

Creating dataframe

Wiki table is saved into an excel file. Now we'll use pandas to import it to a dataframe

In [33]:

```
import pandas as pd
df=pd.read_excel('canada_table_wiki.xlsx')
print('Size= ', df.shape)
df.head()
```

```
Size=  (180, 3)
```

Out[33]:

	Postal code	Borough	Neighborhood
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	Regent Park, Harbourfront

Now, we'll cleanup the table. First, drop rows with 'Not assigned' values in Borough column

In [36]:

```
df.drop(df[df['Borough']=='Not assigned'].index,inplace=True)
```

In [46]:

```
df.reset_index(drop=True,inplace=True)
print('Size= ', df.shape)
df.head()
```

```
Size=  (103, 3)
```

Out[46]:

	Postal code	Borough	Neighborhood
0	M3A	North York	Parkwoods
1	M4A	North York	Victoria Village
2	M5A	Downtown Toronto	Regent Park, Harbourfront
3	M6A	North York	Lawrence Manor, Lawrence Heights
4	M7A	Downtown Toronto	Queen's Park, Ontario Provincial Government

Check whether there are cells with 'Not assigned' value in Neighborhoods column.

In [64]:

```
df[df['Neighborhood']=='Not assigned'].shape
```

Out[64]:

```
(0, 3)
```

Since there are none, data clean-up is complete. Lets check the size

In [66]:

```
df.shape
```

Out[66]:

```
(103, 3)
```

Part 2

Adding geolocation

Import the geospatial data

In [72]:

```
df2=pd.read_csv('Geospatial_Coordinates.csv')
df2.head()
```

Out[72]:

	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

Merge the two dataframes using the column: Postal code

In [84]:

```
df3=pd.merge(df,df2,on='Postal code')
print('Size = ', df3.shape)
df3.head()
```

```
Size =  (103, 5)
```

Out[84]:

	Postal code	Borough	Neighborhood	Latitude	Longitude
0	M3A	North York	Parkwoods	43.753259	-79.329656
1	M4A	North York	Victoria Village	43.725882	-79.315572
2	M5A	Downtown Toronto	Regent Park, Harbourfront	43.654260	-79.360636
3	M6A	North York	Lawrence Manor, Lawrence Heights	43.718518	-79.464763
4	M7A	Downtown Toronto	Queen's Park, Ontario Provincial Government	43.662301	-79.389494

Part 3

Clustering Neighborhoods

Import folium

In [91]:

```
import folium
```

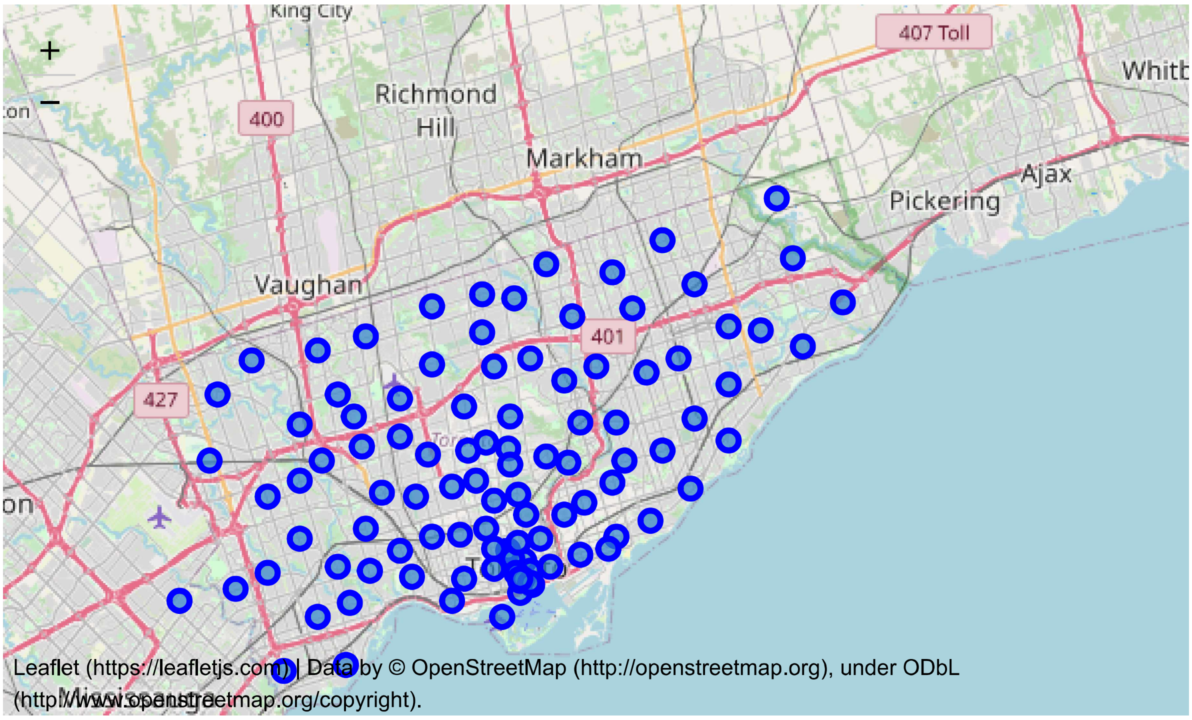
Create map and show neighborhoods on it

In [178]:

```
lat=df3.loc[0,'Latitude']
lon=df3.loc[0,'Longitude']
map_can = folium.Map(location=[lat, lon], zoom_start=10)
for lat, lng, borough, neighborhood in zip(df3['Latitude'],df3['Longitude'],df3['Borough'],
    label = '{}, {}'.format(neighborhood, borough)
    label = folium.Popup(label, parse_html=True)
    folium.CircleMarker(
        [lat, lng],
        radius=5,
        popup=label,
        color='blue',
        fill=True,
        fill_color='#3186cc',
        fill_opacity=0.7,
        parse_html=False).add_to(map_can)

map_can
```

Out[178]:



Leaflet (https://leafletjs.com) | Data by © OpenStreetMap (http://openstreetmap.org), under ODbL (http://www.openstreetmap.org/copyright).

Get Nearby venues from Foursquares

all neighborhoods will be used getting json data, cleaning up, getting frequencies of venues, etc are simillar to that done in lab

In [102]:

```
import json
import requests
```

In [101]:

```
CLIENT_ID = '0MNFD3SH3LTZ40QG5XQHM401WVOXZKTLQGBZFIHCF2ERNTCA' # your Foursquare ID
CLIENT_SECRET = 'VHCKL4GWYPJZBVL45NARX2I43ESIMI1ZPRADPSUR0TDN5BX0' # your Foursquare Secret
VERSION = '20180605' # Foursquare API version
LIMIT = 100 # A default Foursquare API limit value

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

```
Your credentails:
CLIENT_ID: 0MNFD3SH3LTZ40QG5XQHM401WVOXZKTLQGBZFIHCF2ERNTCA
CLIENT_SECRET:VHCKL4GWYPJZBVL45NARX2I43ESIMI1ZPRADPSUR0TDN5BX0
```

In [119]:

```
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 = ['Neighborhood',
                  'Neighborhood Latitude',
                  'Neighborhood Longitude',
                  'Venue',
                  'Venue Latitude',
                  'Venue Longitude',
                  'Venue Category']

    return(nearby_venues)
```

In [120]:

```
can_venues = getNearbyVenues(names=df3['Neighborhood'],
                                   latitudes=df3['Latitude'],
                                   longitudes=df3['Longitude']
                                  )
```

...

In [124]:

```
print('Size = ', can_venues.shape)
print('There are {} uniques categories.'.format(len(can_venues['Venue Category'].unique()))
can_venues.head()
```

```
Size =  (2136, 7)
There are 273 uniques categories.
```

Out[124]:

	Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
0	Parkwoods	43.753259	-79.329656	Brookbanks Park	43.751976	-79.332140	Park
1	Parkwoods	43.753259	-79.329656	Variety Store	43.751974	-79.333114	Food & Drink Shop
2	Victoria Village	43.725882	-79.315572	Victoria Village Arena	43.723481	-79.315635	Hockey Arena
3	Victoria Village	43.725882	-79.315572	Portugril	43.725819	-79.312785	Portuguese Restaurant
4	Victoria Village	43.725882	-79.315572	Tim Hortons	43.725517	-79.313103	Coffee Shop

In [128]:

```
# one hot encoding
can_onehot = pd.get_dummies(can_venues[['Venue Category']], prefix="", prefix_sep="")

# add neighborhood column back to dataframe
can_onehot['Neighborhood'] = can_venues['Neighborhood']

# move neighborhood column to the first column
fixed_columns = [can_onehot.columns[-1]] + list(can_onehot.columns[:-1])
can_onehot = can_onehot[fixed_columns]
print('Size = ', can_onehot.shape)
can_onehot.head()
```

```
Size =  (2136, 273)
```

Out[128]:

	Yoga Studio	Accessories Store	Afghan Restaurant	Airport	Airport Food Court	Airport Gate	Airport Lounge	Airport Service	Airport Terminal	Ame Restau
0	0	0	0	0	0	0	0	0	0	
1	0	0	0	0	0	0	0	0	0	
2	0	0	0	0	0	0	0	0	0	
3	0	0	0	0	0	0	0	0	0	
4	0	0	0	0	0	0	0	0	0	

5 rows × 273 columns

In [127]:

```
can_grouped = can_onehot.groupby('Neighborhood').mean().reset_index()
print('Size = ', can_grouped.shape)
can_grouped.head()
```

```
Size =  (96, 273)
```

Out[127]:

	Neighborhood	Yoga Studio	Accessories Store	Afghan Restaurant	Airport	Airport Food Court	Airport Gate	Airport Lounge	Airport Service
0	Agincourt	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1	Alderwood, Long Branch	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2	Bathurst Manor, Wilson Heights, Downsview North	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
3	Bayview Village	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
4	Bedford Park, Lawrence Manor East	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

5 rows × 273 columns

In [135]:

```
import numpy as np
```

In [137]:

```
def return_most_common_venues(row, num_top_venues):
    row_categories = row.iloc[1:]
    row_categories_sorted = row_categories.sort_values(ascending=False)

    return row_categories_sorted.index.values[0:num_top_venues]

num_top_venues = 10

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

# create columns according to number of top venues
columns = ['Neighborhood']
for ind in np.arange(num_top_venues):
    try:
        columns.append('{}{} Most Common Venue'.format(ind+1, indicators[ind]))
    except:
        columns.append('{}th Most Common Venue'.format(ind+1))

# create a new dataframe
venues_sorted = pd.DataFrame(columns=columns)
venues_sorted['Neighborhood'] = can_grouped['Neighborhood']

for ind in np.arange(can_grouped.shape[0]):
    venues_sorted.iloc[ind, 1:] = return_most_common_venues(can_grouped.iloc[ind, :], num_t

venues_sorted.head()
```

Out[137]:

	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	Agincourt	Lounge	Skating Rink	Latin American Restaurant	Breakfast Spot	Clothing Store	Drugstore	Discount Store	Distribution Center
1	Alderwood, Long Branch	Pizza Place	Pharmacy	Gym	Sandwich Place	Coffee Shop	Pub	Distribution Center	Dessert Shop
2	Bathurst Manor, Wilson Heights, Downsview North	Coffee Shop	Bank	Frozen Yogurt Shop	Shopping Mall	Bridal Shop	Sandwich Place	Diner	Restaurant
3	Bayview Village	Café	Japanese Restaurant	Chinese Restaurant	Bank	Women's Store	Discount Store	Distribution Center	Dog Run

K-means clustering

In [129]:

```
from sklearn.cluster import KMeans
```

In [138]:

```
can_cl = can_grouped.drop('Neighborhood', 1)

# run k-means clustering
kmeans = KMeans(n_clusters=5, random_state=0).fit(can_cl)
```

In [160]:

```
#venues_sorted.insert(0, 'Cluster Labels', kmeans.labels_)

df_merged = df3

# merge manhattan_grouped with manhattan_data to add latitude/longitude for each neighborhc
df_merged = df_merged.join(venues_sorted.set_index('Neighborhood'), on='Neighborhood')

print('size= ',df_merged.shape)
df_merged.head()
```

```
size=  (103, 16)
```

Out[160]:

	Postal code	Borough	Neighborhood	Latitude	Longitude	Cluster Labels	1st Most Common Venue	2nd Most Common Venue	3rd Mo Comm Ver
0	M3A	North York	Parkwoods	43.753259	-79.329656	4.0	Food & Drink Shop	Park	Drugst
1	M4A	North York	Victoria Village	43.725882	-79.315572	2.0	Pizza Place	Coffee Shop	Hocl Are
2	M5A	Downtown Toronto	Regent Park, Harbourfront	43.654260	-79.360636	1.0	Coffee Shop	Park	F
3	M6A	North York	Lawrence Manor, Lawrence Heights	43.718518	-79.464763	1.0	Clothing Store	Women's Store	Cof Sh
4	M7A	Downtown Toronto	Queen's Park, Ontario Provincial Government	43.662301	-79.389494	1.0	Coffee Shop	Yoga Studio	Ba

In [183]:

```
#drop NaN values
df_merged.dropna(axis=0,inplace=True)
df_merged.shape
```

Out[183]:

```
(100, 16)
```

In [147]:

```
import matplotlib.cm as cm
import matplotlib.colors as colors
```

In [182]:

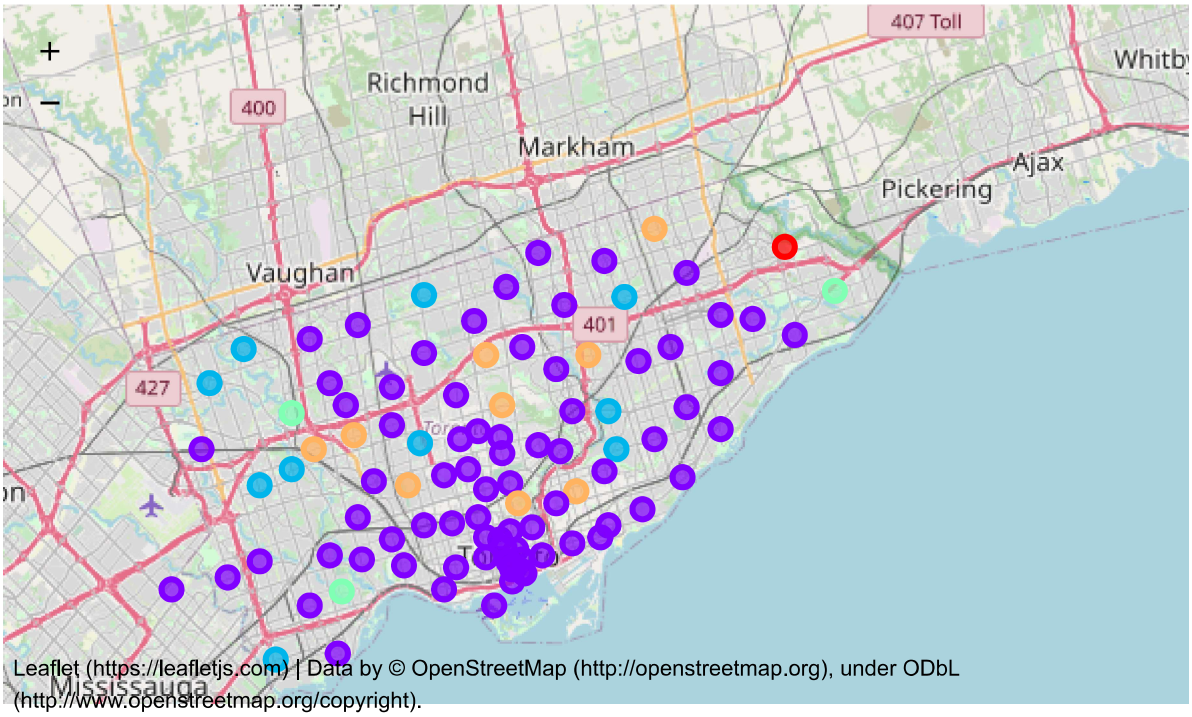
```
# create map
lat=df_merged.loc[0,'Latitude']
lon=df_merged.loc[0,'Longitude']
map_clusters = folium.Map(location=[lat, lon], zoom_start=10)

# set color scheme for the clusters
x = np.arange(5)
ys = [i + x + (i*x)**2 for i in range(5)]
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(df_merged['Latitude'], df_merged['Longitude'],df_merged['
    label = folium.Popup(str(poi) + ' Cluster ' + str(cluster), parse_html=True)
    folium.CircleMarker(
        [lat, lon],
        radius=5,
        popup=label,
        color=rainbow[int(cluster-1)],
        fill=True,
        fill_color=rainbow[int(cluster-1)],
        fill_opacity=0.7).add_to(map_clusters)

map_clusters
```

Out[182]:



Leaflet (https://leafletjs.com) | Data by © OpenStreetMap (http://openstreetmap.org), under ODbL (http://www.openstreetmap.org/copyright).

In [ ]: