The Battle of Neighborhoods

Table of contents

Introduction/Business Problem

Toronto, one of the famous places in world which is diverse and multicultural. I'm planning to move into Toronto but I'm not sure of the exact neighborhood which would be a best fit for me. I would like to explore how much they are similar or dissimilar neighborhoods are aspects from a tourist point of view regarding food, accommodation, beautiful places, and many more.

You should be able to choose, compare different neighborhoods in terms of a service, search for potential explanation of why a neighborhood is popular etc., . Hence the name of the capstone project will be the **Battle of the neighborhoods.**

Data section

In order to explore the similar or dissimilar in aspects of the neighborhoods, I would need **Foursquare location data** to fetch the Venue Category and Boroughs of Toronto.

We will segment it into different neighborhoods using the geographical coordinates of the center of each neighborhood, and then using a combination of location data and machine learning.

Building a recomendation system for finding best clusters of neighborhood based on certain criteria is valuable analytical problem that perfectly fits into Clustering type of Data Science problems which could be solved by unsupervised learning algorithms.

Import required libraries

In [1]:

```
import pandas as pd
import numpy as np
pd.set_option('display.max_columns', None)
pd.set_option('display.max_rows', None)
import requests
from bs4 import BeautifulSoup
import geocoder
import os

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

#!conda install -c conda-forge geopy --yes

from geopy.geocoders import Nominatim # convert an address into latitude and longitude
 values
# Matplotlib and associated plotting modules
import matplotlib.pyplot as plt
import matplotlib.cm as cm
import matplotlib.colors as colors
%matplotlib inline


print('Libraries imported.')
```

```
Libraries imported.
```

To get geo location of address

In [2]:

```
def geo_location(address):
    # get geo location of address
    geolocator = Nominatim(user_agent="ny_explorer")
    location = geolocator.geocode(address)
    latitude = location.latitude
    longitude = location.longitude
    return latitude,longitude

#address = 'Marunji, Pune'
#geo_location(address)
```

To fetch Postcode Borough Neighbourhood Latitude Longitude

In [3]:

```
page = requests.get("https://en.wikipedia.org/wiki/List_of_postal_codes_of_Canada:_M")
soup = BeautifulSoup(page.content, 'html.parser')
table = soup.find('tbody')
rows = table.select('tr')
row = [r.get_text() for r in rows]
```

Create a Data frame

In [4]:

```
df = pd.DataFrame(row)
df1 = df[0].str.split('\n', expand=True)
df2 = df1.rename(columns=df1.iloc[0])
df3 = df2.drop(df2.index[0])
df4 = df3[df3.Borough != 'Not assigned']
df5 = df4.groupby(['Postcode', 'Borough'], sort = False).agg(','.join)
df5.reset_index(inplace = True)
for index, row in df5.iterrows():
    if row["Neighbourhood"] == "Not assigned":
        row["Neighbourhood"] = row["Borough"]

coordinates = pd.read_csv("Geospatial_Coordinates.csv")
coordinates.rename(columns={"Postal Code": "Postcode"}, inplace=True)
df6 = df5.merge(coordinates, on="Postcode", how="left")

df6.head()
```

Out[4]:

	Postcode	Borough	Neighbourhood	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	Harbourfront,Regent Park	43.654260	-79.360636
3	M6A	North York	Lawrence Heights,Lawrence Manor	43.718518	-79.464763
4	M7A	Queen's Park	Queen's Park	43.662301	-79.389494

Use Foursquare API to fetch Borough Venue Venue Latitude Venue Longitude Venue Category for the given geo coordinates

In [23]:

```
CLIENT_ID = '1ZStttttSB3HWF1Z' # my Foursquare ID
CLIENT_SECRET = 'YPMMTBtttYCBJBSHOIJ' # my Foursquare Secret
VERSION = '20180605' # Foursquare API version
LIMIT = 100 # limit of number of venues returned by Foursquare API
radius = 2000 # define radius

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

```
Your credentails:
CLIENT_ID: 1ZStttttSB3HWF1Z
CLIENT_SECRET:YPMMTBtttYCBJBSHOIJ
```

Function to fetch venue categories

In [6]:

```
def getNearbyVenues(names, latitudes, longitudes, radius=500):

    venues_list=[]
    for name, lat, lng in zip(names, latitudes, longitudes):

        # create the API request URL
        url = 'https://api.foursquare.com/v2/venues/explore?&client_id={}&client_secret
={}&v={}&ll={},{}&radius={}&limit={}'.format(
            CLIENT_ID,
            CLIENT_SECRET,
            VERSION,
            lat,
            lng,
            radius,
            LIMIT)

        # make the GET request
        results = requests.get(url).json()["response"]['groups'][0]['items']

        # return only relevant information for each nearby venue
        venues_list.append([(
            name,
            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 = ['Borough',
                  'Venue',
                  'Venue Latitude',
                  'Venue Longitude',
                  'Venue Category' ]

    return(nearby_venues)
```

Create a dataframe

In [7]:

```
toronto_venues = getNearbyVenues(names=df6['Borough'],
                                 latitudes=df6['Latitude'],
                                 longitudes=df6['Longitude']
                                  )
toronto_venues.head(10)
```

Out[7]:

	Borough	Venue	Venue Latitude	Venue Longitude	Venue Category
0	North York	Brookbanks Park	43.751976	-79.332140	Park
1	North York	KFC	43.754387	-79.333021	Fast Food Restaurant
2	North York	Variety Store	43.751974	-79.333114	Food & Drink Shop
3	North York	Victoria Village Arena	43.723481	-79.315635	Hockey Arena
4	North York	Tim Hortons	43.725517	-79.313103	Coffee Shop
5	North York	Portugril	43.725819	-79.312785	Portuguese Restaurant
6	North York	The Frig	43.727051	-79.317418	French Restaurant
7	North York	Eglinton Ave E & Sloane Ave/Bermondsey Rd	43.726086	-79.313620	Intersection
8	Downtown Toronto	Roselle Desserts	43.653447	-79.362017	Bakery
9	Downtown Toronto	Tandem Coffee	43.653559	-79.361809	Coffee Shop

Analysis

Number of unique categories

In [8]:

```
print('The number of unique categories is {}.'.format(len(toronto_venues['Venue Categor
y'].unique())))
```

```
The number of unique categories is 276.
```

Grouping rows by district and by the mean of the frequency of occurrence of each category

In [9]:

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

# add neighborhood column back to dataframe
toronto_onehot['Borough'] = toronto_venues['Borough']

# move district column to the first column
cols=list(toronto_onehot.columns.values)
cols.pop(cols.index('Borough'))
toronto_onehot=toronto_onehot[['Borough']+cols]

# rename Neighborhood for Districts so that future merge works
#Barcelona_onehot.rename(columns = {'District': 'District'}, inplace = True)
toronto_wc = toronto_onehot.groupby('Borough').mean().reset_index()
toronto_wc

toronto_wc.head(15)
```

Out[9]:

	Borough	Accessories Store	Afghan Restaurant	Airport	Airport Food Court	Airport Gate	Airport Lounge	Airport Service
0	Central Toronto	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
1	Downtown Toronto	0.000000	0.000767	0.000767	0.000767	0.000767	0.001534	0.001534
2	East Toronto	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
3	East York	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
4	Etobicoke	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
5	Mississauga	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
6	North York	0.004098	0.000000	0.004098	0.000000	0.000000	0.000000	0.000000
7	Queen's Park	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
8	Scarborough	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
9	West Toronto	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
10	York	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000

In [10]:

```
toronto_wc.shape
```

Out[10]:

(11, 277)

Printing Borough along with the top 5 most common venues

In [11]:

```
num_top_venues = 5

print('Example')
for hood in toronto_wc['Borough'][:5]:
    print("----"+hood+"----")
    temp = toronto_wc[toronto_wc['Borough'] == hood].T.reset_index()
    temp.columns = ['venue','freq']
    temp = temp.iloc[1:]
    temp['freq'] = temp['freq'].astype(float)
    temp = temp.round({'freq': 2})
    print(temp.sort_values('freq', ascending=False).reset_index(drop=True).head(num_top
_venues))
    print('\n')
```

```
Example
----Central Toronto----
              venue  freq
0       Coffee Shop  0.07
1       Pizza Place  0.06
2    Sandwich Place  0.06
3              Park  0.05
4  Sushi Restaurant  0.04


----Downtown Toronto----
                venue  freq
0         Coffee Shop  0.10
1                Café  0.05
2          Restaurant  0.03
3  Italian Restaurant  0.03
4               Hotel  0.03


----East Toronto----
                venue  freq
0         Coffee Shop  0.07
1    Greek Restaurant  0.07
2  Italian Restaurant  0.05
3                Café  0.04
4      Ice Cream Shop  0.04


----East York----
                 venue  freq
0          Coffee Shop  0.06
1         Burger Joint  0.05
2                 Park  0.05
3  Sporting Goods Shop  0.04
4                 Bank  0.04


----Etobicoke----
            venue  freq
0     Pizza Place  0.12
1  Sandwich Place  0.07
2        Pharmacy  0.05
3     Coffee Shop  0.05
4  Discount Store  0.04
```

In [12]:

```
def get_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]
```

In [13]:

```
num_top_venues = 10

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

# create columns according to number of top venues
columns = ['Borough']
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
Borough_venues_sorted = pd.DataFrame(columns=columns)
Borough_venues_sorted['Borough'] = toronto_wc['Borough']

for ind in np.arange(toronto_wc.shape[0]):
    Borough_venues_sorted.iloc[ind, 1:] = get_most_common_venues(toronto_wc.iloc[ind,
:], num_top_venues)

Borough_venues_sorted.head()
```

Out[13]:

	Borough	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
0	Central Toronto	Coffee Shop	Sandwich Place	Pizza Place	Park	Café	Sushi Restaurant	Gym
1	Downtown Toronto	Coffee Shop	Café	Restaurant	Hotel	Italian Restaurant	Bakery	Bar
2	East Toronto	Coffee Shop	Greek Restaurant	Italian Restaurant	Café	Ice Cream Shop	Brewery	Yoga Studio
3	East York	Coffee Shop	Burger Joint	Park	Sandwich Place	Bank	Pharmacy	Pizza Place
4	Etobicoke	Pizza Place	Sandwich Place	Pharmacy	Coffee Shop	Discount Store	Fast Food Restaurant	Grocery Store

Machine Learning - KMeans Clustering

A Clustering Algorithm tries to analyse natural groups of data on the basis of some similarity. It locates the centroid of the group of data points. To carry out effective clustering, the algorithm evaluates the distance between each point from the centroid of the cluster.

K-means Clustering will group these locations of maximum prone areas into clusters and define a cluster center for each clusters. These Clusters centers are the centroids of each cluster and are at a minimum distance from all the points of a particular cluster.

Clustering Borough

In [14]:

```
from sklearn.cluster import KMeans

kclusters = 5

toronto_grouped_clustering = toronto_wc.drop('Borough', 1)

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

# check cluster labels generated for each row in the dataframe
kmeans.labels_[1:10]
```

Out[14]:

```
array([0, 0, 4, 4, 1, 0, 3, 4, 0])
```

merge toronto_wc with df6 to add latitude/longitude for each neighborhood

In [15]:

```
# add clustering labels
Borough_venues_sorted['Cluster Labels'] = kmeans.labels_

Borough_venues_sorted.head(5)

#toronto_merged = toronto_venues
toronto_merged = df6
toronto_merged = toronto_merged.join(Borough_venues_sorted.set_index('Borough'), on='Bo
rough')
toronto_merged.head(10)
```

Out[15]:

	Postcode	Borough	Neighbourhood	Latitude	Longitude	1st Most Common Venue	2nd Most Common Venue	(
0	M3A	North York	Parkwoods	43.753259	-79.329656	Coffee Shop	Clothing Store	F R
1	M4A	North York	Victoria Village	43.725882	-79.315572	Coffee Shop	Clothing Store	F R
2	M5A	Downtown Toronto	Harbourfront,Regent Park	43.654260	-79.360636	Coffee Shop	Café	R
3	M6A	North York	Lawrence Heights,Lawrence Manor	43.718518	-79.464763	Coffee Shop	Clothing Store	F R
4	M7A	Queen's Park	Queen's Park	43.662301	-79.389494	Coffee Shop	Park	
5	M9A	Etobicoke	Islington Avenue	43.667856	-79.532242	Pizza Place	Sandwich Place	F
6	M1B	Scarborough	Rouge,Malvern	43.806686	-79.194353	Breakfast Spot	Fast Food Restaurant	R
7	M3B	North York	Don Mills North	43.745906	-79.352188	Coffee Shop	Clothing Store	F R
8	M4B	East York	Woodbine Gardens,Parkview Hill	43.706397	-79.309937	Coffee Shop	Burger Joint	
9	M5B	Downtown Toronto	Ryerson,Garden District	43.657162	-79.378937	Coffee Shop	Café	R

Visualization

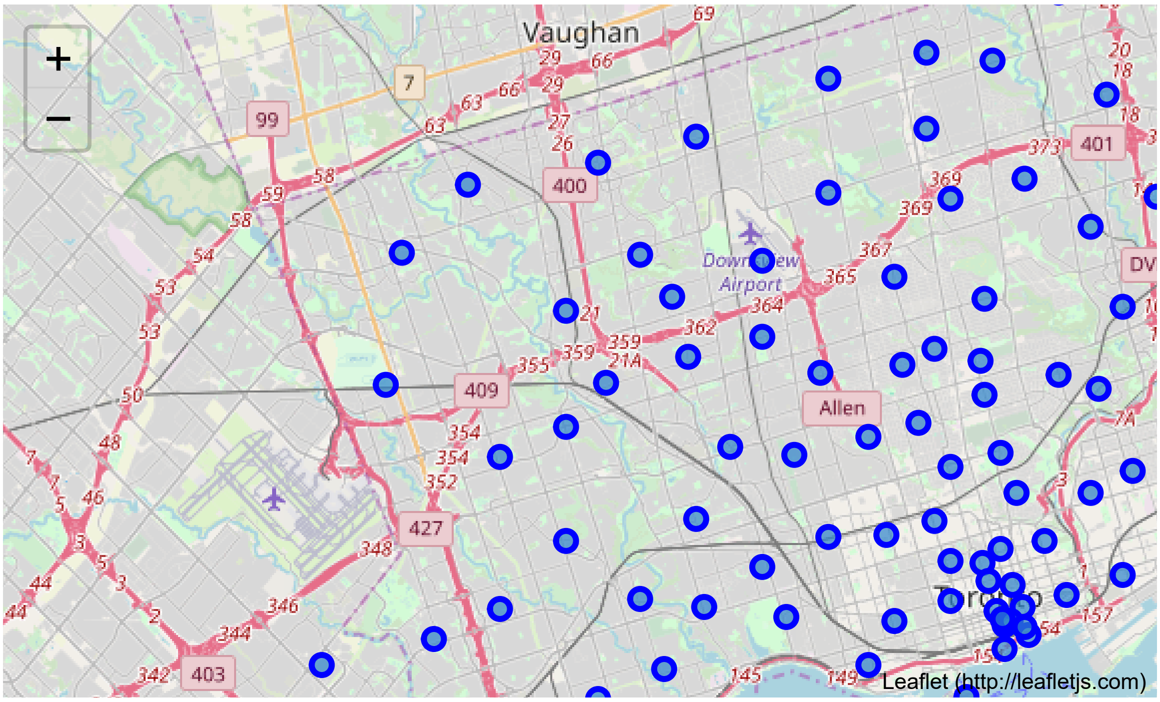
Visualization of Toronto's Borough

Screenshot : https://github.com/akshayca/personal-portfolio/blob/master/Machine%20Learning%20Projects/Clustering/Capstone%20Project%20-%20The%20Battle%20of%20Neighborhoods/toronto_map.PNG (https://github.com/akshayca/personal-portfolio/blob/master/Machine%20Learning%20Projects/Clustering/Capstone%20Project%20-%20The%20Battle%20of%20Neighborhoods/toronto_map.PNG)

In [16]:

```
# create map of Toronto using latitude and longitude values above:
ll= geo_location('Toronto')
toronto_map = folium.Map(location=[ll[0], ll[1]], zoom_start=11)

# add markers to map
for lat, lng, label in zip(toronto_merged['Latitude'], toronto_merged['Longitude'], tor
onto_merged['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[16]:



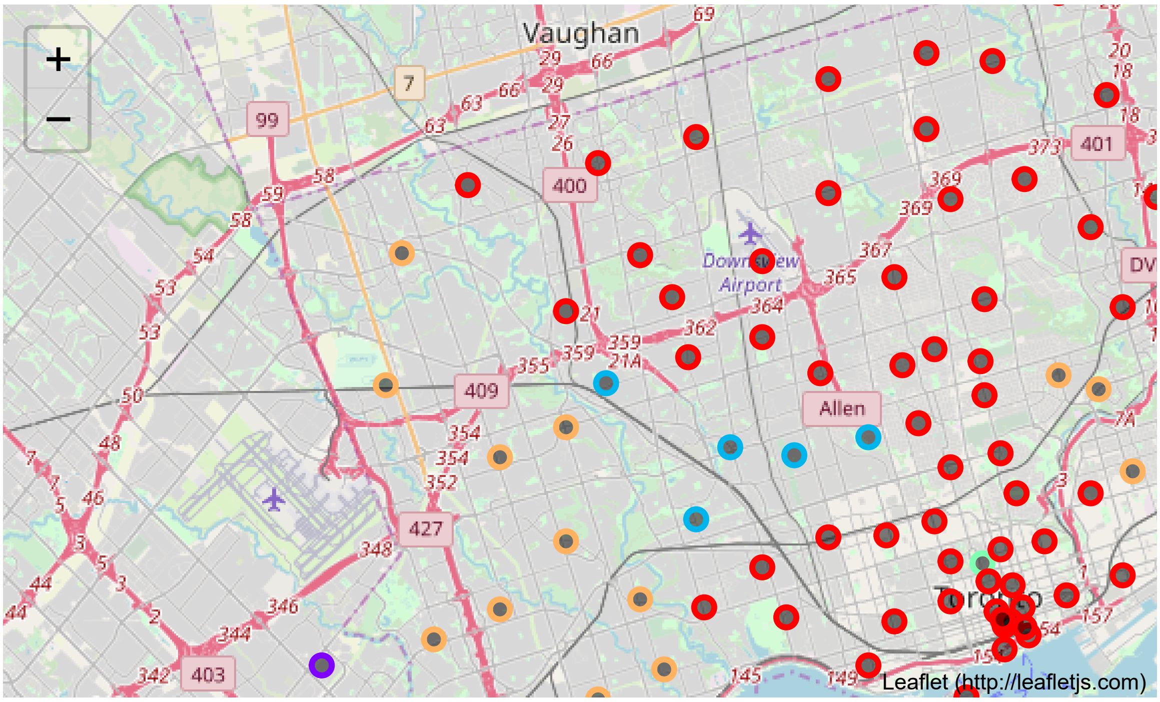
Clusters visualization of Toronto's Borough

Screenshot: [https://github.com/akshayca/personal-portfolio/blob/master/Machine%20Learning%20Projects/Clustering/Capstone%20Project%20-%20The%20Battle%20of%20Neighborhoods/clusters_toronto_map.PNG (https://github.com/akshayca/personal-portfolio/blob/master/Machine%20Learning%20Projects/Clustering/Capstone%20Project%20-%20The%20Battle%20of%20Neighborhoods/clusters_toronto_map.PNG)](https://github.com/akshayca/personal-portfolio/blob/master/Machine%20Learning%20Projects/Clustering/Capstone%20Project%20-%20The%20Battle%20of%20Neighborhoods/clusters_toronto_map.PNG)

In [17]:

```
map_clusters = folium.Map(location=[ll[0], ll[1]], zoom_start=11)

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

# add markers to the map
markers_colors = []
for lat, lon, poi, cluster in zip(toronto_merged['Latitude'], toronto_merged['Longitud
e'], toronto_merged['Borough'], toronto_merged['Cluster Labels']):

    label = '{}, cluster {}'.format(poi, cluster)
    label = folium.Popup(label, parse_html=True)
    folium.CircleMarker(
        [lat, lon],
        radius=5,
        popup=label,
        color=rainbow[cluster-1],
        fill=True,
        fill_color='black',
        fill_opacity=0.5).add_to(map_clusters)

map_clusters
```

Out[17]:



Results

Now let's try to fetch insights from the data.

The following are the highlights of the 5 clusters above:

Cluster #0

Most common venues: Restaurants and Coffee Shop

In [18]:

```
Borough_venues_sorted.loc[Borough_venues_sorted['Cluster Labels'] == 0,
                          Borough_venues_sorted.columns[[0] + list(range(1, Borough_ven
ues_sorted.shape[1]))]]
```

Out[18]:

	Borough	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	
0	Central Toronto	Coffee Shop	Sandwich Place	Pizza Place	Park	Café	Sushi Restaurant	Gym	
1	Downtown Toronto	Coffee Shop	Café	Restaurant	Hotel	Italian Restaurant	Bakery	Bar	F
2	East Toronto	Coffee Shop	Greek Restaurant	Italian Restaurant	Café	Ice Cream Shop	Brewery	Yoga Studio	F
6	North York	Coffee Shop	Clothing Store	Fast Food Restaurant	Japanese Restaurant	Restaurant	Park	Grocery Store	
9	West Toronto	Bar	Café	Coffee Shop	Bakery	Italian Restaurant	Restaurant	Breakfast Spot	

Cluster #1

Most common venues: Hotels and Gym/Fitness center

In [19]:

```
Borough_venues_sorted.loc[Borough_venues_sorted['Cluster Labels'] == 1,
                          Borough_venues_sorted.columns[[0] + list(range(1, Borough_ven
ues_sorted.shape[1]))]]
```

Out[19]:

	Borough	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
5	Mississauga	Hotel	Coffee Shop	Gym / Fitness Center	Mediterranean Restaurant	Fried Chicken Joint	Middle Eastern Restaurant	Sandwich Place

Cluster #2

Most common venues: Park, Convenience Store and Check Cashing Service

In [20]:

```
Borough_venues_sorted.loc[Borough_venues_sorted['Cluster Labels'] == 2,
                          Borough_venues_sorted.columns[[0] + list(range(1, Borough_ven
ues_sorted.shape[1]))]]
```

Out[20]:

	Borough	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	8 C
10	York	Park	Convenience Store	Check Cashing Service	Trail	Restaurant	Caribbean Restaurant	Bus Line	S

Cluster #3

Most common venues: Park and Gym

In [21]:

```
Borough_venues_sorted.loc[Borough_venues_sorted['Cluster Labels'] == 3,
                          Borough_venues_sorted.columns[[0] + list(range(1, Borough_ven
ues_sorted.shape[1]))]]
```

Out[21]:

	Borough	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 Mc Comm Ven
7	Queen's Park	Coffee Shop	Park	Gym	Diner	Seafood Restaurant	Sandwich Place	Salad Place	Burg Jo

Cluster #4

Most common venues: Fast Food Restaurants

In [22]:

```
Borough_venues_sorted.loc[Borough_venues_sorted['Cluster Labels'] == 4,
                          Borough_venues_sorted.columns[[0] + list(range(1, Borough_ven
ues_sorted.shape[1]))]]
```

Out[22]:

	Borough	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
3	East York	Coffee Shop	Burger Joint	Park	Sandwich Place	Bank	Pharmacy	Pizza Place
4	Etobicoke	Pizza Place	Sandwich Place	Pharmacy	Coffee Shop	Discount Store	Fast Food Restaurant	Grocery Store
8	Scarborough	Breakfast Spot	Fast Food Restaurant	Chinese Restaurant	Pizza Place	Coffee Shop	Bakery	Indian Restaurant

Conclusion:

My personal preference would be a home around Fast Food Restaurants so Cluster #4 Neighborhoods - East York, Etobicoke and Scarborough would be best for me :)

In conclusion, this project would have had better results if there were more available data in terms of actual land pricing data within the area, public transportation access and allowance of more venues exploration with the Foursquare (limited venues for free calls).

In [ ]:

```

```