Capstone Project - The Battle of the Neighborhoods

Applied Data Science Capstone by IBM/Coursera

Table of contents

Introduction

Background

Toronto is Canada's largest city, the fourth largest in North America, and home to a diverse population of about 2.8 million people. It is a global centre for business, finance, arts and culture and is consistently ranked one of the world's most livable cities.

Problem

When you are looking to open a restaurant in a popular city as Tonronto city, how to build a successful restaurant. Of course, food and service are important to the success of a restaurant, but the location can be just as crucial. Therefore, target audience of this project will be people who are looking to open a new restaurant. This project will segment the neighborhoods of Toronto into major clusters and examine their food. This quantifiable analysis can be used to understand the distribution of different cultures and food over Canada's largest city. Also, it can be utilized by a new **food vendor** who want to open his or her restaurant or by a **government authority** to examine and study their city’s culture diversity better.

Data

Toronto City Dataset

Data will be scraped from https://en.wikipedia.org/wiki/List_of_postal_codes_of_Canada:_M (https://en.wikipedia.org/wiki/List_of_postal_codes_of_Canada:_M). After Toronto City data is scraped, data will be preprocessed. Data is consist of **Post Code**, **Borough**, and **Neighborhood**.

In [1]:

```
from bs4 import BeautifulSoup
from pattern.web import download
import pandas as pd
import numpy as np
```

In [2]:

```
html_doc = download('https://en.wikipedia.org/wiki/List_of_postal_codes_of_Canada:_M')
soup = BeautifulSoup(html_doc,'lxml')
wiki_table = soup.find('table',class_='wikitable').find_all('tr')
toronto_data = []
for index, row in enumerate(wiki_table):
    if index == 0:
        pass
    else:
        data = row.find_all('td')
        postcode = data[0].text
        borough = data[1].text
        neighborhood = data[2].text.strip()
        toronto_data.append([postcode,borough,neighborhood])
toronto_df = pd.DataFrame(toronto_data, columns=['PostalCode','Borough','Neighborhood'
])
toronto_df.head()
```

Out[2]:

	PostalCode	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	Harbourfront

Only process the cells that have an assigned borough. Ignore cells with a borough that is Not assigned.

In [3]:

```
toronto_df = toronto_df[toronto_df['Borough'] != 'Not assigned']
toronto_df.head()
```

Out[3]:

	PostalCode	Borough	Neighborhood
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

More than one neighborhood can exist in one postal code area.

In [4]:

```
toronto_df['Neighborhood'] = toronto_df.groupby(['PostalCode','Borough']).transform(lam
bda x: ', '.join(x))
toronto_df = toronto_df.drop_duplicates().reset_index(drop=True)
toronto_df.head()
```

Out[4]:

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

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.

In [5]:

```
toronto_df['Neighborhood'] = toronto_df.apply(lambda row: row['Borough'] if row['Neighb
orhood'] == 'Not assigned' else row['Neighborhood'], axis=1)
toronto_df.head()
```

Out[5]:

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

In [6]:

```
toronto_df.shape
```

Out[6]:

```
(103, 3)
```

Geographical Coordinates

Toronto City data will be mapped with the geographical coordinates of each postal code of Toronto City. Geographical Coordinates data is consist of **Post Code**, **Latitude**, and **Longitude**. Link: [http://cocl.us/Geospatial_data (http://cocl.us/Geospatial_data)](http://cocl.us/Geospatial_data)

In [7]:

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

Out[7]:

	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

In [8]:

```
toronto_df = pd.merge(toronto_df, geographical_coordinates_df, left_on='PostalCode', ri
ght_on='Postal Code')
toronto_df = toronto_df[['PostalCode', 'Borough', 'Neighborhood', 'Latitude', 'Longitud
e']]
toronto_df.head()
```

Out[8]:

	PostalCode	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	Harbourfront	43.654260	-79.360636
3	M6A	North York	Lawrence Heights, Lawrence Manor	43.718518	-79.464763
4	M7A	Downtown Toronto	Queen's Park	43.662301	-79.389494

In [9]:

```
toronto_df.shape
```

Out[9]:

```
(103, 5)
```

Foursquare API

Foursquare API, a location data provider, will be used to find the venues on each postal code zone using a radius based on the area cover by each neighborhoods. Data from Foursquare API is consist of **Venue Name**, **Venue Latitude**, **Venue Longitude**, and **Venue Category**.

In [10]:

```
import requests
```

Define Foursquare Credentials and Version

In [11]:

```
CLIENT_ID = 'LJD5RVRNHFATKRW32JNUHPRAMOEL02ZTBRA1VFMNCT4DU55Y' # your Foursquare ID
CLIENT_SECRET = 'MJ1XJZU0ZWGUJJHUUY3LRGDM13SWTXQZJR2GO1RWL5NKCS45' # your Foursquare Se
cret
VERSION = '20180605' # Foursquare API version
LIMIT = 100 # limit of number of venues returned by Foursquare API
print('Your credentails:')
print('CLIENT_ID: ' + CLIENT_ID)
print('CLIENT_SECRET:' + CLIENT_SECRET)
```

```
Your credentails:
CLIENT_ID: LJD5RVRNHFATKRW32JNUHPRAMOEL02ZTBRA1VFMNCT4DU55Y
CLIENT_SECRET:MJ1XJZU0ZWGUJJHUUY3LRGDM13SWTXQZJR2GO1RWL5NKCS45
```

In [12]:

```
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
={}&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_
list])
    nearby_venues.columns = ['Neighborhood',
                  'Neighborhood Latitude',
                  'Neighborhood Longitude',
                  'Venue',
                  'Venue Latitude',
                  'Venue Longitude',
                  'Venue Category']
    return(nearby_venues)
```

In [13]:

```
toronto_venues_df = getNearbyVenues(names=toronto_df['Neighborhood'], latitudes=toronto
_df['Latitude'],
                                    longitudes=toronto_df['Longitude'])
toronto_venues_df.head()
```

Out[13]:

	Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
0	Parkwoods	43.753259	-79.329656	Brookbanks Park	43.751976	-79.332140	Par
1	Parkwoods	43.753259	-79.329656	Variety Store	43.751974	-79.333114	Food & Drink Sho
2	Victoria Village	43.725882	-79.315572	Victoria Village Arena	43.723481	-79.315635	Hocke Aren
3	Victoria Village	43.725882	-79.315572	Tim Hortons	43.725517	-79.313103	Coffe Sho
4	Victoria Village	43.725882	-79.315572	Portugril	43.725819	-79.312785	Portugues Restaurar

In [14]:

```
toronto_venues_df.to_csv('toronto_venues.csv')
```

Methodology

Exploratory Data Analysis

In [15]:

```
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

Use geopy library to get the latitude and longitude values of Toronto City.

In [16]:

```
from geopy.geocoders import Nominatim # convert an address into latitude and longitude
 values

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

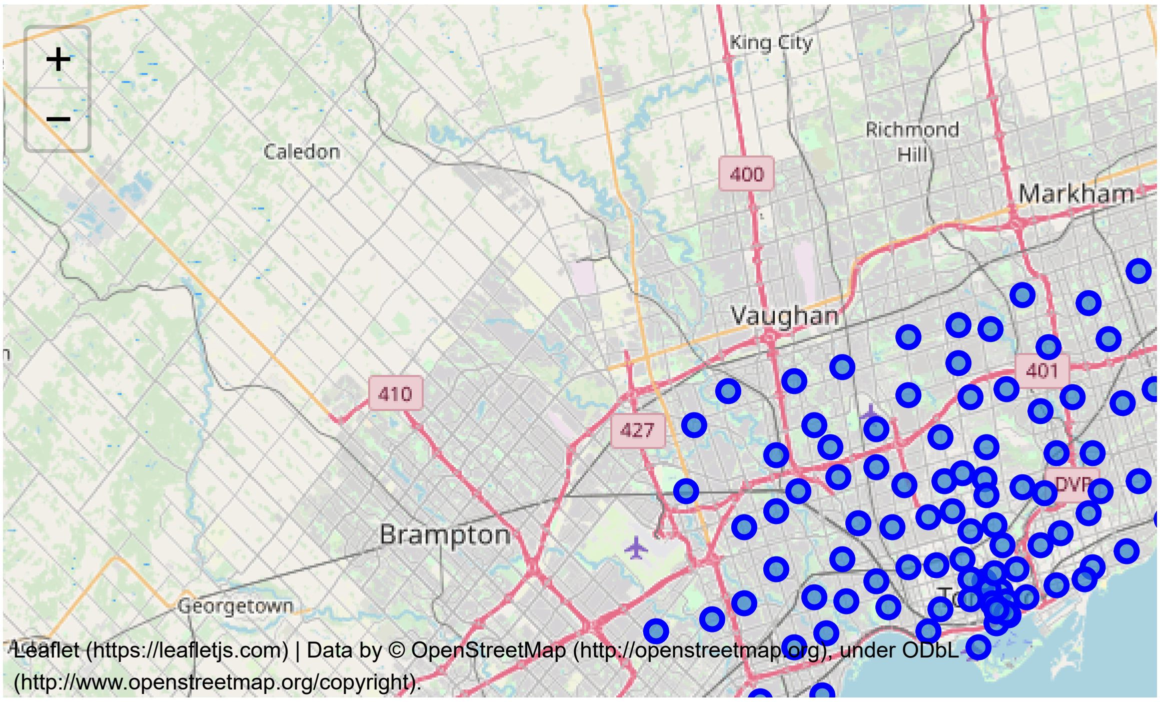
```
The geograpical coordinate of Toronto are 43.653963, -79.387207.
```

Create a map of Toronto with neighborhoods superimposed on top.

In [17]:

```
import folium
# create map of Toronto using latitude and longitude values
map_toronto = folium.Map(location=[latitude, longitude], zoom_start=10)
# add markers to map
for lat, lng, borough, neighborhood in zip(toronto_df['Latitude'], toronto_df['Longitud
e'], toronto_df['Borough'], toronto_df['Neighborhood']):
    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_toronto)
map_toronto
```

Out[17]:



How many venues in Toronto city?

In [18]:

```
toronto_venues_df.shape
```

Out[18]:

(2209, 7)

It is found that there is a total of 2209 venues. How many unique categories?

In [19]:

```
unique_category = toronto_venues_df['Venue Category'].unique()
print('There are '+str(len(unique_category))+' unique categories')
```

```
There are 273 unique categories
```

Data Cleaning

Because of our objective is to understand the distribution of different cultures and food, so we have to remove all the venues which is generalized categories.

In [20]:

```
general_category = []
food_category = []
for cat in unique_category:
    if 'Restaurant' in cat and cat != 'Restaurant':
        food_category.append(cat)
    else:
        general_category.append(cat)
print('There are '+str(len(food_category))+' food categories.')
print('There are '+str(len(general_category))+' general categories.')
```

```
There are 49 food categories.
There are 224 general categories.
```

There are some category about food in general categories. Then, manually select and add them to food categories.

In [21]:

```
food_category += ['Food & Drink Shop', 'Coffee Shop', 'Bakery', 'Breakfast Spot', 'Choc
olate Shop', 'Dessert Shop', 'Café',
                  'Ice Cream Shop', 'Beer Store', 'Health Food Store', 'Beer Bar', 'Bur
ger Joint', 'Fried Chicken Joint',
                  'Nightclub', 'Smoothie Shop', 'Sandwich Place', 'Bar', 'Gastropub',
'Pizza Place', 'Tea Room', 'Taco Place',
                  'Steakhouse', 'Juice Bar', 'Bubble Tea Shop', 'Wine Bar', 'BBQ Joint'
, 'Food Truck', 'Cocktail Bar',
                  'Jazz Club', 'Cheese Shop', 'Fish & Chips Shop', 'Gourmet Shop', 'Iri
sh Pub', 'Salad Place', 'Donut Shop',
                  'Candy Store', 'Frozen Yogurt Shop', 'Food Court', 'Noodle House', 'C
upcake Shop', 'Fruit & Vegetable Store',
                  'Gay Bar', 'Wine Shop']
print('There are '+str(len(food_category))+' food categories.')
```

```
There are 92 food categories.
```

Remove all the venues which is generalized categories.

In [22]:

```
toronto_food_venues_df = toronto_venues_df[toronto_venues_df['Venue Category'].isin(foo
d_category)]
toronto_food_venues_df.head()
```

Out[22]:

	Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
1	Parkwoods	43.753259	-79.329656	Variety Store	43.751974	-79.333114	Food & Drink Shop
3	Victoria Village	43.725882	-79.315572	Tim Hortons	43.725517	-79.313103	Coffee Shop
4	Victoria Village	43.725882	-79.315572	Portugril	43.725819	-79.312785	Portuguese Restaurant
5	Victoria Village	43.725882	-79.315572	The Frig	43.727051	-79.317418	French Restaurant
7	Harbourfront	43.654260	-79.360636	Roselle Desserts	43.653447	-79.362017	Bakery

How many food venues in Toronto city?

In [23]:

```
toronto_food_venues_df.shape
```

Out[23]:

```
(1238, 7)
```

It is found that there is a total of 1238 food venues.

Feature Engineering

First of all, using one hot encoding to convert categorical variables which are venue categories into a form that could be provided to ML algorithms to do a better job in prediction.

In [24]:

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

# add neighborhood column back to dataframe
toronto_onehot['Neighborhood'] = toronto_food_venues_df['Neighborhood']

# 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()
```

Out[24]:

	Neighborhood	Afghan Restaurant	American Restaurant	Asian Restaurant	BBQ Joint	Bakery	Bar	Beer Bar	Beer Store	B Resta
1	Parkwoods	0	0	0	0	0	0	0	0	
3	Victoria Village	0	0	0	0	0	0	0	0	
4	Victoria Village	0	0	0	0	0	0	0	0	
5	Victoria Village	0	0	0	0	0	0	0	0	
7	Harbourfront	0	0	0	0	1	0	0	0	

5 rows × 93 columns

And let's examine the new dataframe size.

In [25]:

```
toronto_onehot.shape
```

Out[25]:

```
(1238, 93)
```

Next, let's group rows by neighborhood and by taking the mean of the frequency of occurrence of each category

In [26]:

```
toronto_grouped = toronto_onehot.groupby('Neighborhood').mean().reset_index()
toronto_grouped.head()
```

Out[26]:

	Neighborhood	Afghan Restaurant	American Restaurant	Asian Restaurant	BBQ Joint	Bakery	Bar	Beer Bar	Beer Store
0	Adelaide, King, Richmond	0.0	0.032787	0.04918	0.0	0.032787	0.065574	0.0	0.0
1	Agincourt	0.0	0.000000	0.00000	0.0	0.000000	0.000000	0.0	0.0
2	Albion Gardens, Beaumond Heights, Humbergate, ...	0.0	0.000000	0.00000	0.0	0.000000	0.000000	0.0	0.2
3	Alderwood, Long Branch	0.0	0.000000	0.00000	0.0	0.000000	0.000000	0.0	0.0
4	Bathurst Manor, Downsview North, Wilson Heights	0.0	0.000000	0.00000	0.0	0.000000	0.000000	0.0	0.0

5 rows × 93 columns

Let's confirm the new size

In [27]:

```
toronto_grouped.shape
```

Out[27]:

```
(82, 93)
```

Now let's create the new dataframe and display the top 10 food venues for each neighborhood.

In [28]:

```
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]
```

In [29]:

```
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
neighborhoods_venues_sorted = pd.DataFrame(columns=columns)
neighborhoods_venues_sorted['Neighborhood'] = toronto_grouped['Neighborhood']

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

neighborhoods_venues_sorted.head()
```

Out[29]:

	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 M Comr Ve
0	Adelaide, King, Richmond	Coffee Shop	Steakhouse	Café	Bar	Burger Joint	Asian Restaurant	Restau
1	Agincourt	Breakfast Spot	Latin American Restaurant	Ethiopian Restaurant	Cupcake Shop	Dessert Shop	Dim Sum Restaurant	Dc Restau
2	Albion Gardens, Beaumond Heights, Humbergate, ...	Sandwich Place	Fried Chicken Joint	Beer Store	Pizza Place	Fast Food Restaurant	Wine Shop	Cu Restau
3	Alderwood, Long Branch	Pizza Place	Coffee Shop	Sandwich Place	Eastern European Restaurant	Comfort Food Restaurant	Cuban Restaurant	Cupc S
4	Bathurst Manor, Downsview North, Wilson Heights	Coffee Shop	Middle Eastern Restaurant	Sushi Restaurant	Fast Food Restaurant	Frozen Yogurt Shop	Pizza Place	Sandv P

Cluster Neighborhoods

In [30]:

```
from sklearn.cluster import KMeans
import matplotlib.cm as cm
import matplotlib.colors as colors
```

In [31]:

```
# set number of clusters
kclusters = 4

toronto_grouped_clustering = toronto_grouped.drop('Neighborhood', 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_[0:10]
```

Out[31]:

```
array([1, 1, 1, 1, 1, 1, 1, 1, 1, 1])
```

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

In [32]:

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

toronto_merged = toronto_df.drop('PostalCode', 1)

# merge toronto_grouped with toronto_data to add latitude/longitude for each neighborho
od
toronto_merged = toronto_merged.join(neighborhoods_venues_sorted.set_index('Neighborhoo
d'), on='Neighborhood')

# Drop the rows where cluster labels are missing
toronto_merged.dropna(inplace=True)

toronto_merged.head()
```

Out[32]:

	Borough	Neighborhood	Latitude	Longitude	Cluster Labels	1st Most Common Venue	2nd Most Common Venue	3rd Mos Commoı Venuı
0	North York	Parkwoods	43.753259	-79.329656	1.0	Food & Drink Shop	Wine Shop	Ethiopiaı Restauran
1	North York	Victoria Village	43.725882	-79.315572	1.0	Coffee Shop	French Restaurant	Portuguesı Restauran
2	Downtown Toronto	Harbourfront	43.654260	-79.360636	1.0	Coffee Shop	Bakery	Mexicaı Restauran
3	North York	Lawrence Heights, Lawrence Manor	43.718518	-79.464763	2.0	Coffee Shop	Vietnamese Restaurant	Ethiopiaı Restauran
4	Downtown Toronto	Queen's Park	43.662301	-79.389494	1.0	Coffee Shop	Mexican Restaurant	Smoothiı Shoı

In [33]:

```
toronto_merged.shape
```

Out[33]:

```
(83, 15)
```

Results and Discussion

Let's visualize the resulting clusters

In [34]:

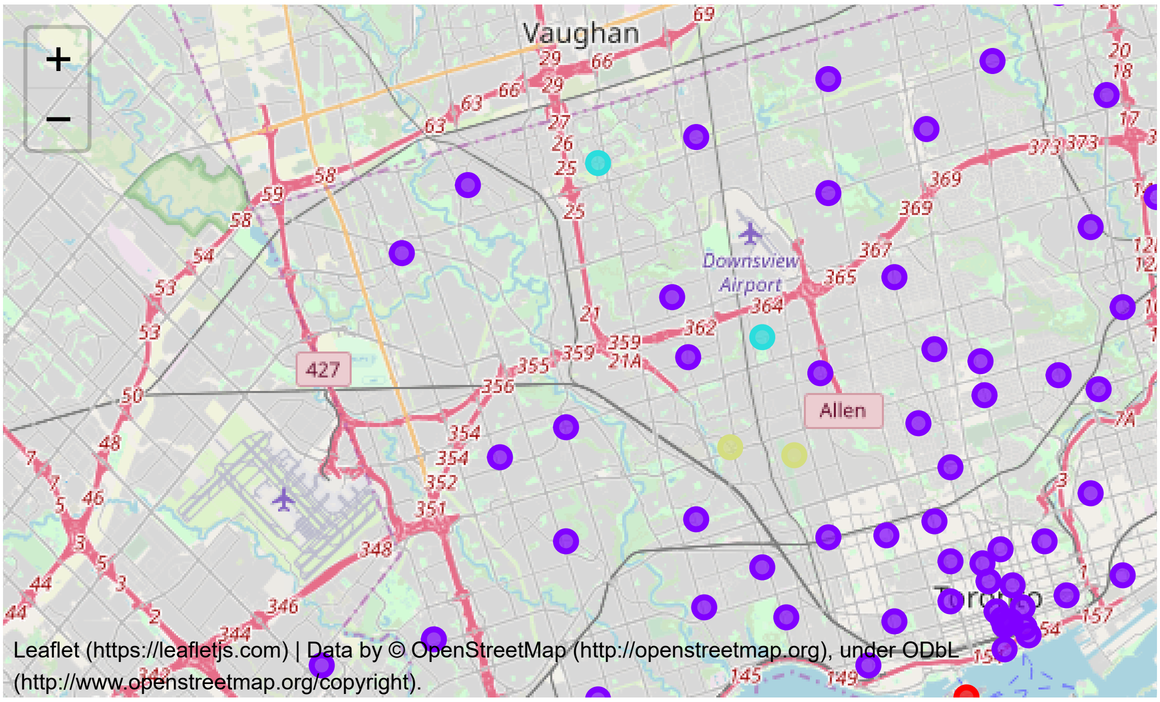
```
# create map
map_clusters = folium.Map(location=[latitude, longitude], zoom_start=11)

# 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(toronto_merged['Latitude'], toronto_merged['Longitud
e'], toronto_merged['Neighborhood'], toronto_merged['Cluster Labels']):
    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[34]:



Examine Clusters

Now, we can examine each cluster and determine the discriminating venue categories that distinguish each cluster.

Cluster 0

In [35]:

```
cluster_0 = toronto_merged.loc[toronto_merged['Cluster Labels'] == 0, toronto_merged.co
lumns[[1] + list(range(5, toronto_merged.shape[1]))]]
for col in ['1st Most Common Venue', '2nd Most Common Venue']:
    print(cluster_0[col].value_counts(ascending=False))
    print('-------------------------------------------')
```

```
Bar    2
Name: 1st Most Common Venue, dtype: int64
-------------------------------------------
Wine Shop    2
Name: 2nd Most Common Venue, dtype: int64
-------------------------------------------
```

So, Cluster 0 is a combination of "Bar" and "Wine Shop".

Cluster 1

In [36]:

```
cluster_1 = toronto_merged.loc[toronto_merged['Cluster Labels'] == 1, toronto_merged.co
lumns[[1] + list(range(5, toronto_merged.shape[1]))]]
for col in ['1st Most Common Venue', '2nd Most Common Venue']:
    print(cluster_1[col].value_counts(ascending=False))
    print('--------------------------------------------')
```

```
Coffee Shop                 27
Café                         8
Pizza Place                  6
Sandwich Place               5
Bakery                       4
Caribbean Restaurant         2
Burger Joint                 2
Asian Restaurant             2
Indian Restaurant            1
Bar                          1
Mediterranean Restaurant     1
Chinese Restaurant           1
Empanada Restaurant          1
Ramen Restaurant             1
Food Truck                   1
Italian Restaurant           1
Food & Drink Shop            1
Breakfast Spot               1
Fried Chicken Joint          1
American Restaurant          1
Sushi Restaurant             1
Greek Restaurant             1
Fast Food Restaurant         1
Health Food Store            1
Falafel Restaurant           1
Name: 1st Most Common Venue, dtype: int64
-------------------------------------------
Coffee Shop                  10
Café                         10
Wine Shop                     9
Pizza Place                   4
Fast Food Restaurant          4
Sandwich Place                3
Japanese Restaurant           3
Fried Chicken Joint           3
Bakery                        3
Mexican Restaurant            3
Dessert Shop                  2
Breakfast Spot                2
Middle Eastern Restaurant     2
Vietnamese Restaurant         2
Burger Joint                  2
Asian Restaurant              2
Steakhouse                    1
Beer Store                    1
French Restaurant             1
Sushi Restaurant              1
Thai Restaurant               1
Latin American Restaurant     1
Indian Restaurant             1
Food & Drink Shop             1
Cocktail Bar                  1
Name: 2nd Most Common Venue, dtype: int64
-------------------------------------------
```

So, Cluster 1 is a "Coffee Shop" dominant cluster.

Cluster 2

In [37]:

```
cluster_2 = toronto_merged.loc[toronto_merged['Cluster Labels'] == 2, toronto_merged.co
lumns[[1] + list(range(5, toronto_merged.shape[1]))]]
for col in ['1st Most Common Venue', '2nd Most Common Venue']:
    print(cluster_2[col].value_counts(ascending=False))
    print('--------------------------------------------')
```

```
Coffee Shop    4
Name: 1st Most Common Venue, dtype: int64
-------------------------------------------
Ethiopian Restaurant     2
Vietnamese Restaurant    1
Korean Restaurant        1
Name: 2nd Most Common Venue, dtype: int64
-------------------------------------------
```

So, Cluster 2 is a "Coffee Shop" dominant cluster.

Cluster 3

In [38]:

```
cluster_3 = toronto_merged.loc[toronto_merged['Cluster Labels'] == 3, toronto_merged.co
lumns[[1] + list(range(5, toronto_merged.shape[1]))]]
for col in ['1st Most Common Venue', '2nd Most Common Venue']:
    print(cluster_3[col].value_counts(ascending=False))
    print('--------------------------------------------')
```

```
Fast Food Restaurant    3
Pizza Place             1
Name: 1st Most Common Venue, dtype: int64
-------------------------------------------
Wine Shop               2
Sandwich Place          1
Fast Food Restaurant    1
Name: 2nd Most Common Venue, dtype: int64
-------------------------------------------
```

So, Cluster 3 is a "Fast Food Restaurant" dominant cluster.

In [39]:

```
summary = [['0',
            cluster_0['1st Most Common Venue'].value_counts(ascending=False).index[0],
            cluster_0['2nd Most Common Venue'].value_counts(ascending=False).index[0],
            cluster_0['Neighborhood'].value_counts(ascending=False).index[0]],
           ['1',
           cluster_1['1st Most Common Venue'].value_counts(ascending=False).index[0],
           cluster_1['2nd Most Common Venue'].value_counts(ascending=False).index[0],
           cluster_1['Neighborhood'].value_counts(ascending=False).index[0]],
           ['2',
           cluster_2['1st Most Common Venue'].value_counts(ascending=False).index[0],
           cluster_2['2nd Most Common Venue'].value_counts(ascending=False).index[0],
           cluster_2['Neighborhood'].value_counts(ascending=False).index[0]],
           ['3',
           cluster_3['1st Most Common Venue'].value_counts(ascending=False).index[0],
           cluster_3['2nd Most Common Venue'].value_counts(ascending=False).index[0],
           cluster_3['Neighborhood'].value_counts(ascending=False).index[0]]]
summary_table = pd.DataFrame(summary, columns=['Cluster', '1st Most Common Venue', '2nd
Most Common Venue', 'Neighborhood'])
summary_table
```

Out[39]:

	Cluster	1st Most Common Venue	2nd Most Common Venue	Neighborhood
0	0	Bar	Wine Shop	Highland Creek, Rouge Hill, Port Union
1	1	Coffee Shop	Coffee Shop	Queen's Park
2	2	Coffee Shop	Ethiopian Restaurant	Woburn
3	3	Fast Food Restaurant	Wine Shop	Caledonia-Fairbanks

Coffe Shop is the most common venue across all the clusters or neighborhoods.

Conclusion

In conclusion, the neighborhoods of Toronto City can be segmented into 4 clusters and upon analysis, it was possible to rename them basis upon the categories of venues in and around that neighborhood. Along with Coffee Shop, Fast Food Restaurant, Bar and Wine Shop are very dominant in Toronto City. This project can also be adjusted to use with other business.