Exploring and clustering the neighborhoods in Toronto

Part 1

In [2]:
```
import pandas as pd
import numpy as np
import types
from botocore.client import Config
import ibm_boto3
```

We copied the data from Wikipedia to excel file and downloaded this file to my Notebook, using the following code:

In [3]:
```
def __iter__(self): return 0

# @hidden_cell
# The following code accesses a file in your IBM Cloud Object Storage. It includes yo
ur credentials.
# You might want to remove those credentials before you share the notebook.
client_11aaefccbd32400f8784ede2c5c844af = ibm_boto3.client(service_name='s3',
    ibm_api_key_id='-VN2-M-BMVHsGanr--Racx2_rFeWPk1Z0a6ddeo0oE0n',
    ibm_auth_endpoint="https://iam.ng.bluemix.net/oidc/token",
    config=Config(signature_version='oauth'),
    endpoint_url='https://s3-api.us-geo.objectstorage.service.networklayer.com')

body = client_11aaefccbd32400f8784ede2c5c844af.get_object(Bucket='courseracapstone-do
notdelete-pr-uzusml2razjnkt',Key='Toronto.xlsx')['Body']
# add missing __iter__ method, so pandas accepts body as file-like object
if not hasattr(body, "__iter__"): body.__iter__ = types.MethodType( __iter__, body )

df = pd.read_excel(body)
df.head()
```

Out[3]:

	Postal code	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 [4]:
```
df.shape
```

Out[4]:
```
(288, 3)
```

We are required to iqnore cells with a borough that is Not assigned. We drop these cells, using the following code:

In [5]:
```
df = df[df.Borough != 'Not assigned']
df.head()
```

Out[5]:

	Postal code	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

In [6]:
```
df.shape
```

Out[6]:
```
(211, 3)
```

In [7]:
```
df.reset_index(drop=True, inplace = True)
df.head()
```

Out[7]:

	Postal code	Borough	Neighbourhood
0	M3A	North York	Parkwoods
1	M4A	North York	Victoria Village
2	M5A	Downtown Toronto	Harbourfront
3	M5A	Downtown Toronto	Regent Park
4	M6A	North York	Lawrence Heights

In [8]:
```
df.dtypes
```

Out[8]:
```
Postal code      object
Borough          object
Neighbourhood    object
dtype: object
```

In order to group by postal codes and neigbourhoods we use the following code:

In [9]:
```
df = df.groupby('Postal code').agg({'Borough':'min','Neighbourhood':','.join}).reset_
index()
df.head(11)
```

Out[9]:

	Postal code	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
5	M1J	Scarborough	Scarborough Village
6	M1K	Scarborough	East Birchmount Park,Ionview,Kennedy Park
7	M1L	Scarborough	Clairlea,Golden Mile,Oakridge
8	M1M	Scarborough	Cliffcrest,Cliffside,Scarborough Village West
9	M1N	Scarborough	Birch Cliff,Cliffside West
10	M1P	Scarborough	Dorset Park,Scarborough Town Centre,Wexford He...

We should find the Neigbourhood, which is not assigned yet

In [10]:
```
a = df.loc[df.Neighbourhood=='Not assigned']
print (a)
```

```
   Postal code       Borough Neighbourhood
85         M7A  Queen's Park  Not assigned
```

As it's the only one cell, we simply may replace "not assigned' with "Queen's Park"

In [11]:
```
df.replace({'Neighbourhood': 'Not assigned'}, {'Neighbourhood': "Queen's Park"}, inpl
ace = True)
```

We made a cross check to ensure that we don't have any rows with "not assigned"

In [12]:
```
a = df.loc[df.Neighbourhood=='Not assigned']
print (a)
```

```
Empty DataFrame
Columns: [Postal code, Borough, Neighbourhood]
Index: []
```

In [13]:
```
df.head()
```

Out[13]:

	Postal code	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 [14]:
```
df.shape
```

Out[14]:
```
(103, 3)
```

In [17]:
```
df['Postal code'].head()
```

Out[17]:
```
0    M1B
1    M1C
2    M1E
3    M1G
4    M1H
Name: Postal code, dtype: object
```

Part 2

We download the data from the csv file:

In [26]:
```
body = client_11aaefccbd32400f8784ede2c5c844af.get_object(Bucket='courseracapstone-donotdelete-pr-uzusml2razjnkt',Key='Geospatial_Coordinates.csv')['Body']
# add missing __iter__ method, so pandas accepts body as file-like object
if not hasattr(body, "__iter__"): body.__iter__ = types.MethodType( __iter__, body )

# If you are reading an Excel file into a pandas DataFrame, replace `read_csv` by `read_excel` in the next statement.
df_geo= pd.read_csv(body)
df_geo.head()
```

Out[26]:

	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 [27]:
```
df_geo.shape
```

Out[27]:
```
(103, 3)
```

We rename the columns in order to facilitate process of merging below

In [28]:
```
df_geo = df_geo.rename(columns={"Postal Code":"Postal code"})
df_geo.head()
```

Out[28]:

	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

We merge 2 tables together

In [31]:
```
df_new = pd.merge(df, df_geo, on='Postal code')
df_new.head()
```

Out[31]:

	Postal code	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 [30]:
```
df_new.shape
```

Out[30]: (103, 5)

Part 3

Import all relevant libraries

In [32]:

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

import pandas as pd # library for data analsysis
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 compl
eted the Foursquare API lab
from geopy.geocoders import Nominatim # convert an address into latitude and longitud
e values

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

# 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

print('Libraries imported.')
```

```
Solving environment: done

## Package Plan ##

  environment location: /opt/conda/envs/Python36

  added / updated specs:
    - geopy


The following packages will be downloaded:

    package                    |            build
    ---------------------------|-----------------
    geopy-1.20.0               |              py_0          57 KB  conda-forge
    ca-certificates-2019.9.11  |       hecc5488_0         144 KB  conda-forge
    geographiclib-1.50         |              py_0          34 KB  conda-forge
    certifi-2019.9.11          |           py36_0         147 KB  conda-forge
    openssl-1.1.1d             |       h516909a_0         2.1 MB  conda-forge
    ------------------------------------------------------------
                                           Total:         2.5 MB

The following NEW packages will be INSTALLED:

    geographiclib:   1.50-py_0         conda-forge
    geopy:           1.20.0-py_0       conda-forge

The following packages will be UPDATED:

    certifi:         2019.9.11-py36_0              --> 2019.9.11-py36_0     conda-fo
rge

The following packages will be DOWNGRADED:

    ca-certificates: 2019.10.16-0                  --> 2019.9.11-hecc5488_0 conda-fo
rge
    openssl:         1.1.1d-h7b6447c_3             --> 1.1.1d-h516909a_0    conda-fo
rge


Downloading and Extracting Packages
geopy-1.20.0         | 57 KB     | ##################################### | 100%
ca-certificates-2019 | 144 KB    | ##################################### | 100%
geographiclib-1.50   | 34 KB     | ##################################### | 100%
certifi-2019.9.11    | 147 KB    | ##################################### | 100%
openssl-1.1.1d       | 2.1 MB    | ##################################### | 100%
Preparing transaction: done
Verifying transaction: done
Executing transaction: done
Solving environment: done

## Package Plan ##

  environment location: /opt/conda/envs/Python36

  added / updated specs:
    - folium=0.5.0


The following packages will be downloaded:

    package                    |            build
    ---------------------------|-----------------
    vincent-0.4.4              |              py_1          28 KB  conda-forge
    altair-3.2.0               |           py36_0         770 KB  conda-forge
```

```
    branca-0.3.1               |           py_0          25 KB  conda-forge
    folium-0.5.0               |           py_0          45 KB  conda-forge
    ------------------------------------------------------------
                                           Total:         868 KB

The following NEW packages will be INSTALLED:

    altair:  3.2.0-py36_0 conda-forge
    branca:  0.3.1-py_0   conda-forge
    folium:  0.5.0-py_0   conda-forge
    vincent: 0.4.4-py_1   conda-forge


Downloading and Extracting Packages
vincent-0.4.4        | 28 KB     | ##################################### | 100%
altair-3.2.0         | 770 KB    | ##################################### | 100%
branca-0.3.1         | 25 KB     | ##################################### | 100%
folium-0.5.0         | 45 KB     | ##################################### | 100%
Preparing transaction: done
Verifying transaction: done
Executing transaction: done
Libraries imported.
```

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

In [33]:
```
address = 'Toronto'
geolocator = Nominatim(user_agent="ny_explorer")
location = geolocator.geocode(address,timeout=60, exactly_one=True)
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

Let's shoose only those Boroughs that contains "Toronto"

In [65]:
```
df_toronto = df_new[df_new['Borough'].str.contains("Toronto")==True]
df_toronto = df_toronto.rename(columns={"Neighbourhood":"Neighborhood"})
df_toronto.head()
```

Out[65]:

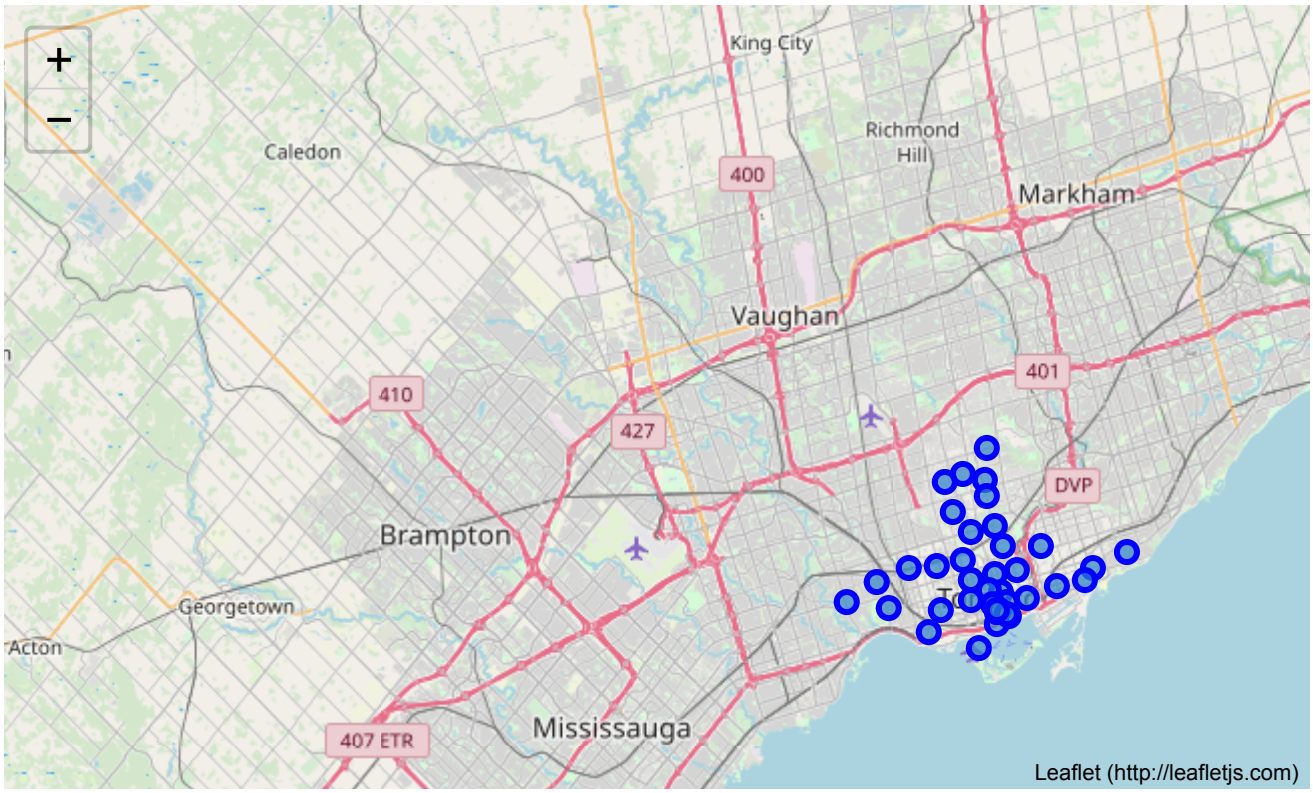
	Postal code	Borough	Neighborhood	Latitude	Longitude
37	M4E	East Toronto	The Beaches	43.676357	-79.293031
41	M4K	East Toronto	The Danforth West,Riverdale	43.679557	-79.352188
42	M4L	East Toronto	The Beaches West,India Bazaar	43.668999	-79.315572
43	M4M	East Toronto	Studio District	43.659526	-79.340923
44	M4N	Central Toronto	Lawrence Park	43.728020	-79.388790

In [66]:
```
# 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(df_toronto['Latitude'], df_toronto['Longit
ude'], df_toronto['Borough'], df_toronto['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[66]:



Let's exctract the data from Foursquare

In [67]:
```
CLIENT_ID = 'HA2LTVYBAF4QMMJEEBTEFH1VGD2B1GIZULF1EHABR0KGWJYH' # your Foursquare ID
CLIENT_SECRET = 'JLOCPKMTSPTGYN2MG234XHUAYJEOM22EXIGWFXPFWBGA30M5' # your Foursquare
 Secret
VERSION = '20191105' # Foursquare API version
LIMIT = 100

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

```
Your credentails:
CLIENT_ID: HA2LTVYBAF4QMMJEEBTEFH1VGD2B1GIZULF1EHABR0KGWJYH
CLIENT_SECRET:JLOCPKMTSPTGYN2MG234XHUAYJEOM22EXIGWFXPFWBGA30M5
```

In [68]:

```
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_sec
ret={}&v={}&ll={},{}&radius={}&limit={}'.format(
        url = 'https://api.foursquare.com/v2/venues/explore?&client_id={}&client_secr
et={}&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 venu
e_list])
    nearby_venues.columns = ['Neighborhood',
                  'Neighborhood Latitude',
                  'Neighborhood Longitude',
                  'Venue',
                  'Venue Latitude',
                  'Venue Longitude',
                  'Venue Category']

    return(nearby_venues)
```

In [70]:
```
toronto_venues = getNearbyVenues(names=df_toronto['Neighborhood'],
                                 latitudes=df_toronto['Latitude'],
                                 longitudes=df_toronto['Longitude']
                                )
```

```
The Beaches
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
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 Spadina,Railway Lan
ds,South Niagara
Stn A PO Boxes 25 The Esplanade
First Canadian Place,Underground city
Christie
Dovercourt Village,Dufferin
Little Portugal,Trinity
Brockton,Exhibition Place,Parkdale Village
High Park,The Junction South
Parkdale,Roncesvalles
Runnymede,Swansea
Business Reply Mail Processing Centre 969 Eastern
```

In [71]:
```
print(toronto_venues.shape)
toronto_venues.head()
```

```
(1693, 7)
```

Out[71]:

	Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
0	The Beaches	43.676357	-79.293031	Glen Manor Ravine	43.676821	-79.293942	Trail
1	The Beaches	43.676357	-79.293031	The Big Carrot Natural Food Market	43.678879	-79.297734	Health Food Store
2	The Beaches	43.676357	-79.293031	Grover Pub and Grub	43.679181	-79.297215	Pub
3	The Beaches	43.676357	-79.293031	Glen Stewart Ravine	43.676300	-79.294784	Other Great Outdoors
4	The Beaches	43.676357	-79.293031	Upper Beaches	43.680563	-79.292869	Neighborhood

In [72]:
```
toronto_venues.groupby('Neighborhood').count()
```

Out[72]:

Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
Adelaide,King,Richmond	100	100	100	100	100	100
Berczy Park	56	56	56	56	56	56
Brockton,Exhibition Place,Parkdale Village	24	24	24	24	24	24
Business Reply Mail Processing Centre 969 Eastern	20	20	20	20	20	20
CN Tower,Bathurst Quay,Island airport,Harbourfront West,King and Spadina,Railway Lands,South Niagara	14	14	14	14	14	14
Cabbagetown,St. James Town	44	44	44	44	44	44
Central Bay Street	82	82	82	82	82	82
Chinatown,Grange Park,Kensington Market	94	94	94	94	94	94
Christie	17	17	17	17	17	17
Church and Wellesley	86	86	86	86	86	86
Commerce Court,Victoria Hotel	100	100	100	100	100	100
Davisville	32	32	32	32	32	32
Davisville North	10	10	10	10	10	10
Deer Park,Forest Hill SE,Rathnelly,South Hill,Summerhill West	15	15	15	15	15	15
Design Exchange,Toronto Dominion Centre	100	100	100	100	100	100
Dovercourt Village,Dufferin	15	15	15	15	15	15
First Canadian Place,Underground city	100	100	100	100	100	100
Forest Hill North,Forest Hill West	4	4	4	4	4	4
Harbord,University of Toronto	35	35	35	35	35	35
Harbourfront East,Toronto Islands,Union Station	100	100	100	100	100	100
Harbourfront,Regent Park	49	49	49	49	49	49
High Park,The Junction South	24	24	24	24	24	24
Lawrence Park	3	3	3	3	3	3
Little Portugal,Trinity	63	63	63	63	63	63
Moore Park,Summerhill East	2	2	2	2	2	2
North Toronto West	22	22	22	22	22	22
Parkdale,Roncesvalles	15	15	15	15	15	15
Rosedale	4	4	4	4	4	4
Roselawn	2	2	2	2	2	2
Runnymede,Swansea	35	35	35	35	35	35
Ryerson,Garden District	100	100	100	100	100	100
St. James Town	100	100	100	100	100	100
Stn A PO Boxes 25 The Esplanade	98	98	98	98	98	98

Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
Studio District	39	39	39	39	39	39
The Annex,North Midtown,Yorkville	21	21	21	21	21	21
The Beaches	5	5	5	5	5	5
The Beaches West,India Bazaar	21	21	21	21	21	21
The Danforth West,Riverdale	42	42	42	42	42	42

In [73]:
```
print('There are {} uniques categories.'.format(len(toronto_venues['Venue Category'].unique())))
```

```
There are 231 uniques categories.
```

Clustering

In [74]:
```
# one hot encoding
toronto_onehot = pd.get_dummies(toronto_venues[['Venue Category']], prefix="", prefix_sep="")

# add neighborhood column back to dataframe
toronto_onehot['Neighborhood'] = toronto_venues['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[74]:

	Yoga Studio	Afghan Restaurant	Airport	Airport Food Court	Airport Gate	Airport Lounge	Airport Service	Airport Terminal	American Restaurant	Antique Shop	Aquariu
0	0	0	0	0	0	0	0	0	0	0	
1	0	0	0	0	0	0	0	0	0	0	
2	0	0	0	0	0	0	0	0	0	0	
3	0	0	0	0	0	0	0	0	0	0	
4	0	0	0	0	0	0	0	0	0	0	

In [75]:
```
toronto_grouped = toronto_onehot.groupby('Neighborhood').mean().reset_index()
toronto_grouped
```

Out[75]:

	Neighborhood	Yoga Studio	Afghan Restaurant	Airport	Airport Food Court	Airport Gate	Airport Lounge	Airport Service	Air Term
0	Adelaide,King,Richmond	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
1	Berczy Park	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
2	Brockton,Exhibition Place,Parkdale Village	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
3	Business Reply Mail Processing Centre 969 Eastern	0.050000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
4	CN Tower,Bathurst Quay,Island airport,Harbourf...	0.000000	0.000000	0.071429	0.071429	0.071429	0.142857	0.142857	0.142
5	Cabbagetown,St. James Town	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
6	Central Bay Street	0.012195	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
7	Chinatown,Grange Park,Kensington Market	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
8	Christie	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
9	Church and Wellesley	0.011628	0.011628	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
10	Commerce Court,Victoria Hotel	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
11	Davisville	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
12	Davisville North	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
13	Deer Park,Forest Hill SE,Rathnelly,South Hill,...	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
14	Design Exchange,Toronto Dominion Centre	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
15	Dovercourt Village,Dufferin	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
16	First Canadian Place,Underground city	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
17	Forest Hill North,Forest Hill West	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
18	Harbord,University of Toronto	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
19	Harbourfront East,Toronto Islands,Union Station	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
20	Harbourfront,Regent Park	0.020408	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
21	High Park,The Junction South	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
22	Lawrence Park	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
23	Little Portugal,Trinity	0.015873	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
24	Moore Park,Summerhill East	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
25	North Toronto West	0.045455	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
26	Parkdale,Roncesvalles	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
27	Rosedale	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
28	Roselawn	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000

	Neighborhood	Yoga Studio	Afghan Restaurant	Airport	Airport Food Court	Airport Gate	Airport Lounge	Airport Service	Air Term
29	Runnymede,Swansea	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
30	Ryerson,Garden District	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
31	St. James Town	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
32	Stn A PO Boxes 25 The Esplanade	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
33	Studio District	0.025641	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
34	The Annex,North Midtown,Yorkville	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
35	The Beaches	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
36	The Beaches West,India Bazaar	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
37	The Danforth West,Riverdale	0.023810	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000

In [76]:
```
toronto_grouped.shape
```

Out[76]:
```
(38, 231)
```

In [82]:
```
# set number of clusters
kclusters = 5

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[82]:
```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0], dtype=int32)
```

In [83]:
```
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 [84]:
```
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_gro
uped.iloc[ind, :], num_top_venues)

neighborhoods_venues_sorted.head()
```

Out[84]:

	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	
0	Adelaide,King,Richmond	Coffee Shop	Café	Bar	Steakhouse	Restaurant	Cosmetics Shop	Bakery	R
1	Berczy Park	Coffee Shop	Seafood Restaurant	Beer Bar	Farmers Market	Cocktail Bar	Bakery	Café	
2	Brockton,Exhibition Place,Parkdale Village	Breakfast Spot	Café	Coffee Shop	Bakery	Climbing Gym	Sandwich Place	Burrito Place	R
3	Business Reply Mail Processing Centre 969 Eastern	Light Rail Station	Gym / Fitness Center	Spa	Skate Park	Restaurant	Recording Studio	Pizza Place	
4	CN Tower,Bathurst Quay,Island airport,Harbourf...	Airport Lounge	Airport Terminal	Airport Service	Boat or Ferry	Sculpture Garden	Bar	Boutique	

In [85]:
```
# add clustering labels
neighborhoods_venues_sorted.insert(0, 'Cluster Labels', kmeans.labels_)

toronto_merged = df_toronto

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

toronto_merged.head()
```

Out[85]:

	Postal code	Borough	Neighborhood	Latitude	Longitude	Cluster Labels	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4t Co
37	M4E	East Toronto	The Beaches	43.676357	-79.293031	0	Pub	Other Great Outdoors	Health Food Store	
41	M4K	East Toronto	The Danforth West,Riverdale	43.679557	-79.352188	0	Greek Restaurant	Coffee Shop	Ice Cream Shop	Res
42	M4L	East Toronto	The Beaches West,India Bazaar	43.668999	-79.315572	0	Sandwich Place	Park	Pub	Li
43	M4M	East Toronto	Studio District	43.659526	-79.340923	0	Café	Coffee Shop	American Restaurant	
44	M4N	Central Toronto	Lawrence Park	43.728020	-79.388790	4	Park	Swim School	Bus Line	

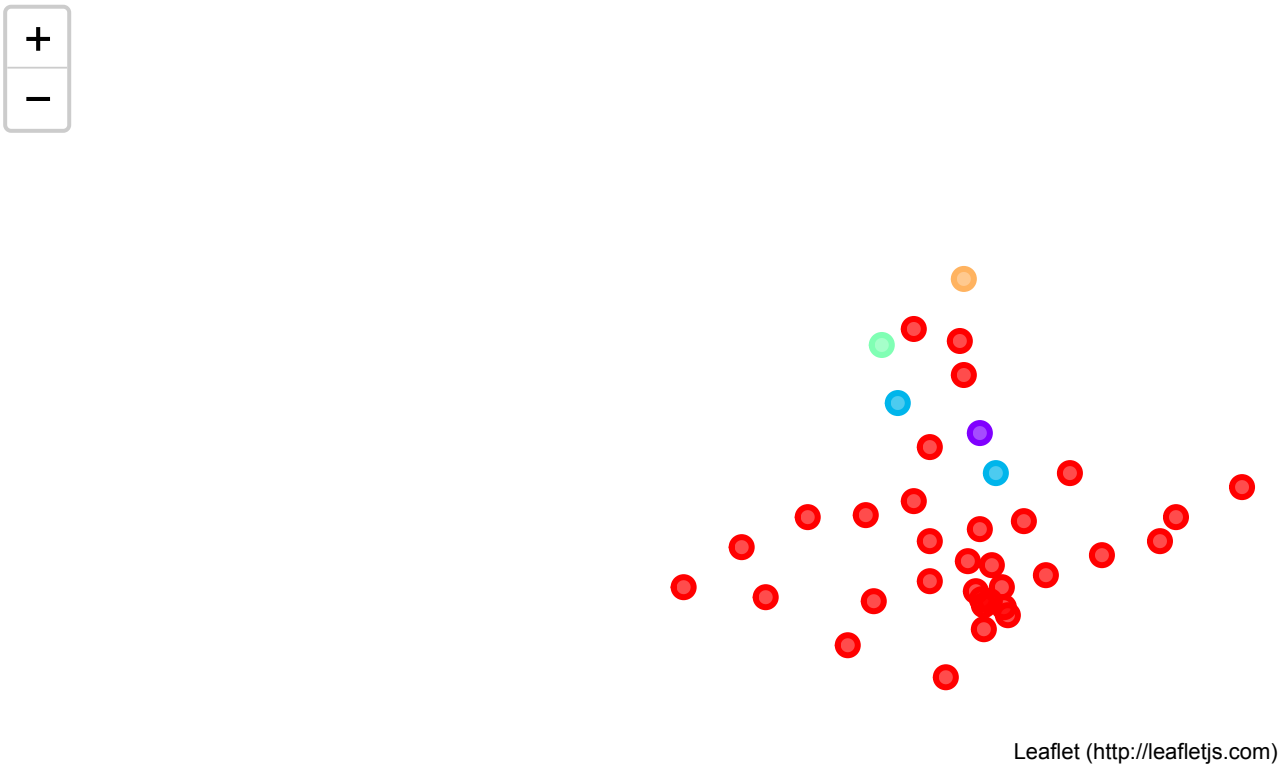
In [86]:
```
# create map
map_clusters = folium.Map(location=[latitude, longitude], zoom_start=11)

# 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['Longitu
de'], 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[cluster-1],
        fill=True,
        fill_color=rainbow[cluster-1],
        fill_opacity=0.7).add_to(map_clusters)

map_clusters
```

Out[86]:



In [130]:
```
toronto_clusters = toronto_merged[['Borough','Cluster Labels']]
toronto_clusters["value"]=1
pivot = pd.pivot_table(toronto_clusters, values="value", index=["Borough"], columns=
"Cluster Labels", fill_value=0)
pivot
```

```
/opt/conda/envs/Python36/lib/python3.6/site-packages/ipykernel/__main__.py:2: Settin
gWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/in
dexing.html#indexing-view-versus-copy
  from ipykernel import kernelapp as app
```

Out[130]:

Cluster Labels	0	1	2	3	4
Borough					
Central Toronto	1	1	1	1	1
Downtown Toronto	1	0	1	0	0
East Toronto	1	0	0	0	0
West Toronto	1	0	0	0	0

Conclusion: As it's seen above, Downtown, East and West Toronto are within one cluster, which means that they are more or less similar.

In [ ]: