Peer-graded Assignment: Capstone Project - The Battle of Neighborhoods

1. Introduction

Everyone needs a place to live, a place to call home. Sometimes we want to live near our workplace. On the other hand the need to be close to nature is growing stronger and people choose suburban existence.

Surroundings and places nearby are strong factors when choosing neighbourhoods while looking for property.

Business Problem: Determine the attractiveness of neighbourhoods in Toronto, considering environmental and facilities factors, to justify real estates prices and make data-driven decisions on certain investments.

In the previous assignment neighbourhoods data with postal codes was we scraped from Wikipedia. The official distribution of neighbourhoods in Toronto is different from information given on Wikipedia. To create dataframe contaning neighbourhoods and corresponding coordinates I will use datasets provided by Toronto Open Data portal.

Target audience

Knowledge generated in this project would be useful for anyone looking for real esates, from big investors choosing areas for new investments in housing, families looking for best places to raise their children or anyone seeking rentals in "good" neighbourhoods.

By further analysis I will try find the answer what actually "good" neighbourhood means.

2. Data

Datasets used in this project:

- neighbourhoods info (name, longitude, latitude)
- environmental features of neighbourhoods
- venues information gathered through Foursquare API

3. Methodology

This part consists of the following compononets:

- Importing data
- Cleaning data
- Exploratory analysis
- Statistical method of choosing best naighbourhoods by rating
- Clustering with K-Mean algorithm

Data preparation

Importing neighbourhoods dataset

In [2]:
```
import pandas as pd
df = pd.read_csv('~/Downloads/neighbourhoods-2.csv').sort_values('A
REA_SHORT_CODE')
df.head()
```

Out[2]:

	_id	AREA_ID	AREA_ATTR_ID	PARENT_AREA_ID	AREA_SHORT_CODE	AREA_LONG
63	1184	25886718	25926725	49885	1	
20	1141	25886715	25926682	49885	2	
56	1177	25886723	25926718	49885	3	
40	1161	25886730	25926702	49885	4	
112	1233	25886733	25926774	49885	5	

Imported dataframe contains geospatial data that can be used for map visualizations. Columns needed for further analysis:

- AREA_SHORT_CODE
- AREA_NAME
- LONGITUDE
- LATITUDE

In [3]:
```
df1 = df[['AREA_SHORT_CODE', 'AREA_NAME', 'LONGITUDE', 'LATITUDE']]
.reset_index(drop=True)
df1.head()
```

Out[3]:

	AREA_SHORT_CODE	AREA_NAME	LONGITUDE	LATITUDE
0	1	West Humber-Clairville (1)	-79.596356	43.716180
1	2	Mount Olive-Silverstone-Jamestown (2)	-79.587259	43.746868
2	3	Thistletown-Beaumond Heights (3)	-79.563491	43.737988
3	4	Rexdale-Kipling (4)	-79.566228	43.723725
4	5	Elms-Old Rexdale (5)	-79.548983	43.721519

Importing environment dataset

In [4]:
```
df2 = pd.read_csv('~/Downloads/environment.csv', index_col=None, de
limiter =';')
df2.head()
```

Out[4]:

	Neighbourhood	Neighbourhood Id	Green Rebate Programs	Green Spaces	Pollutant Carcinogenic TEP Score	Pollutant Non-Carcinogenic TEP Score	Pc F
0	West Humber-Clairville	1	428	2,078835532	5737,87	18658529,73	
1	Mount Olive-Silverstone-Jamestown	2	250	1,048870056	29,76	2015	
2	Thistletown-Beaumond Heights	3	118	0,939107957	0	0	
3	Rexdale-Kipling	4	121	0,240663012	0	37632	
4	Elms-Old Rexdale	5	73	0,730089694	0	309	

Before joining this two datasets it would be helpful to check sizes of each dataframe.

In [5]:
```
df1.shape
```
Out[5]: (140, 4)

In [6]:
```
df2.shape
```
Out[6]: (140, 8)

Both dataframes have 140 rows, for 140 neighbourhoods in Toronto (as official Toronto websites claim). Both dataframes contain neighbourhood name but names in *df1* also have id in the name in brackets. To make sure we are joining the same neighbourhoods join operation will be performed on id column. In df1 it is called 'AREA_SHORT_CODE' while in df2 the name of the corresponding column is 'Neighbourhood Id'.

Joining datasets

In [7]:
```
data = pd.merge(
    df1,
    df2,
    left_on='AREA_SHORT_CODE',
    right_on='Neighbourhood Id')

data.head()
```
Out[7]:

	AREA_SHORT_CODE	AREA_NAME	LONGITUDE	LATITUDE	Neighbourhood	Neighbourh
0	1	West Humber-Clairville (1)	-79.596356	43.716180	West Humber-Clairville	
1	2	Mount Olive-Silverstone-Jamestown (2)	-79.587259	43.746868	Mount Olive-Silverstone-Jamestown	
2	3	Thistletown-Beaumond Heights (3)	-79.563491	43.737988	Thistletown-Beaumond Heights	
3	4	Rexdale-Kipling (4)	-79.566228	43.723725	Rexdale-Kipling	
4	5	Elms-Old Rexdale (5)	-79.548983	43.721519	Elms-Old Rexdale	

Data cleaning

Since it is official toronto data, all rows match when it comes to neighbourhood names. Now dataframe *data* contains 2 columns with neighbourhood name so we can drop 'AREA_NAME' coulumn to keep dataframe clean. Also 'AREA_SHORT_CODE' is the same as 'Neighbourhood id' so it can be dropped.

In [8]:
```
data.drop(['AREA_NAME', 'AREA_SHORT_CODE'], axis=1, inplace=True)

data.head()
```

Out[8]:

	LONGITUDE	LATITUDE	Neighbourhood	Neighbourhood Id	Green Rebate Programs	Green Spaces	P Carcir TEI
0	-79.596356	43.716180	West Humber-Clairville	1	428	2,078835532	
1	-79.587259	43.746868	Mount Olive-Silverstone-Jamestown	2	250	1,048870056	
2	-79.563491	43.737988	Thistletown-Beaumond Heights	3	118	0,939107957	
3	-79.566228	43.723725	Rexdale-Kipling	4	121	0,240663012	
4	-79.548983	43.721519	Elms-Old Rexdale	5	73	0,730089694	

Columns explanation

• LONGITUDE, LATITUDE - geospatial coordinates • Neighbourhood - neighbourhood name • Neighbourhood Id - helps identify neighbourhood

Columns with environmental information (description below was provided with dataset from various Toronto units and organizations):

• Green Rebate Programs - Water-Saving Rebate Program (toilets and washing machines) -This is a Time Series showing a Score (1 to 100) of the change in this indicator between the Reference Period 2008 and Reference Period 2011. Time series indicator scores are calculated from the percent change between reference periods, using a formula which categorizes negative and positive percent change into 1-100 scores

• Green Spaces - Green/Open Spaces - City of Toronto, Parks Forestry & Recreation, Jan 2012 data. Extracted from GCC SDE geospatial repository (layer TCL3.UPARK) in March 2013. Total land area (in square kilometres) designated as parkland or green space (including utility corridors and utility areas such as soccer fields)

• Pollutant Carcinogenic TEP Score - Total Carcinogenic Toxic Equivalency Potentials (TEP) Score for All Chemicals - Total Carcinogenic Toxic Equivalency Potentials (TEP) Score for All Chemicals as provided by the City of Toronto's ChemTRAC chemical tracking program for 2012 (www.toronto.ca/chemtrac). A full explanation of TEPs can be found here: http://scorecard.goodguide.com/env-releases/def/tep_gen.html (http://scorecard.goodguide.com/env-releases/def/tep_gen.html)

• Pollutant Non-Carcinogenic TEP Score - Total Non-Carcinogenic Toxic Equivalency Potentials (TEP) Score for All Chemicals - Total Non-Carcinogenic Toxic Equivalency Potentials (TEP) Score for All Chemicals as provided by the City of Toronto's ChemTRAC chemical tracking program for 2012 (www.toronto.ca/chemtrac). A full explanation of TEPs can be found here: http://scorecard.goodguide.com/env-releases/def/tep_gen.html (http://scorecard.goodguide.com/env-releases/def/tep_gen.html)

• Pollutants Released to Air - Total Pollutants Released to Air 2012 as tracked by the City of Toronto's ChemTRAC program for 25 different pollutant chemicals

• Tree Cover - City of Toronto, Parks Forestry & Recreation, Forestry Management, 2008 data. Indicator presents total area (in square metres) of tree foliage cover identified using satellite imaging.

Importing Foursquare API data

In [9]:
```
import requests # library to handle requests
```

In [100]:
```
# @hidden_cell
CLIENT_ID = 'PGNGQ1TOCDTFFEFA4FEPJY5RBOKTSFI13LLGLOL0C4BXMH2O' # yo
ur Foursquare ID
CLIENT_SECRET = 'RZZ1F2N1RBGCP0SEFHSZCRSHIVFBB3NDSDJB3LRFVHY53LZN'
# your Foursquare Secret
VERSION = '20180605' # Foursquare API version

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

```
Your credentails:
CLIENT_ID: PGNGQ1TOCDTFFEFA4FEPJY5RBOKTSFI13LLGLOL0C4BXMH2O
CLIENT_SECRET:RZZ1F2N1RBGCP0SEFHSZCRSHIVFBB3NDSDJB3LRFVHY53LZN
```

In [11]:
```
def getNearbyVenues(names, latitudes, longitudes, radius=500, LIMIT
=100):

    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,
            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_lis
t for item in venue_list])
    nearby_venues.columns = ['Neighbourhood',
                  'Neighbourhood Latitude',
                  'Neighbourhood Longitude',
                  'Venue',
                  'Venue Latitude',
                  'Venue Longitude',
                  'Venue Category']

    return(nearby_venues)
```

In [12]:
```
toronto_venues = getNearbyVenues(
                    names = data['Neighbourhood'],
                    latitudes = data['LATITUDE'],
                    longitudes = data['LONGITUDE']
                            )
toronto_venues.head()
```

Out[12]:

	Neighbourhood	Neighbourhood Latitude	Neighbourhood Longitude	Venue	Venue Latitude	Venue Longitude	(
0	West Humber-Clairville	43.71618	-79.596356	Woodbine Racetrack	43.717929	-79.598574	Ra
1	West Humber-Clairville	43.71618	-79.596356	Mandarin Buffet	43.719798	-79.595820	R
2	West Humber-Clairville	43.71618	-79.596356	Tim Hortons	43.714657	-79.593716	Co
3	West Humber-Clairville	43.71618	-79.596356	Xawaash	43.715786	-79.593053	Medi R
4	West Humber-Clairville	43.71618	-79.596356	Winners	43.719819	-79.594923	Clotł

In [13]:
```
toronto_venues.shape
```

Out[13]: (2109, 7)

There are 2109 venues in 140 neighbourhoods in *toronto_venues* dataframe.

Exploratory analysis of neighbourhoods

In this part I will check distinct features of datasets to choose top neighbourhoods in Toronto. Features chosen for this analysis are: Green Rebate Programs, Green Spaces and Tree Cover. The environmental norms considering pollutants have changed after cration of this datasets and analysis regarding these features might be scope of another project.

For each feature 'Rating' will be attributed. The smaller its value, the better neighbourhood is. Rating is created by taking index after sorting.

Toronto environmental dataset *data*

1. Green Rebate Programs top neighbourhoods

In [14]:
```
data = data.sort_values('Green Rebate Programs', ascending=False).r
eset_index(drop=True)
data['Rating 1'] = data.index
data.head(10)
```

Out[14]:

	LONGITUDE	LATITUDE	Neighbourhood	Neighbourhood Id	Green Rebate Programs	Green Spaces	Carc Tl
0	-79.275009	43.820691	Milliken	130	815	0,512021903	
1	-79.266712	43.805441	Agincourt North	129	775	0,250098151	
2	-79.314084	43.795716	L'Amoreaux	117	745	0,598731588	
3	-79.186343	43.821201	Rouge	131	725	14,27145522	
4	-79.321207	43.812959	Steeles	116	692	0,450219119	
5	-79.228586	43.766740	Woburn	137	600	1,354011781	
6	-79.222517	43.803658	Malvern	132	549	0,690798137	
7	-79.265612	43.788658	Agincourt South-Malvern West	128	470	0,2397495	
8	-79.596356	43.716180	West Humber-Clairville	1	428	2,078835532	
9	-79.298637	43.748572	Wexford/Maryvale	119	396	0,47116816	

2. Green spaces top neighbourhoods

In [15]:
```
data = data.sort_values('Green Spaces', ascending=False).reset_inde
x(drop=True)
data['Rating 2'] = data.index
data.head(10)
```

Out[15]:

	LONGITUDE	LATITUDE	Neighbourhood	Neighbourhood Id	Green Rebate Programs	Green Spaces	P Carcir TEI
0	-79.580445	43.658017	Eringate-Centennial-West Deane	11	313	2,891261519	
1	-79.335651	43.649292	South Riverdale	70	388	2,776069153	
2	-79.377202	43.633880	Waterfront Communities-The Island	77	187	2,567218437	
3	-79.596356	43.716180	West Humber-Clairville	1	428	2,078835532	
4	-79.186343	43.821201	Rouge	131	725	14,27145522	
5	-79.176676	43.767490	West Hill	136	338	1,993607468	10.
6	-79.207041	43.782399	Morningside	135	235	1,854249005	
7	-79.467872	43.645065	High Park-Swansea	87	252	1,832356973	
8	-79.378904	43.731013	Bridle Path-Sunnybrook-York Mills	41	85	1,732820954	
9	-79.235530	43.721121	Cliffcrest	123	238	1,432132979	

3. Tree Cover top neighbourhoods

In [16]:
```
data = data.sort_values('Tree Cover', ascending=False).reset_index(
drop=True)
data['Rating 3'] = data.index
data.head(10)
```

Out[16]:

	LONGITUDE	LATITUDE	Neighbourhood	Neighbourhood Id	Green Rebate Programs	Green Spaces	P Carcir TEI
0	-79.548983	43.721519	Elms-Old Rexdale	5	73	0,730089694	
1	-79.437409	43.720345	Englemount-Lawrence	32	379	0,20662549	
2	-79.408007	43.681852	Casa Loma	96	80	0,173001423	
3	-79.334948	43.786982	Pleasant View	46	301	0,155828919	
4	-79.485589	43.701326	Brookhaven-Amesbury	30	113	0,405956959	
5	-79.260382	43.725556	Kennedy Park	124	200	0,119007268	
6	-79.515723	43.702716	Weston	113	94	0,240153499	
7	-79.480758	43.715574	Maple Leaf	29	82	0,167833165	
8	-79.496045	43.657420	Lambton Baby Point	114	80	0,440985674	
9	-79.404001	43.671585	Annex	95	235	0,11249685	

To find top neighbourhoods for all three factors we need to sum all the ratings.

In [17]:
```
data['Rating'] = data[['Rating 1', 'Rating 2', 'Rating 3']].sum(axi
s=1)
```

In [18]:
```
data.head(10)
```

Out[18]:

	LONGITUDE	LATITUDE	Neighbourhood	Neighbourhood Id	Green Rebate Programs	Green Spaces	P Carcir TEI
0	-79.548983	43.721519	Elms-Old Rexdale	5	73	0,730089694	
1	-79.437409	43.720345	Englemount-Lawrence	32	379	0,20662549	
2	-79.408007	43.681852	Casa Loma	96	80	0,173001423	
3	-79.334948	43.786982	Pleasant View	46	301	0,155828919	
4	-79.485589	43.701326	Brookhaven-Amesbury	30	113	0,405956959	
5	-79.260382	43.725556	Kennedy Park	124	200	0,119007268	
6	-79.515723	43.702716	Weston	113	94	0,240153499	
7	-79.480758	43.715574	Maple Leaf	29	82	0,167833165	
8	-79.496045	43.657420	Lambton Baby Point	114	80	0,440985674	
9	-79.404001	43.671585	Annex	95	235	0,11249685	

Since the rating was given in ascending order we need to find neighbourhoods with lowest values of Rating.

Data printed below shows 10 best neighbourhoods considering environmental factors.

In [19]:
```
top_neighbourhoods = data[['Neighbourhood', 'Neighbourhood Id', 'Rating']].sort_values('Rating', ascending = True).reset_index(drop=True)
top_neighbourhoods.head(10)
```

Out[19]:

	Neighbourhood	Neighbourhood Id	Rating
0	West Humber-Clairville	1	71
1	West Hill	136	72
2	Banbury-Don Mills	42	79
3	Woburn	137	81
4	Englemount-Lawrence	32	98
5	Eringate-Centennial-West Deane	11	99
6	Parkwoods-Donalda	45	101
7	High Park-Swansea	87	104
8	South Riverdale	70	113
9	Islington-City Centre West	14	116

Toronto venues dataset *toronto_venues*

Displaying most popular venues category

In [20]:
```
venue1 = toronto_venues.groupby(['Venue Category']).count().reset_i
ndex()
venue1.rename(columns={'Venue': 'Number of venues'}, inplace=True)
venue1[['Venue Category', 'Number of venues']].sort_values(by=['Num
ber of venues'], ascending = False).reset_index(drop=True).head(10)
```

Out[20]:

	Venue Category	Number of venues
0	Coffee Shop	155
1	Park	84
2	Pizza Place	83
3	Café	83
4	Sandwich Place	62
5	Italian Restaurant	53
6	Fast Food Restaurant	44
7	Bar	43
8	Bakery	43
9	Restaurant	40

Displaying top neighbourhoods sorted by the number of venues. Rating4 is the number indicating standing of neighbourhood in this category.

In [21]:
```
venue2 = toronto_venues.groupby(['Neighbourhood']).count().reset_in
dex()

venue2.rename(columns={'Venue': 'Number of venues'}, inplace=True)

venue2 = venue2[['Neighbourhood', 'Number of venues']].sort_values(
by=['Number of venues'], ascending = False).reset_index(drop=True)

venue2['Rating4']=venue2.index
venue2.head(10)
```

Out[21]:

	Neighbourhood	Number of venues	Rating4
0	Church-Yonge Corridor	100	0
1	Bay Street Corridor	95	1
2	Kensington-Chinatown	94	2
3	Mount Pleasant West	65	3
4	Junction Area	61	4
5	Dufferin Grove	60	5
6	Yonge-St.Clair	56	6
7	The Beaches	55	7
8	Playter Estates-Danforth	52	8
9	Lawrence Park North	50	9

For environmental factors we calculated rating of each neighbourhood. Now this is time to take 'Rating4' into account by summing this 2 columns together.

In [23]:
```
data2 = pd.merge(top_neighbourhoods, venue2)
data2.head()
```

Out[23]:

	Neighbourhood	Neighbourhood Id	Rating	Number of venues	Rating4
0	West Humber-Clairville	1	71	17	36
1	West Hill	136	72	4	96
2	Banbury-Don Mills	42	79	23	26
3	Woburn	137	81	7	69
4	Englemount-Lawrence	32	98	8	67

In [24]:
```
data2['Rating'] = data2[['Rating', 'Rating4']].sum(axis=1)
data2=data2[['Neighbourhood', 'Rating']].sort_values('Rating').rese
t_index(drop=True)
data2.head(10)
```

Out[24]:

	Neighbourhood	Rating
0	Banbury-Don Mills	105
1	West Humber-Clairville	107
2	Rouge	136
3	High Park-Swansea	146
4	Woburn	150
5	Islington-City Centre West	156
6	Yorkdale-Glen Park	163
7	Englemount-Lawrence	165
8	Pleasant View	167
9	West Hill	168

Clustering

To check different neighbourhoods in Toronto we will perform clustering.

In [78]:
```
cluster_df=pd.merge(data, venue2)
cluster_df.head()
```

Out[78]:

	LONGITUDE	LATITUDE	Neighbourhood	Neighbourhood Id	Green Rebate Programs	Green Spaces	P Carcir TEI
0	-79.548983	43.721519	Elms-Old Rexdale	5	73	0,730089694	
1	-79.437409	43.720345	Englemount-Lawrence	32	379	0,20662549	
2	-79.408007	43.681852	Casa Loma	96	80	0,173001423	
3	-79.334948	43.786982	Pleasant View	46	301	0,155828919	
4	-79.485589	43.701326	Brookhaven-Amesbury	30	113	0,405956959	

In [57]:
```
toronto_clustering=cluster_df[['Rating 1', 'Rating 2', 'Rating 3',
'Rating4']].reset_index(drop=True)
from sklearn.cluster import KMeans
# set number of clusters
k = 10

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

# check cluster labels generated for each row in the dataframe
kmeans.labels_[0:10]
```

Out[57]:
```
array([2, 3, 9, 3, 2, 3, 9, 2, 2, 3], dtype=int32)
```

In [79]:
```
cluster_df.insert(0, 'Cluster Labels', kmeans.labels_)

cluster_df.head()
```

Out[79]:

	Cluster Labels	LONGITUDE	LATITUDE	Neighbourhood	Neighbourhood Id	Green Rebate Programs	Gree Space
0	2	-79.548983	43.721519	Elms-Old Rexdale	5	73	0,73008969
1	3	-79.437409	43.720345	Englemount-Lawrence	32	379	0,2066254
2	9	-79.408007	43.681852	Casa Loma	96	80	0,17300142
3	3	-79.334948	43.786982	Pleasant View	46	301	0,15582891
4	2	-79.485589	43.701326	Brookhaven-Amesbury	30	113	0,40595695

Instaling Folium package for map visialization

In [89]:
```
!conda install -c conda-forge folium=0.5.0 --yes
import folium # map rendering library

print('Libraries imported.')
```

```
Solving environment: done


==> WARNING: A newer version of conda exists. <==
  current version: 4.5.12
  latest version: 4.7.12

Please update conda by running

    $ conda update -n base -c defaults conda
```

```
## Package Plan ##

  environment location: /anaconda3

  added / updated specs:
    - folium=0.5.0


The following packages will be downloaded:

    package                    |            build
    ---------------------------|-----------------
    conda-4.7.12               |           py37_0         3.0 MB
conda-forge
    conda-package-handling-1.6.0|   py37h01d97ff_0         1.4 MB
conda-forge
    branca-0.3.1               |             py_0          25 KB
conda-forge
    folium-0.5.0               |             py_0          45 KB
conda-forge
    vincent-0.4.4              |             py_1          28 KB
conda-forge
    altair-3.2.0               |           py37_0         773 KB
conda-forge
    ------------------------------------------------------------
                                           Total:         5.3 MB

The following NEW packages will be INSTALLED:

    altair:                 3.2.0-py37_0         conda-forge
    branca:                 0.3.1-py_0           conda-forge
    conda-package-handling: 1.6.0-py37h01d97ff_0 conda-forge
    folium:                 0.5.0-py_0           conda-forge
    vincent:                0.4.4-py_1           conda-forge

The following packages will be UPDATED:

    conda:                  4.5.12-py37_0                    --> 4
.7.12-py37_0 conda-forge


Downloading and Extracting Packages
conda-4.7.12         | 3.0 MB    | ###############################
###### | 100%
conda-package-handli | 1.4 MB    | ###############################
###### | 100%
branca-0.3.1         | 25 KB     | ###############################
###### | 100%
folium-0.5.0         | 45 KB     | ###############################
###### | 100%
vincent-0.4.4        | 28 KB     | ###############################
###### | 100%
altair-3.2.0         | 773 KB    | ###############################
###### | 100%
Preparing transaction: done
Verifying transaction: done
```

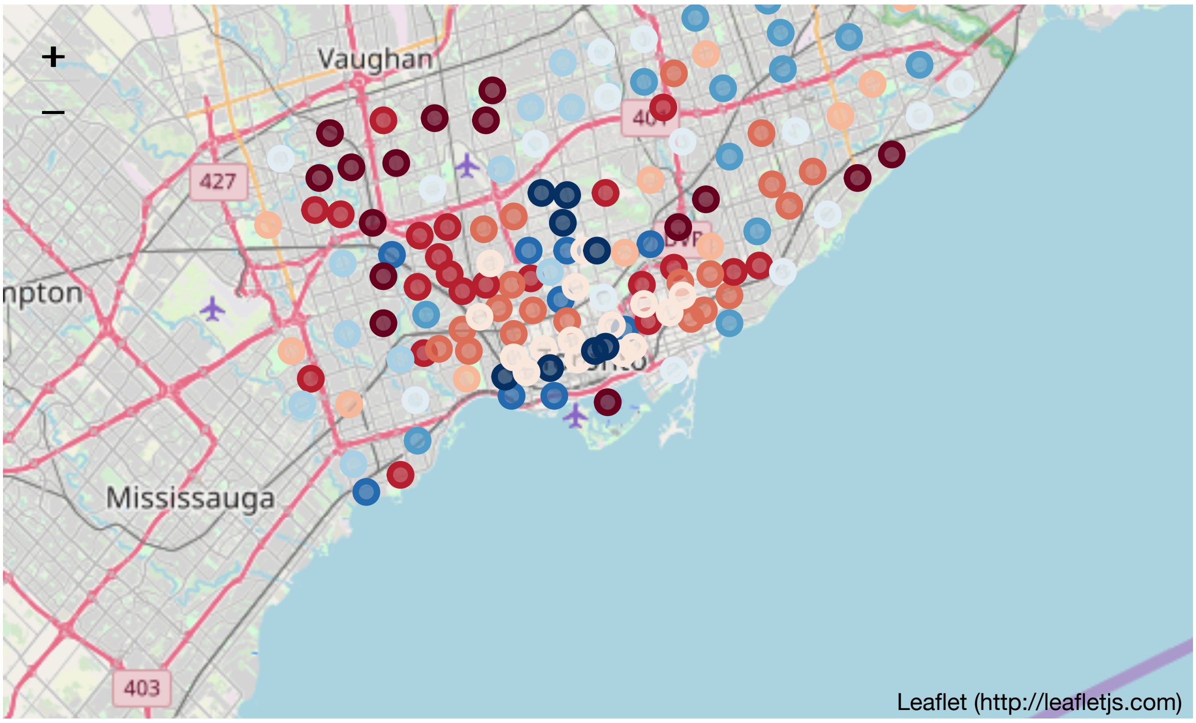
```
Executing transaction: done
Libraries imported.
```

4. Results

In [96]:

```python
import numpy as np
import matplotlib.cm as cm
import matplotlib.colors as colors

# create map
latitude =  43.653908
longitude = -79.384293
map_clusters = folium.Map(location=[latitude, longitude], zoom_start=10)

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

# add markers to the map
markers_colors = []
for lat, lon, poi, cluster in zip(cluster_df['LATITUDE'], cluster_df['LONGITUDE'], cluster_df['Neighbourhood'], cluster_df['Cluster Labels']):
    label = folium.Popup(str(poi) + ' Cluster ' + str(cluster), parse_html=True)
    folium.CircleMarker(
        [lat, lon],
        radius=5,
        popup=label,
        color=RdBu[cluster-1],
        fill=True,
        fill_color=RdBu[cluster-1],
        fill_opacity=0.7).add_to(map_clusters)

map_clusters
```

Out[96]:



Comparing clustering results with top neighbourhoods ordered by rating.

In [98]:

```
top=cluster_df[['Neighbourhood', 'Rating', 'Cluster Labels']].sort_
values('Rating').reset_index(drop=True)
top.head(20)
```

Out[98]:

	Neighbourhood	Rating	Cluster Labels
0	West Humber-Clairville	71	4
1	West Hill	72	6
2	Banbury-Don Mills	79	4
3	Woburn	81	4
4	Englemount-Lawrence	98	3
5	Eringate-Centennial-West Deane	99	4
6	Parkwoods-Donalda	101	6
7	High Park-Swansea	104	4
8	South Riverdale	113	6
9	Islington-City Centre West	116	4
10	Cliffcrest	119	6
11	L'Amoreaux	122	4
12	Stonegate-Queensway	123	6
13	Rouge	124	4
14	Bendale	124	6
15	Pleasant View	126	3
16	Morningside	127	4
17	Dovercourt-Wallace Emerson-Juncti	131	3
18	Yorkdale-Glen Park	132	3
19	Downsview-Roding-CFB	137	6

As shown above top neighbourhoods are only clusters labeled 3, 4 and 6.

We can also check the worst neighbourhoods (which have higher rating)

In [99]:
```
worst=cluster_df[['Neighbourhood', 'Rating', 'Cluster Labels']].sort_values('Rating', ascending=False).reset_index(drop=True)
worst.head(20)
```

Out[99]:

	Neighbourhood	Rating	Cluster Labels
0	Regent Park	395	5
1	Little Portugal	363	5
2	University	348	5
3	Forest Hill South	344	7
4	Clanton Park	338	7
5	Weston-Pellam Park	336	5
6	Church-Yonge Corridor	327	0
7	Kensington-Chinatown	326	5
8	Moss Park	323	5
9	Briar Hill-Belgravia	318	5
10	Playter Estates-Danforth	307	5
11	Bay Street Corridor	306	0
12	Yonge-St.Clair	303	5
13	Blake-Jones	298	5
14	North St.James Town	297	5
15	Lawrence Park North	293	0
16	Humber Summit	292	1
17	Scarborough Village	290	1
18	Dufferin Grove	287	5
19	Willowdale West	286	7

Worst neighbourhoods have labels 0, 1, 5 and 7.

5. Discussion

Best neighbourhoods Toronto based on performed analysis are:

- West Humber-Clairville
- West Hill
- Banbury-Don Mills
- Woburn
- Englemount-Lawrence

These results show neighbourhoods in Toronto as combination of districts that have highest environmental factors and number of venues.

6. Conclusion

Choosing best place to live or to buy property as investment is not always easy task. There are common factors like price, number of rooms, time of getting to work that are always important. But sometimes we have many options, especially in big city like Toronto, so other factors like venues nearby or verdancy can help make decicions.

```
In [ ]:
```