Coursera_DataScientist_Course9_Week3_Assignment_HSD_20190612 (Part 3 of 3)

COMMENTS: In this notebook I have documented the objective and the steps indicated in the assignment in order to follow step by step the instructions and keep track of the activities step by step

IMPORTANT: This is notebook HSD_Week3_Part_3_of_3. This is Coursera Capstone Project - Exercise for Week 3. I use the same file used by Part 1 of 3, Part 2 of 3, ad now Part 3 of 3. Therefore, this is an incremental file containing all the contents. For This part (3 of 3) go to the bottom of the Notebook

========== PART 1 OF 3 ==========

OBJECTIVE OF THE ACTIVITY: Explore and cluster the neighborhoods in Toronto.

STEP 1: Create a new Notebook for this assignment. ---> STATUS: step completed. It is this notebook

STEP 2: Build the code to scrape the Wikipedia page, https://en.wikipedia.org/wiki/List_of_postal_codes_of_Canada:_M, -> STATUS: completed. See the code below

In [2]:
```
#import the libraries
import numpy as np    # import library to managed daa in vectorized way
import pandas as pd   # library for data analysis
import requests       # library for requests
```

In [3]:
```
#Load the html file from internet to a local file for later processing with Beautiful Soup Library
```

Segmenting and Clustering Neig | HSD_Week3_Part_3_of_3 - IBM W | hectorsilvadiniz/Coursera_Capsto

https://jp-tok.dataplatform.cloud.ibm.com/analytics/notebooks/v2/0a15442b-0f6d-4477-a0e8-8426d...

IBM Watson Studio | Upgrade | Hector Silva Diniz's Account

My Projects / Coursera_DataScience_8_Machin... / HSD_Week3_Part_3_of_3

File Edit View Insert Cell Kernel Help | Trusted | Python 3.5

Run | Format Code

```
import pandas as pd  # library for data analysis
import requests      # library for requests
```

In [3]:
```
#load the html file from internet to a local file for later processing with Beautiful Soup library
!wget -q -O 'file_with_input_info' https://en.wikipedia.org/wiki/List_of_postal_codes_of_Canada:_M
```

In [4]:
```
#import the library that will help to get the contents from html file
from bs4 import BeautifulSoup
```

In [5]:
```
#read the content of the html file using de BS library
with open("file_with_input_info") as html_file:
    soup = BeautifulSoup(html_file,'lxml')
#print(soup.prettify())     #this line is commented. I activated when testing the code in order to have the look and feel of the content
```

In [6]:
```
#store the table in a variable in order to scan contents at level table / body / line / field
my_table = soup.find('table')
#print(my_table) #this line is commented. I activated when testing the code in order to have the look and feel of the content
my_body = my_table.tbody
#print(my_body) #this line is commented. I activated when testing the code in order to have the look and feel of the content
```

In [7]:
```
# With this loop the table is recognized as a whole
#Create an empty dataframe in order to store the information taht will be read from input source
my_df = pd.DataFrame(columns=('PostalCode','Borough','Neighborhood'))
# Need special treatment (try & except) to isolate records with <th> labels from <td> labels that are the ones taht have the content
my_counter_of_records = 0
my_fields_content = [' ',' ',' ']
for my_table_record in my_table.tbody:
    my_list = ""
    my_ok_to_save = False
    try:
        my_field_index = 0
        for my_field in my_table_record.find_all('td'):
            my_fields_content[my_field_index] = my_field.text
```

Type here to search | ESP | 11:38 PM 6/16/2019

```
my_ok_to_save = False
try:
    my_field_index = 0
    for my_field in my_table_record.find_all('td'):
        my_fields_content[my_field_index] = my_field.text
        if my_field_index == 2:
            my_fields_content[my_field_index] = my_fields_content[my_field_index][:-1]    #take out "\n" char at the end of line (in 3rd field)
        my_list = my_list + "," + my_field.text
        my_field_index = my_field_index + 1
        my_ok_to_save = True
except:
    my_counter_of_records = my_counter_of_records + 0   # counter remains the same. this is for making it explicit
if my_ok_to_save == True:
    # print(my_counter_of_records, my_list)             #line only used when testing the code
    my_df.loc[len(my_df)] = my_fields_content           #inserts fields in the record of the dataframe
    my_counter_of_records = my_counter_of_records + 1   #increments the record number for next iteration
```

In [8]:
```
my_df.shape
```

Out[8]: (288, 3)

In [9]:
```
my_df.head(5)
```

Out[9]:

	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

3	M4A	North York	Victoria Village
4	M5A	Downtown Toronto	Harbourfront

STEP 3: Create the dataframe

Step 3.1 The dataframe will consist of three columns: PostalCode, Borough, and Neighborhood --> STATUS DONE. It was done at the ned of step 2

Step 3.2 Only process the cells that have an assigned borough. Ignore cells with a borough that is Not assigned. --> STATUS DONE. See code below

In [10]:
```
print(my_df.shape)
```

```
(288, 3)
```

In [11]:
```
my_df2 = pd.DataFrame(columns=('PostalCode','Borough','Neighborhood'))
my_df2 = my_df[my_df['Borough']!='Not assigned']     #copy data in the new df only lines with Borough different from 'Not assigned' value
my_df2.reset_index(drop=True,inplace=True)
```

In [12]:
```
print(my_df2.shape)
```

```
(211, 3)
```

Step 3.3 More than one neighborhood can exist in one postal code area. For example, in the table on the Wikipedia page, you will notice that M5A is listed twice and has two neighborhoods: Harbourfront and Regent Park. These two rows will be combined into one row with the neighborhoods separated with a comma --> STATUS DONE. See code below

In [13]:
```
#just checking the example mentioned in the exercise instructions..  It is true, there are more than 1 Neighborhood per PostaCode
print(my_df2[my_df2['PostalCode']=='M5A'])
```

```
  PostalCode           Borough  Neighborhood
```

Segmenting and Clustering Neig | HSD_Week3_Part_3_of_3 - IBM W | hectorsilvadiniz/Coursera_Capsto

https://jp-tok.dataplatform.cloud.ibm.com/analytics/notebooks/v2/0a15442b-0f6d-4477-a0e8-8426d...

IBM Watson Studio | Upgrade | Hector Silva Diniz's Account

My Projects / Coursera_DataScience_8_Machin... / HSD_Week3_Part_3_of_3

File Edit View Insert Cell Kernel Help | Trusted | Python 3.5

Run | Format Code

In [13]:
```
#just checking the example mentioned in the exercise instructions..  It is true, there are more than 1 Neighborhood per PostaCode
print(my_df2[my_df2['PostalCode']=='M5A'])
```

```
  PostalCode           Borough  Neighborhood
2        M5A  Downtown Toronto  Harbourfront
3        M5A  Downtown Toronto   Regent Park
```

In [14]:
```
#Now we will set up a process to combine the Neighborhood names that are associated with the same PostCode
my_df3 = pd.DataFrame(columns=('PostalCode','Borough','Neighborhood'))
for my_i in range(0,len(my_df2)):
    my_df2_postal_code = my_df2.loc[my_i][0]          #capture the postalcode from the original dataframe
    postalcode_is_new = True                          #assumes it will be a new code in the destination dataframe
    for my_j in range (0,len(my_df3)):                #loop to determine if PostalCode is new in destination dataframe
        my_df3_postal_code = my_df3.loc[my_j][0]
        my_df3_neighborhood = my_df3.loc[my_j][2]
        if my_df2_postal_code == my_df3_postal_code:
            postalcode_is_new = False
            my_df3_INDEX = my_j                       #saves the reference to the line in the destination dataframe, the one that matched
    if postalcode_is_new == True:                     #if it is a new item, it is added into the destination dataframe
        my_df3_postal_code =  my_df2.loc[my_i][0]
        my_df3_borough =  my_df2.loc[my_i][1]
        my_df3_neighborhood =  my_df2.loc[my_i][2]
        my_df3.loc[len(my_df3)] = [my_df3_postal_code, my_df3_borough, my_df3_neighborhood]
    else:                                             #it it is not a new item, checks it combines the content of neighborhood
        my_df3_neighborhood = my_df2.loc[my_i][2] + ", " +  my_df3_neighborhood
        my_df3.loc[my_df3_INDEX][2] = my_df3_neighborhood
```

In [15]:
```
print(my_df3.shape)
```

```
(103, 3)
```

In [16]:
```
#just re-checking the example mentioned in the exercise instructions..  It is true, there are more than 1 Neighborhood per PostaCode
print(my_df3[my_df3['PostalCode']=='M5A'])
```

Type here to search | ESP | 11:38 PM 6/16/2019

Segmenting and Clustering Neig | HSD_Week3_Part_3_of_3 - IBM W | hectorsilvadiniz/Coursera_Capsto

https://jp-tok.dataplatform.cloud.ibm.com/analytics/notebooks/v2/0a15442b-0f6d-4477-a0e8-8426d...

IBM Watson Studio | Upgrade | Hector Silva Diniz's Account

My Projects / Coursera_DataScience_8_Machin... / HSD_Week3_Part_3_of_3

File Edit View Insert Cell Kernel Help | Trusted | Python 3.5

Run | Format Code

In [16]:
```
#just re-checking the example mentioned in the exercise instructions..  It is true, there are more than 1 Neighborhood per PostaCode
print(my_df3[my_df3['PostalCode']=='M5A'])
```

```
  PostalCode           Borough              Neighborhood
2        M5A  Downtown Toronto  Regent Park, Harbourfront
```

In [17]:
```
my_df3.head()
```

Out[17]:

	PostalCode	Borough	Neighborhood
0	M3A	North York	Parkwoods
1	M4A	North York	Victoria Village
2	M5A	Downtown Toronto	Regent Park, Harbourfront
3	M6A	North York	Lawrence Manor, Lawrence Heights
4	M7A	Queen's Park	Not assigned

Step 3.4 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. --> STATUS DONE. See code below

In [18]:
```
#Check the cases indicated as cases to be fixed
for my_i in range(0,len(my_df3)):
    my_df3_neighborhood = my_df3.loc[my_i][2]         #capture the neighborhood
    if my_df3_neighborhood == 'Not assigned':
        my_df3_postal_code =  my_df3.loc[my_i][0]
        my_df3_borough =  my_df3.loc[my_i][1]
        my_df3_neighborhood =  my_df3.loc[my_i][2]
        print("record to be transformed: ", my_df3_postal_code," , ", my_df3_borough, " , ", my_df3_neighborhood)
```

```
record to be transformed:  M7A  ,  Queen's Park  ,  Not assigned
```

Type here to search | ESP | 11:38 PM 6/16/2019

Segmenting and Clustering Neig | HSD_Week3_Part_3_of_3 - IBM W | hectorsilvadiniz/Coursera_Capsto

https://jp-tok.dataplatform.cloud.ibm.com/analytics/notebooks/v2/0a15442b-0f6d-4477-a0e8-8426d...

IBM Watson Studio | Upgrade | Hector Silva Diniz's Account

My Projects / Coursera_DataScience_8_Machin... / HSD_Week3_Part_3_of_3

File Edit View Insert Cell Kernel Help | Trusted | Python 3.5

Run | Format Code

```
        my_df3_borough =  my_df3.loc[my_i][1]
        my_df3_neighborhood =  my_df3.loc[my_i][2]
        print("record to be transformed: ", my_df3_postal_code," , ", my_df3_borough, " , ", my_df3_neighborhood)
```

```
record to be transformed:  M7A  ,  Queen's Park  ,  Not assigned
```

In [19]:
```
#Transform the record..
for my_i in range(0,len(my_df3)):
    my_df3_neighborhood = my_df3.loc[my_i][2]         #capture the neighborhood
    if my_df3_neighborhood == 'Not assigned':
        my_df3_postal_code =  my_df3.loc[my_i][0]
        my_df3_borough =  my_df3.loc[my_i][1]
        my_df3_neighborhood =  my_df3.loc[my_i][2]
        my_df3.loc[my_i][2] = my_df3.loc[my_i][1]     #transform the record
```

In [20]:
```
#re-Check the cases indicated as cases to be fixed
for my_i in range(0,len(my_df3)):
    my_df3_neighborhood = my_df3.loc[my_i][2]         #capture the neighborhood
    if my_df3_neighborhood == 'Not assigned':
        my_df3_postal_code =  my_df3.loc[my_i][0]
        my_df3_borough =  my_df3.loc[my_i][1]
        my_df3_neighborhood =  my_df3.loc[my_i][2]
        print("record to be transformed: ", my_df3_postal_code," , ", my_df3_borough, " , ", my_df3_neighborhood)
```

Step 3.5 Clean your Notebook and add Markdown cells to explain your work and any assumptions you are making. --> STATUS: DONE. See code above. I have added comments

Ste 3.6 In the last cell of your notebook, use the .shape method to print the number of rows of your dataframe. --> STATUS: DONE. See code below

In [21]:
```
my_df3.shape
```

Out[21]: (103, 3)

Type here to search | ESP 11:38 PM 6/16/2019

In [21]:
```
my_df3.shape
```

Out[21]: (103, 3)

In [22]:
```
my_df3.head(103)
```

Out[22]:

	PostalCode	Borough	Neighborhood
0	M3A	North York	Parkwoods
1	M4A	North York	Victoria Village
2	M5A	Downtown Toronto	Regent Park, Harbourfront
3	M6A	North York	Lawrence Manor, Lawrence Heights
4	M7A	Queen's Park	Queen's Park
5	M9A	Etobicoke	Islington Avenue
6	M1B	Scarborough	Malvern, Rouge
7	M3B	North York	Don Mills North
8	M4B	East York	Parkview Hill, Woodbine Gardens
9	M5B	Downtown Toronto	Garden District, Ryerson
10	M6B	North York	Glencairn
11	M9B	Etobicoke	West Deane Park, Princess Gardens, Martin Grov...
12	M1C	Scarborough	Port Union, Rouge Hill, Highland Creek
13	M3C	North York	Don Mills South, Flemingdon Park
14	M4C	East York	Woodbine Heights
15	M5C	Downtown Toronto	St. James Town
16	M6C	York	Humewood-Cedarvale

14	M4C	East York	Woodbine Heights
15	M5C	Downtown Toronto	St. James Town
16	M6C	York	Humewood-Cedarvale
17	M9C	Etobicoke	Old Burnhamthorpe, Markland Wood, Eringate, Bl...
18	M1E	Scarborough	West Hill, Morningside, Guildwood
19	M4E	East Toronto	The Beaches
20	M5E	Downtown Toronto	Berczy Park
21	M6E	York	Caledonia-Fairbanks
22	M1G	Scarborough	Woburn
23	M4G	East York	Leaside
24	M5G	Downtown Toronto	Central Bay Street
25	M6G	Downtown Toronto	Christie
26	M1H	Scarborough	Cedarbrae
27	M2H	North York	Hillcrest Village
28	M3H	North York	Wilson Heights, Downsview North, Bathurst Manor
29	M4H	East York	Thorncliffe Park
...	...	...	...
73	M4R	Central Toronto	North Toronto West
74	M5R	Central Toronto	Yorkville, North Midtown, The Annex
75	M6R	West Toronto	Roncesvalles, Parkdale
76	M7R	Mississauga	Canada Post Gateway Processing Centre
77	M9R	Etobicoke	St. Phillips, Richview Gardens, Martin Grove G...
78	M1S	Scarborough	Agincourt

76	M7R	Mississauga	Canada Post Gateway Processing Centre
77	M9R	Etobicoke	St. Phillips, Richview Gardens, Martin Grove G...
78	M1S	Scarborough	Agincourt
79	M4S	Central Toronto	Davisville
80	M5S	Downtown Toronto	University of Toronto, Harbord
81	M6S	West Toronto	Swansea, Runnymede
82	M1T	Scarborough	Tam O'Shanter, Sullivan, Clarks Corners
83	M4T	Central Toronto	Summerhill East, Moore Park
84	M5T	Downtown Toronto	Kensington Market, Grange Park, Chinatown
85	M1V	Scarborough	Steeles East, Milliken, L'Amoreaux East, Aginc...
86	M4V	Central Toronto	Summerhill West, South Hill, Rathnelly, Forest...
87	M5V	Downtown Toronto	South Niagara, Railway Lands, King and Spadina...
88	M8V	Etobicoke	New Toronto, Mimico South, Humber Bay Shores
89	M9V	Etobicoke	Thistletown, South Steeles, Silverstone, Mount...
90	M1W	Scarborough	L'Amoreaux West
91	M4W	Downtown Toronto	Rosedale
92	M5W	Downtown Toronto	Stn A PO Boxes 25 The Esplanade
93	M8W	Etobicoke	Long Branch, Alderwood
94	M9W	Etobicoke	Northwest
95	M1X	Scarborough	Upper Rouge
96	M4X	Downtown Toronto	St. James Town, Cabbagetown
97	M5X	Downtown Toronto	Underground city, First Canadian Place
98	M8X	Etobicoke	Old Mill North, Montgomery Road, The Kingsway

96	M4X	Downtown Toronto	St. James Town, Cabbagetown
97	M5X	Downtown Toronto	Underground city, First Canadian Place
98	M8X	Etobicoke	Old Mill North, Montgomery Road, The Kingsway
99	M4Y	Downtown Toronto	Church and Wellesley
100	M7Y	East Toronto	Business Reply Mail Processing Centre 969 Eastern
101	M8Y	Etobicoke	Sunnylea, Royal York South East, The Queensway...
102	M8Z	Etobicoke	South of Bloor, Royal York South West, The Que...

103 rows × 3 columns

STEP 4: Submit link to your Notebook on the Github repository --> STATUS: Done

I added the link of this notebook in the coursera instruction section

========== PART 2 OF 3 ==========

STEP A

... Geocoder Python package ... The problem with this Package is you have to be persistent sometimes in order to get the geographical coordinates of a given postal code. So you can make a call to get the latitude and longitude coordinates of a given postal code and the result would be None, and then make the call again and you would get the coordinates. So, in order to make sure that you get the coordinates for all of our neighborhoods, you can run a while loop for each postal code. Given that this package can be very unreliable, in case you are not able to get the geographical coordinates of the neighborhoods using the Geocoder package, here is a link to a csv file that has the geographical coordinates of each postal code: http://cocl.us/Geospatial_data --> STATUS: DONE. See code below

In [23]:
```
# I use the cvs file to add latirud and longitud to the dataframe created in part 1 o 3 above.
```

geographical coordinates of each postal code: http://cocl.us/Geospatial_data --> STATUS: DONE. See code below

In [23]:
```
# I use the cvs file to add latirud and longitud to the dataframe created in part 1 o 3 above.
```

In [24]:
```
my_dfLL = pd.read_csv(r'http://cocl.us/Geospatial_data')
```

In [25]:
```
my_dfLL.head()
```

Out[25]:

	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 [26]:
```
#add 2 columns to destination dataframe
my_df3['Latitude'] = " "
my_df3['Longitude'] = " "
```

In [27]:
```
#Now we will set up a process to add on the orignal dataframe the content of Long and Lat based on PostalCode
for my_i in range(0,len(my_df3)):
    my_df3_postal_code = my_df3.loc[my_i][0]          #capture the postalcode from the original dataframe
    LL_is_found = False                               #assumes there is not LL code in the LL dataframe
    for my_j in range (0,len(my_dfLL)):               #loop to determine if LL is found in LL dataframe
        my_dfLL_postal_code = my_dfLL.loc[my_j][0]
        if my_df3_postal_code == my_dfLL_postal_code:
            LL_is_found = True
            my_dfLL_INDEX = my_j                      #saves the reference to the line in the destination dataframe, the one that matched
            my_latitud = my_dfLL.loc[my_j][1]
```

```
        if my_df3_postal_code == my_dfLL_postal_code:
            LL_is_found = True
            my_dfLL_INDEX = my_j                    #saves the reference to the line in the destination dataframe, the one that matched
            my_latitud = my_dfLL.loc[my_j][1]
            my_longitud = my_dfLL.loc[my_j][2]
    if LL_is_found == True:                         #if it was found the LL, it is added into the original dataframe
        my_df3.loc[my_i][3] = my_latitud
        my_df3.loc[my_i][4] = my_longitud
```

In [28]:
```
my_df3.head(103)
```

Out[28]:

	PostalCode	Borough	Neighborhood	Latitude	Longitude
0	M3A	North York	Parkwoods	43.7533	-79.3297
1	M4A	North York	Victoria Village	43.7259	-79.3156
2	M5A	Downtown Toronto	Regent Park, Harbourfront	43.6543	-79.3606
3	M6A	North York	Lawrence Manor, Lawrence Heights	43.7185	-79.4648
4	M7A	Queen's Park	Queen's Park	43.6623	-79.3895
5	M9A	Etobicoke	Islington Avenue	43.6679	-79.5322
6	M1B	Scarborough	Malvern, Rouge	43.8067	-79.1944
7	M3B	North York	Don Mills North	43.7459	-79.3522
8	M4B	East York	Parkview Hill, Woodbine Gardens	43.7064	-79.3099
9	M5B	Downtown Toronto	Garden District, Ryerson	43.6572	-79.3789
10	M6B	North York	Glencairn	43.7096	-79.4451
11	M9B	Etobicoke	West Deane Park, Princess Gardens, Martin Grov...	43.6509	-79.5547
12	M1C	Scarborough	Port Union, Rouge Hill, Highland Creek	43.7845	-79.1605
13	M3C	North York	Don Mills South, Flemingdon Park	43.7259	-79.3409

11	M9B	Etobicoke	West Deane Park, Princess Gardens, Martin Grov...	43.6509	-79.5547
12	M1C	Scarborough	Port Union, Rouge Hill, Highland Creek	43.7845	-79.1605
13	M3C	North York	Don Mills South, Flemingdon Park	43.7259	-79.3409
14	M4C	East York	Woodbine Heights	43.6953	-79.3184
15	M5C	Downtown Toronto	St. James Town	43.6515	-79.3754
16	M6C	York	Humewood-Cedarvale	43.6938	-79.4282
17	M9C	Etobicoke	Old Burnhamthorpe, Markland Wood, Eringate, Bl...	43.6435	-79.5772
18	M1E	Scarborough	West Hill, Morningside, Guildwood	43.7636	-79.1887
19	M4E	East Toronto	The Beaches	43.6764	-79.293
20	M5E	Downtown Toronto	Berczy Park	43.6448	-79.3733
21	M6E	York	Caledonia-Fairbanks	43.689	-79.4535
22	M1G	Scarborough	Woburn	43.771	-79.2169
23	M4G	East York	Leaside	43.7091	-79.3635
24	M5G	Downtown Toronto	Central Bay Street	43.658	-79.3874
25	M6G	Downtown Toronto	Christie	43.6695	-79.4226
26	M1H	Scarborough	Cedarbrae	43.7731	-79.2395
27	M2H	North York	Hillcrest Village	43.8038	-79.3635
28	M3H	North York	Wilson Heights, Downsview North, Bathurst Manor	43.7543	-79.4423
29	M4H	East York	Thorncliffe Park	43.7054	-79.3494
...	...	...	...	...	...
73	M4R	Central Toronto	North Toronto West	43.7154	-79.4057
74	M5R	Central Toronto	Yorkville, North Midtown, The Annex	43.6727	-79.4057
75	M6R	West Toronto	Roncesvalles, Parkdale	43.649	-79.4563

73	M4R	Central Toronto	North Toronto West	43.7154	-79.4057
74	M5R	Central Toronto	Yorkville, North Midtown, The Annex	43.6727	-79.4057
75	M6R	West Toronto	Roncesvalles, Parkdale	43.649	-79.4563
76	M7R	Mississauga	Canada Post Gateway Processing Centre	43.637	-79.6158
77	M9R	Etobicoke	St. Phillips, Richview Gardens, Martin Grove G...	43.6889	-79.5547
78	M1S	Scarborough	Agincourt	43.7942	-79.262
79	M4S	Central Toronto	Davisville	43.7043	-79.3888
80	M5S	Downtown Toronto	University of Toronto, Harbord	43.6627	-79.4
81	M6S	West Toronto	Swansea, Runnymede	43.6516	-79.4844
82	M1T	Scarborough	Tam O'Shanter, Sullivan, Clarks Corners	43.7816	-79.3043
83	M4T	Central Toronto	Summerhill East, Moore Park	43.6896	-79.3832
84	M5T	Downtown Toronto	Kensington Market, Grange Park, Chinatown	43.6532	-79.4
85	M1V	Scarborough	Steeles East, Milliken, L'Amoreaux East, Aginc...	43.8153	-79.2846
86	M4V	Central Toronto	Summerhill West, South Hill, Rathnelly, Forest...	43.6864	-79.4
87	M5V	Downtown Toronto	South Niagara, Railway Lands, King and Spadina...	43.6289	-79.3944
88	M8V	Etobicoke	New Toronto, Mimico South, Humber Bay Shores	43.6056	-79.5013
89	M9V	Etobicoke	Thistletown, South Steeles, Silverstone, Mount...	43.7394	-79.5884
90	M1W	Scarborough	L'Amoreaux West	43.7995	-79.3184
91	M4W	Downtown Toronto	Rosedale	43.6796	-79.3775
92	M5W	Downtown Toronto	Stn A PO Boxes 25 The Esplanade	43.6464	-79.3748
93	M8W	Etobicoke	Long Branch, Alderwood	43.6024	-79.5435
94	M9W	Etobicoke	Northwest	43.7067	-79.5941
95	M1X	Scarborough	Upper Rouge	43.8361	-79.2056

93	M8W	Etobicoke	Long Branch, Alderwood	43.6024	-79.5435
94	M9W	Etobicoke	Northwest	43.7067	-79.5941
95	M1X	Scarborough	Upper Rouge	43.8361	-79.2056
96	M4X	Downtown Toronto	St. James Town, Cabbagetown	43.668	-79.3677
97	M5X	Downtown Toronto	Underground city, First Canadian Place	43.6484	-79.3823
98	M8X	Etobicoke	Old Mill North, Montgomery Road, The Kingsway	43.6537	-79.5069
99	M4Y	Downtown Toronto	Church and Wellesley	43.6659	-79.3832
100	M7Y	East Toronto	Business Reply Mail Processing Centre 969 Eastern	43.6627	-79.3216
101	M8Y	Etobicoke	Sunnylea, Royal York South East, The Queensway...	43.6363	-79.4985
102	M8Z	Etobicoke	South of Bloor, Royal York South West, The Que...	43.6288	-79.521

103 rows × 5 columns

Step B: Once you are able to create the above dataframe, submit a link to the new Notebook on your Github repository. (2 marks) --> STATUS: DONE.

I added the link of this notebook in the coursera instruction section

========== PART 3 OF 3 ==========

Objective: Explore and cluster the neighborhoods in Toronto. You can decide to work with only boroughs that contain the word Toronto and then replicate the same analysis we did to the New York City data. It is up to you. Just make sure:to add enough Markdown cells to explain what you decided to do and to report any observations you make. When finished, submit a link to the new Notebook on your Github repository. --> STATUS: DONE. See code below

1.DOWNLOAD AND EXPLORE DATA SET

observations you make. When finished, submit a link to the new Notebook on your Github repository. --> STATUS: DONE. See code below

1.DOWNLOAD AND EXPLORE DATA SET

In [40]:
```
print(my_df3.shape)
```

```
(103, 5)
```

In [41]:
```
my_df3.head()
```

Out[41]:

	PostalCode	Borough	Neighborhood	Latitude	Longitude
0	M3A	North York	Parkwoods	43.7533	-79.3297
1	M4A	North York	Victoria Village	43.7259	-79.3156
2	M5A	Downtown Toronto	Regent Park, Harbourfront	43.6543	-79.3606
3	M6A	North York	Lawrence Manor, Lawrence Heights	43.7185	-79.4648
4	M7A	Queen's Park	Queen's Park	43.6623	-79.3895

2.EXPLORE NEIGHBORHOODS IN TORONTO

In [39]:
```
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 completed the Foursquare API lab
```

```
import json # library to handle JSON files

!conda install -c conda-forge geopy --yes # uncomment this line if you haven't completed the Foursquare API lab
from geopy.geocoders import Nominatim # convert an address into latitude and longitude values

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

# 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

# All requested packages already installed.

Solving environment: done

## Package Plan ##

  environment location: /opt/conda/envs/DSX-Python35

  added / updated specs:
    - folium=0.5.0


The following packages will be downloaded:
```

```
  added / updated specs:
    - folium=0.5.0


The following packages will be downloaded:

    package                    |            build
    ---------------------------|-----------------
    folium-0.5.0               |           py_0          45 KB  conda-forge
    branca-0.3.1               |           py_0          25 KB  conda-forge
    vincent-0.4.4              |           py_1          28 KB  conda-forge
    altair-2.2.2               |         py35_1         462 KB  conda-forge
    ------------------------------------------------------------
                                          Total:         560 KB

The following NEW packages will be INSTALLED:

    altair:  2.2.2-py35_1 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
folium-0.5.0         | 45 KB     | ##################################### | 100%
branca-0.3.1         | 25 KB     | ##################################### | 100%
vincent-0.4.4        | 28 KB     | ##################################### | 100%
altair-2.2.2         | 462 KB    | ##################################### | 100%
Preparing transaction: done
Verifying transaction: done
Executing transaction: done
Libraries imported.
```

In [44]:
```
my_df3.head()
```

Out[44]:

Segmenting and Clustering Neig | HSD_Week3_Part_3_of_3 - IBM W | hectorsilvadiniz/Coursera_Capsto

https://jp-tok.dataplatform.cloud.ibm.com/analytics/notebooks/v2/0a15442b-0f6d-4477-a0e8-8426d...

IBM Watson Studio | Upgrade | Hector Silva Diniz's Account

My Projects / Coursera_DataScience_8_Machin... / HSD_Week3_Part_3_of_3

File Edit View Insert Cell Kernel Help | Trusted | Python 3.5

Run | Format Code

In [44]:
```
my_df3.head()
```

Out[44]:

	PostalCode	Borough	Neighborhood	Latitude	Longitude
0	M3A	North York	Parkwoods	43.7533	-79.3297
1	M4A	North York	Victoria Village	43.7259	-79.3156
2	M5A	Downtown Toronto	Regent Park, Harbourfront	43.6543	-79.3606
3	M6A	North York	Lawrence Manor, Lawrence Heights	43.7185	-79.4648
4	M7A	Queen's Park	Queen's Park	43.6623	-79.3895

In [ ]:
```
# Use geopy library to get the latitude and longitude values of Toronto
```

In [47]:
```
address = 'Toronto'
geolocator = Nominatim(user_agent="to_explorer")
location = geolocator.geocode(address)
latitude = location.latitude
longitude = location.longitude
print('The geograpical coordinate of Toronto City are {}, {}.'.format(latitude, longitude))
```

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

In [46]:
```
#### Create a map of Toronto with neighborhoods superimposed on top.
```

In [48]:
```
# 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(my_df3['Latitude'], my_df3['Longitude'], my_df3['Borough'], my_df3['Neighborhood']):
    label = '{}, {}'.format(neighborhood, borough)
```

Type here to search | ESP 11:39 PM 6/16/2019

Segmenting and Clustering Neig | HSD_Week3_Part_3_of_3 - IBM W | hectorsilvadiniz/Coursera_Capsto

https://jp-tok.dataplatform.cloud.ibm.com/analytics/notebooks/v2/0a15442b-0f6d-4477-a0e8-8426d...

IBM Watson Studio | Upgrade | Hector Silva Diniz's Account

My Projects / Coursera_DataScience_8_Machin... / HSD_Week3_Part_3_of_3

File Edit View Insert Cell Kernel Help | Trusted | Python 3.5

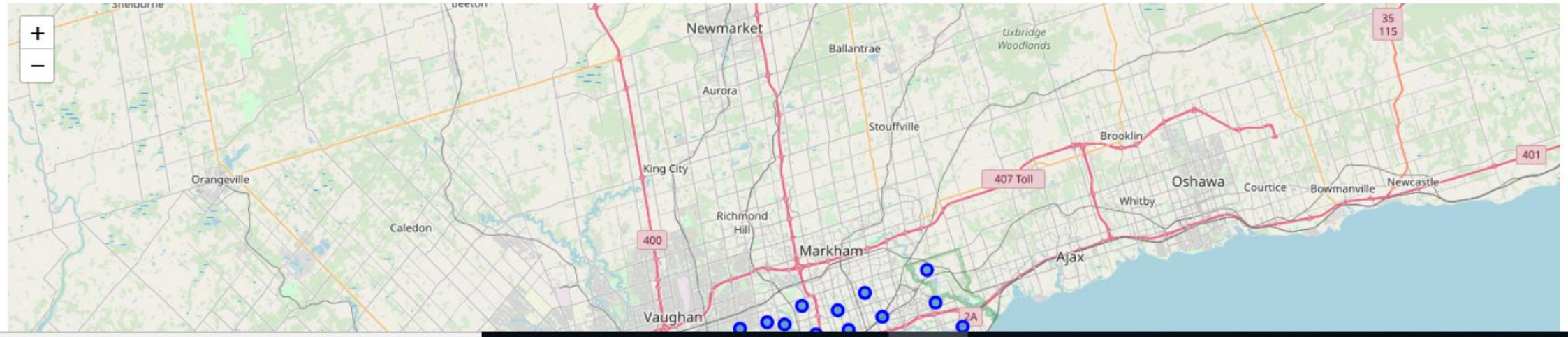
Run | Format Code

```
map_toronto = folium.Map(location=[latitude, longitude], zoom_start=10)

# add markers to map
for lat, lng, borough, neighborhood in zip(my_df3['Latitude'], my_df3['Longitude'], my_df3['Borough'], my_df3['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[48]:



Type here to search | ESP | 11:39 PM 6/16/2019

Segmenting and Clustering Neig | HSD_Week3_Part_3_of_3 - IBM W | hectorsilvadiniz/Coursera_Capsto

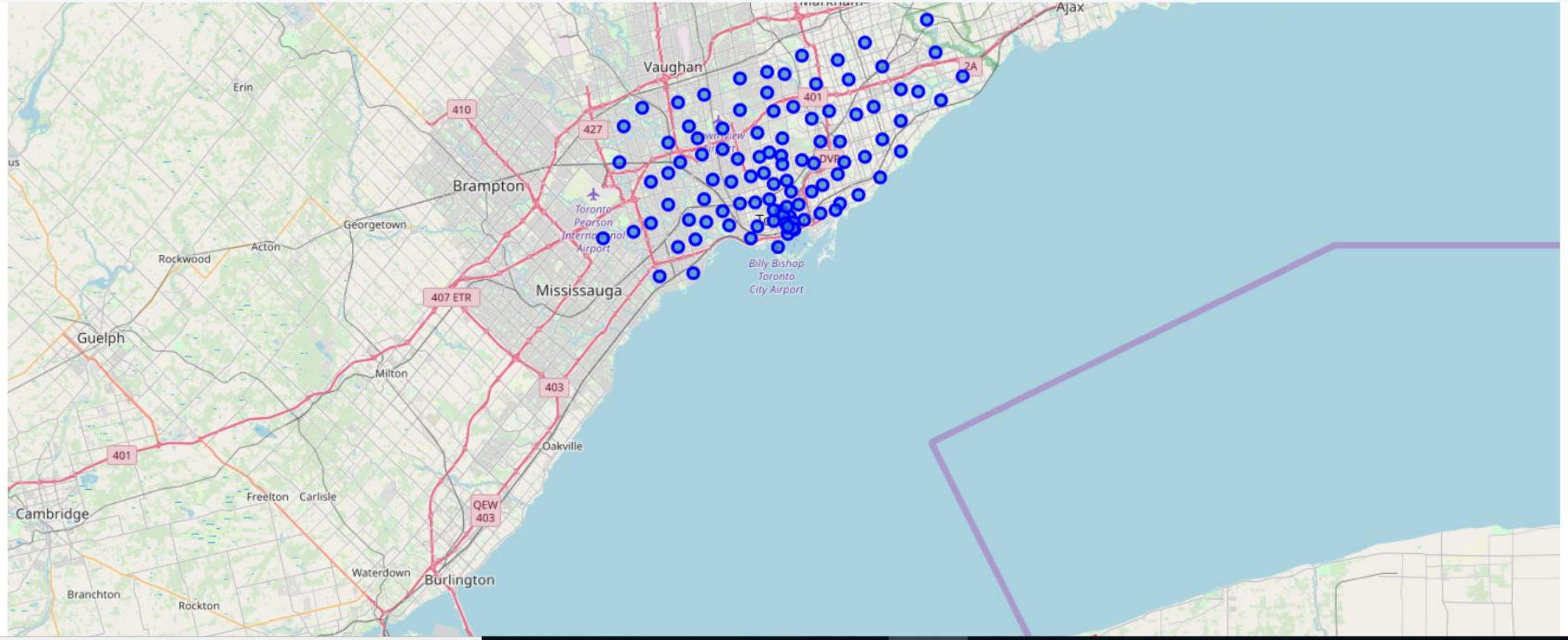
https://jp-tok.dataplatform.cloud.ibm.com/analytics/notebooks/v2/0a15442b-0f6d-4477-a0e8-8426d...

IBM Watson Studio | Upgrade | Hector Silva Diniz's Account | HS

My Projects / Coursera_DataScience_8_Machin... / HSD_Week3_Part_3_of_3

File Edit View Insert Cell Kernel Help | Trusted | Python 3.5

Run | Format Code



Type here to search | ESP | 11:39 PM 6/16/2019

Simplifying the above map and segment and cluster only the neighborhoods in Downtown Toronto...

In [49]:
```
toronto_data = my_df3[my_df3['Borough'] == 'Downtown Toronto'].reset_index(drop=True)
toronto_data.head()
```

Out[49]:

	PostalCode	Borough	Neighborhood	Latitude	Longitude
0	M5A	Downtown Toronto	Regent Park, Harbourfront	43.6543	-79.3606
1	M5B	Downtown Toronto	Garden District, Ryerson	43.6572	-79.3789
2	M5C	Downtown Toronto	St. James Town	43.6515	-79.3754
3	M5E	Downtown Toronto	Berczy Park	43.6448	-79.3733
4	M5G	Downtown Toronto	Central Bay Street	43.658	-79.3874

In [50]:
```
address = 'Downtown Toronto'
geolocator = Nominatim(user_agent="to_explorer")
location = geolocator.geocode(address)
latitude = location.latitude
longitude = location.longitude
print('The geograpical coordinate of Downtown Toronto are {}, {}.'.format(latitude, longitude))
```

```
The geograpical coordinate of Downtown Toronto are 43.6541737, -79.3808116451341.
```

In [51]:
```
# create map of Toronto using latitude and longitude values
```

```
print('The geograpical coordinate of Downtown Toronto are {}, {}.'.format(latitude, longitude))
```

```
The geograpical coordinate of Downtown Toronto are 43.6541737, -79.3808116451341.
```

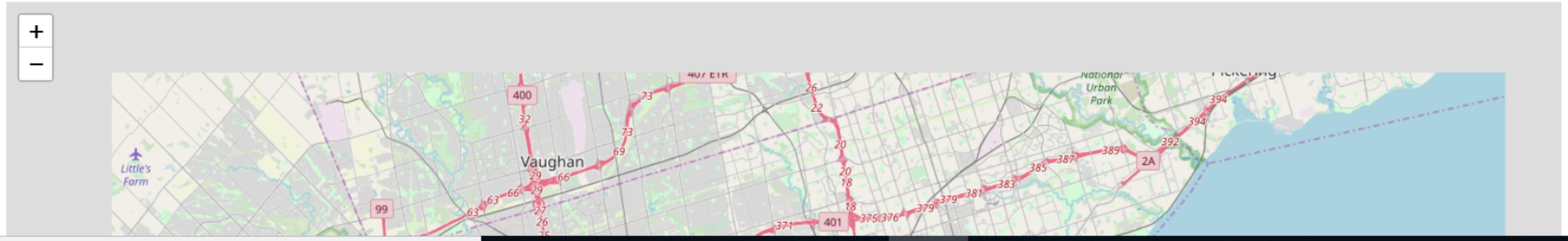
In [51]:

```
# create map of Toronto using latitude and longitude values
map_downtown_toronto = folium.Map(location=[latitude, longitude], zoom_start=11)

# add markers to map
for lat, lng, borough, neighborhood in zip(toronto_data['Latitude'], toronto_data['Longitude'], toronto_data['Borough'], toronto_data['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_downtown_toronto)

map_downtown_toronto
```

Out[51]:



Segmenting and Clustering Neig | HSD_Week3_Part_3_of_3 - IBM W | hectorsilvadiniz/Coursera_Capsto

https://jp-tok.dataplatform.cloud.ibm.com/analytics/notebooks/v2/0a15442b-0f6d-4477-a0e8-8426d...

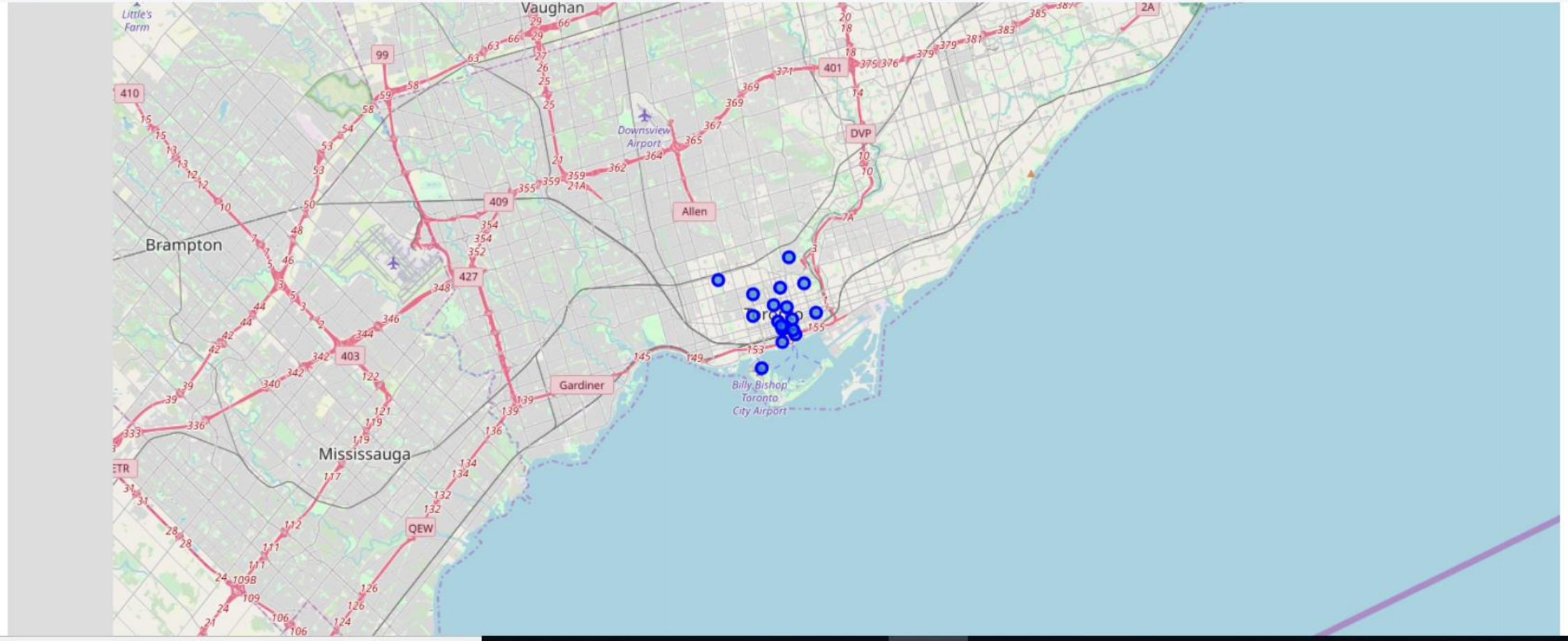
IBM Watson Studio | Upgrade | Hector Silva Diniz's Account

My Projects / Coursera_DataScience_8_Machin... / HSD_Week3_Part_3_of_3

File Edit View Insert Cell Kernel Help

Trusted | Python 3.5

Run Format Code



Type here to search

ESP 11:39 PM 6/16/2019

Define Foursquare Credentials and Version

In [91]:
```
# @hidden_cell
CLIENT_ID = '1XCXJVJL3QADGOU2MEK2NVEPWHTG1TR1GVYES4XXNAOUPTX0' # your Foursquare ID
CLIENT_SECRET = '2EDVWSHJQ1YKRNCRUZPVFGKESJYI33OTUZ1ZRCHCKIJ4UDMI' # your Foursquare Secret
VERSION = '20180605' # Foursquare API version

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

Exploring the first neighborhood in the Toronto dataframe...

In [53]:
```
toronto_data.loc[0, 'Neighborhood']
```

Out[53]: 'Regent Park, Harbourfront'

In [54]:
```
neighborhood_latitude = toronto_data.loc[0, 'Latitude'] # neighborhood latitude value
neighborhood_longitude = toronto_data.loc[0, 'Longitude'] # neighborhood longitude value
```

Segmenting and Clustering Neig | HSD_Week3_Part_3_of_3 - IBM W | hectorsilvadiniz/Coursera_Capsto

https://jp-tok.dataplatform.cloud.ibm.com/analytics/notebooks/v2/0a15442b-0f6d-4477-a0e8-8426d...

IBM Watson Studio | Upgrade | Hector Silva Diniz's Account

My Projects / Coursera_DataScience_8_Machin... / HSD_Week3_Part_3_of_3

File Edit View Insert Cell Kernel Help | Trusted | Python 3.5

Run | Format Code

In [54]:
```
neighborhood_latitude = toronto_data.loc[0, 'Latitude'] # neighborhood latitude value
neighborhood_longitude = toronto_data.loc[0, 'Longitude'] # neighborhood longitude value

neighborhood_name = toronto_data.loc[0, 'Neighborhood'] # neighborhood name

print('Latitude and longitude values of {} are {}, {}.'.format(neighborhood_name,
                                                               neighborhood_latitude,
                                                               neighborhood_longitude))
```

```
Latitude and longitude values of Regent Park, Harbourfront are 43.6542599, -79.3606359.
```

Now looking the venues...

In [55]:
```
LIMIT = 100 # limit of number of venues returned by Foursquare API
radius = 500 # define radius
url = 'https://api.foursquare.com/v2/venues/explore?&client_id={}&client_secret={}&v={}&ll={},{}&radius={}&limit={}'.format(
    CLIENT_ID,
    CLIENT_SECRET,
    VERSION,
    neighborhood_latitude,
    neighborhood_longitude,
    radius,
    LIMIT)
url # display URL
```

Out[55]:
```
'https://api.foursquare.com/v2/venues/explore?&client_id=1XCXJVJL3QADGOU2MEK2NVEPWHTG1TR1GVYES4XXNAOUPTX0&client_secret=2EDVWSHJQ1YKRNCRUZPVFGKESJYI33OTUZ1ZRCHCKIJ4UDMI&v=20180605&ll=43.6542599,-79.3606359&radius=500&limit=100'
```

In [56]:
```
results = requests.get(url).json()
results
```

Out[56]:
```
{'meta': {'code': 200, 'requestId': '5d06ddb0f19f440025fd10e0'},
```

Type here to search | ESP 11:39 PM 6/16/2019

Segmenting and Clustering Neig | HSD_Week3_Part_3_of_3 - IBM W | hectorsilvadiniz/Coursera_Capsto

https://jp-tok.dataplatform.cloud.ibm.com/analytics/notebooks/v2/0a15442b-0f6d-4477-a0e8-8426d...

IBM Watson Studio | Upgrade | Hector Silva Diniz's Account

My Projects / Coursera_DataScience_8_Machin... / HSD_Week3_Part_3_of_3

File Edit View Insert Cell Kernel Help | Trusted | Python 3.5

Run | Format Code

In [56]:
```
results = requests.get(url).json()
results
```

Out[56]:
```
{'meta': {'code': 200, 'requestId': '5d06ddb0f19f440025fd10e0'},
 'response': {'groups': [{'items': [{'reasons': {'count': 0,
       'items': [{'reasonName': 'globalInteractionReason',
         'summary': 'This spot is popular',
         'type': 'general'}]},
     'referralId': 'e-0-54ea41ad498e9a11e9e13308-0',
     'venue': {'categories': [{'icon': {'prefix': 'https://ss3.4sqi.net/img/categories_v2/food/bakery_',
         'suffix': '.png'},
        'id': '4bf58dd8d48988d16a941735',
        'name': 'Bakery',
        'pluralName': 'Bakeries',
        'primary': True,
        'shortName': 'Bakery'}],
      'id': '54ea41ad498e9a11e9e13308',
      'location': {'address': '362 King St E',
       'cc': 'CA',
       'city': 'Toronto',
       'country': 'Canada',
       'crossStreet': 'Trinity St',
       'distance': 143,
```

In [57]:
```
# function that extracts the category of the venue
def get_category_type(row):
    try:
        categories_list = row['categories']
    except:
        categories_list = row['venue.categories']

    if len(categories_list) == 0:
        return None
    else:
        return categories_list[0]['name']
```

Type here to search | ESP | 11:39 PM 6/16/2019

```
    if len(categories_list) == 0:
        return None
    else:
        return categories_list[0]['name']
```

In [58]:

```
venues = results['response']['groups'][0]['items']

nearby_venues = json_normalize(venues) # flatten JSON

# filter columns
filtered_columns = ['venue.name', 'venue.categories', 'venue.location.lat', 'venue.location.lng']
nearby_venues =nearby_venues.loc[:, filtered_columns]

# filter the category for each row
nearby_venues['venue.categories'] = nearby_venues.apply(get_category_type, axis=1)

# clean columns
nearby_venues.columns = [col.split(".")[-1] for col in nearby_venues.columns]

nearby_venues.head()
```

Out[58]:

	name	categories	lat	lng
0	Roselle Desserts	Bakery	43.653447	-79.362017
1	Tandem Coffee	Coffee Shop	43.653559	-79.361809
2	Toronto Cooper Koo Family Cherry St YMCA Centre	Gym / Fitness Center	43.653191	-79.357947
3	Body Blitz Spa East	Spa	43.654735	-79.359874
4	Morning Glory Cafe	Breakfast Spot	43.653947	-79.361149

```
Now exploring Neighborhoods in Downtown Toronto
```

```
Now exploring Neighborhoods in Downtown Toronto
```

In [60]:
```
# using a a function to repeat the same process to all the neighborhoods in Toronto
```

In [62]:
```
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',
```

Segmenting and Clustering Neig | HSD_Week3_Part_3_of_3 - IBM W | hectorsilvadiniz/Coursera_Capsto

https://jp-tok.dataplatform.cloud.ibm.com/analytics/notebooks/v2/0a15442b-0f6d-4477-a0e8-8426d...

IBM Watson Studio | Upgrade | Hector Silva Diniz's Account

My Projects / Coursera_DataScience_8_Machin... / HSD_Week3_Part_3_of_3

File Edit View Insert Cell Kernel Help | Trusted | Python 3.5

Run | Format Code

```
    nearby_venues.columns = ['Neighborhood',
                  'Neighborhood Latitude',
                  'Neighborhood Longitude',
                  'Venue',
                  'Venue Latitude',
                  'Venue Longitude',
                  'Venue Category']
    return(nearby_venues)
```

In [65]:

```
downtown_toronto_venues = getNearbyVenues(names=toronto_data['Neighborhood'],
                                          latitudes=toronto_data['Latitude'],
                                          longitudes=toronto_data['Longitude']
                                         )
```

```
Regent Park, Harbourfront
Garden District, Ryerson
St. James Town
Berczy Park
Central Bay Street
Christie
Richmond, King, Adelaide
Union Station, Toronto Islands, Harbourfront East
Toronto Dominion Centre, Design Exchange
Victoria Hotel, Commerce Court
University of Toronto, Harbord
Kensington Market, Grange Park, Chinatown
South Niagara, Railway Lands, King and Spadina, Harbourfront West, Island airport, Bathurst Quay, CN Tower
Rosedale
Stn A PO Boxes 25 The Esplanade
St. James Town, Cabbagetown
Underground city, First Canadian Place
Church and Wellesley
```

In [66]:

```
# Checking the size of the resulting dataframe
```

Type here to search | ESP | 11:39 PM 6/16/2019

Underground city, First Canadian Place
Church and Wellesley

In [66]:
```
# Checking the size of the resulting dataframe
print(downtown_toronto_venues.shape)
downtown_toronto_venues.head()
```

```
(1286, 7)
```

Out[66]:

	Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
0	Regent Park, Harbourfront	43.65426	-79.360636	Roselle Desserts	43.653447	-79.362017	Bakery
1	Regent Park, Harbourfront	43.65426	-79.360636	Tandem Coffee	43.653559	-79.361809	Coffee Shop
2	Regent Park, Harbourfront	43.65426	-79.360636	Toronto Cooper Koo Family Cherry St YMCA Centre	43.653191	-79.357947	Gym / Fitness Center
3	Regent Park, Harbourfront	43.65426	-79.360636	Body Blitz Spa East	43.654735	-79.359874	Spa
4	Regent Park, Harbourfront	43.65426	-79.360636	Morning Glory Cafe	43.653947	-79.361149	Breakfast Spot

In [67]:
```
# taking a look at the quantity of venues per neighborhood
downtown_toronto_venues.groupby('Neighborhood').count()
```

Out[67]:

Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
Berczy Park	55	55	55	55	55	55
Central Bay Street	88	88	88	88	88	88
Christie	15	15	15	15	15	15
Church and Wellesley	87	87	87	87	87	87
Garden District, Ryerson	100	100	100	100	100	100
Kensington Market, Grange Park, Chinatown	100	100	100	100	100	100

Church and Wellesley	87	87	87	87	87	87
Garden District, Ryerson	100	100	100	100	100	100
Kensington Market, Grange Park, Chinatown	100	100	100	100	100	100
Regent Park, Harbourfront	48	48	48	48	48	48
Richmond, King, Adelaide	100	100	100	100	100	100
Rosedale	4	4	4	4	4	4
South Niagara, Railway Lands, King and Spadina, Harbourfront West, Island airport, Bathurst Quay, CN Tower	15	15	15	15	15	15
St. James Town	100	100	100	100	100	100
St. James Town, Cabbagetown	46	46	46	46	46	46
Stn A PO Boxes 25 The Esplanade	94	94	94	94	94	94
Toronto Dominion Centre, Design Exchange	100	100	100	100	100	100
Underground city, First Canadian Place	100	100	100	100	100	100
Union Station, Toronto Islands, Harbourfront East	100	100	100	100	100	100
University of Toronto, Harbord	34	34	34	34	34	34
Victoria Hotel, Commerce Court	100	100	100	100	100	100

In [69]:
```
# Finding out how many unique categories can be curated from all the returned venues...
print('There are {} uniques categories.'.format(len(downtown_toronto_venues['Venue Category'].unique())))
```

```
There are 207 uniques categories.
```

Analyzing Each Neighborhood in Toronto

In [70]:
```
# one hot encoding
```

Segmenting and Clustering Neigl | HSD_Week3_Part_3_of_3 - IBM W | hectorsilvadiniz/Coursera_Capsto

https://jp-tok.dataplatform.cloud.ibm.com/analytics/notebooks/v2/0a15442b-0f6d-4477-a0e8-8426d...

IBM Watson Studio | Upgrade | Hector Silva Diniz's Account | HS

My Projects / Coursera_DataScience_8_Machin... / HSD_Week3_Part_3_of_3

File Edit View Insert Cell Kernel Help | Trusted | Python 3.5

Run | Format | Code

Analyzing Each Neighborhood in Toronto

In [70]:

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

# add neighborhood column back to dataframe
downtown_toronto_onehot['Neighborhood'] = downtown_toronto_venues['Neighborhood']

# move neighborhood column to the first column
fixed_columns = [downtown_toronto_onehot.columns[-1]] + list(downtown_toronto_onehot.columns[:-1])
downtown_toronto_onehot = downtown_toronto_onehot[fixed_columns]

downtown_toronto_onehot.head()
```

Out[70]:

porting Goods Shop	Sports Bar	Steakhouse	Strip Club	Supermarket	Sushi Restaurant	Taco Place	Tailor Shop	Taiwanese Restaurant	Tanning Salon	Tea Room	Thai Restaurant	Theater	Theme Restaurant	Thrift / Vintage Store	Toy / Game Store	Trail	Train Station	Vegetarian / Vegan Restaurant	Video Game Store	Video Store	Vietnamese Restaurant	Wine Bar	Wings Joint	Women's Store
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

In [72]:

```
#visualizing the size of the dataframe
```

Type here to search | ESP | 11:40 PM | 6/16/2019

Segmenting and Clustering Neig | HSD_Week3_Part_3_of_3 - IBM W | hectorsilvadiniz/Coursera_Capsto

https://jp-tok.dataplatform.cloud.ibm.com/analytics/notebooks/v2/0a15442b-0f6d-4477-a0e8-8426d...

IBM Watson Studio | Upgrade | Hector Silva Diniz's Account

My Projects / Coursera_DataScience_8_Machin... / HSD_Week3_Part_3_of_3

File Edit View Insert Cell Kernel Help | Trusted | Python 3.5

Run | Format Code

In [72]:
```
#visualizing the size of the dataframe
downtown_toronto_onehot.shape
```

Out[72]: (1286, 207)

In [73]:
```
# Grouping rows by neighborhood and by taking the mean of the frequency of occurrence of each category
downtown_toronto_grouped = downtown_toronto_onehot.groupby('Neighborhood').mean().reset_index()
downtown_toronto_grouped
```

Out[73]:

	Neighborhood	Yoga Studio	Adult Boutique	Afghan Restaurant	Airport	Airport Food Court	Airport Gate	Airport Lounge	Airport Service	Airport Terminal	American Restaurant	Antique Shop	Aquarium	Art Gallery	Art Museum	Arts & Crafts Store	Asian Restaurant	BBQ Joint	Baby Store	Bagel Shop	Bakery	Ba
0	Berczy Park	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00	0.018182	0.000000	0.000000	0.00	0.00	0.000000	0.018182	0.036364	0.000
1	Central Bay Street	0.011364	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.011364	0.000000	0.00	0.000000	0.011364	0.011364	0.00	0.00	0.000000	0.000000	0.022727	0.000
2	Christie	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00	0.000000	0.000000	0.000000	0.00	0.00	0.066667	0.000000	0.000000	0.000
3	Church and Wellesley	0.011494	0.011494	0.011494	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.011494	0.000000	0.00	0.000000	0.000000	0.011494	0.00	0.00	0.000000	0.000000	0.000000	0.000
4	Garden District, Ryerson	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.010000	0.000000	0.00	0.010000	0.000000	0.000000	0.00	0.00	0.000000	0.000000	0.010000	0.010
5	Kensington Market, Grange Park, Chinatown	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00	0.000000	0.000000	0.010000	0.00	0.00	0.000000	0.010000	0.040000	0.000
6	Regent Park, Harbourfront	0.020833	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.020833	0.00	0.020833	0.000000	0.000000	0.00	0.00	0.000000	0.000000	0.062500	0.020

Type here to search | ESP | 11:40 PM 6/16/2019

	Chinatown																					
6	Regent Park, Harbourfront	0.020833	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.020833	0.00	0.020833	0.000000	0.000000	0.00	0.00	0.000000	0.000000	0.062500	0.020
7	Richmond, King, Adelaide	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.040000	0.000000	0.00	0.010000	0.010000	0.000000	0.02	0.00	0.000000	0.000000	0.030000	0.000
8	Rosedale	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00	0.000000	0.000000	0.000000	0.00	0.00	0.000000	0.000000	0.000000	0.000
9	South Niagara, Railway Lands, King and Spadina...	0.000000	0.000000	0.000000	0.066667	0.066667	0.066667	0.133333	0.133333	0.133333	0.000000	0.000000	0.00	0.000000	0.000000	0.000000	0.00	0.00	0.000000	0.000000	0.000000	0.000
10	St. James Town	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.020000	0.000000	0.00	0.010000	0.000000	0.000000	0.00	0.01	0.000000	0.010000	0.030000	0.010
11	St. James Town, Cabbagetown	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00	0.000000	0.000000	0.000000	0.00	0.00	0.000000	0.000000	0.043478	0.021
12	Stn A PO Boxes 25 The Esplanade	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.010638	0.00	0.021277	0.000000	0.000000	0.00	0.00	0.000000	0.010638	0.021277	0.000
13	Toronto Dominion Centre, Design Exchange	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.030000	0.000000	0.00	0.010000	0.000000	0.000000	0.01	0.00	0.000000	0.000000	0.030000	0.000
14	Underground city, First Canadian Place	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.030000	0.000000	0.00	0.010000	0.000000	0.000000	0.02	0.00	0.000000	0.000000	0.030000	0.000
15	Union Station, Toronto Islands, Harbourfront East	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.05	0.010000	0.000000	0.000000	0.00	0.00	0.000000	0.000000	0.030000	0.000
16	University of Toronto	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00	0.000000	0.000000	0.000000	0.00	0.00	0.000000	0.000000	0.058824	0.000

Segmenting and Clustering Neig | HSD_Week3_Part_3_of_3 - IBM W | hectorsilvadiniz/Coursera_Capsto

https://jp-tok.dataplatform.cloud.ibm.com/analytics/notebooks/v2/0a15442b-0f6d-4477-a0e8-8426d...

IBM Watson Studio | Upgrade | Hector Silva Diniz's Account

My Projects / Coursera_DataScience_8_Machin... / HSD_Week3_Part_3_of_3

File Edit View Insert Cell Kernel Help | Trusted | Python 3.5

Run | Format Code

15	Islands, Harbourfront East	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.05	0.010000	0.000000	0.000000	0.00	0.00	0.000000	0.000000	0.030000	0.000
16	University of Toronto, Harbord	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00	0.000000	0.000000	0.000000	0.00	0.00	0.000000	0.000000	0.058824	0.000
17	Victoria Hotel, Commerce Court	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.030000	0.000000	0.00	0.010000	0.000000	0.000000	0.00	0.00	0.000000	0.000000	0.030000	0.000

In [74]:
```
#rechecking the new size of the dataframe...
downtown_toronto_grouped.shape
```

Out[74]: (18, 207)

In [75]:
```
# Printing each neighborhood along with the top 5 most common venues
num_top_venues = 5

for hood in downtown_toronto_grouped['Neighborhood']:
    print("----"+hood+"----")
    temp = downtown_toronto_grouped[downtown_toronto_grouped['Neighborhood'] == 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')
```

```
----Berczy Park----
                venue  freq
0         Coffee Shop  0.09
1        Cocktail Bar  0.05
2  Italian Restaurant  0.04
```

Type here to search | ESP | 11:40 PM 6/16/2019

```
                  venue  freq
0         Coffee Shop  0.09
1        Cocktail Bar  0.05
2  Italian Restaurant  0.04
3               Café  0.04
4         Cheese Shop  0.04


----Central Bay Street----
                  venue  freq
0         Coffee Shop  0.16
1               Café  0.05
2  Italian Restaurant  0.05
3        Burger Joint  0.03
4      Sandwich Place  0.03


----Christie----
                venue  freq
```

In [76]:
```
# Puting the information into a datafram
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 [77]:
```
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:
```

```
# 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'] = downtown_toronto_grouped['Neighborhood']

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

neighborhoods_venues_sorted.head()
```

Out[77]:

	Neighborhood	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue	7th Most Common Venue	8th Most Common Venue	9th Most Common Venue	10th Most Common Venue
0	Berczy Park	Coffee Shop	Cocktail Bar	Seafood Restaurant	Bakery	Farmers Market	Steakhouse	Beer Bar	Cheese Shop	Café	Italian Restaurant
1	Central Bay Street	Coffee Shop	Café	Italian Restaurant	Sandwich Place	Burger Joint	Middle Eastern Restaurant	Japanese Restaurant	Bubble Tea Shop	Bar	Bakery
2	Christie	Grocery Store	Café	Park	Nightclub	Diner	Baby Store	Italian Restaurant	Restaurant	Coffee Shop	Convenience Store
3	Church and Wellesley	Coffee Shop	Japanese Restaurant	Sushi Restaurant	Gay Bar	Restaurant	Café	Pub	Hotel	Fast Food Restaurant	Gym
4	Garden District, Ryerson	Coffee Shop	Clothing Store	Café	Cosmetics Shop	Middle Eastern Restaurant	Tea Room	Italian Restaurant	Bubble Tea Shop	Pizza Place	Diner

Clustering Neighborhoods

Clustering Neighborhoods

In [ ]:
```
#Running  k-means to cluster the neighborhood into 5 clusters.
```

In [78]:
```
# set number of clusters
kclusters = 5

downtown_toronto_grouped_clustering = downtown_toronto_grouped.drop('Neighborhood', 1)

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

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

Out[78]: array([0, 0, 3, 0, 0, 4, 0, 0, 1, 2], dtype=int32)

In [79]:
```
# Creating a new dataframe that includes the cluster as well as the top 10 venues for each neighborhood.
# add clustering labels
neighborhoods_venues_sorted.insert(0, 'Cluster Labels', kmeans.labels_)

downtown_toronto_merged = toronto_data    #########

downtown_toronto_merged = downtown_toronto_merged.join(neighborhoods_venues_sorted.set_index('Neighborhood'), on='Neighborhood')

downtown_toronto_merged.head()
```

Out[79]:

	PostalCode	Borough	Neighborhood	Latitude	Longitude	Cluster Labels	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue	7th Most Common Venue	8th Most Common Venue	9th Most Common Venue	10th Most Common Venue
0	M5A	Downtown	Regent Park,	43.6543	-79.3606	0	Coffee Shop	Pub	Park	Bakery	Breakfast Spot	Restaurant	Theater	Café	Mexican	Electronics

	PostalCode	Borough	Neighborhood	Latitude	Longitude	Cluster Labels	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue	7th Most Common Venue	8th Most Common Venue	9th Most Common Venue	10th Most Common Venue
0	M5A	Downtown Toronto	Regent Park, Harbourfront	43.6543	-79.3606	0	Coffee Shop	Pub	Park	Bakery	Breakfast Spot	Restaurant	Theater	Café	Mexican Restaurant	Electronics Store
1	M5B	Downtown Toronto	Garden District, Ryerson	43.6572	-79.3789	0	Coffee Shop	Clothing Store	Café	Cosmetics Shop	Middle Eastern Restaurant	Tea Room	Italian Restaurant	Bubble Tea Shop	Pizza Place	Diner
2	M5C	Downtown Toronto	St. James Town	43.6515	-79.3754	0	Coffee Shop	Café	Hotel	Restaurant	Cosmetics Shop	Clothing Store	Cocktail Bar	Bakery	Gastropub	Breakfast Spot
3	M5E	Downtown Toronto	Berczy Park	43.6448	-79.3733	0	Coffee Shop	Cocktail Bar	Seafood Restaurant	Bakery	Farmers Market	Steakhouse	Beer Bar	Cheese Shop	Café	Italian Restaurant
4	M5G	Downtown Toronto	Central Bay Street	43.658	-79.3874	0	Coffee Shop	Café	Italian Restaurant	Sandwich Place	Burger Joint	Middle Eastern Restaurant	Japanese Restaurant	Bubble Tea Shop	Bar	Bakery

In [84]:

```
# Visualizing the resulting clusters
# create map
map_clusters = folium.Map(location=[latitude, longitude], zoom_start=12)

# 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(downtown_toronto_merged['Latitude'], downtown_toronto_merged['Longitude'], downtown_toronto_merged['Neighborhood'], downtown_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],
```

Segmenting and Clustering Neig | HSD_Week3_Part_3_of_3 - IBM W | hectorsilvadiniz/Coursera_Capsto

https://jp-tok.dataplatform.cloud.ibm.com/analytics/notebooks/v2/0a15442b-0f6d-4477-a0e8-8426d...

IBM Watson Studio | Upgrade | Hector Silva Diniz's Account

My Projects / Coursera_DataScience_8_Machin... / HSD_Week3_Part_3_of_3

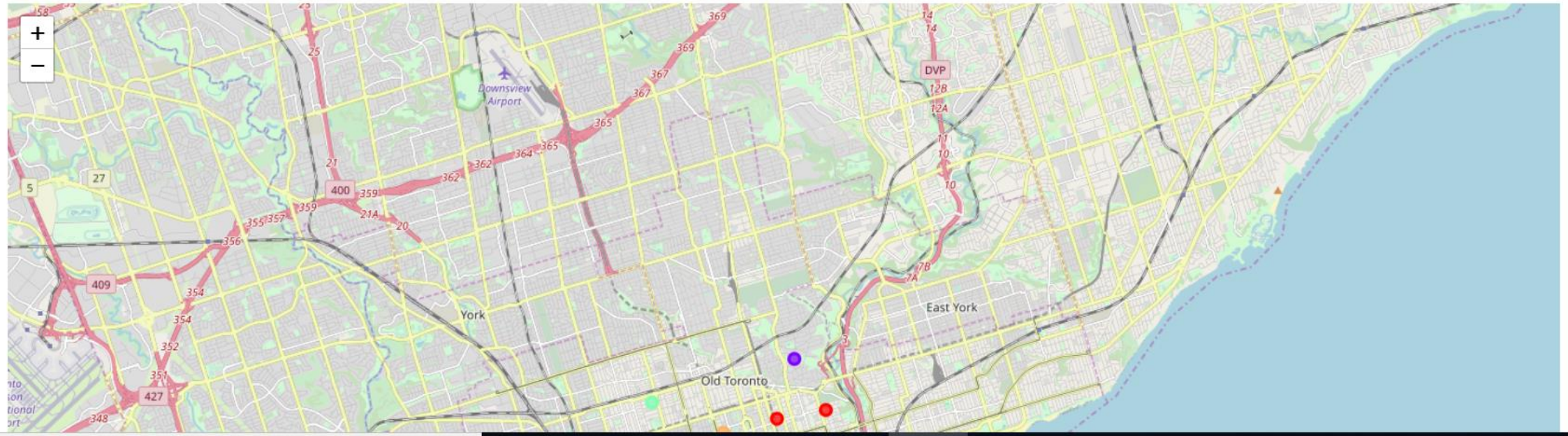
File Edit View Insert Cell Kernel Help | Trusted | Python 3.5

Run | Format Code

```
        [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[84]:



Type here to search | ESP | 11:40 PM 6/16/2019

Segmenting and Clustering Neig | HSD_Week3_Part_3_of_3 - IBM W | hectorsilvadiniz/Coursera_Capsto

https://jp-tok.dataplatform.cloud.ibm.com/analytics/notebooks/v2/0a15442b-0f6d-4477-a0e8-8426d...

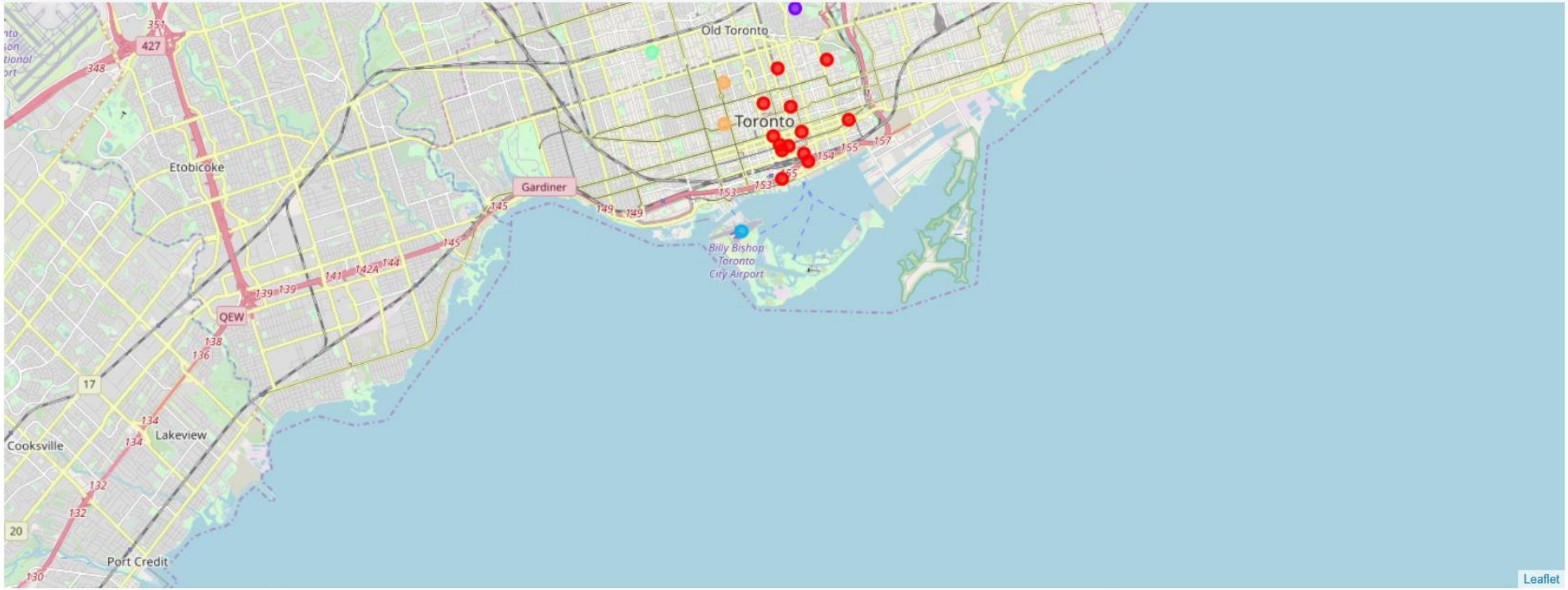
IBM Watson Studio | Upgrade | Hector Silva Diniz's Account | HS

My Projects / Coursera_DataScience_8_Machin... / HSD_Week3_Part_3_of_3

File Edit View Insert Cell Kernel Help

Trusted | Python 3.5

Run | Format Code



Examining Clusters

Type here to search | ESP | 11:40 PM 6/16/2019

Examining Clusters

In [86]:
```
#Cluster 1
downtown_toronto_merged.loc[downtown_toronto_merged['Cluster Labels'] == 0, \
                            downtown_toronto_merged.columns[[1] + list(range(5, downtown_toronto_merged.shape[1]))]]
```

Out[86]:

	Borough	Cluster Labels	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue	7th Most Common Venue	8th Most Common Venue	9th Most Common Venue	10th Most Common Venue
0	Downtown Toronto	0	Coffee Shop	Pub	Park	Bakery	Breakfast Spot	Restaurant	Theater	Café	Mexican Restaurant	Electronics Store
1	Downtown Toronto	0	Coffee Shop	Clothing Store	Café	Cosmetics Shop	Middle Eastern Restaurant	Tea Room	Italian Restaurant	Bubble Tea Shop	Pizza Place	Diner
2	Downtown Toronto	0	Coffee Shop	Café	Hotel	Restaurant	Cosmetics Shop	Clothing Store	Cocktail Bar	Bakery	Gastropub	Breakfast Spot
3	Downtown Toronto	0	Coffee Shop	Cocktail Bar	Seafood Restaurant	Bakery	Farmers Market	Steakhouse	Beer Bar	Cheese Shop	Café	Italian Restaurant
4	Downtown Toronto	0	Coffee Shop	Café	Italian Restaurant	Sandwich Place	Burger Joint	Middle Eastern Restaurant	Japanese Restaurant	Bubble Tea Shop	Bar	Bakery
6	Downtown Toronto	0	Coffee Shop	Café	American Restaurant	Bar	Steakhouse	Bakery	Gym	Restaurant	Hotel	Cosmetics Shop
7	Downtown Toronto	0	Coffee Shop	Aquarium	Hotel	Restaurant	Italian Restaurant	Café	Brewery	Bakery	Pizza Place	Sporting Goods Shop
8	Downtown Toronto	0	Coffee Shop	Hotel	Café	Restaurant	Italian Restaurant	Gastropub	Deli / Bodega	Bakery	American Restaurant	Beer Bar
9	Downtown Toronto	0	Coffee Shop	Café	Hotel	Restaurant	Deli / Bodega	Gastropub	Seafood Restaurant	Bakery	Italian Restaurant	American Restaurant

8	Toronto	0	Coffee Shop	Hotel	Cafe	Restaurant	Italian Restaurant	Gastropub	Deli / Bodega	Bakery	Restaurant	Beer Bar
9	Downtown Toronto	0	Coffee Shop	Café	Hotel	Restaurant	Deli / Bodega	Gastropub	Seafood Restaurant	Bakery	Italian Restaurant	American Restaurant
14	Downtown Toronto	0	Coffee Shop	Café	Restaurant	Italian Restaurant	Beer Bar	Seafood Restaurant	Hotel	Cocktail Bar	Art Gallery	Cheese Shop
15	Downtown Toronto	0	Coffee Shop	Restaurant	Park	Pub	Café	Bakery	Italian Restaurant	Pizza Place	Breakfast Spot	Butcher
16	Downtown Toronto	0	Coffee Shop	Café	Hotel	Restaurant	Steakhouse	Seafood Restaurant	American Restaurant	Bar	Bakery	Deli / Bodega
17	Downtown Toronto	0	Coffee Shop	Japanese Restaurant	Sushi Restaurant	Gay Bar	Restaurant	Café	Pub	Hotel	Fast Food Restaurant	Gym

In [87]:
```
#Cluster 2
downtown_toronto_merged.loc[downtown_toronto_merged['Cluster Labels'] == 1, \
                            downtown_toronto_merged.columns[[1] + list(range(5, downtown_toronto_merged.shape[1]))]]
```

Out[87]:

	Borough	Cluster Labels	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue	7th Most Common Venue	8th Most Common Venue	9th Most Common Venue	10th Most Common Venue
13	Downtown Toronto	1	Park	Playground	Trail	Women's Store	Department Store	Ethiopian Restaurant	Electronics Store	Eastern European Restaurant	Dumpling Restaurant	Donut Shop

In [88]:
```
#Cluster 3
downtown_toronto_merged.loc[downtown_toronto_merged['Cluster Labels'] == 2, \
                            downtown_toronto_merged.columns[[1] + list(range(5, downtown_toronto_merged.shape[1]))]]
```

Out[88]:

	Borough	Cluster Labels	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue	7th Most Common Venue	8th Most Common Venue	9th Most Common Venue	10th Most Common Venue
12	Downtown Toronto	2	Airport Lounge	Airport Service	Airport Terminal	Bar	Boutique	Boat or Ferry	Sculpture Garden	Airport	Airport Food Court	Airport Gate

	Borough	Cluster Labels	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue	7th Most Common Venue	8th Most Common Venue	9th Most Common Venue	10th Most Common Venue
12	Downtown Toronto	2	Airport Lounge	Airport Service	Airport Terminal	Bar	Boutique	Boat or Ferry	Sculpture Garden	Airport	Airport Food Court	Airport Gate

In [89]:
```
#Cluster 4
downtown_toronto_merged.loc[downtown_toronto_merged['Cluster Labels'] == 3, \
                            downtown_toronto_merged.columns[[1] + list(range(5, downtown_toronto_merged.shape[1]))]]
```

Out[89]:

	Borough	Cluster Labels	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue	7th Most Common Venue	8th Most Common Venue	9th Most Common Venue	10th Most Common Venue
5	Downtown Toronto	3	Grocery Store	Café	Park	Nightclub	Diner	Baby Store	Italian Restaurant	Restaurant	Coffee Shop	Convenience Store

In [90]:
```
#Cluster 5
downtown_toronto_merged.loc[downtown_toronto_merged['Cluster Labels'] == 4, \
                            downtown_toronto_merged.columns[[1] + list(range(5, downtown_toronto_merged.shape[1]))]]
```

Out[90]:

	Borough	Cluster Labels	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue	7th Most Common Venue	8th Most Common Venue	9th Most Common Venue	10th Most Common Venue
10	Downtown Toronto	4	Café	Restaurant	Bookstore	Japanese Restaurant	Bar	Italian Restaurant	Bakery	Nightclub	Chinese Restaurant	Sandwich Place
11	Downtown Toronto	4	Café	Vegetarian / Vegan Restaurant	Coffee Shop	Mexican Restaurant	Dumpling Restaurant	Bar	Bakery	Vietnamese Restaurant	Chinese Restaurant	Caribbean Restaurant

	Borough	Cluster Labels	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue	7th Most Common Venue	8th Most Common Venue	9th Most Common Venue	10th Most Common Venue
12	Downtown Toronto	2	Airport Lounge	Airport Service	Airport Terminal	Bar	Boutique	Boat or Ferry	Sculpture Garden	Airport	Airport Food Court	Airport Gate

In [89]:
```
#Cluster 4
downtown_toronto_merged.loc[downtown_toronto_merged['Cluster Labels'] == 3, \
                            downtown_toronto_merged.columns[[1] + list(range(5, downtown_toronto_merged.shape[1]))]]
```

Out[89]:

	Borough	Cluster Labels	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue	7th Most Common Venue	8th Most Common Venue	9th Most Common Venue	10th Most Common Venue
5	Downtown Toronto	3	Grocery Store	Café	Park	Nightclub	Diner	Baby Store	Italian Restaurant	Restaurant	Coffee Shop	Convenience Store

In [90]:
```
#Cluster 5
downtown_toronto_merged.loc[downtown_toronto_merged['Cluster Labels'] == 4, \
                            downtown_toronto_merged.columns[[1] + list(range(5, downtown_toronto_merged.shape[1]))]]
```

Out[90]:

	Borough	Cluster Labels	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue	7th Most Common Venue	8th Most Common Venue	9th Most Common Venue	10th Most Common Venue
10	Downtown Toronto	4	Café	Restaurant	Bookstore	Japanese Restaurant	Bar	Italian Restaurant	Bakery	Nightclub	Chinese Restaurant	Sandwich Place
11	Downtown Toronto	4	Café	Vegetarian / Vegan Restaurant	Coffee Shop	Mexican Restaurant	Dumpling Restaurant	Bar	Bakery	Vietnamese Restaurant	Chinese Restaurant	Caribbean Restaurant