Applied Data Science Capstone: Find the best place to open up a high-end steakhouse in Toronto

Load Libraries and import Toronto Postal Code Data

In [2]:
```
import numpy as np
import pandas as pd
import requests
from bs4 import BeautifulSoup

res = requests.get('https://en.wikipedia.org/wiki/List_of_postal_codes_
of_Canada:_M')
soup = BeautifulSoup(res.content,'lxml')
table = soup.find_all('table')[0]
df = pd.read_html(str(table))

data = pd.DataFrame(df[0])

data = data.rename(columns={0:'Postal Code', 1:'Borough', 2:'Neighbourh
ood'})

data = data.iloc[1:]

data = data[~data['Borough'].str.contains('Not assigned')]

df2=data.groupby(['Postal Code', 'Borough']).apply(lambda group: ', '.j
oin(group['Neighbourhood']))

df2=df2.to_frame().reset_index()
df2 = df2.rename(columns={0:'Neighborhood'})
```

```
df2.loc[df2.Neighborhood == 'Not assigned', 'Neighborhood' ] = df2.Boro
ugh

df2.head()
```

Out[2]:

	Postal Code	Borough	Neighborhood
0	M1B	Scarborough	Malvern, Rouge
1	M1C	Scarborough	Rouge Hill, Port Union, Highland Creek
2	M1E	Scarborough	Guildwood, Morningside, West Hill
3	M1G	Scarborough	Woburn
4	M1H	Scarborough	Cedarbrae

Load geospatial cooridinates for Toronto and merge with Toronto Postal Code Data

In [3]:
```
!wget -O to_geo_space.csv http://cocl.us/Geospatial_data

gs = pd.read_csv('to_geo_space.csv')

gs = gs.rename(columns={'Postal Code':'Postal Code'})

gs1 = pd.merge(df2, gs, on='Postal Code', how='inner')

gs1.head()
```

```
--2021-01-03 05:28:10--  http://cocl.us/Geospatial_data
Resolving cocl.us (cocl.us)... 169.63.96.194, 169.63.96.176
Connecting to cocl.us (cocl.us)|169.63.96.194|:80... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: https://cocl.us/Geospatial_data [following]
--2021-01-03 05:28:10--  https://cocl.us/Geospatial_data
Connecting to cocl.us (cocl.us)|169.63.96.194|:443... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: https://ibm.box.com/shared/static/9afzr83pps4pwf2smjjcf1y5m
```

```
vgb18rr.csv [following]
--2021-01-03 05:28:11--  https://ibm.box.com/shared/static/9afzr83pps
4pwf2smjjcf1y5mvgb18rr.csv
Resolving ibm.box.com (ibm.box.com)... 107.152.29.197
Connecting to ibm.box.com (ibm.box.com)|107.152.29.197|:443... connec
ted.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: /public/static/9afzr83pps4pwf2smjjcf1y5mvgb18rr.csv [follow
ing]
--2021-01-03 05:28:11--  https://ibm.box.com/public/static/9afzr83pps
4pwf2smjjcf1y5mvgb18rr.csv
Reusing existing connection to ibm.box.com:443.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: https://ibm.ent.box.com/public/static/9afzr83pps4pwf2smjjcf
1y5mvgb18rr.csv [following]
--2021-01-03 05:28:11--  https://ibm.ent.box.com/public/static/9afzr8
3pps4pwf2smjjcf1y5mvgb18rr.csv
Resolving ibm.ent.box.com (ibm.ent.box.com)... 107.152.29.201
Connecting to ibm.ent.box.com (ibm.ent.box.com)|107.152.29.201|:44
3... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://public.boxcloud.com/d/1/b1!kYDTUivzqtf_WRCMyouSx6v-
mmhQ_LWkk4RI7PQYcMMTeFPFtAOC11XOuRP9hbhYWFEXpylBcYwsb8S0u-cEBa8KS1Bad
IrNv1XZatVLHABguPdvJrym60MlXzKDbuWUeFVjjtaI7lUit-csBYRlC05XBTBw83BTQF
bzNwihIAhlruyq3e2axCjsYuwWuS9fWoR7A3z3Ll41QhGOTfk716SWiOGkx3GvDKbs4EJ
cim9w9SAm9PsJLAZznyaHeI39hfQvh0t95Ly7AOAO5lROfMXQkUc8ssQw0Mi8JOrxFdtN
BEUFWKzgb4aqWiRw7PUw_ieV_EQUWrXj-OIMjsw1yASy5BpRwJc1VYfo-3Pcc6iJq8DB6
I3y57Qn6OyGk4LbDNjJ157czWmBxaowYfF8WcEDofqqbfW_FTUHeqZnmjG2jnjryaFcof
k3PZEk8CeMRsA4slNJfLOu9PIR59nE4gPSo57cvSlmOvURdn5Xr3KVzodlH83lVh9czJg
sxm5JpjZsB2yoZhCGC1Cg-VPayRXxLCaLlP5a-BNL4yd38JfJ0vvzEfwkpjoKwNydn1fw
SHXziqCekO4hXQW_CV-zYN6t3v2v39Uq0crsrz_jksp-qQ6bSeGliQ_-D8FtTnwGQMK1C
oadtA0pwTnkN0yBus-rKaUS2H7bPoXA1erQJ5O4mEGwzgMOECSZUPhcZVzymzLDxdxZrQ
a7lquO9Yxyfu2mnQokhePVZkw5sHdipV2alHysSYxP5amn-oDJtftbMRqGJ6rW7r0l7nS
OaOUfv5cyiyonZjdnDdtmbqJlJM07rJQEWslzTL7Z7ShCSpfhJ0-uOxqMWNWzmMWho6tC
LPt1mPte6T06_kXxnEoauiuVwv3V0t7rNrop_oB9V9n5zMLqh28r8IAkYx764mGGoH_R5
TC9BaCa2FX9XW7IxAYoGmZdN8yXmBVg2ZbBiFyntPjRiwn11vy1NXtC9WuyZn48qgr8RY
icEe2es6Cg7f_hVfHv_2xsRyn_Fs-2ZYx1ReFVxKoxhj0MYP4RNVAdNKqVSwKfH6z-fhE
ZfmnCiWRhwsv3xc_lXipcQYrsUC_iFMb-9NN7-1zyA2xwtyLRhIl5wANYnpPoecBV2aSB
sie4DXufYUkGJS_Eeb5g6AJRbPequPEHUjvGHuxgNIRRW2L-OcNpDIHPFcH4XbkXIAlO1
```

```
NzRQ2-sFFDivVy-jGw88cYXvmywlVpLqYwfyRI51If-pURAVy6Z3iOufjpSHRuxc3nbeB
7Eo82UhOiz9fktWxgy-ituVp1kMrk3AM-a4L4Wg2__Y_JDu5_tADPeT5dVn8cEFYliyiT
33mUchx3Hgi0FPmf3_NPo-WHnt_xsGaQiodqgGYlCxB-XpfPjS90ydrzpiWLdSgi_EHy2
R_F46mDFfdnJDNsbrrcUfrjsiLCWsjDrUSZWccVjwt3B1xlA/download [following]
--2021-01-03 05:28:12--  https://public.boxcloud.com/d/1/b1!kYDTUivzq
tf_WRCMyouSx6v-mmhQ_LWkk4RI7PQYcMMTeFPFtAOC11XOuRP9hbhYWFEXpylBcYwsb8
S0u-cEBa8KS1BadIrNv1XZatVLHABguPdvJrym60MlXzKDbuWUeFVjjtaI7lUit-csBYR
lC05XBTBw83BTQFbzNwihIAhlruyq3e2axCjsYuwWuS9fWoR7A3z3Ll41QhGOTfk716SW
iOGkx3GvDKbs4EJcim9w9SAm9PsJLAZznyaHeI39hfQvh0t95Ly7AOAO5lROfMXQkUc8s
sQw0Mi8JOrxFdtNBEUFWKzgb4aqWiRw7PUw_ieV_EQUWrXj-OIMjsw1yASy5BpRwJc1VY
fo-3Pcc6iJq8DB6I3y57Qn6OyGk4LbDNjJ157czWmBxaowYfF8WcEDofqqbfW_FTUHeqZ
nmjG2jnjryaFcofk3PZEk8CeMRsA4slNJfLOu9PIR59nE4gPSo57cvSlmOvURdn5Xr3KV
zodlH83lVh9czJgsxm5JpjZsB2yoZhCGC1Cg-VPayRXxLCaLlP5a-BNL4yd38JfJ0vvzE
fwkpjoKwNydn1fwSHXziqCekO4hXQW_CV-zYN6t3v2v39Uq0crsrz_jksp-qQ6bSeGliQ
_-D8FtTnwGQMK1CoadtA0pwTnkN0yBus-rKaUS2H7bPoXA1erQJ5O4mEGwzgMOECSZUPh
cZVzymzLDxdxZrQa7lquO9Yxyfu2mnQokhePVZkw5sHdipV2alHysSYxP5amn-oDJtftb
MRqGJ6rW7r0l7nSOaOUfv5cyiyonZjdnDdtmbqJlJM07rJQEWslzTL7Z7ShCSpfhJ0-uO
xqMWNWzmMWho6tCLPt1mPte6T06_kXxnEoauiuVwv3V0t7rNrop_oB9V9n5zMLqh28r8I
AkYx764mGGoH_R5TC9BaCa2FX9XW7IxAYoGmZdN8yXmBVg2ZbBiFyntPjRiwn11vy1NXt
C9WuyZn48qgr8RYicEe2es6Cg7f_hVfHv_2xsRyn_Fs-2ZYx1ReFVxKoxhj0MYP4RNVAd
NKqVSwKfH6z-fhEZfmnCiWRhwsv3xc_lXipcQYrsUC_iFMb-9NN7-1zyA2xwtyLRhIl5w
ANYnpPoecBV2aSBsie4DXufYUkGJS_Eeb5g6AJRbPequPEHUjvGHuxgNIRRW2L-OcNpDI
HPFcH4XbkXIAlO1NzRQ2-sFFDivVy-jGw88cYXvmywlVpLqYwfyRI51If-pURAVy6Z3iO
ufjpSHRuxc3nbeB7Eo82UhOiz9fktWxgy-ituVp1kMrk3AM-a4L4Wg2__Y_JDu5_tADPe
T5dVn8cEFYliyiT33mUchx3Hgi0FPmf3_NPo-WHnt_xsGaQiodqgGYlCxB-XpfPjS90yd
rzpiWLdSgi_EHy2R_F46mDFfdnJDNsbrrcUfrjsiLCWsjDrUSZWccVjwt3B1xlA/downl
oad
Resolving public.boxcloud.com (public.boxcloud.com)... 107.152.29.200
Connecting to public.boxcloud.com (public.boxcloud.com)|107.152.29.20
0|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2891 (2.8K) [text/csv]
Saving to: ‘to_geo_space.csv’

to_geo_space.csv    100%[===================>]   2.82K  --.-KB/s    i
n 0s

2021-01-03 05:28:12 (46.2 MB/s) - ‘to_geo_space.csv’ saved [2891/289
1]
```

Out[3]:

	Postal Code	Borough	Neighborhood	Latitude	Longitude
0	M1B	Scarborough	Malvern, Rouge	43.806686	-79.194353
1	M1C	Scarborough	Rouge Hill, Port Union, Highland Creek	43.784535	-79.160497
2	M1E	Scarborough	Guildwood, Morningside, West Hill	43.763573	-79.188711
3	M1G	Scarborough	Woburn	43.770992	-79.216917
4	M1H	Scarborough	Cedarbrae	43.773136	-79.239476

Toronto neighborhoods populations by their postal code

In [4]:

```
import ssl
ssl._create_default_https_context = ssl._create_unverified_context

df_pop = pd.read_csv('https://www12.statcan.gc.ca/census-recensement/20
16/dp-pd/hlt-fst/pd-pl/Tables/File.cfm?T=1201&SR=1&RPP=9999&PR=0&CMA=0&
CSD=0&S=22&O=A&Lang=Eng&OFT=CSV',encoding = 'unicode_escape')

df_pop = df_pop.rename(columns={'Geographic code':'Postal Code', 'Geogr
aphic name':'Postal Code2', 'Province or territory':'Province', 'Incomp
letely enumerated Indian reserves and Indian settlements, 2016':'Incomp
lete', 'Population, 2016':'Population_2016', 'Total private dwellings,
 2016':'TotalPrivDwellings', 'Private dwellings occupied by usual resid
ents, 2016':'PrivDwellingsOccupied'})
df_pop= df_pop.drop(columns=['Postal Code2', 'Province', 'Incomplete',
'TotalPrivDwellings', 'PrivDwellingsOccupied'])

df_pop = df_pop.iloc[1:]
df_pop.head()
```

Out[4]:

	Postal Code	Population_2016

	Postal Code	Population_2016
1	A0A	46587.0
2	A0B	19792.0
3	A0C	12587.0
4	A0E	22294.0
5	A0G	35266.0

Merge Postal Codes with their corresponding populations

In [5]:
```
gs1
gs1 = pd.merge(df_pop, gs1, on='Postal Code', how='right')
gs1 = gs1.sort_values(by=['Population_2016'], ascending=False)

gs1.head()
```

Out[5]:

	Postal Code	Population_2016	Borough	Neighborhood	Latitude	Longitude
22	M2N	75897.0	North York	Willowdale, Willowdale East	43.770120	-79.408493
0	M1B	66108.0	Scarborough	Malvern, Rouge	43.806686	-79.194353
18	M2J	58293.0	North York	Fairview, Henry Farm, Oriole	43.778517	-79.346556
100	M9V	55959.0	Etobicoke	South Steeles, Silverstone, Humbergate, Jamest...	43.739416	-79.588437
14	M1V	54680.0	Scarborough	Milliken, Agincourt North, Steeles East, L'Amo...	43.815252	-79.284577

Toronto Neighborhoods based on Average After Tax Income by Postal Codes

In [6]:
```
df_income = pd.read_csv(body)
df_income = pd.read_csv('IncomeToronto.csv',encoding = 'unicode_escape'
)
df_income = df_income.rename(columns={"Average After Tax Income":"AvgAf
terTaxIncome"})
df_income.head()
```

Out[6]:

	Postal Code	Average After Tax Income
0	M1B	30801
1	M1C	34837
2	M1E	43848
3	M1G	27341
4	M1H	None

Merge Postal Codes with average incomes

In [7]:
```
gs1 = pd.merge(df_income, gs1, on='Postal Code', how='right')

gs1 = gs1.replace('None', 0)
```

In [8]:
```
gs1['Average After Tax Income'] = gs1['Average After Tax Income'].astyp
e('float64')
```

In [9]:
```
gs1 = gs1.sort_values(by=['Average After Tax Income'], ascending=False)

gs1.to_csv('TO_Affluence.csv')

gs1.head(10)
```

Out[9]:

	Postal Code	Average After Tax Income	Population_2016	Borough	Neighborhood	Latitude	Longitude

	Postal Code	Average After Tax Income	Population_2016	Borough	Neighborhood	Latitude	Longitude
20	M2L	193454.0	11717.0	North York	York Mills, Silver Hills	43.757490	-79.374714
48	M4T	134865.0	10463.0	Central Toronto	Moore Park, Summerhill East	43.689574	-79.383160
49	M4V	115033.0	18241.0	Central Toronto	Summerhill West, Rathnelly, South Hill, Forest...	43.686412	-79.400049
89	M8X	97836.0	10787.0	Etobicoke	The Kingsway, Montgomery Road, Old Mill North	43.653654	-79.506944
44	M4N	95343.0	15330.0	Central Toronto	Lawrence Park	43.728020	-79.388790
62	M5M	85678.0	25975.0	North York	Bedford Park, Lawrence Manor East	43.733283	-79.419750
38	M4G	85496.0	19076.0	East York	Leaside	43.709060	-79.363452
65	M5R	80138.0	26496.0	Central Toronto	The Annex, North Midtown, Yorkville	43.672710	-79.405678
92	M9A	72156.0	35594.0	Etobicoke	Islington Avenue, Humber Valley Village	43.667856	-79.532242
23	M2P	70885.0	7843.0	North York	York Mills West	43.752758	-79.400049

In [10]:
```
CLIENT_ID = 'Hidden'


CLIENT_SECRET = 'Hidden'


VERSION = '20180604'
```

In [11]:
```
import requests
from pandas.io.json import json_normalize

LIMIT = 200

radius = 500

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

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

        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)

        results = requests.get(url).json()["response"]['groups'][0]['it
ems']

        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 fo
r item in venue_list])
    nearby_venues.columns = ['Neighborhood',
                  'Neighborhood Latitude',
                  'Neighborhood Longitude',
                  'Venue',
                  'Venue Latitude',
                  'Venue Longitude',
                  'Venue Category']

    return(nearby_venues)
```

In [12]:
```
Data1 = gs1
Data1.head()
```

Out[12]:

	Postal Code	Average After Tax Income	Population_2016	Borough	Neighborhood	Latitude	Longitude
20	M2L	193454.0	11717.0	North York	York Mills, Silver Hills	43.757490	-79.374714
48	M4T	134865.0	10463.0	Central Toronto	Moore Park, Summerhill East	43.689574	-79.383160
49	M4V	115033.0	18241.0	Central Toronto	Summerhill West, Rathnelly, South Hill, Forest...	43.686412	-79.400049
89	M8X	97836.0	10787.0	Etobicoke	The Kingsway, Montgomery Road, Old Mill North	43.653654	-79.506944
44	M4N	95343.0	15330.0	Central Toronto	Lawrence Park	43.728020	-79.388790

In [14]:
```
Venues1 = getNearbyVenues(names=Data1['Neighborhood'],
                                   latitudes=Data1['Latitude'],
                                   longitudes=Data1['Longitude']
                                  )
```

```
York Mills, Silver Hills
Moore Park, Summerhill East
Summerhill West, Rathnelly, South Hill, Forest Hill SE, Deer Park
The Kingsway, Montgomery Road, Old Mill North
Lawrence Park
```

```
Bedford Park, Lawrence Manor East
Leaside
The Annex, North Midtown, Yorkville
Islington Avenue, Humber Valley Village
York Mills West
Davisville North
Runnymede, Swansea
The Beaches
Brockton, Parkdale Village, Exhibition Place
Church and Wellesley
CN Tower, King and Spadina, Railway Lands, Harbourfront West, Bathurst
Quay, South Niagara, Island airport
Don Mills
Forest Hill North & West, Forest Hill Road Park
The Danforth West, Riverdale
Eringate, Bloordale Gardens, Old Burnhamthorpe, Markland Wood
High Park, The Junction South
Guildwood, Morningside, West Hill
Central Bay Street
Humewood-Cedarvale
Berczy Park
Birch Cliff, Cliffside West
Mimico NW, The Queensway West, South of Bloor, Kingsway Park South Wes
t, Royal York South West
Victoria Village
Bathurst Manor, Wilson Heights, Downsview North
India Bazaar, The Beaches West
Bayview Village
University of Toronto, Harbord
Parkdale, Roncesvalles
Roselawn
West Deane Park, Princess Gardens, Martin Grove, Islington, Cloverdale
New Toronto, Mimico South, Humber Bay Shores
Westmount
Little Portugal, Trinity
Regent Park, Harbourfront
East Toronto, Broadview North (Old East York)
Kingsview Village, St. Phillips, Martin Grove Gardens, Richview Gardens
Willowdale, Willowdale East
```

```
Cliffside, Cliffcrest, Scarborough Village West
Alderwood, Long Branch
Woodbine Heights
Hillcrest Village
Downsview
Kensington Market, Chinatown, Grange Park
Parkwoods
Rouge Hill, Port Union, Highland Creek
Dufferin, Dovercourt Village
Christie
Humberlea, Emery
Glencairn
Malvern, Rouge
Wexford, Maryvale
Caledonia-Fairbanks
Lawrence Manor, Lawrence Heights
Downsview
Willowdale, Newtonbrook
Clarks Corners, Tam O'Shanter, Sullivan
Del Ray, Mount Dennis, Keelsdale and Silverthorn
Runnymede, The Junction North
Northwest, West Humber - Clairville
Willowdale, Willowdale West
Weston
St. James Town, Cabbagetown
Agincourt
Dorset Park, Wexford Heights, Scarborough Town Centre
Scarborough Village
Steeles West, L'Amoreaux West
Kennedy Park, Ionview, East Birchmount Park
Woburn
Humber Summit
Milliken, Agincourt North, Steeles East, L'Amoreaux East
Northwood Park, York University
Parkview Hill, Woodbine Gardens
South Steeles, Silverstone, Humbergate, Jamestown, Mount Olive, Beaumon
d Heights, Thistletown, Albion Gardens
Thorncliffe Park
Don Mills
```

```
Downsview
Golden Mile, Clairlea, Oakridge
North Park, Maple Leaf Park, Upwood Park
Harbourfront East, Union Station, Toronto Islands
Cedarbrae
Old Mill South, King's Mill Park, Sunnylea, Humber Bay, Mimico NE, The
Queensway East, Royal York South East, Kingsway Park South East
Richmond, Adelaide, King
Fairview, Henry Farm, Oriole
Upper Rouge
Business reply mail Processing Centre, South Central Letter Processing
Plant Toronto
Queen's Park, Ontario Provincial Government
St. James Town
Toronto Dominion Centre, Design Exchange
Downsview
Studio District
North Toronto West, Lawrence Park
Davisville
Rosedale
Garden District, Ryerson
First Canadian Place, Underground city
Stn A PO Boxes
Commerce Court, Victoria Hotel
Canada Post Gateway Processing Centre
```

In [15]:
```
print('Unique Venue Categories:')
list(Venues1['Venue Category'].unique())
```

```
Unique Venue Categories:
```

Out[15]:
```
['Trail',
 'Restaurant',
 'Tennis Court',
 'Playground',
 'Supermarket',
 'Liquor Store',
 'Sushi Restaurant',
 'American Restaurant',
 'Coffee Shop',
 'Pub',
 'Fried Chicken Joint',
 'Vietnamese Restaurant',
 'Pizza Place',
 'Bank',
 'Light Rail Station',
 'Sandwich Place',
 'Bagel Shop',
 'River',
 'Park',
 'Business Service',
 'Swim School',
 'Bus Line',
 'Café',
 'Indian Restaurant',
 'Italian Restaurant',
 'Thai Restaurant',
 'Juice Bar',
 'Comfort Food Restaurant',
 'Greek Restaurant',
 'Pharmacy',
 'Grocery Store',
 'Butcher',
 'Japanese Restaurant',
 'Spa',
 'Toy / Game Store',
 'Sports Bar',
 'Sporting Goods Shop',
 'Fish & Chips Shop',
 'Bike Shop',
 'Pet Store',
 'Burger Joint',
 'Smoothie Shop',
 'Shopping Mall',
 'Dessert Shop',
 'Brewery',
 'Department Store',
 'Beer Store',
 'Breakfast Spot',
 'Furniture / Home Store',
 'Mexican Restaurant',
 'BBQ Joint',
 'Donut Shop',
 'History Museum',
 'Middle Eastern Restaurant',
 'Convenience Store',
 'Food & Drink Shop',
 'Hotel',
 'Gym / Fitness Center',
 'Burrito Place',
 'Bookstore',
 'Falafel Restaurant',
 'French Restaurant',
 'Latin American Restaurant',
 'Gourmet Shop',
 'Vegetarian / Vegan Restaurant',
 'Indie Movie Theater',
 'Diner',
 'Bar',
 'Tea Room',
 'Health Food Store',
 'Yoga Studio',
 'Electronics Store',
 'Gym',
 'Neighborhood',
 'Asian Restaurant',
 'Climbing Gym',
 'Bakery',
 'Performing Arts Venue',
 'Stadium',
 'Nightclub',
 'Intersection',
 'Theme Restaurant',
 'Dance Studio',
 'Bubble Tea Shop',
 'Beer Bar',
 'Ramen Restaurant',
 'Salon / Barbershop',
 'Creperie',
 'Martial Arts School',
 'Escape Room',
 'Adult Boutique',
 'Ethiopian Restaurant',
 'Hobby Shop',
 'Ice Cream Shop',
 "Men's Store",
 'Gay Bar',
 'Smoke Shop',
 'Steakhouse',
 'Sake Bar',
 'Gastropub',
 'Dog Run',
 'Distribution Center',
 'Mediterranean Restaurant',
 'Theater',
 'Health & Beauty Service',
 'Korean Restaurant',
 'Caribbean Restaurant',
 'Clothing Store',
 'Strip Club',
 'Fast Food Restaurant',
 'Sculpture Garden',
 'Airport',
 'Airport Lounge',
 'Harbor / Marina',
 'Airport Food Court',
 'Airport Terminal',
 'Airport Gate',
 'Plane',
 'Boutique',
 'Airport Service',
 'Rental Car Location',
 'Boat or Ferry',
 'Jewelry Store',
 'Cosmetics Shop',
 'Fruit & Vegetable Store',
 'Tibetan Restaurant',
 'Lounge',
 'Frozen Yogurt Shop',
 'Shopping Plaza',
 'Flea Market',
 'Arts & Crafts Store',
 'Antique Shop',
 'Speakeasy',
 'Music Venue',
 'Discount Store',
 'Cajun / Creole Restaurant',
 'Medical Center',
 'Modern European Restaurant',
 'Seafood Restaurant',
 'Art Museum',
 'Poke Place',
 'Miscellaneous Shop',
 'Comic Shop',
 'Portuguese Restaurant',
 'Office',
 'Salad Place',
 'Wine Bar',
 'Field',
 'Hockey Arena',
 'Concert Hall',
 'Museum',
 'Farmers Market',
 'Cocktail Bar',
 'Fountain',
 'Bistro',
 'Basketball Stadium',
 'Jazz Club',
 'Cheese Shop',
 'Art Gallery',
 'Fish Market',
 'Tailor Shop',
 'Beach',
 'Irish Pub',
 'Eastern European Restaurant',
 'General Entertainment',
 'Skating Rink',
 'College Stadium',
 'Wings Joint',
 'Supplement Shop',
 'Hardware Store',
 'Flower Shop',
 'Tanning Salon',
 'Deli / Bodega',
 'Bridal Shop',
 'Gas Station',
 'Gift Shop',
 'Chinese Restaurant',
 'Movie Theater',
 'College Gym',
 'Video Game Store',
 'College Arts Building',
 'Noodle House',
 'Cuban Restaurant',
 'Garden',
 'New American Restaurant',
 'Record Shop',
 'Malay Restaurant',
 'Cupcake Shop',
 'Historic Site',
 'Chocolate Shop',
 'Event Space',
 'Shoe Store',
 'Plaza',
 'Motel',
 'Curling Ice',
 'Video Store',
 'Golf Course',
 'Pool',
 'Athletics & Sports',
 'Organic Grocery',
 'Belgian Restaurant',
 'Gaming Cafe',
 'Filipino Restaurant',
 'Doner Restaurant',
 'Food Court',
 'Massage Studio',
 'Poutine Place',
 'Hospital',
 'Bed & Breakfast',
 'Construction & Landscaping',
 'Candy Store',
 'Baby Store',
 'Baseball Field',
 'Food Service',
 'Print Shop',
 'Auto Garage',
 "Women's Store",
 'Accessories Store',
 'Carpet Store',
 'Home Service',
 'Food Truck',
 'Drugstore',
 'Garden Center',
 'Taiwanese Restaurant',
 'Market',
 'Snack Place',
 'Gym Pool',
 'Korean BBQ Restaurant',
 'Warehouse Store',
 'Housing Development',
 'Dim Sum Restaurant',
 'Metro Station',
 'Bus Station',
 'Soccer Field',
 'Basketball Court',
 'Lake',
 'IT Services',
 'Roof Deck',
 'Train Station',
 'Aquarium',
 'Monument / Landmark',
 'Scenic Lookout',
 'Baseball Stadium',
 'Hotel Bar',
 'Hakka Restaurant',
 'Opera House',
 'Colombian Restaurant',
 'Brazilian Restaurant',
 'Gluten-free Restaurant',
 'Building',
 'Soup Place',
 'Mobile Phone Shop',
 'Luggage Store',
 'Skate Park',
 'College Auditorium',
 'College Cafeteria',
 'German Restaurant',
 'Lingerie Store',
 'Moroccan Restaurant',
 'General Travel',
 'Taco Place',
 'Other Repair Shop',
 'Stationery Store',
 'Coworking Space',
 'College Rec Center',
 'Other Great Outdoors',
 'Hookah Bar',
 'Molecular Gastronomy Restaurant',
 'Church',
 'Optical Shop']
```

In [61]:
```
restuarant_list = ['Steakhouse', 'Coffee Shop', 'Café', 'Ramen Restaurant', 'Indonesian Restaurant', 'Restaurant', 'Japanese Restaurant',
             'Fast Food Restaurant', 'Sushi Restaurant', 'Vietnamese Restaurant', 'Pizza Place', 'Sandwich Place', 'Middle Eastern Restaurant'
,
             'Burger Joint', 'American Restaurant', 'Food Court', 'Wings Joint', 'Burrito Place', 'Asian Restaurant', 'Deli / Bodega',
             'Greek Restaurant', 'Fried Chicken Joint', 'Airport Food Court', 'Chinese Restaurant', 'Breakfast Spot', 'Mexican Restaurant',
             'Indian Restaurant', 'Latin American Restaurant', 'Bar',
'Pub', 'Italian Restaurant', 'French Restaurant', 'Ice Cream Shop',
             'Caribbean Restaurant', 'Gastropub', 'Thai Restaurant', 'C
ajun / Creole Restaurant', 'Diner', 'Dim Sum Restaurant', 'Seafood Rest
aurant',
             'Food & Drink Shop', 'Noodle House', 'Food', 'Fish & Chips
 Shop', 'Falafel Restaurant', 'Gourmet Shop', 'Vegetarian / Vegan Resta
urant',
             'South American Restaurant', 'Korean Restaurant', 'Cuban R
estaurant', 'New American Restaurant', 'Malay Restaurant', 'Mac & Chees
e Joint',
             'Bistro', 'Southern / Soul Food Restaurant', 'Tapas Restau
rant',  'Sports Bar', 'Polish Restaurant', 'Ethiopian Restaurant',
             'Creperie', 'Sake Bar', 'Persian Restaurant', 'Afghan Rest
aurant','Mediterranean Restaurant', 'BBQ Joint', 'Jewish Restaurant',
             'Comfort Food Restaurant',  'Hakka Restaurant', 'Food Truc
k', 'Taiwanese Restaurant',  'Snack Place', 'Eastern European Restauran
t',
             'Dumpling Restaurant', 'Belgian Restaurant', 'Arepa Restau
rant', 'Taco Place', 'Doner Restaurant', 'Filipino Restaurant',
             'Hotpot Restaurant', 'Poutine Place', 'Salad Place',  'Por
tuguese Restaurant', 'Modern European Restaurant', 'Empanada Restauran
t',
             'Irish Pub', 'Molecular Gastronomy Restaurant', 'German Re
staurant', 'Brazilian Restaurant', 'Gluten-free Restaurant', 'Soup Plac
e']

restuarant_pd = pd.DataFrame(restuarant_list)

restuarant_pd = restuarant_pd.rename(columns={0:'Venue Category'})

Newframe = pd.merge(Venues1, restuarant_pd, on='Venue Category', how='r
ight')

Newframe.groupby('Neighborhood').count()
```

Out[61]:

	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
Neighborhood						

Agincourt	2	2	2	2	2	2
Alderwood, Long Branch	5	5	5	5	5	5
Bathurst Manor, Wilson Heights, Downsview North	12	12	12	12	12	12
Bayview Village	3	3	3	3	3	3

	Neighborhood Latitude	**Neighborhood Longitude**	**Venue**	**Venue Latitude**	**Venue Longitude**	**Venue Category**
Neighborhood						
Bedford Park, Lawrence Manor East	17	17	17	17	17	17
...	...	...	...	...	...	...
Westmount	5	5	5	5	5	5
Wexford, Maryvale	3	3	3	3	3	3
Willowdale, Willowdale East	20	20	20	20	20	20
Willowdale, Willowdale West	2	2	2	2	2	2
Woburn	2	2	2	2	2	2

79 rows × 6 columns

In [58]:
```
Newframe = Newframe.dropna(axis=0, subset=['Venue'])
```

In [62]:
```
Onehot1 = pd.get_dummies(Newframe[['Venue Category']], prefix="", prefi
x_sep="")

Onehot1['Neighborhood'] = Newframe['Neighborhood']


# move neighborhood column to the first column
fixed_columns = [Onehot1.columns[-1]] + list(Onehot1.columns[:-1])
```

```
Onehot1 = Onehot1[fixed_columns]

Onehot1.head()
```

Out[62]:

	Neighborhood	Afghan Restaurant	Airport Food Court	American Restaurant	Arepa Restaurant	Asian Restaurant	BBQ Joint	Bar	Belgian Restaurant
0	Moore Park, Summerhill East	0	0	0	0	0	0	0	0
1	Summerhill West, Rathnelly, South Hill, Forest...	0	0	0	0	0	0	0	0
2	Bedford Park, Lawrence Manor East	0	0	0	0	0	0	0	0
3	Leaside	0	0	0	0	0	0	0	0
4	Runnymede, Swansea	0	0	0	0	0	0	0	0

5 rows × 91 columns

In [63]:

```
Group1 = Onehot1.groupby('Neighborhood').mean().reset_index()
Group1.shape


Group1.head()
```

Out[63]:

	Neighborhood	Afghan Restaurant	Airport Food Court	American Restaurant	Arepa Restaurant	Asian Restaurant	BBQ Joint	Bar	Belgian Restaurant
0	Agincourt	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	0.0

	Neighborhood	Afghan Restaurant	Airport Food Court	American Restaurant	Arepa Restaurant	Asian Restaurant	BBQ Joint	Bar	Belgian Restaurant
1	Alderwood, Long Branch	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	0.0
2	Bathurst Manor, Wilson Heights, Downsview North	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	0.0
3	Bayview Village	0.0	0.0	0.000000	0.0	0.0	0.0	0.0	0.0
4	Bedford Park, Lawrence Manor East	0.0	0.0	0.058824	0.0	0.0	0.0	0.0	0.0

5 rows × 91 columns

Utilize Sillhouette Score to Segment Data

In [87]:

```
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score
import numpy as np


Groupedclustering = Group1.drop('Neighborhood', 1)

kclusters = np.arange(2,10)
results = {}
for size in kclusters:
    model = KMeans(n_clusters = size).fit(Groupedclustering)
    predictions = model.predict(Groupedclustering)
    results[size] = silhouette_score(Groupedclustering, predictions)

Sizefit = max(results, key=results.get)
Sizefit
```

```
Out[87]: 8
```

```
In [88]: from sklearn.cluster import KMeans

kclusters = Sizefit

kmeans = KMeans(n_clusters=kclusters, random_state=0).fit(Groupedcluste
ring)

kmeans.labels_[0:10]
```

```
Out[88]: array([0, 2, 0, 0, 0, 0, 0, 0, 0, 1], dtype=int32)
```

```
In [89]: 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]

num_top_venues = 10

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

columns = ['Neighborhood']
for ind in np.arange(num_top_venues):
    try:
        columns.append('{}{} Most Common Venue'.format(ind+1, indicator
s[ind]))
    except:
        columns.append('{}th Most Common Venue'.format(ind+1))

neighborhoods_sortedvenues = pd.DataFrame(columns=columns)
neighborhoods_sortedvenues['Neighborhood'] = Group1['Neighborhood']

for ind in np.arange(Group1.shape[0]):
    neighborhoods_sortedvenues.iloc[ind, 1:] = return_most_common_venue
s(Group1.iloc[ind, :], num_top_venues)

neighborhoods_sortedvenues.head()
```

Out[89]:

	Neighborhood	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue	7th Most Common Venue
0	Agincourt	Breakfast Spot	Latin American Restaurant	Food	Doner Restaurant	Dumpling Restaurant	Eastern European Restaurant	Empanada Restaurant
1	Alderwood, Long Branch	Pizza Place	Pub	Coffee Shop	Sandwich Place	Falafel Restaurant	Dim Sum Restaurant	Diner
2	Bathurst Manor, Wilson Heights, Downsview North	Coffee Shop	Pizza Place	Sandwich Place	Diner	Middle Eastern Restaurant	Chinese Restaurant	Restaurant
3	Bayview Village	Chinese Restaurant	Japanese Restaurant	Café	Wings Joint	Fish & Chips Shop	Dumpling Restaurant	Eastern European Restaurant
4	Bedford Park, Lawrence Manor East	Italian Restaurant	Coffee Shop	Sandwich Place	Indian Restaurant	Pizza Place	Pub	Restaurant

In [90]:

```
Labeled = pd.merge(Data1,Group1, on='Neighborhood', how='right')

Labeled.shape


Labeled = Labeled.drop(columns=['Steakhouse', 'Coffee Shop', 'Café', 'R
amen Restaurant', 'Indonesian Restaurant', 'Restaurant', 'Japanese Rest
aurant',
                'Fast Food Restaurant', 'Sushi Restaurant', 'Vietnamese Re
staurant', 'Pizza Place', 'Sandwich Place', 'Middle Eastern Restaurant'
,
                'Burger Joint', 'American Restaurant', 'Food Court', 'Wing
s Joint', 'Burrito Place', 'Asian Restaurant', 'Deli / Bodega',
                'Greek Restaurant', 'Fried Chicken Joint', 'Airport Food C
ourt', 'Chinese Restaurant', 'Breakfast Spot', 'Mexican Restaurant',
              'Indian Restaurant', 'Latin American Restaurant', 'Bar',
'Pub', 'Italian Restaurant', 'French Restaurant', 'Ice Cream Shop',
              'Caribbean Restaurant', 'Gastropub', 'Thai Restaurant', 'C
ajun / Creole Restaurant', 'Diner', 'Dim Sum Restaurant', 'Seafood Rest
aurant',
              'Food & Drink Shop', 'Noodle House', 'Food', 'Fish & Chips
 Shop', 'Falafel Restaurant', 'Gourmet Shop', 'Vegetarian / Vegan Resta
urant',
              'South American Restaurant', 'Korean Restaurant', 'Cuban R
estaurant', 'New American Restaurant', 'Malay Restaurant', 'Mac & Chees
e Joint',
              'Bistro', 'Southern / Soul Food Restaurant', 'Tapas Restau
rant',  'Sports Bar', 'Polish Restaurant', 'Ethiopian Restaurant',
              'Creperie', 'Sake Bar', 'Persian Restaurant', 'Afghan Rest
aurant','Mediterranean Restaurant', 'BBQ Joint', 'Jewish Restaurant',
              'Comfort Food Restaurant',  'Hakka Restaurant', 'Food Truc
k', 'Taiwanese Restaurant',  'Snack Place', 'Eastern European Restauran
t',
              'Dumpling Restaurant', 'Belgian Restaurant', 'Arepa Restau
rant', 'Taco Place', 'Doner Restaurant', 'Filipino Restaurant',
              'Hotpot Restaurant', 'Poutine Place', 'Salad Place',  'Por
tuguese Restaurant', 'Modern European Restaurant', 'Empanada Restauran
t',
              'Irish Pub', 'Molecular Gastronomy Restaurant', 'German Re
staurant', 'Brazilian Restaurant', 'Gluten-free Restaurant', 'Soup Plac
e'])
Labeled.head()
```

Out[90]:

	Postal Code	Average After Tax Income	Population_2016	Borough	Neighborhood	Latitude	Longitude
0	M4T	134865.0	10463.0	Central Toronto	Moore Park, Summerhill East	43.689574	-79.383160
1	M4V	115033.0	18241.0	Central Toronto	Summerhill West, Rathnelly, South Hill, Forest...	43.686412	-79.400049
2	M5M	85678.0	25975.0	North York	Bedford Park, Lawrence Manor East	43.733283	-79.419750

3	M4G	85496.0	19076.0	East York	Leaside	43.709060	-79.363452
4	M5R	80138.0	26496.0	Central Toronto	The Annex, North Midtown, Yorkville	43.672710	-79.405678

In [97]:
```
Merged1 = Labeled

#Merged1['Cluster Labels'] = kmeans.labels_

Merged1 = Merged1.join(neighborhoods_sortedvenues.set_index('Neighborho
od'), on='Neighborhood')
print(kmeans.labels_)

Merged1.head()
```

```
[0 2 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 5 0 2 0 0 4 0 0 0 0 0
0 6 2
 0 0 0 2 0 0 0 6 0 7 0 0 1 0 0 2 3 0 0 0 1 0 2 0 0 0 0 0 0 0 0 2 0 0
0 0 0
 2 0 0 2 0]
```

Out[97]:

	Postal Code	Average After Tax Income	Population_2016	Borough	Neighborhood	Latitude	Longitude	1st Mos Commo Venu
0	M4T	134865.0	10463.0	Central Toronto	Moore Park, Summerhill East	43.689574	-79.383160	Restaura
1	M4V	115033.0	18241.0	Central Toronto	Summerhill West, Rathnelly, South Hill, Forest...	43.686412	-79.400049	Coffe Sho
2	M5M	85678.0	25975.0	North York	Bedford Park, Lawrence Manor East	43.733283	-79.419750	Italia Restaura

	Postal Code	Average After Tax Income	Population_2016	Borough	Neighborhood	Latitude	Longitude	1st Mos Commo Venu
3	M4G	85496.0	19076.0	East York	Leaside	43.709060	-79.363452	Coffe Sho
4	M5R	80138.0	26496.0	Central Toronto	The Annex, North Midtown, Yorkville	43.672710	-79.405678	Caf

In [116]:
```
Cluster_0_coorid = Merged1[['Latitude', 'Longitude']]
Cluster_0_coorid = list(Cluster_0_coorid.values)
lat = []
long = []



for l in Cluster_0_coorid:
  lat.append(l[0])
  long.append(l[1])



Blatitude = sum(lat)/len(lat)
Blongitude = sum(long)/len(long)
print(Blatitude)
print(Blongitude)
```

```
43.701714456626505
-79.39324610240962
```

In [115]:
```
!pip install opencage
from opencage.geocoder import OpenCageGeocode
from pprint import pprint

pprint(results)
```

```
Requirement already satisfied: opencage in /opt/conda/envs/Python-3.7-m
ain/lib/python3.7/site-packages (1.2.2)
Requirement already satisfied: pyopenssl>=0.15.1 in /opt/conda/envs/Pyt
hon-3.7-main/lib/python3.7/site-packages (from opencage) (19.1.0)
Requirement already satisfied: Requests>=2.2.0 in /opt/conda/envs/Pytho
n-3.7-main/lib/python3.7/site-packages (from opencage) (2.24.0)
Requirement already satisfied: six>=1.4.0 in /opt/conda/envs/Python-3.7
-main/lib/python3.7/site-packages (from opencage) (1.15.0)
Requirement already satisfied: backoff>=1.10.0 in /opt/conda/envs/Pytho
n-3.7-main/lib/python3.7/site-packages (from opencage) (1.10.0)
Requirement already satisfied: cryptography>=2.8 in /opt/conda/envs/Pyt
hon-3.7-main/lib/python3.7/site-packages (from pyopenssl>=0.15.1->openc
age) (3.2.1)
Requirement already satisfied: chardet<4,>=3.0.2 in /opt/conda/envs/Pyt
hon-3.7-main/lib/python3.7/site-packages (from Requests>=2.2.0->opencag
e) (3.0.4)
Requirement already satisfied: certifi>=2017.4.17 in /opt/conda/envs/Py
thon-3.7-main/lib/python3.7/site-packages (from Requests>=2.2.0->openca
ge) (2020.12.5)
Requirement already satisfied: idna<3,>=2.5 in /opt/conda/envs/Python-
3.7-main/lib/python3.7/site-packages (from Requests>=2.2.0->opencage)
(2.9)
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1
in /opt/conda/envs/Python-3.7-main/lib/python3.7/site-packages (from Re
quests>=2.2.0->opencage) (1.25.9)
Requirement already satisfied: cffi!=1.11.3,>=1.8 in /opt/conda/envs/Py
thon-3.7-main/lib/python3.7/site-packages (from cryptography>=2.8->pyop
enssl>=0.15.1->opencage) (1.14.0)
Requirement already satisfied: pycparser in /opt/conda/envs/Python-3.7-
main/lib/python3.7/site-packages (from cffi!=1.11.3,>=1.8->cryptography
>=2.8->pyopenssl>=0.15.1->opencage) (2.20)
{2: 0.15585874455713347,
 3: 0.16969064732471367,
 4: 0.19188364869285318,
 5: 0.057621136614974665,
 6: 0.1997041382209193,
 7: 0.08293214201469745,
 8: 0.22066544188682236,
 9: 0.17386543570695392}
```

In [102]:
```
!pip install folium
```

```
Collecting folium
  Downloading folium-0.11.0-py2.py3-none-any.whl (93 kB)
     |████████████████████████████████| 93 kB 3.6 MB/s  eta 0:00:01
Requirement already satisfied: numpy in /opt/conda/envs/Python-3.7-mai
n/lib/python3.7/site-packages (from folium) (1.18.5)
Requirement already satisfied: jinja2>=2.9 in /opt/conda/envs/Python-3.
7-main/lib/python3.7/site-packages (from folium) (2.11.2)
Collecting branca>=0.3.0
  Downloading branca-0.4.2-py3-none-any.whl (24 kB)
Requirement already satisfied: requests in /opt/conda/envs/Python-3.7-m
ain/lib/python3.7/site-packages (from folium) (2.24.0)
Requirement already satisfied: MarkupSafe>=0.23 in /opt/conda/envs/Pyth
on-3.7-main/lib/python3.7/site-packages (from jinja2>=2.9->folium) (1.
1.1)
Requirement already satisfied: idna<3,>=2.5 in /opt/conda/envs/Python-
3.7-main/lib/python3.7/site-packages (from requests->folium) (2.9)
Requirement already satisfied: certifi>=2017.4.17 in /opt/conda/envs/Py
thon-3.7-main/lib/python3.7/site-packages (from requests->folium) (202
0.12.5)
Requirement already satisfied: chardet<4,>=3.0.2 in /opt/conda/envs/Pyt
hon-3.7-main/lib/python3.7/site-packages (from requests->folium) (3.0.
4)
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1
in /opt/conda/envs/Python-3.7-main/lib/python3.7/site-packages (from re
quests->folium) (1.25.9)
Installing collected packages: branca, folium
Successfully installed branca-0.4.2 folium-0.11.0
```

In [110]:
```
Merged1.head()
```

Out[110]:

	Postal Code	Average After Tax Income	Population_2016	Borough	Neighborhood	Latitude	Longitude	1st Most Common Venue

	Postal Code	Average After Tax Income	Population_2016	Borough	Neighborhood	Latitude	Longitude	1st Most Common Venue
0	M4T	134865.0	10463.0	Central Toronto	Moore Park, Summerhill East	43.689574	-79.383160	Restaurant
1	M4V	115033.0	18241.0	Central Toronto	Summerhill West, Rathnelly, South Hill, Forest...	43.686412	-79.400049	Coffee Shop
2	M5M	85678.0	25975.0	North York	Bedford Park, Lawrence Manor East	43.733283	-79.419750	Italian Restaurant
3	M4G	85496.0	19076.0	East York	Leaside	43.709060	-79.363452	Coffee Shop
4	M5R	80138.0	26496.0	Central Toronto	The Annex, North Midtown, Yorkville	43.672710	-79.405678	Café

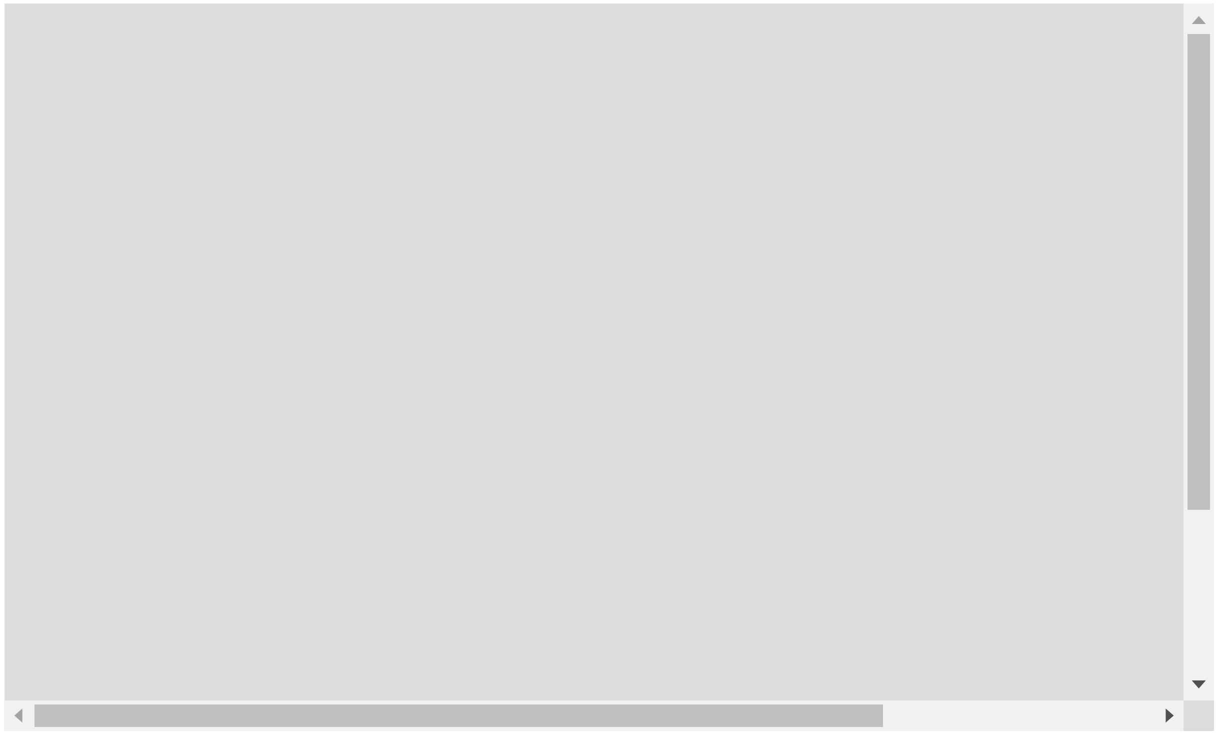
In [126]:

```
# getfolium
import folium
# Matplotlib and associated plotting modules
import matplotlib.cm as cm
import matplotlib.colors as colors

# create map
map_clusters = folium.Map(location=[43.689574, -79.383160], zoom_start=
11)
for lat, lng, label in zip(Merged1['Latitude'], Merged1['Longitude'], M
erged1['Neighborhood']):
    label = folium.Popup(label, parse_html=True)
    folium.CircleMarker(
        [43.689574, -79.383160],
        radius=5,
        popup=label,
        color='blue',
        fill=True,
        fill_color='#3186cc',
        fill_opacity=0.7,
        parse_html=False).add_to(map_clusters)

map_clusters
```

Out[126]:



The best location for a steakhouse is at this location

In [ ]: