Caribbbean Restaurant: Data Science for Desicion Making

Clustering the Neighbourhoods of London and Toronto to select the perfect location for a Caribbean Restaurant

Introduction

In this Assignment, I would like to create a scenario, where I would like to compare between the cities of London and Toronto for decide which is the most ideal city to locate a Caribbean Restaurant. There are many varieties of foods of Latin origin but I would like to know specifically Caribbean food. Also with this project it will be possible to see which communities or neighborhoods frequently visit or reside the Latino community or people of other ethnic groups who like Latino food.

Business Problem

It is a challenge to establish a food business and it is more complex in a place where it may not consume as much. The Latino community continues to grow in these countries, which represents a long-term, prosperous business. My Project wants to establish a Caribbean food concept that fits the city's environment. Most of the Latin food options are more Mexican food oriented but not so much Caribbean food. In part, it is intended that the Latino residents of these cities have options where to eat or brings a gastronomic offer distant from those that already exist. My priority is to know which would be the city where my gastronomic offer could have the greatest acceptance. Using data science and machine learning and data visualization methods and tools will help me establish better decision-making.

Data Description

We require geographical location data for both London and Toronto. Postal codes in each city serve as a starting point. Using Postal codes we use can find out the neighborhoods, boroughs, venues and their most popular venue categories specially food places.

London Data

To derive our solution, We scrape our data from https://en.wikipedia.org/wiki/List_of_areas_of_London

This wikipedia page has information about all the neighbourhoods, we limit it London.

1. borough : Name of Neighbourhood
2. town : Name of borough
3. post_code : Postal codes for London

This wikipedia page have limited information about the geographical locations. To solve this problem we use ArcGIS API

Toronto Data

To derive our solution, We scrape our data fromhttps://en.wikipedia.org/wiki/List_of_postal_codes_of_Canada:_M

This wikipedia page has information about all the neighbourhoods, we limit it Toronto.

1. borough : Name of Neighbourhood
2. town : Name of borough
3. post_code : Postal codes for Canada

This wikipedia page lacks information about the geographical locations. To solve this problem we use ArcGIS API

ArcGIS API

ArcGIS Online enables you to connect people, locations, and data using interactive maps. Work with smart, data-driven styles and intuitive analysis tools that deliver location intelligence. Share your insights with the world or specific groups.

More specifically, we use ArcGIS to get the geo locations of the neighbourhoods of London and Toronto. The following columns are added to our initial dataset which prepares our data.

1. latitude : Latitude for Neighbourhood
2. longitude : Longitude for Neighbourhood

Foursquare API Data

After finding the list of neighbourhoods, we then connect to the Foursquare API to gather information about venues inside each and every neighbourhood. For each neighbourhood, we have chosen the radius to be 500 meters.

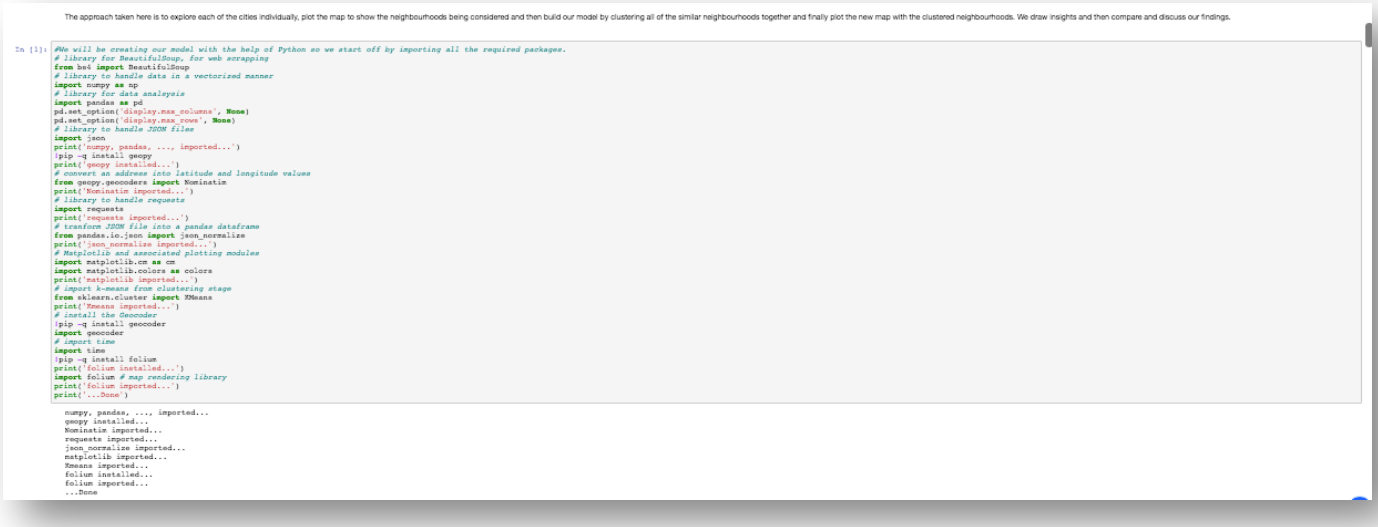
The data retrieved from Foursquare contained information of venues within a specified distance of the longitude and latitude of the postcodes. The information obtained per venue as follows:

1. Neighbourhood : Name of the Neighbourhood
2. Neighbourhood Latitude : Latitude of the Neighbourhood
3. Neighbourhood Longitude : Longitude of the Neighbourhood
4. Venue : Name of the Venue
5. Venue Latitude : Latitude of Venue
6. Venue Longitude : Longitude of Venue
7. Venue Category : Category of Venue

Methodology

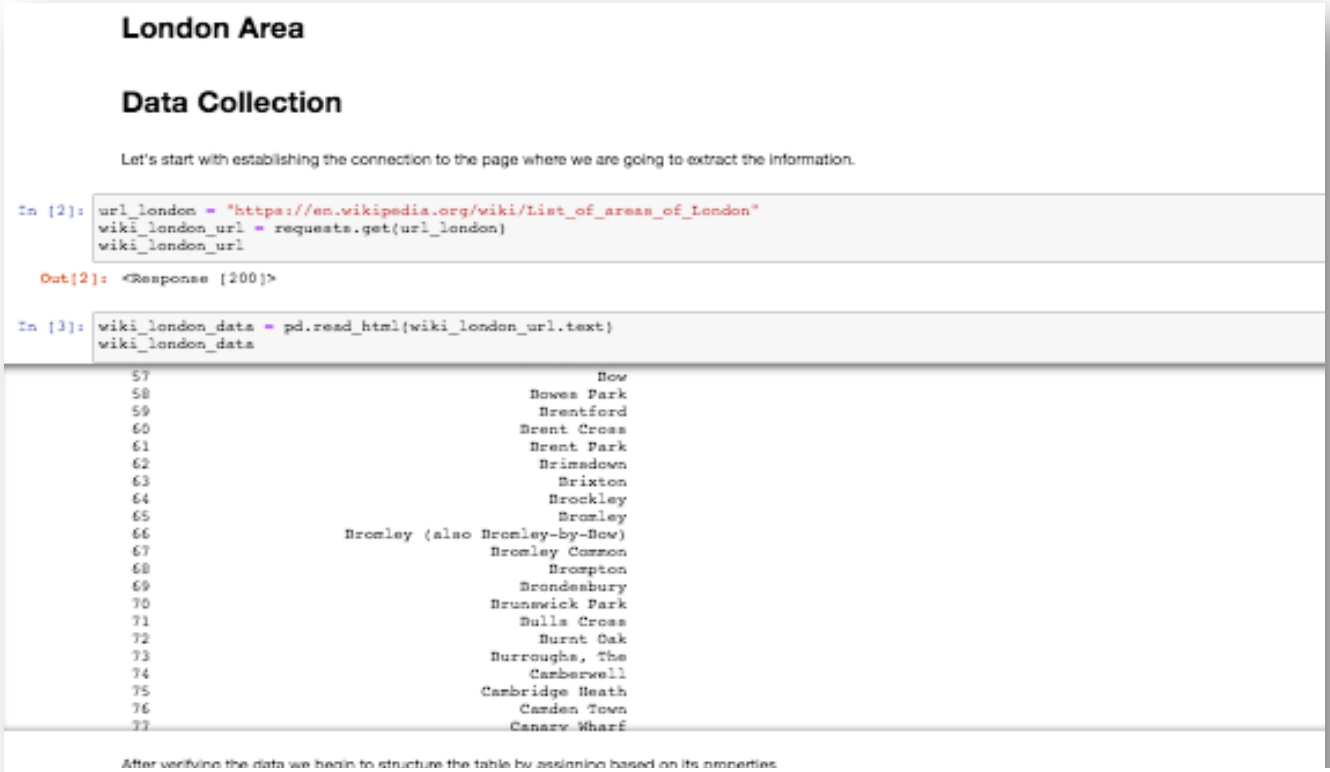
The approach taken here is to explore each of the cities individually, plot the map to show the neighbourhoods being considered and then build our model by clustering all of the similar neighbourhoods together and finally plot the new map with the clustered neighbourhoods. We draw insights and then compare and discuss our findings.

The libraries are imported in order to perform the analysis and visualization of clusters for each city:



Collect Data

The data is collected using information available on publicly accessible and geographically referenced websites of both cities. This data is verified and sorted so that it can be processed for further analysis.



London Area

Data Collection

Let's start with establishing the connection to the page where we are going to extract the information.

```
In [2]: url_london = "https://en.wikipedia.org/wiki/List_of_areas_of_London"
        wiki_london_url = requests.get(url_london)
        wiki_london_url

Out[2]: <Response [200]>

In [3]: wiki_london_data = pd.read_html(wiki_london_url.text)
        wiki_london_data
```

57	How
58	Bowes Park
59	Brentford
60	Brent Cross
61	Brent Park
62	Brimsdown
63	Brixton
64	Brockley
65	Bromley
66	Bromley (also Bromley-by-Bow)
67	Bromley Common
68	Brompton
69	Brondesbury
70	Brunswick Park
71	Bulls Cross
72	Burnt Oak
73	Burroughs, The
74	Camberwell
75	Cambridge Heath
76	Camden Town
77	Canary Wharf

After verifying the data we begin to structure the table by assigning based on its properties

Toronto Area

Data Collection

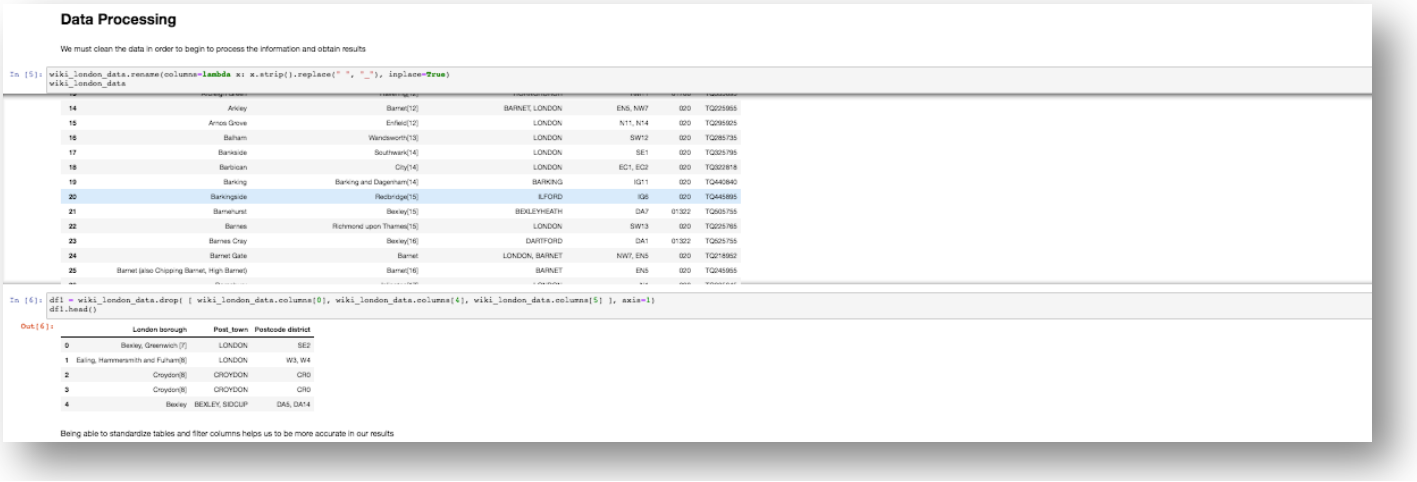
```
In [51]: url = "https://en.wikipedia.org/wiki/List_of_postal_codes_of_Canada:_M"
         wiki_url = requests.get(url)
         wiki_url

Out[51]: <Response [200]>

In [52]: wiki_data = pd.read_html(wiki_url.text)
         wiki_data
```

Out[52]: [

	Postal Code	Borough
0	M1A	Not assigned
1	M2A	Not assigned
2	M3A	North York
3	M4A	North York
4	M5A	Downtown Toronto
5	M6A	North York
6	M7A	Downtown Toronto
7	M8A	Not assigned
8	M9A	Etobicoke
9	M1B	Scarborough
10	M2B	Not assigned
11	M3B	North York
12	M4B	East York
13	M5B	Downtown Toronto
14	M6B	North York
15	M7B	Not assigned
16	M8B	Not assigned
17	M9B	Etobicoke
18	M1C	Scarborough

Data Processing

Parameters are established to measure each of the columns to be evaluated. The columns the town are changed to borough and the neighborhoods are filtered by the city we are analyzing London and Toronto. All characters and numbers are eliminated in order to have the information as clean as possible. This information is filtered so that we can standardize each of the processes we want to search in Foursquare API.



Data Processing

We must clean the data in order to begin to process the information and obtain results

```
In [5]: wiki_london_data.rename(columns=lambda x: x.strip().replace(" ", "_"), inplace=True)
        wiki_london_data
```

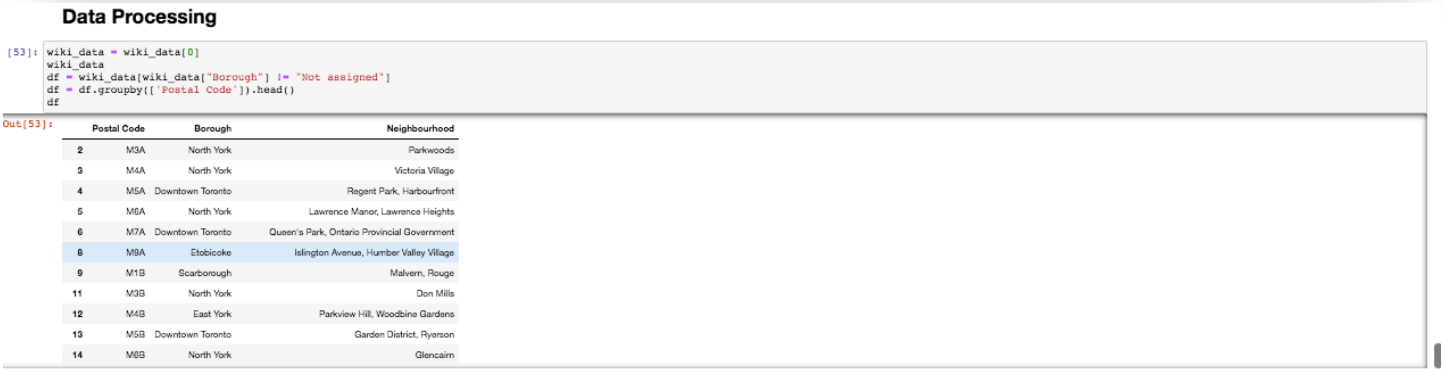
14	Arkley	Barnet[12]	BARNET, LONDON	EN5, NW7	020	TQ2259...
15	Arnos Grove	Enfield[12]	LONDON	N11, N14	020	TQ2995...
16	Balham	Wandsworth[13]	LONDON	SW12	020	TQ2857...
17	Barkside	Southwark[14]	LONDON	SE1	020	TQ3257...
18	Barbican	City[14]	LONDON	EC1, EC2	020	TQ3218...
19	Barking	Barking and Dagenham[14]	BARKING	IG11	020	TQ4408...
20	Barkingside	Redbridge[15]	ILFORD	IG6	020	TQ4458...
21	Barnehurst	Bexley[15]	BEXLEYHEATH	DA7	01322	TQ5057...
22	Barnes	Richmond upon Thames[15]	LONDON	SW13	020	TQ2257...
23	Barnes Cray	Bexley[16]	DARTFORD	DA1	01322	TQ5257...
24	Barnet Gate	Barnet	LONDON, BARNET	NW7, EN5	020	TQ2158...
25	Barnet (also Chipping Barnet, High Barnet)	Barnet[16]	BARNET	EN5	020	TQ2459...

```
In [6]: df1 = wiki_london_data.drop( [ wiki_london_data.columns[0], wiki_london_data.columns[4], wiki_london_data.columns[5] ], axis=1)
        df1.head()
```

Out[6]:

	London borough	Post_town	Postcode district
0	Bexley, Greenwich [7]	LONDON	SE2
1	Ealing, Hammersmith and Fulham[8]	LONDON	W3, W4
2	Croydon[8]	CROYDON	CR0
3	Croydon[8]	CROYDON	CR0
4	Bexley	BEXLEY, SIDCUP	DA5, DA14

Being able to standardize tables and filter columns helps us to be more accurate in our results



Data Processing

```
[53]: wiki_data = wiki_data[0]
      wiki_data
      df = wiki_data[wiki_data["Borough"] != "Not assigned"]
      df = df.groupby(['Postal Code']).head()
      df
```

Out[53]:

	Postal Code	Borough	Neighbourhood
2	M3A	North York	Parkwoods
3	M4A	North York	Victoria Village
4	M5A	Downtown Toronto	Regent Park, Harbourfront
5	M6A	North York	Lawrence Manor, Lawrence Heights
6	M7A	Downtown Toronto	Queen's Park, Ontario Provincial Government
8	M9A	Etobicoke	Islington Avenue, Humber Valley Village
9	M1B	Scarborough	Malvern, Rouge
11	M3B	North York	Don Mills
12	M4B	East York	Parkview Hill, Woodbine Gardens
13	M5B	Downtown Toronto	Garden District, Ryerson
14	M6B	North York	Glencairn

Geolocation Process

The geolocation of each borough was achieved for the city of London using the ArcGIS API, which has a very complete library of precise coordinates of each London neighborhood. While for Toronto I used the same data as for the last lab, the document : "https://cocl.us/Geospatial_data" ,contains complete information on postal codes, cities and can be used to identify the latitudes and longitudes of the boroughs of Toronto.

Geolocations of the London Neighbourhoods

ArcGis API

Its provide a modern and easy to use Pythonic library to perform GIS visualization and analysis, spatial data management and GIS system administration tasks that can run both interactively, and using scripts

In [13]:
```
pip install arcgis
```

```
Requirement already satisfied: arcgis in /opt/conda/envs/Python-3.7-main/lib/python3.7/site-packages (1.6.0)
Note: you may need to restart the kernel to use updated packages.
```

In [14]:
```
from arcgis.geocoding import geocode
from arcgis.gis import GIS
gis = GIS()
```

```
/opt/conda/envs/Python-3.7-main/lib/python3.7/site-packages/arcgis/features/_data/geodataset/utils.py:16: FutureWarning: The pandas.datetime class is deprecated and will be removed from pandas in a future version. Import from datetime module instead.
  pd.datetime,
```

We create a function with which we can set the required properties for the geolocation of London

Longitude

Extracting the longitude from our previously collected coordinates

In [21]:
```
lng_uk = coordinates_latlng_uk.apply(lambda x: x.split(',')[1])
lng_uk
```

Out[21]:
```
0        0.12127000000003818
1        -0.2674599999999714
6        -0.0805799999994081
7        -0.1196799999995982
9        -0.0560299999993399
10       -0.0987699999994496
12       -0.1294499999996298
14        -0.245109999999542
15       -0.1336599999996347
16        -0.1403899999994494
17       -0.0956799999995891
18       -0.0881499999992779
22       -0.2416299999992962
24        -0.245109999999542
26       -0.0869699999995114
27       -0.1628999999997917
28        -0.180479999999318
29       -0.0560299999993399
30      0.015280000000075233
34       -0.2619399999992457
```

In [22]:
```
london_merged = pd.concat([df1,lat_uk.astype(float), lng_uk.astype(float)], axis=1)
london_merged.columns= ['borough','town','post code','latitude','longitude']
```

Latitude

Extracting the latitude from our previously collected coordinates

In [19]:
```
coordinates_latlng_uk
```

Out[19]:
```
0       51.492450000000076,0.12127000000003818
1        51.51324000000005,-0.2674599999999714
6       51.512000000000064,-0.0805799999994081
7       51.516510000000004,-0.1196799999995982
9       51.410090000000008,-0.0560299999993399
10     51.523610000000076,-0.0987699999994496
12       51.563930000000003,-0.1294499999996298
14      51.615680000000054,-0.2451099999999542
15       51.634290000000008,-0.1336599999996347
16        51.44822000000005,-0.1403899999994494
17     51.499960000000044,-0.0956799999995891
18     51.518410000000074,-0.0881499999992779
22       51.474690000000007,-0.2416299999992962
24      51.615680000000054,-0.2451099999999542
26       51.529690000000007,-0.0869699999995114
27       51.467600000000006,-0.1628999999997917
28        51.51494000000008,-0.1804799999999318
29       51.410090000000008,-0.0560299999993399
30      51.509130000000003,0.015280000000075233
34       51.489410000000004,-0.2619399999992457
```

In [56]:
```
data = pd.read_csv("https://cocl.us/Geospatial_data")
data
```

Out[56]:

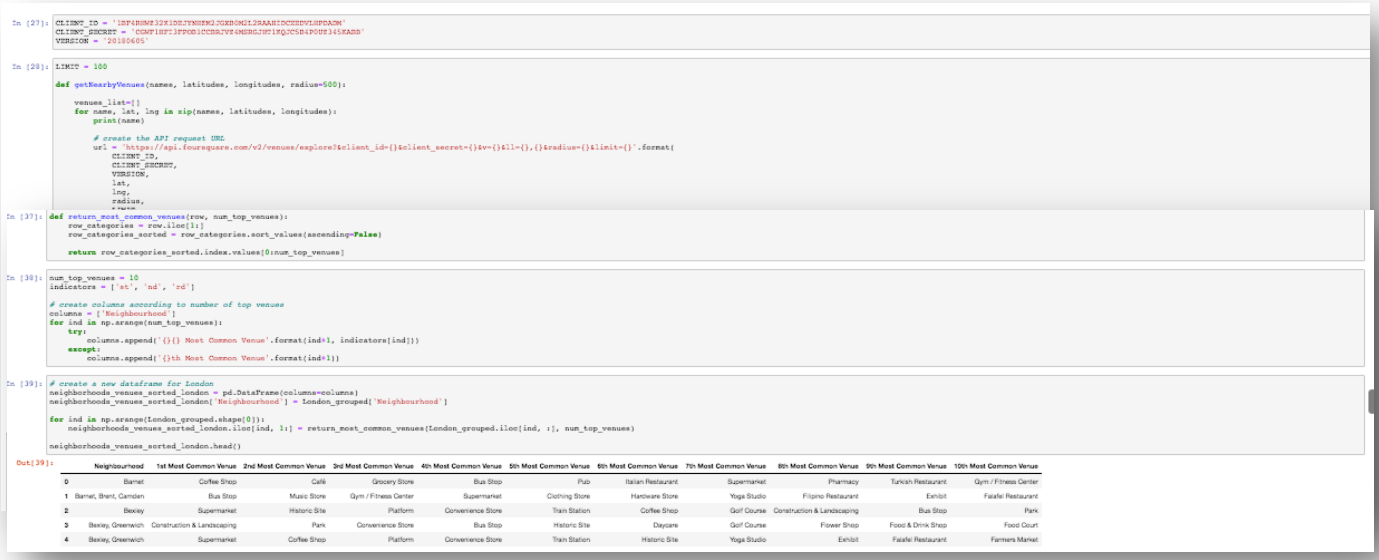
	Postal Code	Latitude	Longitude
0	M1B	43.806686	-79.194353
1	M1C	43.784535	-79.160497
2	M1E	43.763573	-79.188711
3	M1G	43.770992	-79.216917
4	M1H	43.773136	-79.239476
5	M1J	43.744734	-79.239476
6	M1K	43.727929	-79.262029
7	M1L	43.711112	-79.284577
8	M1M	43.716316	-79.239476
9	M1N	43.692657	-79.264848
10	M1P	43.757410	-79.273304

In [58]:
```
address = 'Toronto, Ontario'

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

```
The coordinates of Toronto are 43.6534817, -79.3839347.
```

Foursquare API

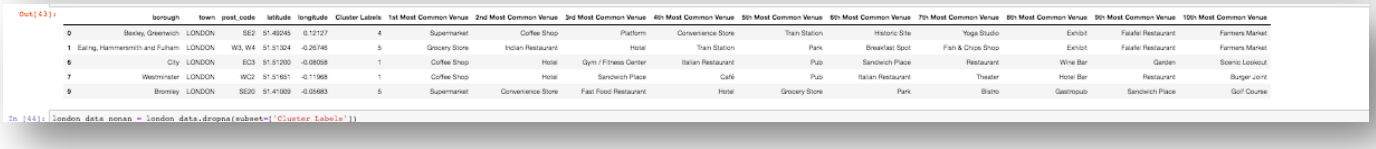
Once we have the geographic coordinates we proceed to enter the Foursquare API account to make our searches. What we are looking for is a general profile by categories of all the services offered by the cities. The services are filtered by the most common uses of the cities and tables are created to define the most common categories in both cities.



```
In [27]: CLIENT_ID = '1BF4RHWZ32K1DEJYNHZM2JGXB0M2L2RAAHIDCEEDVLHPDADM'
         CLIENT_SECRET = 'CGWF1HFI3PPOB1CCBRJVZ4MSRGJHT1KQJC5B4P0U2345KABB'
         VERSION = '20180605'

In [28]: LIMIT = 100

         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,

In [37]: 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 [38]: num_top_venues = 10
         indicators = ['st', 'nd', 'rd']

         # create columns according to number of top venues
         columns = ['Neighbourhood']
         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))

In [39]: # create a new dataframe for London
         neighborhoods_venues_sorted_london = pd.DataFrame(columns=columns)
         neighborhoods_venues_sorted_london['Neighbourhood'] = London_grouped['Neighbourhood']

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

         neighborhoods_venues_sorted_london.head()
```

Out[39]:

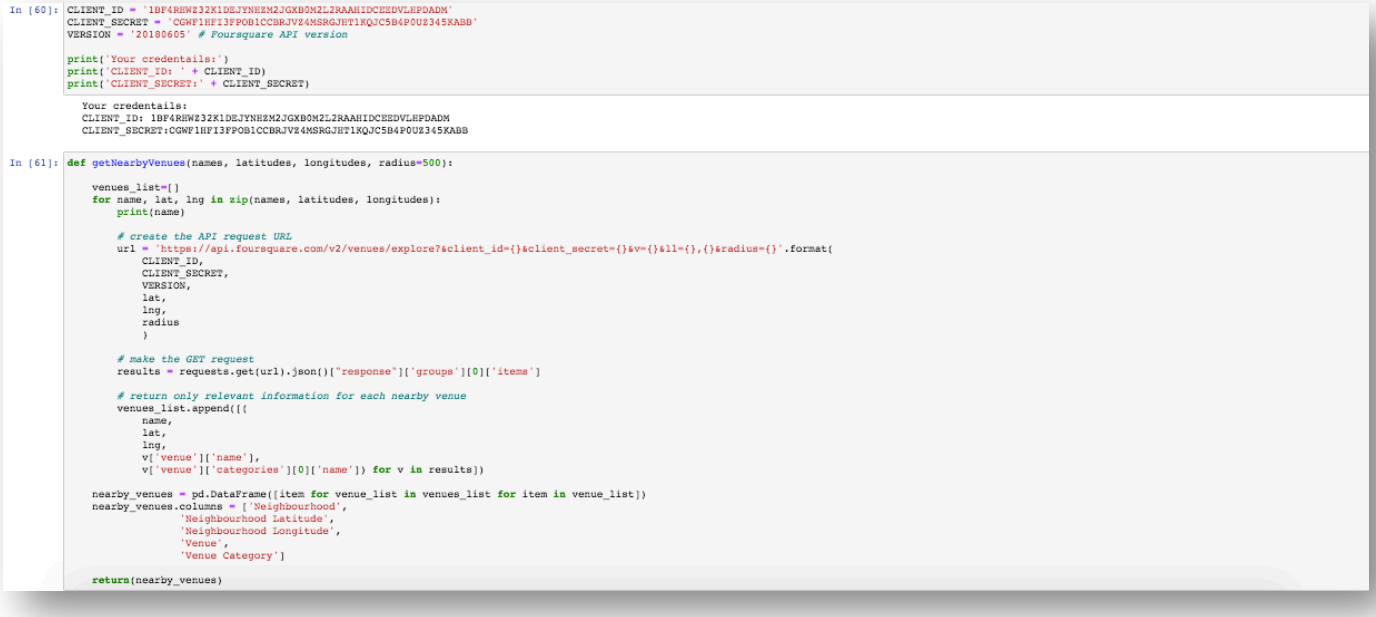
	Neighbourhood	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue	7th Most Common Venue	8th Most Common Venue	9th Most Common Venue	10th Most Common Venue
0	Barnet	Coffee Shop	Café	Grocery Store	Bus Stop	Pub	Italian Restaurant	Supermarket	Pharmacy	Turkish Restaurant	Gym / Fitness Center
1	Barnet, Brent, Camden	Bus Stop	Music Store	Gym / Fitness Center	Supermarket	Clothing Store	Hardware Store	Yoga Studio	Filipino Restaurant	Exhibit	Falafel Restaurant
2	Bexley	Supermarket	Historic Site	Platform	Convenience Store	Train Station	Coffee Shop	Golf Course	Construction & Landscaping	Bus Stop	Park
3	Bexley, Greenwich	Construction & Landscaping	Park	Convenience Store	Bus Stop	Historic Site	Daycare	Golf Course	Flower Shop	Food & Drink Shop	Food Court
4	Bexley, Greenwich	Supermarket	Coffee Shop	Platform	Convenience Store	Train Station	Historic Site	Yoga Studio	Exhibit	Falafel Restaurant	Farmers Market



Out[43]:

	borough	town	post_code	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	Bexley, Greenwich	LONDON	SE2	51.49245	0.12127	4	Supermarket	Coffee Shop	Platform	Convenience Store	Train Station	Historic Site	Yoga Studio	Exhibit	Falafel Restaurant	Farmers Market
1	Ealing, Hammersmith and Fulham	LONDON	W3, W4	51.51324	-0.26746	5	Grocery Store	Indian Restaurant	Hotel	Train Station	Park	Breakfast Spot	Fish & Chips Shop	Exhibit	Falafel Restaurant	Farmers Market
6	City	LONDON	EC3	51.51200	-0.08058	1	Coffee Shop	Hotel	Gym / Fitness Center	Italian Restaurant	Pub	Sandwich Place	Restaurant	Wine Bar	Garden	Scenic Lookout
7	Westminster	LONDON	WC2	51.51651	-0.11968	1	Coffee Shop	Hotel	Sandwich Place	Café	Pub	Italian Restaurant	Theater	Hotel Bar	Restaurant	Burger Joint
9	Bromley	LONDON	SE20	51.41009	-0.05683	5	Supermarket	Convenience Store	Fast Food Restaurant	Hotel	Grocery Store	Park	Bistro	Gastropub	Sandwich Place	Golf Course

```
In [44]: london_data_nonan = london_data.dropna(subset=['Cluster Labels'])
```



```
In [60]: CLIENT_ID = '1BF4RHWZ32K1DEJYNHZM2JGXB0M2L2RAAHIDCEEDVLHPDADM'
         CLIENT_SECRET = 'CGWF1HFI3PPOB1CCBRJVZ4MSRGJHT1KQJC5B4P0U2345KABB'
         VERSION = '20180605' # Foursquare API version

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

Your credentails:
CLIENT_ID: 1BF4RHWZ32K1DEJYNHZM2JGXB0M2L2RAAHIDCEEDVLHPDADM
CLIENT_SECRET:CGWF1HFI3PPOB1CCBRJVZ4MSRGJHT1KQJC5B4P0U2345KABB

In [61]: 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={}'.format(
                     CLIENT_ID,
                     CLIENT_SECRET,
                     VERSION,
                     lat,
                     lng,
                     radius
                     )

                 # 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']['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 = ['Neighbourhood',
                           'Neighbourhood Latitude',
                           'Neighbourhood Longitude',
                           'Venue',
                           'Venue Category']

             return(nearby_venues)
```

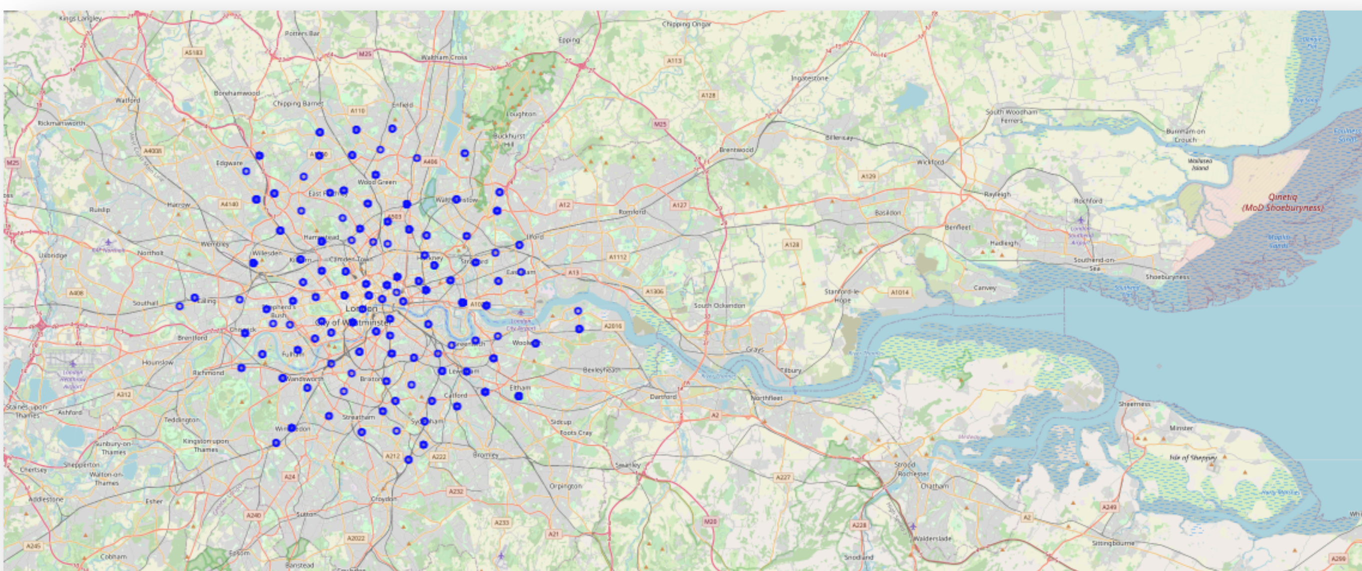
	Neighbourhood	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue	7th Most Common Venue	8th Most Common Venue	9th Most Common Venue	10th Most Common Venue
0	Agincourt	Lounge	Latin American Restaurant	Breakfast Spot	Skating Rink	Yoga Studio	Electronics Store	Eastern European Restaurant	Drugstore	Donut Shop	Dog Run
1	Alderwood, Long Branch	Pizza Place	Gym	Pub	Coffee Shop	Sandwich Place	Skating Rink	Dance Studio	Department Store	Dessert Shop	Dim Sum Restaurant
2	Bathurst Manor, Wilson Heights, Downsview North	Coffee Shop	Bank	Pharmacy	Mobile Phone Shop	Shopping Mall	Diner	Sandwich Place	Deli / Bodega	Intersection	Supermarket
3	Bayview Village	Café	Chinese Restaurant	Bank	Japanese Restaurant	Yoga Studio	Escape Room	Electronics Store	Eastern European Restaurant	Drugstore	Donut Shop
4	Bedford Park, Lawrence Manor East	Coffee Shop	Sandwich Place	Italian Restaurant	Pizza Place	Pharmacy	Juice Bar	Butcher	Sushi Restaurant	Pub	Restaurant

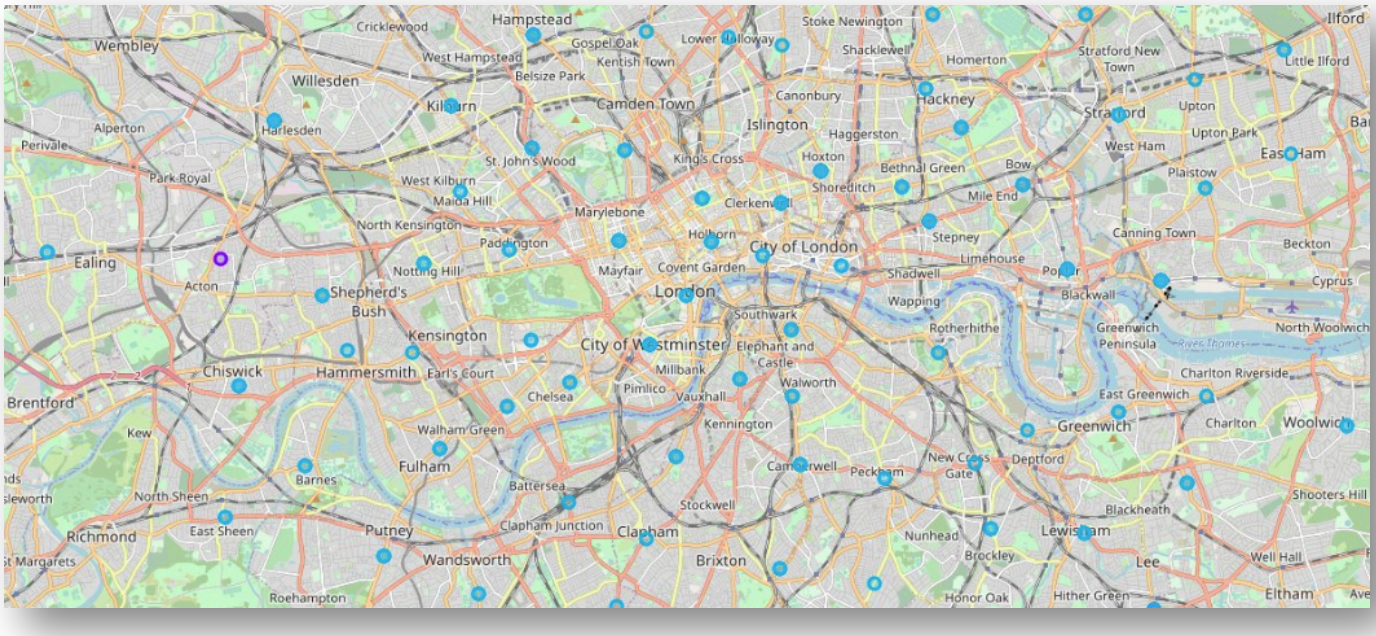
	Postal Code	Borough	Neighbourhood	Latitude	Longitude	Cluster Labels	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue	7th Most Common Venue	8th Most Common Venue	9th Most Common Venue	10th Most Common Venue
0	M3A	North York	Parkwoods	43.753259	-79.329656	1.0	Park	Food & Drink Shop	Yoga Studio	Deli / Bodega	Electronics Store	Eastern European Restaurant	Drugstore	Donut Shop	Dog Run	Distribution Center
1	M4A	North York	Victoria Village	43.725882	-79.315572	0.0	French Restaurant	Hockey Arena	Pizza Place	Coffee Shop	Portuguese Restaurant	Intersection	Dance Studio	Deli / Bodega	Department Store	Dessert Shop
2	M5A	Downtown Toronto	Regent Park, Harbourfront	43.654260	-79.360636	0.0	Coffee Shop	Park	Bakery	Breakfast Spot	Café	Restaurant	Pub	Chocolate Shop	Performing Arts Venue	Yoga Studio
3	M6A	North York	Lawrence Manor, Lawrence Heights	43.718518	-79.464763	0.0	Clothing Store	Accessories Store	Coffee Shop	Furniture / Home Store	Carpet Store	Boutique	Vietnamese Restaurant	Department Store	Electronics Store	Eastern European Restaurant
4	M7A	Downtown Toronto	Queen's Park, Ontario Provincial Government	43.662301	-79.389494	0.0	Coffee Shop	Diner	Sushi Restaurant	Fried Chicken Joint	Italian Restaurant	Bar	Beer Bar	Smoothie Shop	Burrito Place	Sandwich Place

Visualizations of Clusters Maps

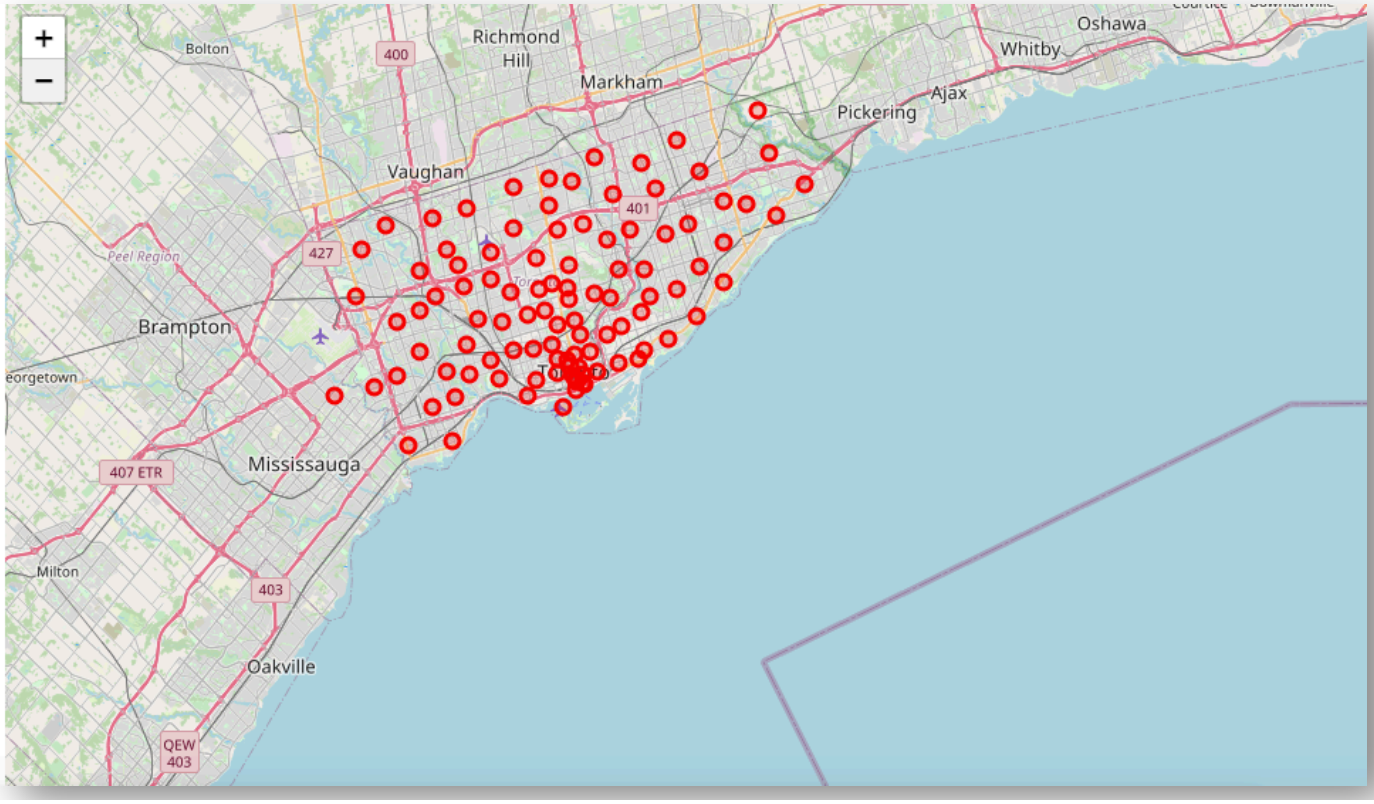
The following maps show the points that represent the service areas distributed in each of the boroughs analyzed.

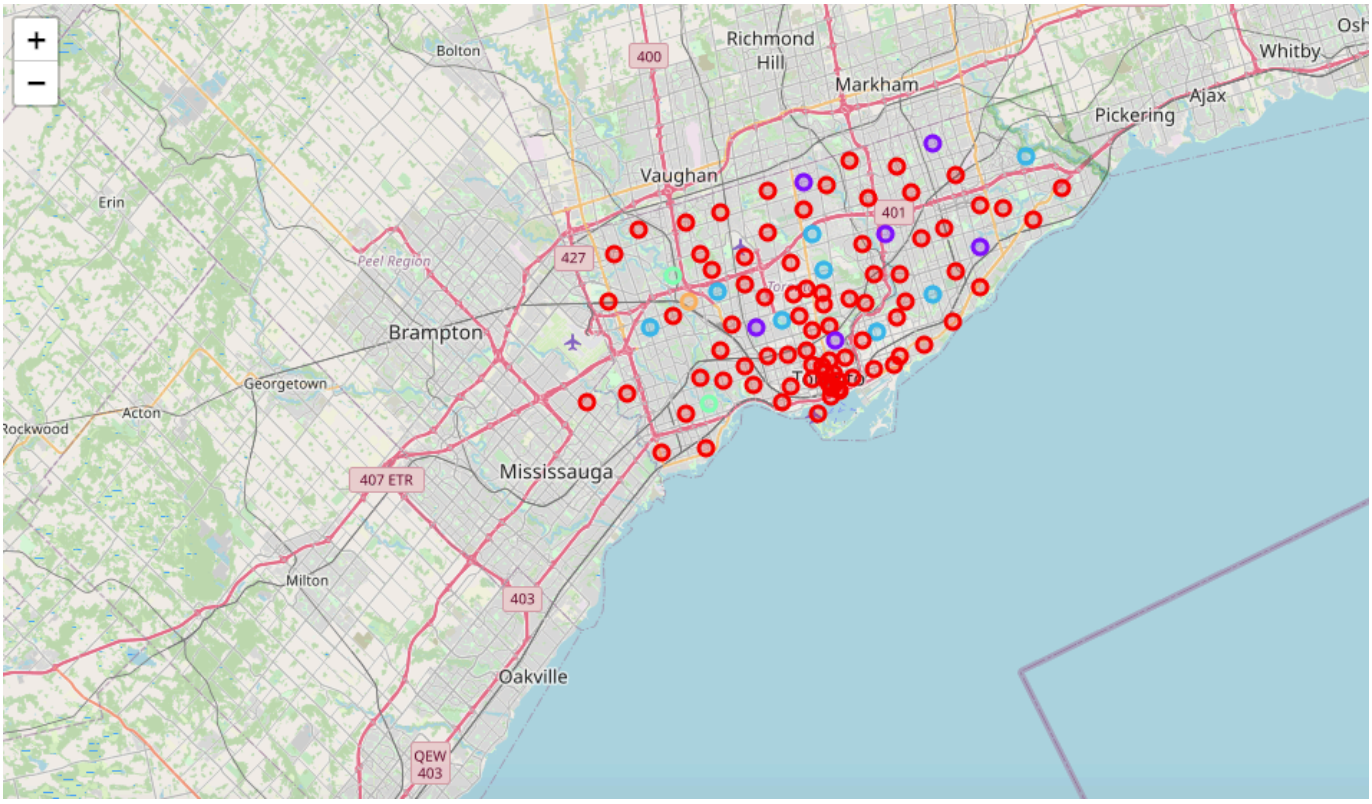
London Map




Hampstead
Cricklewood
Wembley
Willesden
West Hampstead
Gospel Oak
Kentish Town
Belsize Park
Kilburn
Camden Town
Islington
Stoke Newington
Shacklewell
Homerton
Hackney
Stratford New Town
Little Ilford
Upton
Upton Park
West Ham
Plaistow
Bow
Bethnal Green
Shoreditch
Mile End
Haggerston
Hoxton
Kings Cross
St. John's Wood
Alperton
Perivale
Harlesden
Park Royal
West Kilburn
Maida Hill
North Kensington
Paddington
Marylebone
Clerkenwell
Holborn
City of London
Stepney
Limehouse
Canning Town
Beckton
Ealing
Acton
Notting Hill
Mayfair
Covent Garden
London
Shadwell
Wapping
Blackwall
Cyprus
Shepherd's Bush
Kensington
City of Westminster
Southwark
Elephant and Castle
Rotherhithe
Greenwich Peninsula
North Woolwich
Chiswick
Hammersmith
Earl's Court
Millbank
Pimlico
Vauxhall
Walworth
Charlton Riverside
Brentford
Chelsea
East Greenwich
Kew
Walham Green
Kennington
Greenwich
Charlton
Woolwich
Fulham
Barnes
Battersea
Camberwell
Peckham
New Cross Gate
Deptford
North Sheen
Stockwell
Shooters Hill
Blackheath
Richmond
East Sheen
Putney
Clapham Junction
Clapham
Nunhead
Lewisham
Wandsworth
Brixton
Brockley
Lee
Well Hall
Roehampton
Honor Oak
Hither Green
Eltham

Toronto Map


Bolton
400
Richmond Hill
Markham
Oshawa
Whitby
Ajax
Pickering
Vaughan
401
427
Peel Region
Brampton
Mississauga
407 ETR
Milton
403
Oakville
QEW
403



Results and Discussion¶

The cities of London and Toronto are very diverse in all areas. Both cities concentrate most of their services in restaurants and coffee shops. In terms of supermarkets and department stores, there was a higher concentration in Toronto than in London.So based on the observed pattern we can articulate that London is a city more focused on historical tourism where most of its residents live within the city. London's transportation system encourages people to use trains, bicycles and buses more than their own cars. Toronto, on the other hand, is a city with a more complex mobility pattern where there is a flow of residents who travel out of the city to the downtown area to get their groceries or services. Toronto also has more versatility in terms of attractions so it also has international tourism but focuses even more on local tourism and providing services for its residents and visitors.

Particularly in our focus on restaurants, both cities have an interesting variety of gastronomic options.In London the international restaurants are mostly of Indian origin or Mediterranean food where only less than 10 Latin restaurants were found all within 5-8 miles of each other. Of the Latin restaurants in London only 1 specializes in Caribbean food. In Toronto there is a little more gastronomic offer than in London. Especially in the Caribbean food restaurants section I could identify about 7 restaurants, in particular 1 of Cuban origin. Therefore, the offer of Caribbean food consumption in Toronto is much more notable than in London.

Conclusion

Both cities are very attractive for starting a business. In London although there is not much offer of Latin restaurants, I could choose to become one of the few Caribbean restaurants, which would make me an exclusive restaurant. I would have to do a more in-depth market and demographic study to ensure that my business in London could work. While Toronto has more variety of food of Latin and Caribbean origin, I see more competition but a greater chance of acceptance of my gastronomic offer. In Toronto the market study would be more focused on what menu to implement in comparison to the other restaurants in the area that could attract customers.

So it would be wiser to start my business in Toronto, as it is a city where I can have more opportunity to generate income and establish myself more easily.However, I would never rule out in the future having a franchise of my business in London where I can grow the taste for Caribbean food in the residents and visitors.