Capstone Project

The battle of Neighbourhood.

Introduction

Events, Holidays and Travel Plan details to neighborhoods of Toronto.

The report will provide details of Hotels (stay), nearby restaurants, places to visit in the vicinity, places for shopping, cafes etc. This will help any event management firm or a family to plan for a short trip providing them vital information of the neighborhood for their place of visit.

Detailed maps will be provided for the same.

Target audience here is either an Event Management firm or a family that may need details of the neighborhood, which would help them to plan their trips accordingly.

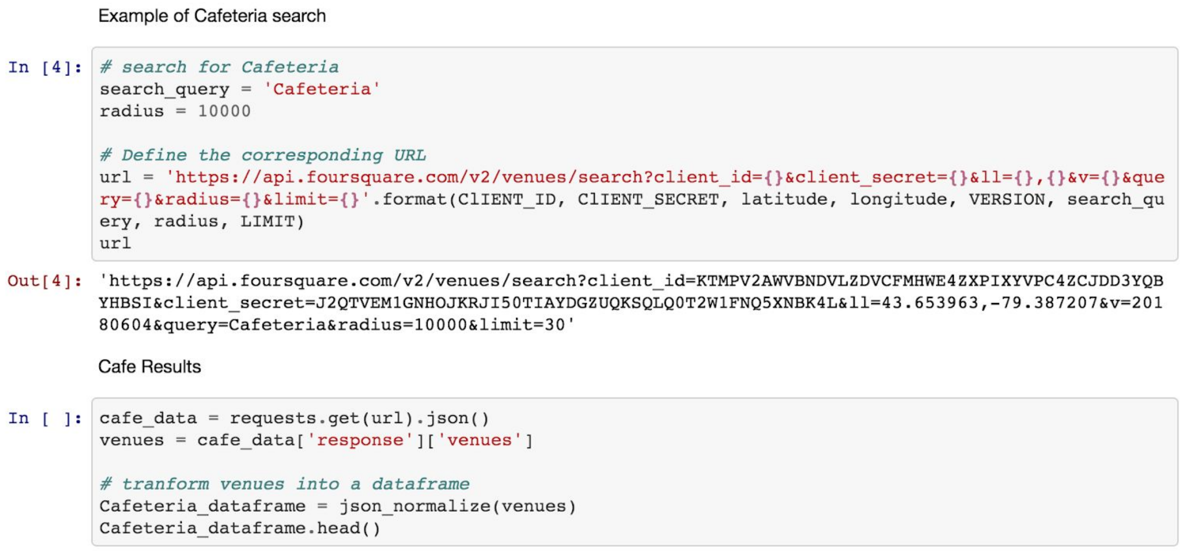
Data

For this report, the detailed data explanation can be seen in the following URL:
https://github.com/NitinLachwani/DSCapstone/blob/master/Capstone_data.ipynb

We first tried to extract the latitude and longitude for the city of Toronto. We then used the foursquare API to extract various types of landmarks. Since the report is for travellers, we tried to extract data for Shops, Restaurants and Hotels.

We had the option to make more searches like Cafes etc but thought of keeping the search restricted to the above 3.

The link above shows how the search is made using the foursquare API.



Example of Cafeteria search

```
In [4]: # search for Cafeteria
        search_query = 'Cafeteria'
        radius = 10000

        # Define the corresponding URL
        url = 'https://api.foursquare.com/v2/venues/search?client_id={}&client_secret={}&ll={},{}&v={}&que
        ry={}&radius={}&limit={}'.format(ClIENT_ID, ClIENT_SECRET, latitude, longitude, VERSION, search_qu
        ery, radius, LIMIT)
        url

Out[4]: 'https://api.foursquare.com/v2/venues/search?client_id=KTMPV2AWVBNDVLZDVCFMHWE4ZXPIXYVPC4ZCJDD3YQB
        YHBSI&client_secret=J2QTVEM1GNHOJKRJI50TIAYDGZUQKSQLQ0T2W1FNQ5XNBK4L&ll=43.653963,-79.387207&v=201
        80604&query=Cafeteria&radius=10000&limit=30'
```

Cafe Results

```
In [ ]: cafe_data = requests.get(url).json()
        venues = cafe_data['response']['venues']

        # tranform venues into a dataframe
        Cafeteria_dataframe = json_normalize(venues)
        Cafeteria_dataframe.head()
```

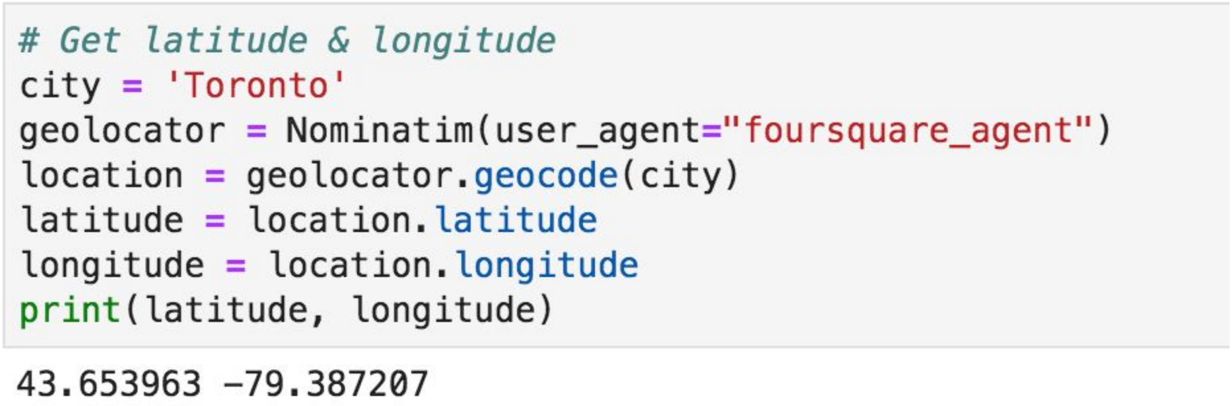
Methodology

This section will include:

1. Get pre-requisite data
2. Search for specific details
3. Extract data
4. Format and Filter the data
5. Make the data presentable
6. Visualize the data using Maps

Get pre-requisite data

For this report, latitude and longitude are the pre-requisite data.
We first try to get these by using geolocator.geocode().



```
# Get latitude & longitude
city = 'Toronto'
geolocator = Nominatim(user_agent="foursquare_agent")
location = geolocator.geocode(city)
latitude = location.latitude
longitude = location.longitude
print(latitude, longitude)
```

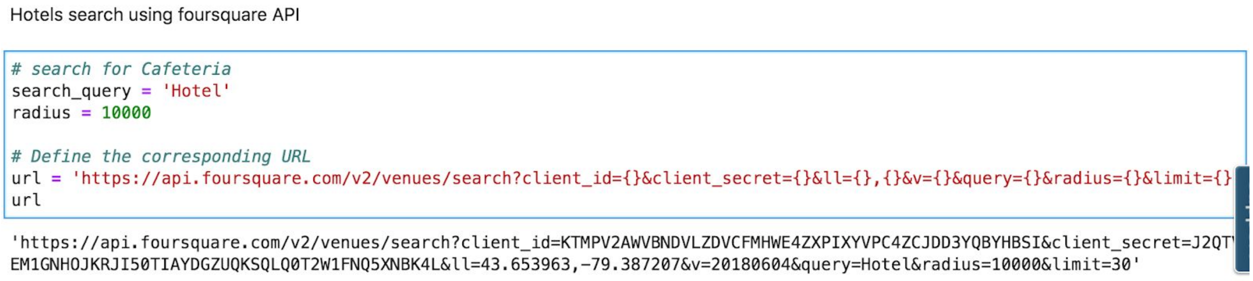
```
43.653963 -79.387207
```

This was the base on which the report data is extracted and structured.

Search for specific details

For this report, we tried to keep our searches to Restaurants, Hotels and Shops. The above three will help travellers or Event management firms to plan their stay/visits.

We used the foursquare API to search for these details.



Hotels search using foursquare API

```
# search for Cafeteria
search_query = 'Hotel'
radius = 10000

# Define the corresponding URL
url = 'https://api.foursquare.com/v2/venues/search?client_id={}&client_secret={}&ll={},{}&v={}&query={}&radius={}&limit={}
url
```

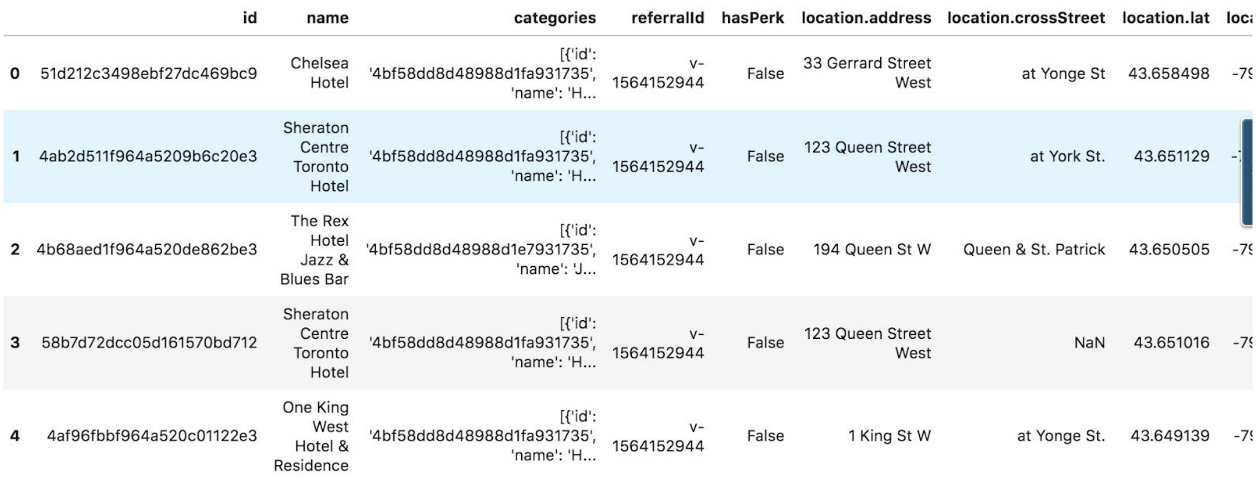
```
'https://api.foursquare.com/v2/venues/search?client_id=KTMPV2AWVBNDVLZDVCFMHWE4ZXPIXYVPC4ZCJDD3YQBYHBSI&client_secret=J2QT
EM1GNHOJKRJI50TIAYDGZUQKSQLQ0T2W1FNQ5XNBK4L&ll=43.653963,-79.387207&v=20180604&query=Hotel&radius=10000&limit=30'
```

The results (in json format) are pushed into a dataframe.

Extract the data

The data for Restaurants, Shops and Hotels was extracted using foursquare API and the json output was pushed into a dataframe.

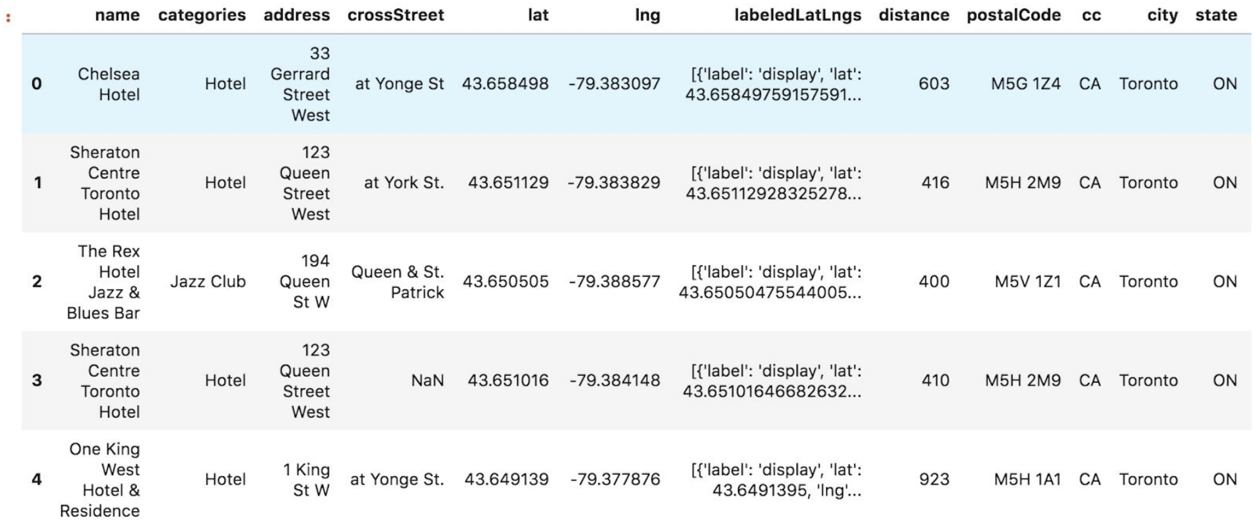
High-level details of the data can be viewed by running the head() method of the dataframe.



	id	name	categories	referralId	hasPerk	location.address	location.crossStreet	location.lat	loca
0	51d212c3498ebf27dc469bc9	Chelsea Hotel	[{'id': '4bf58dd8d48988d1fa931735', 'name': 'H...	v-1564152944	False	33 Gerrard Street West	at Yonge St	43.658498	-79
1	4ab2d511f964a5209b6c20e3	Sheraton Centre Toronto Hotel	[{'id': '4bf58dd8d48988d1fa931735', 'name': 'H...	v-1564152944	False	123 Queen Street West	at York St.	43.651129	-7
2	4b68aed1f964a520de862be3	The Rex Hotel Jazz & Blues Bar	[{'id': '4bf58dd8d48988d1e7931735', 'name': 'J...	v-1564152944	False	194 Queen St W	Queen & St. Patrick	43.650505	-79
3	58b7d72dcc05d161570bd712	Sheraton Centre Toronto Hotel	[{'id': '4bf58dd8d48988d1fa931735', 'name': 'H...	v-1564152944	False	123 Queen Street West	NaN	43.651016	-79
4	4af96fbbf964a520c01122e3	One King West Hotel & Residence	[{'id': '4bf58dd8d48988d1fa931735', 'name': 'H...	v-1564152944	False	1 King St W	at Yonge St.	43.649139	-79

Format and filter the data

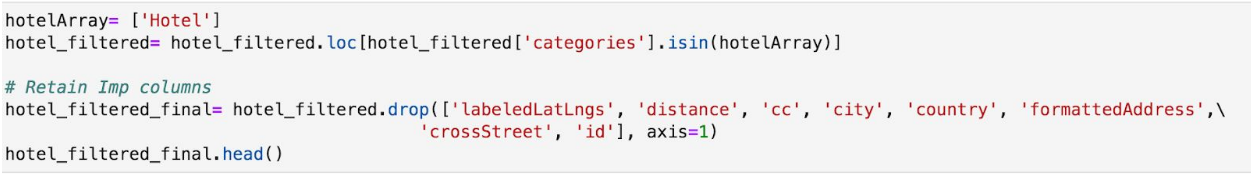
The data as seen above has unformatted data (categories). We need to format the data for categories attribute to ensure we can filter based on it. We loop through each row and try to extract the category of the venue. We would then split the column data and keep only the last term. This would give us appropriate categories which we can use for filtering.



	name	categories	address	crossStreet	lat	lng	labeledLatLngs	distance	postalCode	cc	city	state
0	Chelsea Hotel	Hotel	33 Gerrard Street West	at Yonge St	43.658498	-79.383097	[{'label': 'display', 'lat': 43.65849759157591...	603	M5G 1Z4	CA	Toronto	ON
1	Sheraton Centre Toronto Hotel	Hotel	123 Queen Street West	at York St.	43.651129	-79.383829	[{'label': 'display', 'lat': 43.65112928325278...	416	M5H 2M9	CA	Toronto	ON
2	The Rex Hotel Jazz & Blues Bar	Jazz Club	194 Queen St W	Queen & St. Patrick	43.650505	-79.388577	[{'label': 'display', 'lat': 43.65050475544005...	400	M5V 1Z1	CA	Toronto	ON
3	Sheraton Centre Toronto Hotel	Hotel	123 Queen Street West	NaN	43.651016	-79.384148	[{'label': 'display', 'lat': 43.65101646682632...	410	M5H 2M9	CA	Toronto	ON
4	One King West Hotel & Residence	Hotel	1 King St W	at Yonge St.	43.649139	-79.377876	[{'label': 'display', 'lat': 43.6491395, 'lng'...	923	M5H 1A1	CA	Toronto	ON

We can see categories like Hotel and Jazz Club in the above data frame. However, for travellers, this information would be misleading since they would like to stay in Hotels and hence we would need to filter out Jazz Club while displaying the data for Hotels.

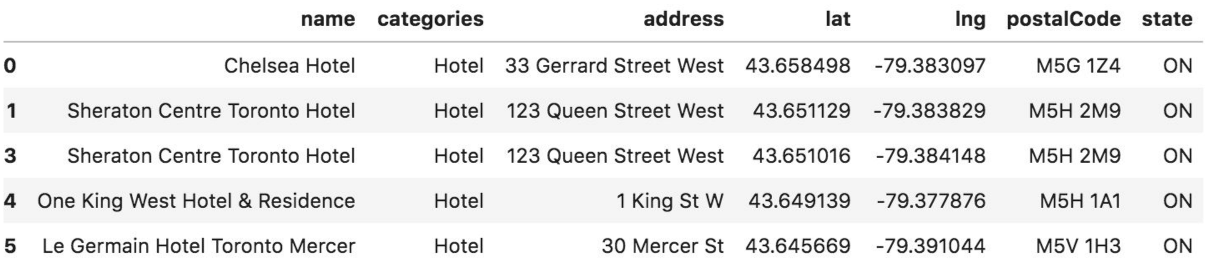
This filtration is achieved by passing an array specifying Hotel.



```
hotelArray= ['Hotel']
hotel_filtered= hotel_filtered.loc[hotel_filtered['categories'].isin(hotelArray)]

# Retain Imp columns
hotel_filtered_final= hotel_filtered.drop(['labeledLatLngs', 'distance', 'cc', 'city', 'country', 'formattedAddress',\
                                          'crossStreet', 'id'], axis=1)
hotel_filtered_final.head()
```

We also drop unwanted columns like crossStreet, id etc from the data frame.

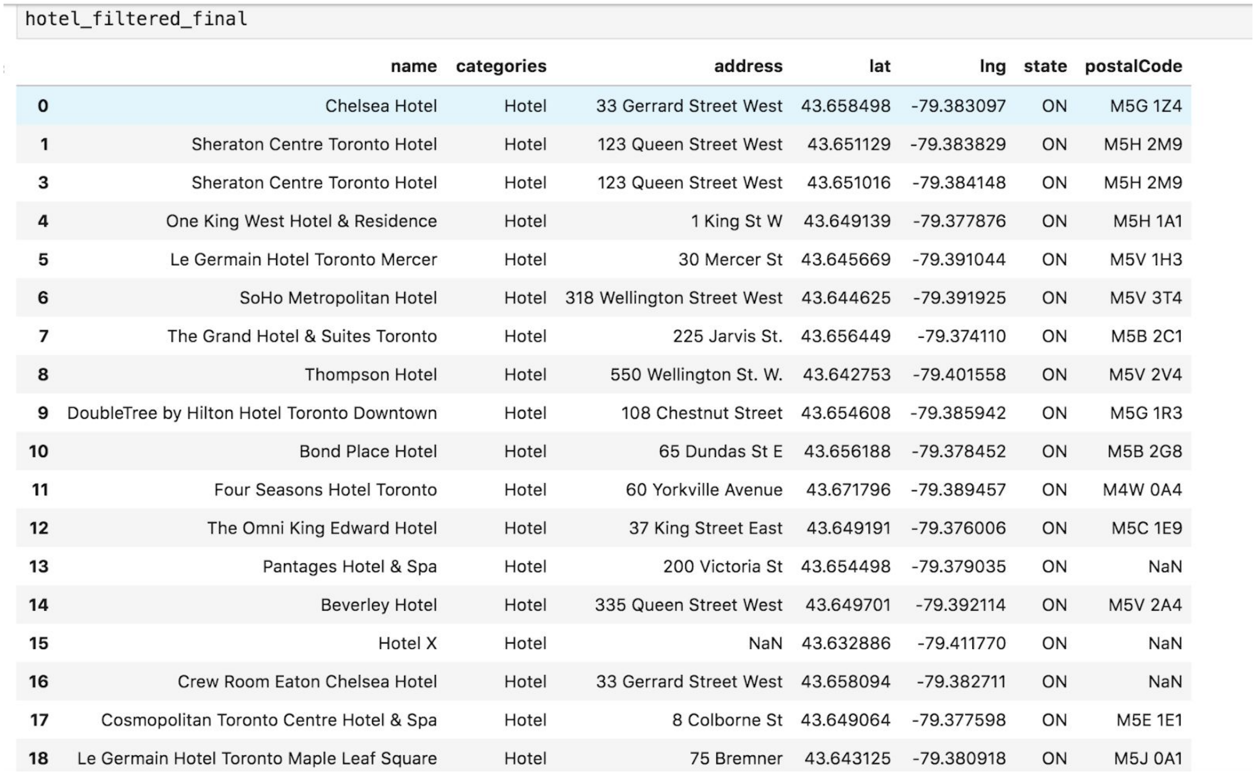


	name	categories	address	lat	lng	postalCode	state
0	Chelsea Hotel	Hotel	33 Gerrard Street West	43.658498	-79.383097	M5G 1Z4	ON
1	Sheraton Centre Toronto Hotel	Hotel	123 Queen Street West	43.651129	-79.383829	M5H 2M9	ON
3	Sheraton Centre Toronto Hotel	Hotel	123 Queen Street West	43.651016	-79.384148	M5H 2M9	ON
4	One King West Hotel & Residence	Hotel	1 King St W	43.649139	-79.377876	M5H 1A1	ON
5	Le Germain Hotel Toronto Mercer	Hotel	30 Mercer St	43.645669	-79.391044	M5V 1H3	ON

The resultant data frame (seen above) is much cleaner and formatted for us to move ahead to work on it to make the data presentable.

Make the data presentable

The data shown below has a flaw.

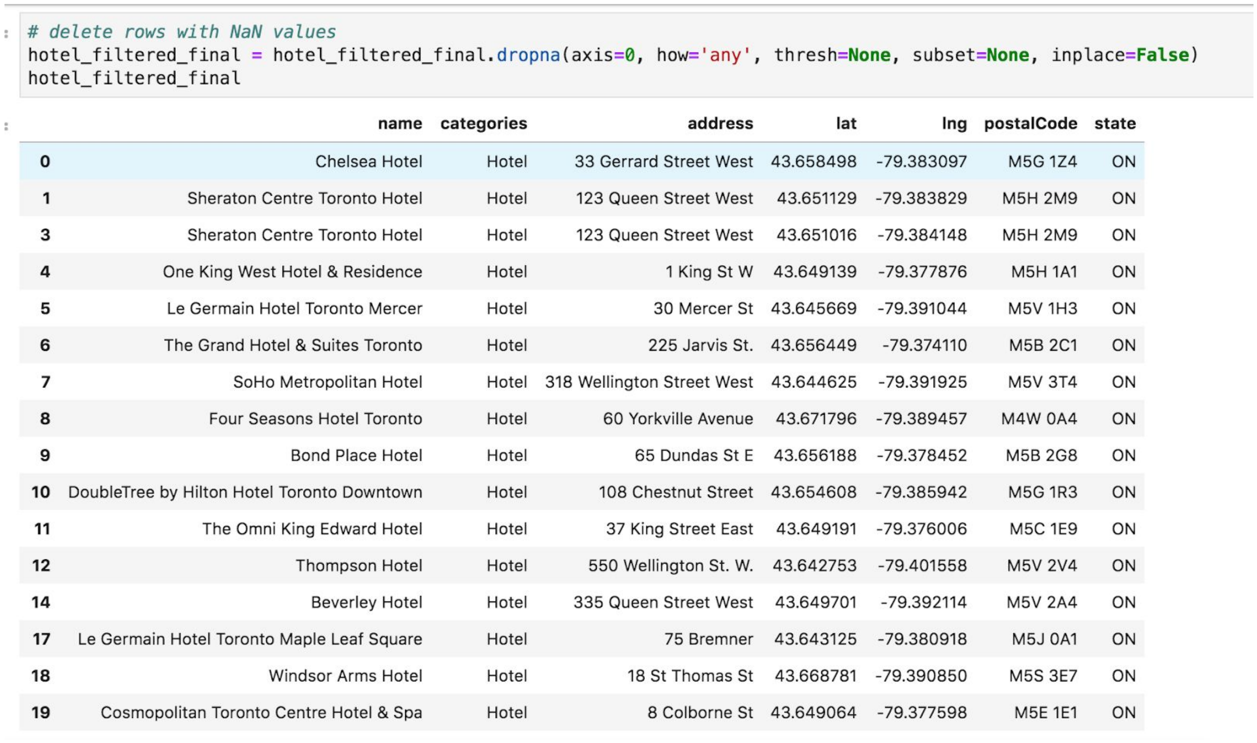


```
hotel_filtered_final
```

	name	categories	address	lat	lng	state	postalCode
0	Chelsea Hotel	Hotel	33 Gerrard Street West	43.658498	-79.383097	ON	M5G 1Z4
1	Sheraton Centre Toronto Hotel	Hotel	123 Queen Street West	43.651129	-79.383829	ON	M5H 2M9
3	Sheraton Centre Toronto Hotel	Hotel	123 Queen Street West	43.651016	-79.384148	ON	M5H 2M9
4	One King West Hotel & Residence	Hotel	1 King St W	43.649139	-79.377876	ON	M5H 1A1
5	Le Germain Hotel Toronto Mercer	Hotel	30 Mercer St	43.645669	-79.391044	ON	M5V 1H3
6	SoHo Metropolitan Hotel	Hotel	318 Wellington Street West	43.644625	-79.391925	ON	M5V 3T4
7	The Grand Hotel & Suites Toronto	Hotel	225 Jarvis St.	43.656449	-79.374110	ON	M5B 2C1
8	Thompson Hotel	Hotel	550 Wellington St. W.	43.642753	-79.401558	ON	M5V 2V4
9	DoubleTree by Hilton Hotel Toronto Downtown	Hotel	108 Chestnut Street	43.654608	-79.385942	ON	M5G 1R3
10	Bond Place Hotel	Hotel	65 Dundas St E	43.656188	-79.378452	ON	M5B 2G8
11	Four Seasons Hotel Toronto	Hotel	60 Yorkville Avenue	43.671796	-79.389457	ON	M4W 0A4
12	The Omni King Edward Hotel	Hotel	37 King Street East	43.649191	-79.376006	ON	M5C 1E9
13	Pantages Hotel & Spa	Hotel	200 Victoria St	43.654498	-79.379035	ON	NaN
14	Beverley Hotel	Hotel	335 Queen Street West	43.649701	-79.392114	ON	M5V 2A4
15	Hotel X	Hotel	NaN	43.632886	-79.411770	ON	NaN
16	Crew Room Eaton Chelsea Hotel	Hotel	33 Gerrard Street West	43.658094	-79.382711	ON	NaN
17	Cosmopolitan Toronto Centre Hotel & Spa	Hotel	8 Colborne St	43.649064	-79.377598	ON	M5E 1E1
18	Le Germain Hotel Toronto Maple Leaf Square	Hotel	75 Bremner	43.643125	-79.380918	ON	M5J 0A1

We can see Nan in the postalCode for rows 15 and 16.

We will need to eliminate all NaN to make the data presentable and move forward with visualization.



```
# delete rows with NaN values
hotel_filtered_final = hotel_filtered_final.dropna(axis=0, how='any', thresh=None, subset=None, inplace=False)
hotel_filtered_final
```

	name	categories	address	lat	lng	postalCode	state
0	Chelsea Hotel	Hotel	33 Gerrard Street West	43.658498	-79.383097	M5G 1Z4	ON
1	Sheraton Centre Toronto Hotel	Hotel	123 Queen Street West	43.651129	-79.383829	M5H 2M9	ON
3	Sheraton Centre Toronto Hotel	Hotel	123 Queen Street West	43.651016	-79.384148	M5H 2M9	ON
4	One King West Hotel & Residence	Hotel	1 King St W	43.649139	-79.377876	M5H 1A1	ON
5	Le Germain Hotel Toronto Mercer	Hotel	30 Mercer St	43.645669	-79.391044	M5V 1H3	ON
6	The Grand Hotel & Suites Toronto	Hotel	225 Jarvis St.	43.656449	-79.374110	M5B 2C1	ON
7	SoHo Metropolitan Hotel	Hotel	318 Wellington Street West	43.644625	-79.391925	M5V 3T4	ON
8	Four Seasons Hotel Toronto	Hotel	60 Yorkville Avenue	43.671796	-79.389457	M4W 0A4	ON
9	Bond Place Hotel	Hotel	65 Dundas St E	43.656188	-79.378452	M5B 2G8	ON
10	DoubleTree by Hilton Hotel Toronto Downtown	Hotel	108 Chestnut Street	43.654608	-79.385942	M5G 1R3	ON
11	The Omni King Edward Hotel	Hotel	37 King Street East	43.649191	-79.376006	M5C 1E9	ON
12	Thompson Hotel	Hotel	550 Wellington St. W.	43.642753	-79.401558	M5V 2V4	ON
14	Beverley Hotel	Hotel	335 Queen Street West	43.649701	-79.392114	M5V 2A4	ON
17	Le Germain Hotel Toronto Maple Leaf Square	Hotel	75 Bremner	43.643125	-79.380918	M5J 0A1	ON
18	Windsor Arms Hotel	Hotel	18 St Thomas St	43.668781	-79.390850	M5S 3E7	ON
19	Cosmopolitan Toronto Centre Hotel & Spa	Hotel	8 Colborne St	43.649064	-79.377598	M5E 1E1	ON

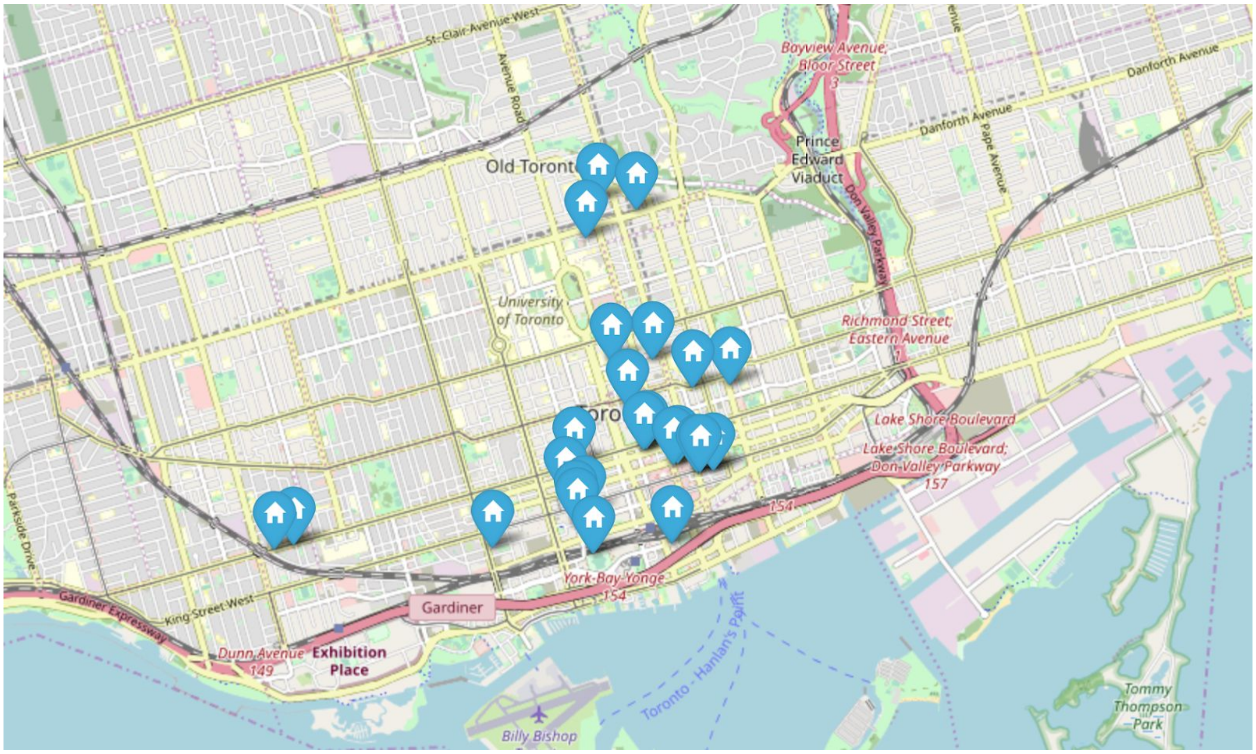
As seen above, we used dropna method to drop all columns with NaN. Column names were renamed as well.

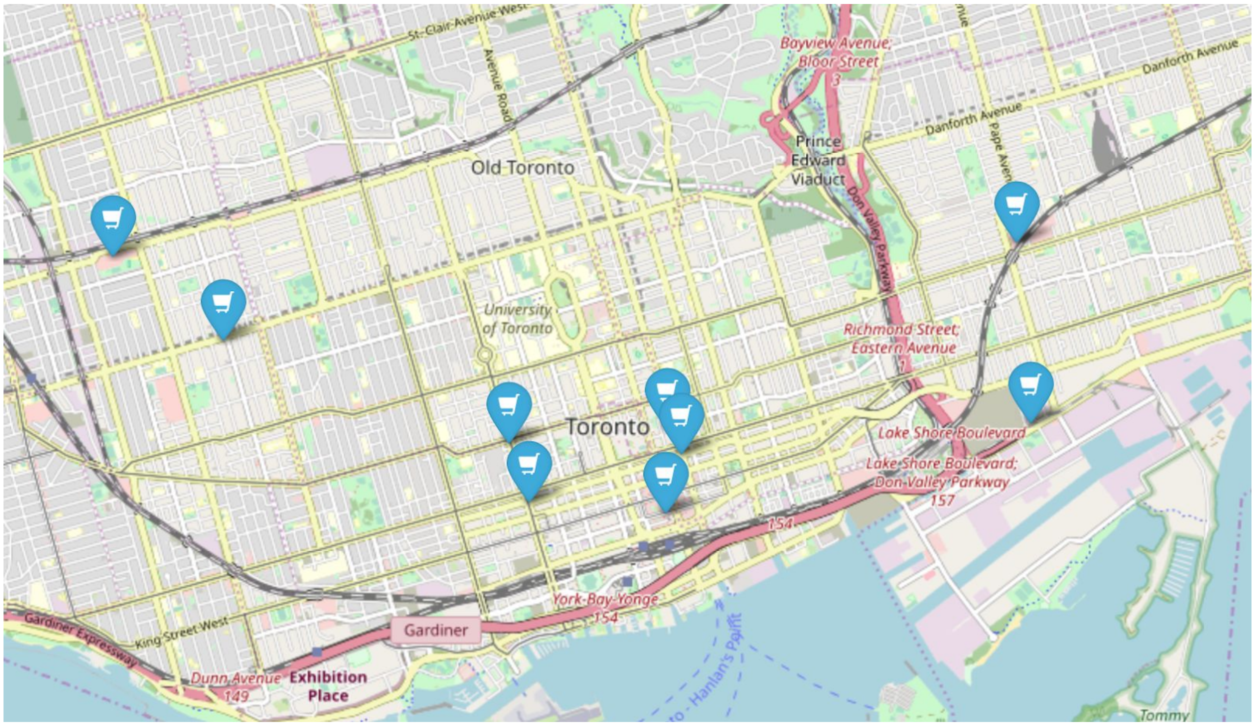
Now the data is clean and presentable and we can proceed with visualization.

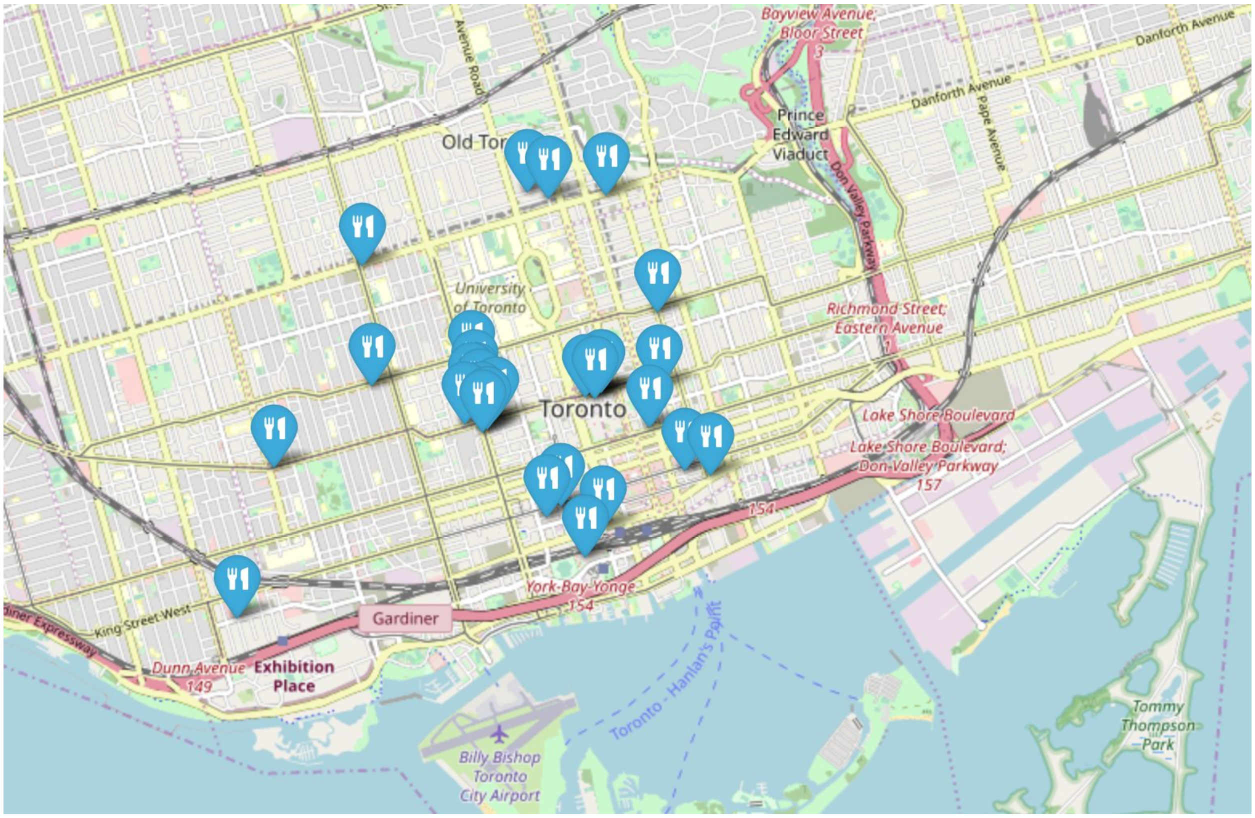
Visualize the data using Maps

Folium API was used to generate Maps for visualization.
We used markers and icons from glyphicon to emphasize markers/icons for each of the venues.

We can see below Maps for Restaurants, Shops and Hotels.

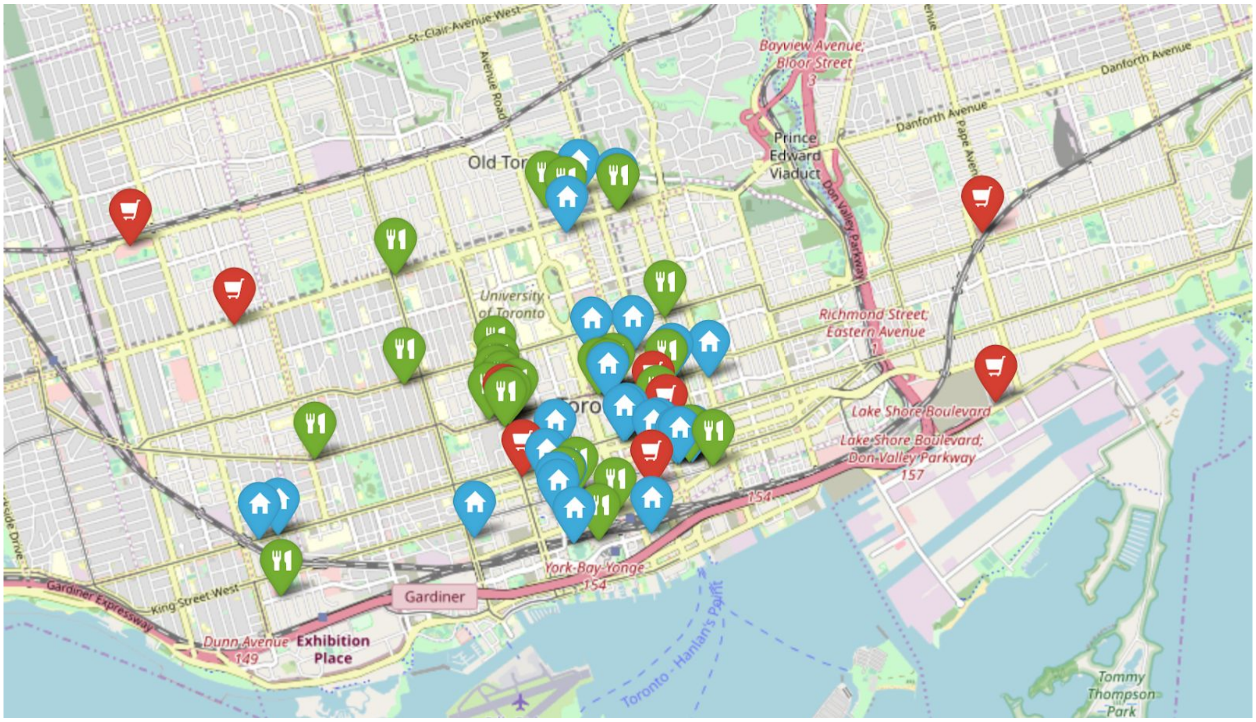






We can see different icons used to differentiate between the three categories.

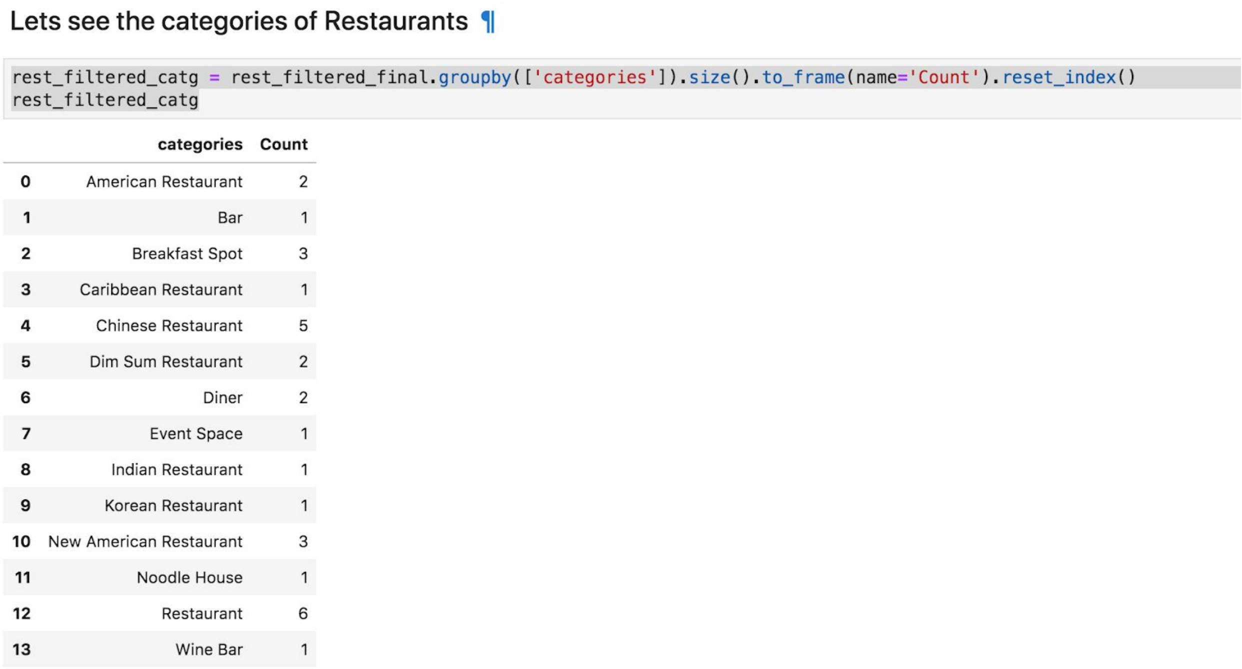
We then consolidated all the information into one Map which would help travellers and event management firms.



Notice the use of different markers and colour highlights from previous maps.

Results

We can observe lots of Hotels in the neighbourhood. However, the number of shops are less and dispersed. However, the restaurants are closely interspersed and near to each other ensuring that people have options. We can also see many restaurants catering to asian cuisine.



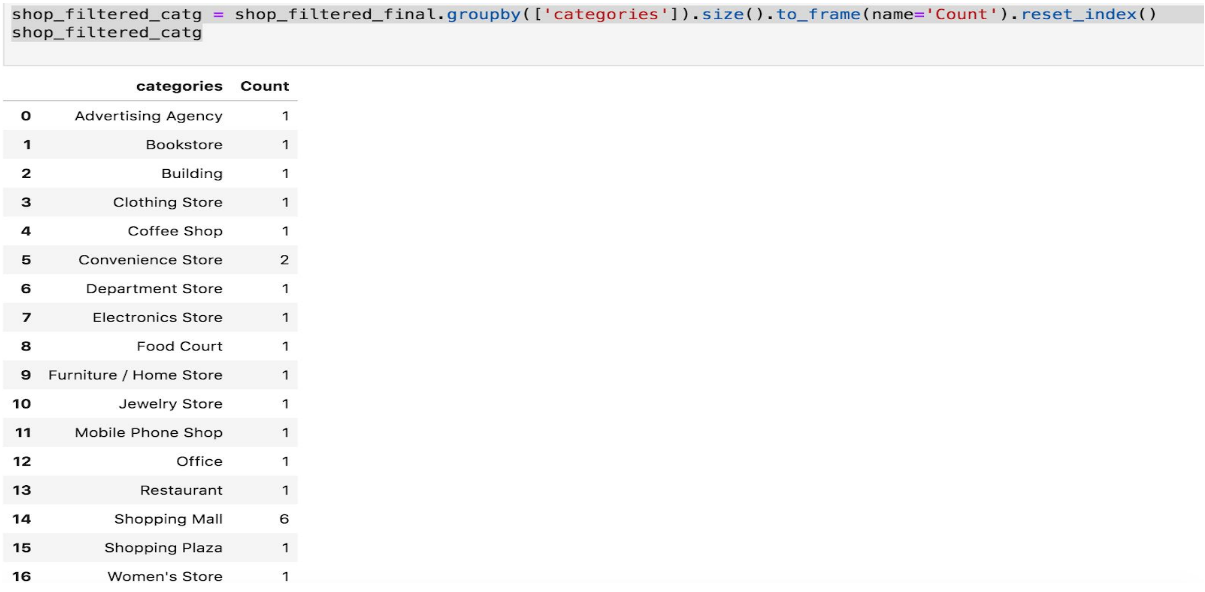
Lets see the categories of Restaurants ¶

```
rest_filtered_catg = rest_filtered_final.groupby(['categories']).size().to_frame(name='Count').reset_index()
rest_filtered_catg
```

	categories	Count
0	American Restaurant	2
1	Bar	1
2	Breakfast Spot	3
3	Caribbean Restaurant	1
4	Chinese Restaurant	5
5	Dim Sum Restaurant	2
6	Diner	2
7	Event Space	1
8	Indian Restaurant	1
9	Korean Restaurant	1
10	New American Restaurant	3
11	Noodle House	1
12	Restaurant	6
13	Wine Bar	1

There are a total of 10 restaurants catering to Asian cuisine (Chinese, Indian, Korean, Dim Sum etc) as compared to 5 catering to American cuisine. This implies that there are many Asian inhabitants in the neighborhood or travellers from Asia visiting the neighborhood.

If we categorize the shops, we can see the following:



```
shop_filtered_catg = shop_filtered_final.groupby(['categories']).size().to_frame(name='Count').reset_index()
shop_filtered_catg
```

	categories	Count
0	Advertising Agency	1
1	Bookstore	1
2	Building	1
3	Clothing Store	1
4	Coffee Shop	1
5	Convenience Store	2
6	Department Store	1
7	Electronics Store	1
8	Food Court	1
9	Furniture / Home Store	1
10	Jewelry Store	1
11	Mobile Phone Shop	1
12	Office	1
13	Restaurant	1
14	Shopping Mall	6
15	Shopping Plaza	1
16	Women's Store	1

There are very few stores competing with each other and there are distinct stores which reduces choices for travellers. We have 6 shopping malls indicating that people like to spend time in malls rather than individual stores. Malls may have similar stores within them.

Discussion and Recommendations

As viewed in the above section, the Shop map shows shops not interspersed like the Restaurants. Travellers would need to plan their shopping appropriately which is never good for a neighborhood and hence more sales would be made in the malls. Restaurants have enough variety of cuisine and travellers have plethora of options since the restaurants are fairly close to each other.

This is a proper business neighborhood with Hotels where one could spend time for meetings and use the restaurants for business lunches.

Conclusion

Using the foursquare venue API, we could get data of remote neighborhoods like Toronto and help make plans for Event management companies and travellers.

We were able to analyze various categories of venues. More deep down analysis can be made and this was a great learning experience as a first time data science project.

Thank you for reading till the end. Have a great day!