Capstone Project Report

On

The Best places to Visit in a city

Submitted by

Ayush Saxena

Table of Contents

Introduction – Business Problem Description

Scenario and Background

Every city has its own different flavors, there are multiple venues which one would like to visit while going into an unknown city. However, sometimes due to time constraints it gets difficult for the person to identify 'top venues' of different categories that a city is famous for. Thus, if a person is able to segment a city on the basis of top venues to visit, it would not only save time but also will give the person opportunity to explore almost all different categories of top venues across the city.

Problem Statement

Can we use data science to segment city based on its 'top venues' for different categories that can help a person in exploring the city better.?

a. Example:

- Suppose a city is famous for restaurants, parks, museum and old monuments.
- We will segment a city based on above venues so that if a person wishes to visit best restaurants, he/she can visit one portion of a city having the best restaurants
- Likewise, if a person wishes to visit parks then he/she can go in the specific area that is famous for parks instead of choosing to go to another area.
- This will save time of the person and will also give the opportunity to explore the best places of the venues he/she wishes to visit.

Target Audience: Interest

It will be having two major target groups:

1. **People visiting new places**
   All those who visit places frequently for business/personal purpose would like to explore the places they visit effectively in the minimum possible time. The project will help them in selecting places to visit as per their choice very conveniently.
2. **Travel Guides and tourism companies**
   Tourism sector can take advantage of this project by segmenting the cities and then helping the people to explore the complete city, covering all the famous and exquisite areas. They can also save time by managing their trips to venues as per their segments.

Data Description

Data Sources

1. Wiki page having the details of the postal codes, Borough and Neighborhood of Canada was used to extract the data and read into data frame using pandas.
   https://en.wikipedia.org/wiki/List_of_postal_codes_of_Canada:_M

2. Google API was used to gather the coordinates for each of the postal code

Data Cleaning

Before processing the gathered data, it was cleaned thoroughly to remove all the non-required information and convert the data ready for use.

1. Initially all the missing column values of the column 'Borough' were removed as they were of no significance.
2. Neighbourhoods having same postal codes were combined into one separated by commas.
3. For each of the postal code respective Latitude and Longitude coordinates were added using Google API.
4. Finally, the postal codes column was dropped as it was irrelevant in the further analysis.

Data Information

Below is the description of the data columns that are used for the analysis:

Postal code	Borough	Neighborhood	Latitude	Longitude

1. **Postal code**: It defines the particular Neighbourhood in a city, it is required for calculating the latitude and longitude coordinates.
2. **Borough**: A town or a district which is an administrative unit, for every Borough there are multiple neighbourhoods
3. **Neighbourhoods**: Basically, a community within a city, there may be multiple neighbourhoods for a particular postal code.
4. **Latitude and Longitude**: The geospatial coordinates defining a particular location, it is required for Foursquare API for analysis and information gathering.

Example of data after cleaning it is ready to use data looked like:

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

Methodology Used (Clustering – k means)

1. After cleaning the data, pandas library was used to import the data frame as shown below:

```
import pandas as pd
df = pd.read_csv("postal_codes_with_coordinates_of_Canada.csv")
df = df.replace(to_replace ='/', value = ',', regex = True)
df.reset_index(drop=True, inplace=True)
df.head()
```

	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

2. Moreover, all the below listed libraries required for the analysis were also imported:
   - **Numpy** - library to handle data in a vectorized manner
   - **Json** - library to handle JSON files
   - **Nominatim** - convert an address into latitude and longitude values
   - **json_normalize** - tranform JSON file into a pandas dataframe
   - **matplotlib** - Library for visualizations
   - **folium** – Map rendering library

3. Geopy library was used to obtain the latitude and longitude values of Toronto as shown below and henceforth a map was rendered with all the neighborhoods imposed on the map as shown below:

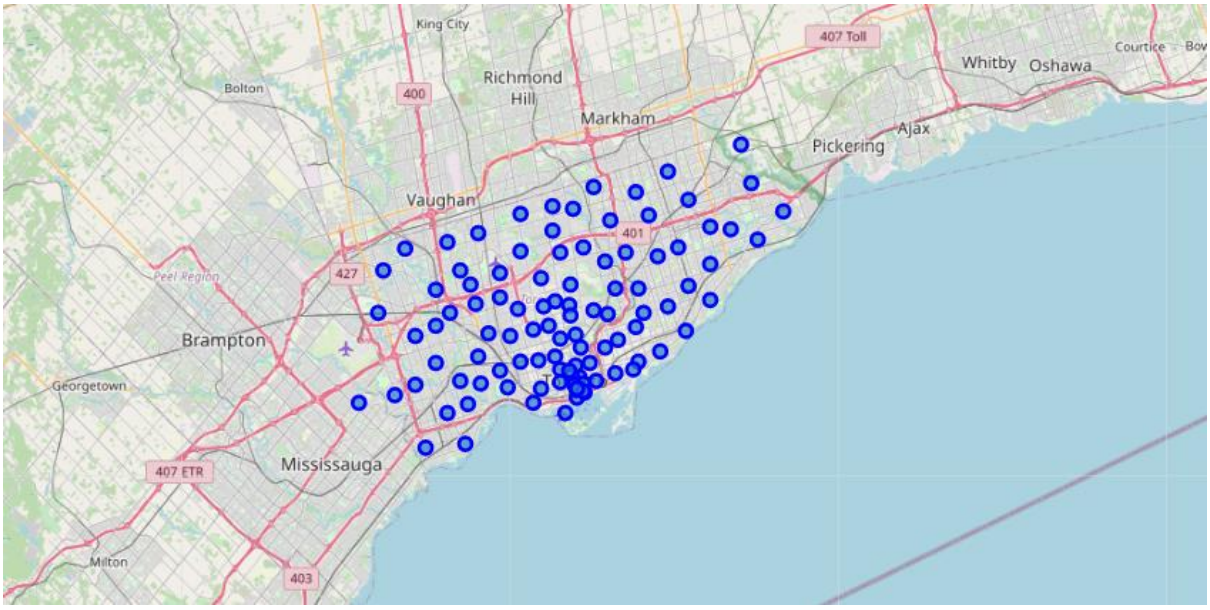
Using geopy library to get the latitude and longitude values of Toronto

```
address = 'Toronto, ON'

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

```
The geograpical coordinate of Toronto are 43.6534817, -79.3839347.
```

4. The blue dots on the map of Toronto below signifies the neighborhoods that are imposed on the map as shown, our task is to categorize into mutually exclusive clusters based on top venues so that for a particular category of a venue a person can visit a specific cluster region instead of wasting time search the best in other areas.



5. For our study in this project we will focus on clustering one of the Borough and will draw insights from it, the same can be replicated to other boroughs and likewise to any other city.

   We have chosen 'Downtown Toronto' as our Borough for study as illustrated below:

```
Downtown_Toronto_data = neighborhoods[neighborhoods['Borough'] == 'Downtown Toronto'].reset_index(drop=True)
Downtown_Toronto_data.head()
```

	Borough	Neighborhood	Latitude	Longitude
0	Downtown Toronto	Rosedale	43.679563	-79.377529
1	Downtown Toronto	St. James Town , Cabbagetown	43.667967	-79.367675
2	Downtown Toronto	Church and Wellesley	43.665860	-79.383160
3	Downtown Toronto	Regent Park , Harbourfront	43.654260	-79.360636
4	Downtown Toronto	Garden District, Ryerson	43.657162	-79.378937

6. Similarly, we will draw the map of Downtown Toronto and will superimpose it's respective neighborhoods on it.

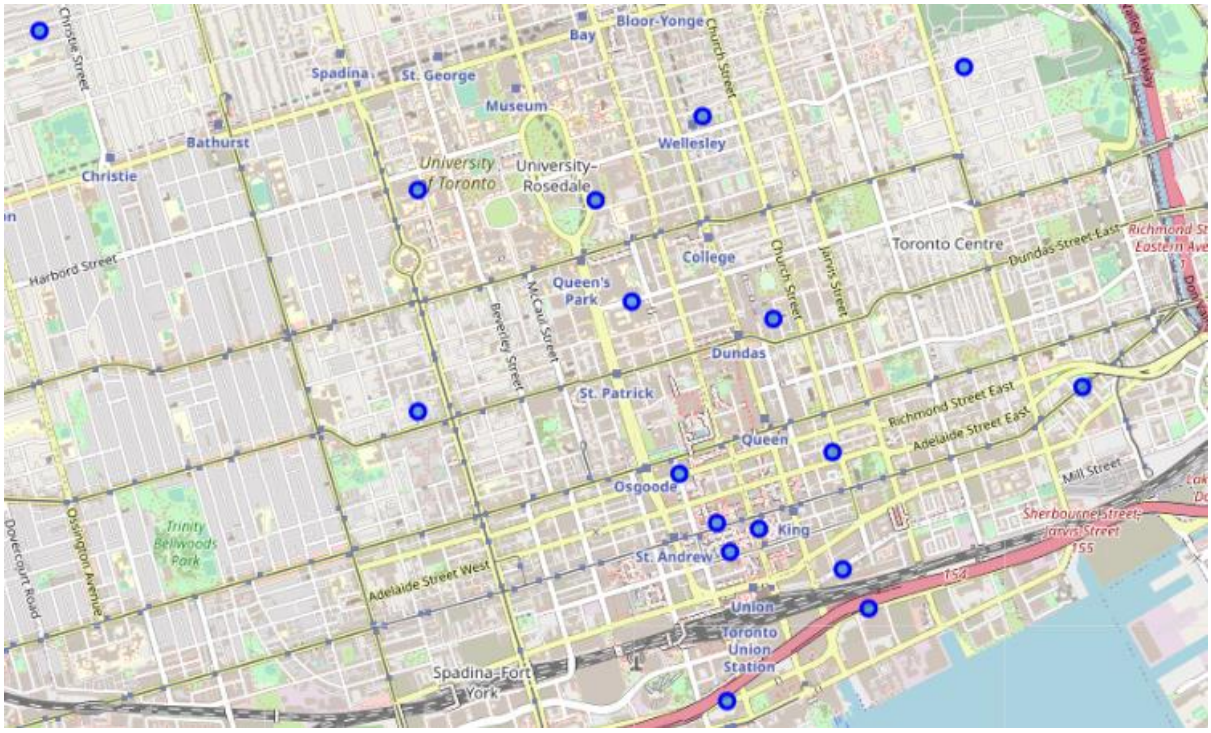
The geographical coordinates of Downtown Toronto

```
address = 'Downtown Toronto, ON'

geolocator = Nominatim(user_agent="ny_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 43.6563221, -79.3809161.
```

Neighborhoods imposed on Map of Downtown Toronto:



7. Some of the Neighborhoods are as follows:

```
Downtown_Toronto_data.loc[0:5, 'Neighborhood']
```

```
0                       Rosedale
1    St. James Town , Cabbagetown
2            Church and Wellesley
3      Regent Park , Harbourfront
4        Garden District, Ryerson
5                  St. James Town
Name: Neighborhood, dtype: object
```

8. Next we will take each neighborhood one by one and will identify the top 100 venues in each of them using GET request of **Foursquare API**

9. We will be exploring the top 100 venues for neighborhoods with the following conditions:
    - Number of venues limit set to 100
    - Radius of 500 meters for venues in each neighborhood

Creating the GET request URL

```
LIMIT = 100 # limit of number of venues returned by Foursquare API

radius = 500 #  radius


# URL creation
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
```

10. Now we will convert the results obtained from Foursquare API (json file) into pandas data frame. Also, as most of the information in json file is stored in items key, we will be using 'get_category_type' function of Foursquare as shown:

```
results = requests.get(url).json()
#results
```

As all the information is in the *items* key. Hence we will borrow the **get_category_type** function from the Foursquare lab.

```
# 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']
```

11. Now we have the json results stored into data frame with each neighborhood having different venue categories with its respective latitude and longitude coordinates.

	Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
0	Rosedale	43.679563	-79.377529	Rosedale Park	43.682328	-79.378934	Playground
1	Rosedale	43.679563	-79.377529	Whitney Park	43.682036	-79.373788	Park
2	Rosedale	43.679563	-79.377529	Alex Murray Parkette	43.678300	-79.382773	Park
3	Rosedale	43.679563	-79.377529	Milkman's Lane	43.676352	-79.373842	Trail
4	St. James Town , Cabbagetown	43.667967	-79.367675	Cranberries	43.667843	-79.369407	Diner

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

Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
Berczy Park	58	58	58	58	58	58
CN Tower , King and Spadina , Railway Lands , Harbourfront West , Bathurst Quay , South Niagara , Island airport	16	16	16	16	16	16
Central Bay Street	64	64	64	64	64	64
Christie	17	17	17	17	17	17
Church and Wellesley	75	75	75	75	75	75
Commerce Court , Victoria Hotel	100	100	100	100	100	100
First Canadian Place , Underground city	100	100	100	100	100	100
Garden District, Ryerson	100	100	100	100	100	100
Harbourfront East , Union Station , Toronto Islands	100	100	100	100	100	100
Kensington Market , Chinatown , Grange Park	57	57	57	57	57	57
Queen's Park , Ontario Provincial Government	40	40	40	40	40	40

Analyzing Each Neighborhood

Now we will use one-hot-encoding for each of the category of the venue and will group by neighborhood using the mean of frequency of the occurrence of each category as shown:

Analyzing Each Neighborhood

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

# add neighborhood column back to dataframe
DT_onehot['Neighborhood'] = DT_venues['Neighborhood']

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

DT_onehot.head()
```

	Yoga Studio	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
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

```
DT_grouped = DT_onehot.groupby('Neighborhood').mean().reset_index()
DT_grouped
```

	Neighborhood	Yoga Studio	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
0	Berczy Park	0.000000	0.0000	0.0000	0.0000	0.000	0.0000	0.000	0.000000	0.000000	0.00	0.017241	0.000000	0.000000	0.000000
1	CN Tower , King and Spadina , Railway Lands , ...	0.000000	0.0625	0.0625	0.0625	0.125	0.1875	0.125	0.000000	0.000000	0.00	0.000000	0.000000	0.000000	0.000000
2	Central Bay Street	0.015625	0.0000	0.0000	0.0000	0.000	0.0000	0.000	0.000000	0.000000	0.00	0.000000	0.015625	0.000000	0.000000
3	Christie	0.000000	0.0000	0.0000	0.0000	0.000	0.0000	0.000	0.000000	0.000000	0.00	0.000000	0.000000	0.000000	0.000000
4	Church and Wellesley	0.026667	0.0000	0.0000	0.0000	0.000	0.0000	0.000	0.013333	0.000000	0.00	0.000000	0.000000	0.013333	0.000000

12. Once we have grouped the results by neighborhood, we will select the top 10 venues for each of the neighborhood.

Creating the new dataframe and display the top 10 venues for each neighborhood

```
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:
        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'] = DT_grouped['Neighborhood']

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

neighborhoods_venues_sorted.head()
```

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
Berczy Park	Coffee Shop	Cocktail Bar	Seafood Restaurant	Italian Restaurant	Beer Bar	Bakery	Restaurant	Cheese Shop	Café	Farmers Market
CN Tower , King and Spadina , Railway Lands , ...	Airport Service	Airport Lounge	Airport Terminal	Coffee Shop	Harbor / Marina	Boat or Ferry	Sculpture Garden	Bar	Boutique	Airport
Central Bay Street	Coffee Shop	Italian Restaurant	Café	Sandwich Place	Bubble Tea Shop	Fried Chicken Joint	Salad Place	Ice Cream Shop	Burger Joint	Japanese Restaurant
Christie	Grocery Store	Café	Park	Baby Store	Coffee Shop	Candy Store	Restaurant	Diner	Italian Restaurant	Nightclub
Church and Wellesley	Sushi Restaurant	Japanese Restaurant	Coffee Shop	Gay Bar	Restaurant	Pub	Hotel	Yoga Studio	Gastropub	Men's Store

Clustering Neighborhoods

We have clustered the neighborhoods using k-means clustering algorithm. Using SSE for different values of k, it has been identified that k=5 has the minimum SSE and hence we have divided the neighborhoods into 5 different clusters.

CLustering into 5 clusters

```
# set number of clusters
kclusters = 5

DT_grouped_clustering = DT_grouped.drop('Neighborhood', 1)

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

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

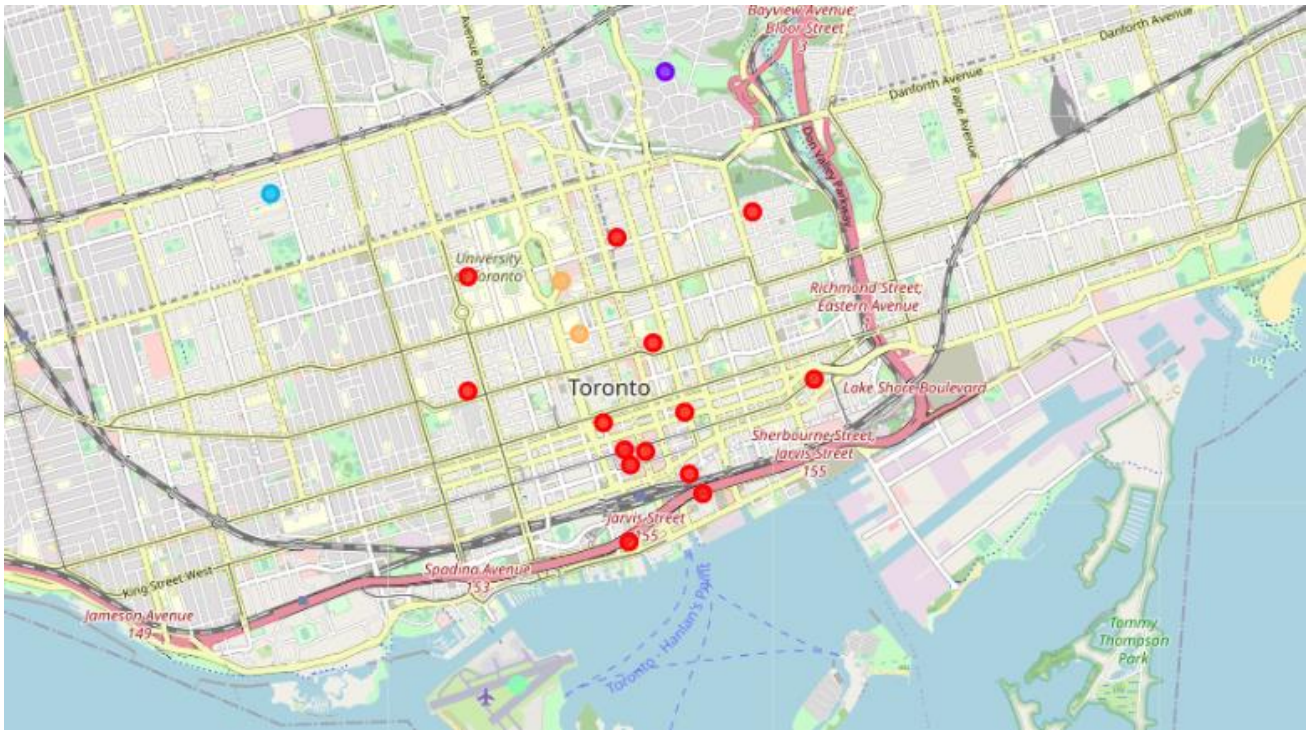
```
array([0, 3, 4, 2, 0, 0, 0, 0, 0, 0], dtype=int32)
```

Borough remains the same as Downtown Toronto and each neighborhood is assigned one o the 5 cluster labels as shown

	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
0	Downtown Toronto	Rosedale	43.679563	-79.377529	1	Park	Trail	Playground	Women's Store	Cupcake Shop
1	Downtown Toronto	St. James Town , Cabbagetown	43.667967	-79.367675	0	Restaurant	Coffee Shop	Pub	Italian Restaurant	Bakery
2	Downtown Toronto	Church and Wellesley	43.665860	-79.383160	0	Sushi Restaurant	Japanese Restaurant	Coffee Shop	Gay Bar	Restaurant
3	Downtown Toronto	Regent Park , Harbourfront	43.654260	-79.360636	0	Coffee Shop	Pub	Bakery	Park	Restaurant
4	Downtown Toronto	Garden District, Ryerson	43.657162	-79.378937	0	Clothing Store	Coffee Shop	Café	Japanese Restaurant	Restaurant

Visualizing Clusters

Clusters generated are plotted on the Toronto map for Downtown Toronto's neighborhoods and its top categories.



Examining Clusters

We will now identify the top venues for each of the neighborhood as per the clusters

Cluster 1

```
DT_merged.loc[DT_merged['Cluster Labels'] == 0, DT_merged.columns[[1] + list(range(5, DT_merged.shape[1]))]]
```

	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
1	St. James Town , Cabbagetown	Coffee Shop	Bakery	Italian Restaurant	Pizza Place	Café	Market	Pub	Restaurant
2	Church and Wellesley	Sushi Restaurant	Coffee Shop	Japanese Restaurant	Restaurant	Gay Bar	Hotel	Pub	Men's Store
3	Regent Park , Harbourfront	Coffee Shop	Pub	Park	Bakery	Breakfast Spot	Theater	Café	Restaurant
4	Garden District, Ryerson	Clothing Store	Coffee Shop	Café	Restaurant	Japanese Restaurant	Bubble Tea Shop	Cosmetics Shop	Middle Eastern Restaurant
5	St. James Town	Coffee Shop	Café	Gastropub	Cocktail Bar	Italian Restaurant	American Restaurant	Gym	Farmers Market
6	Berczy Park	Coffee Shop	Cocktail Bar	Café	Beer Bar	Farmers Market	Bakery	Restaurant	Cheese Shop

Cluster 2

```
DT_merged.loc[DT_merged['Cluster Labels'] == 1, DT_merged.columns[[1] + list(range(5, DT_merged.shape[1]))]]
```

	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
0	Rosedale	Park	Trail	Playground	Cupcake Shop	Donut Shop	Doner Restaurant	Dog Run	Distribution Center

Cluster 3

```
DT_merged.loc[DT_merged['Cluster Labels'] == 2, DT_merged.columns[[1] + list(range(5, DT_merged.shape[1]))]]
```

	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
17	Christie	Grocery Store	Café	Park	Candy Store	Nightclub	Coffee Shop	Restaurant	Gas Station

Cluster 4

```
DT_merged.loc[DT_merged['Cluster Labels'] == 3, DT_merged.columns[[1] + list(range(5, DT_merged.shape[1]))]]
```

	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
14	CN Tower , King and Spadina , Railway Lands , ...	Airport Lounge	Airport Service	Airport Terminal	Plane	Harbor / Marina	Sculpture Garden	Boat or Ferry	Rental Car Location

Cluster 5 ¶

```
DT_merged.loc[DT_merged['Cluster Labels'] == 4, DT_merged.columns[[1] + list(range(5, DT_merged.shape[1]))]]
```

	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
7	Central Bay Street	Coffee Shop	Italian Restaurant	Café	Sandwich Place	Bubble Tea Shop	Fried Chicken Joint	Salad Place	Ice Cream Shop
18	Queen's Park , Ontario Provincial Government	Coffee Shop	Sushi Restaurant	Diner	Beer Bar	Café	Bank	Bar	Burrito Place

Results

We have successfully clustered the top venues based on neighborhoods into 5 distinct clusters namely:

- Cluster 1: **Coffee shops** - It includes top venue categories of, cafes, coffee shops, restaurants.
- Cluster 2: **Parks** - It includes top venue categories of parks, playground etc.
- Cluster 3: **Grocery stores** - It includes top venue categories of grocery stores, candy store etc.
- Cluster 4: **Airport services** - It includes top venue categories of Airport services, airport lounge, airport terminal etc.
- Cluster 5: **Restaurants** - It include top venue categories of international taste restaurants like Sushi, Italian restaurants etc.

Hence, if a person wishes to have explore the best coffee in the town then it can directly visit the cluster 1 of the Downtown Toronto as it is the aggregation of the best available in the city and likewise for any other service as required by the person, respective clusters will serve as a much better map.

Discussion

Below are the assumptions made while working on this project.

- The clusters/groups are created for the neighbourhoods of Toronto with available data. These can be improved or enhanced using refining the data.
- The value of k chosen for the study is using the best possible guess estimate and can be improved further.
- This project can be replicated for analysis of any locations having Foursquare data.
- The project was done for one of the Borough, Downtown Toronto the same can be extended to other Boroughs and their respective neighbourhoods.
- Regular updating or addition of more data to Foursquare database could refine the results more.

Conclusion

With the help of data science, this project has been successfully been able to segment the city into various clusters that can help a person to explore the city in a much better way with least amount of time. Clusters would guide the tourists to appropriate areas where they can explore the venues of their choices. This project can be replicated to any other city as well, the more data available on the city the better will be the analysis and the results. Henceforth, it can be very beneficial to our target audience: people who visit different new places regularly and the tourist guides as it saves their time and gives them a plan that they can follow to help the people visit the best places around the city.