Capstone project

Report

Exploring the Restaurants of Toronto Neighborhoods

Prepared by Richardson T. Pierre

I. Introduction

Toronto is the one of most populous city in the North America, home of many great business. It is one of the most diverse cities on the planet, as it is home to over 6 million people and over 300 languages.

As quoted in an article - What Food Tells Us About Culture:

"Food is not rational Food is culture, habit, craving and identity."

Undoubtedly, Food Diversity is an important part of an ethnically diverse metropolis. The idea of this project is to categorically segment the neighborhoods of Toronto into major clusters and examine their cuisines. A desirable intention is to examine the neighborhood cluster's food habits and taste. Further examination might reveal if food has any relationship with the diversity of a neighborhood.

The purpose of this project is to help to understand the diversity of a neighborhood by leveraging venue data from the Foursquare's location API and k-means clustering machine learning algorithm. Exploratory Data Analysis will help to discover further about the culture and diversity of the neighborhood. Stakeholders would be the one who are interested to use this quantifiable analysis to understand the distribution of different cultures and cuisines over one of the most diverse cities on the planet - Toronto. And, this project can be used by a new food investor who is willing to open his or her restaurant over there. Or by a government or authority to examine and study their city's culture diversity better.

II. Data acquisition and cleaning

In this project the Scarborough dataset is used, which have been scrapped from Wikipedia on Week 3 of the capstone project course.

Link to Data: https://en.wikipedia.org/wiki/List_of_postal_codes_of_Canada:_M

Geolocation data: https://cocl.us/Geospatial_data

After processing the data and associate them to the coordinate location data, the result is as followed:

	PostalCode	Borough	Neighbourhood	Latitude	Longitude
0	M3A	North York	Parkwoods	43.753259	-79.329656
1	M4A	North York	Victoria Village	43.725882	-79.315572
2	M5A	Downtown Toronto	Regent Park, Harbourfront	43.654260	-79.360636
3	M6A	North York	Lawrence Manor, Lawrence Heights	43.718518	-79.464763
4	M7A	Downtown Toronto	Queen's Park, Ontario Provincial Government	43.662301	-79.389494

This is a portion of a data-frame that contains 103 rows and 5 columns of data.

Foursquare API Data

Data about different venues in different neighborhoods of that specific borough have been required. In order to gain that information we will use "Foursquare" locational information. Foursquare is a location data provider with information about all manner of venues and events within an area of interest. Such information includes venue names, locations, menus and even photos. As such, the foursquare location platform will be used as the sole data source since all the stated required information can be obtained through the API.

After finding the list of neighborhoods, we then connect to the Foursquare API to gather information about venues inside each and every neighborhood. For each neighborhood, we have chosen the radius to be 100 meter.

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. Neighborhood
2. Neighborhood Latitude
3. Neighborhood Longitude
4. Venue
5. Name of the venue
6. Venue Latitude
7. Venue Longitude
8. Venue Category

III - Methodology

Methodology Download and Explore Toronto Dataset

In order to segment the neighborhoods of Toronto, a dataset is required that contains the 10 boroughs and the 103 neighborhoods. This dataset is scraped from wikipedia website using the mentioned URL. After, this dataset is merged with the respective dataset of the coordinates data for each neighborhood. Once the the data are ready, it is analyzed to understand the structure of the file. The data returned is transformed, into a pandas dataframe, by looping through the data and filling the dataframe rows one at a time using the following depicted loop.

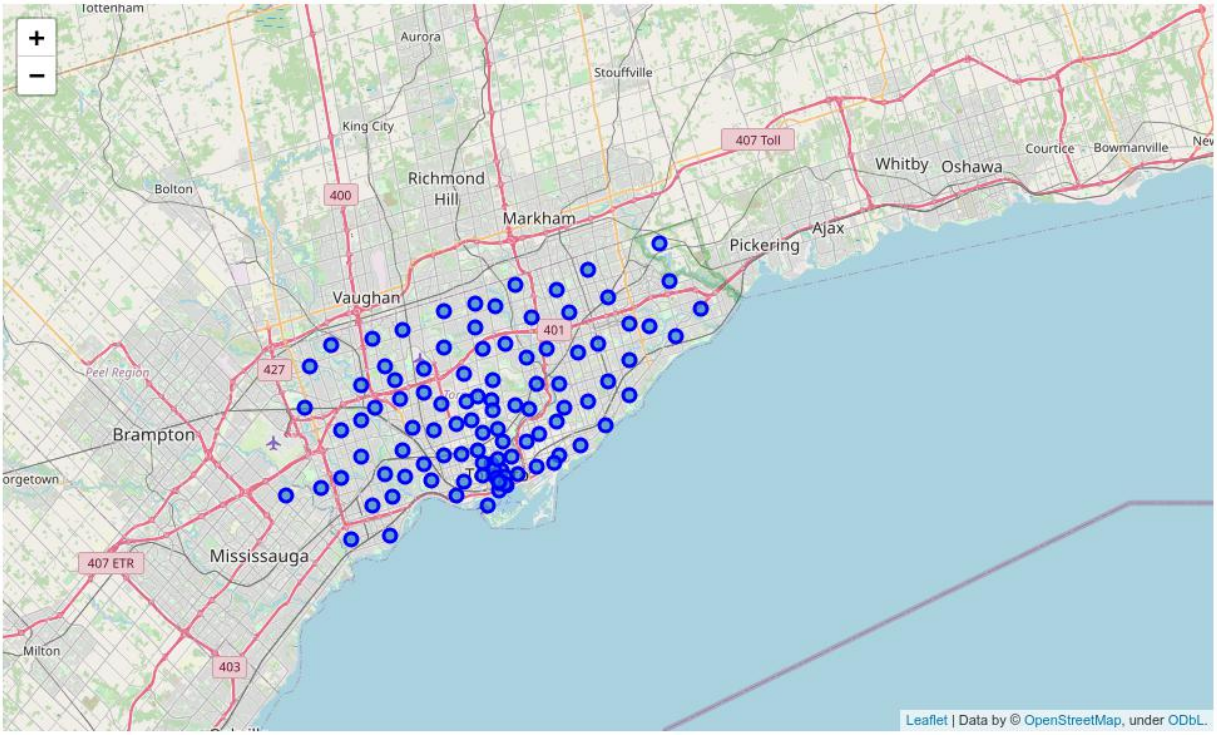
```
# define the dataframe columns
columns = []
for tr in table.find_all('th'):
    columns.append(tr.text.replace('\n', '').replace(' ', '').strip())

# Get the cell values and insert them to the dataframe
table_rows = table.find_all('tr')

res = []
for tr in table_rows:
    td = tr.find_all('td')
    row = [tr.text.strip() for tr in td if tr.text.strip()]
    if row:
        res.append(row)
df = pd.DataFrame(res, columns=columns)
df.head()
```

As a result, a dataframe is created with Borough, Neighborhood, Latitude and Longitude details of the Toronto's neighborhood.

	PostalCode	Borough	Neighbourhood	Latitude	Longitude
0	M3A	North York	Parkwoods	43.753259	-79.329656
1	M4A	North York	Victoria Village	43.725882	-79.315572
2	M5A	Downtown Toronto	Regent Park, Harbourfront	43.654260	-79.360636
3	M6A	North York	Lawrence Manor, Lawrence Heights	43.718518	-79.464763
4	M7A	Downtown Toronto	Queen's Park, Ontario Provincial Government	43.662301	-79.389494

Upon analysis, it is found that the dataframe consists of 10 boroughs and 103 neighborhoods. Further, 'geopy' library is used to get the latitude and longitude values of Toronto, which was returned to be Latitude: 43.6534817, Longitude: -79.3839347.

The following depiction is a map generated using python 'folium' library to visualize the map of Toronto and its neighborhoods.



RESTful API Calls to Foursquare

The Foursquare API is used to explore the neighborhoods and segment them. To access the API, 'CLIENT_ID', 'CLIENT_SECRET' and 'VERSION' is defined. There are many endpoints available on Foursquare for various GET requests. But, to explore the cuisines, it is required that all the venues extracted are from 'Food' category. Foursquare Venue Category Hierarchy is retrieved using the following code block: The returned request is:

```
url = 'https://api.foursquare.com/v2/venues/categories?&client_id={}&client_secret={}&v={}'.format(
    CLIENT_ID,
    CLIENT_SECRET,
    VERSION)
category_results = requests.get(url).json()
```

The returned request is further analyzed:

```
for key, value in category_results['response']['categories'][0].items():
    print(key, len(str(value)))
```

```
id 24
name 20
pluralName 20
shortName 20
icon 98
categories 15910
```

Upon analysis, it is found that there are 10 major or parent categories of venues, under which all the other sub-categories are included. Following depiction shows the 'Category ID' and 'Category Name' retrieved from API:

```
for data in category_list:
    print(data['id'], data['name'])
```

```
4d4b7104d754a06370d81259 Arts & Entertainment
4d4b7105d754a06372d81259 College & University
4d4b7105d754a06373d81259 Event
4d4b7105d754a06374d81259 Food
4d4b7105d754a06376d81259 Nightlife Spot
4d4b7105d754a06377d81259 Outdoors & Recreation
4d4b7105d754a06375d81259 Professional & Other Places
4e67e38e036454776db1fb3a Residence
4d4b7105d754a06378d81259 Shop & Service
4d4b7105d754a06379d81259 Travel & Transport
```

As said earlier, the ‘FOOD’ category in the above depiction is the matter of interest. A function is created to return a dictionary with ‘Category ID’ & ‘Category Name’ of ‘Food’ & it's sub-categories.

```
# function to flatten a 'parent_id' category, returns all categories if checkParentID = False
def flatten_Hierarchy(category_list, checkParentID, category_dict, parent_id = ''):
    for data in category_list:

        if checkParentID == True and data['id'] == parent_id:
            category_dict[data['id']] = data['name']
            flatten_Hierarchy(category_list = data['categories'], checkParentID = False, category_dict = category_d

        elif checkParentID == False:
            category_dict[data['id']] = data['name']
            if len(data['categories']) != 0:
                flatten_Hierarchy(category_list = data['categories'], checkParentID = False, category_dict = catego

    return category_dict
```

To further understand the results of GET Request, the first neighborhood of the ‘Toronto’ dataset is explored. The first neighborhood returned is ‘KFC’ with Latitude 43.7532586 and Longitude -79.3296565. Then, a GET request URL is created to search for Venue with ‘Category ID’ = '4d4b7105d754a06374d81259', which is the ‘Category ID’ for ‘Food’, and radius = 500 meters.

```
LIMIT = 1 # limit of number of venues returned by Foursquare API
radius = 500 # define radius
categoryId = '4d4b7105d754a06374d81259' # category ID for "Food"

# create URL

url = 'https://api.foursquare.com/v2/venues/search?&client_id={}&client_secret={}&v={}&ll={},{}&radius={}&categoryI
    CLIENT_ID,
    CLIENT_SECRET,
    VERSION,
    neighborhood_latitude,
    neighborhood_longitude,
    radius,
    categoryId,
    LIMIT)
url
```

The returned request is then examined, which is as follows:

```
In [23]: results['response']['venues']
```

```
Out[23]: [{'id': '4e6696b6d16433b9ffff47c3',
           'name': 'KFC',
           'location': {'lat': 43.75438666345904,
            'lng': -79.3330206627504,
            'labeledLatLngs': [{'label': 'display',
              'lat': 43.75438666345904,
              'lng': -79.3330206627504}],
            'distance': 298,
            'cc': 'CA',
            'country': 'Canada',
            'formattedAddress': ['Canada']},
           'categories': [{'id': '4bf58dd8d48988d16e941735',
             'name': 'Fast Food Restaurant',
             'pluralName': 'Fast Food Restaurants',
             'shortName': 'Fast Food',
             'icon': {'prefix': 'https://ss3.4sqi.net/img/categories_v2/food/fastfood_',
              'suffix': '.png'},
             'primary': True}],
           'referralId': 'v-1596375624',
           'hasPerk': False}]
```

The request returned the ‘Category Name’ of the venue as 'KFC' which is of 'Food' category. As, the aim is to segment the neighborhoods of Toronto with respect to the ‘Food’ in its vicinity, it is further required to fetch this data from all the 306 neighborhoods' venues.

To overcome the redundancy of the process followed above, a function ‘getNearbyFood’ is created. This functions loop through all the neighborhoods of Toronto and creates an API request URL with radius = 500, LIMIT = 100. By limit, it is defined that maximum 100 nearby venues should be returned. Further, the GET request is made to Foursquare API and only relevant information for each nearby venue is extracted from it. The data is then appended to a python ‘list’. Lastly the python ‘list’ is unfolded or flattened to append it to dataframe being returned by the function.
It is inquisitive to know that Foursquare API returns all the sub-categories, if a top-level category is specified in the GET Request.

```
def getNearbyFood(names, latitudes, longitudes, radius=1000, LIMIT=500):
    not_found = 0
    print('***Start ', end='')
    venues_list=[]
    for name, lat, lng in zip(names, latitudes, longitudes):
        print(' .', end='')

        # create the API request URL
        url = 'https://api.foursquare.com/v2/venues/search?&client_id={}&client_secret={}&v={}&ll={},{}&radius={}&c
            CLIENT_ID,
            CLIENT_SECRET,
            VERSION,
            lat,
            lng,
            radius,
            "4d4b7105d754a06374d81259", # "Food" category id
            LIMIT)

        try:
            # make the GET request
            results = requests.get(url).json()['response']['venues']

            # return only relevant information for each nearby venue
            venues_list.append([(
                name,
                lat,
                lng,
                v['name'],
                v['location']['lat'],
                v['location']['lng'],
                v['categories'][0]['name']) for v in results])
        except:
            not_found += 1
```

Pickle

Pickle is a very important and easy-to-use library. It is used to serialize the information retrieved from GET requests, to make a persistent '.pkl' file. This file can later be deserialized to retrieve an exact python object structure. This is a crucial as it will counter any redundant requests to the Foursquare API, which is chargeable over the threshold limits

```
import pickle # to serialize and deserialize a Python object structure
try:
    with open('toroto_food_venues.pkl', 'rb') as f:
        toronto_venues = pickle.load(f)
    print("---Dataframe Existed and Deserialized---")
except:
    toronto_venues = getNearbyFood(names=df2['Neighbourhood'],
                                   latitudes=df2['Latitude'],
                                   longitudes=df2['Longitude']
                                  )
    with open('toronto_food_venues.pkl', 'wb') as f:
        pickle.dump(toronto_venues, f)
    print("---Dataframe Created and Serialized---")
```

The returned 'dataframe' is as follows:

	Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
0	Parkwoods	43.753259	-79.329656	Allwyn's Bakery	43.759840	-79.324719	Caribbean Restaurant
1	Parkwoods	43.753259	-79.329656	Subway	43.760334	-79.326906	Sandwich Place
2	Parkwoods	43.753259	-79.329656	Allwyn's	43.761000	-79.325478	Caribbean Restaurant
3	Parkwoods	43.753259	-79.329656	Joey	43.753441	-79.321640	Burger Joint
4	Parkwoods	43.753259	-79.329656	A&W	43.760643	-79.326865	Fast Food Restaurant

Exploratory Data Analysis

The merged dataframe ‘toronto_venues’ has all the required information. The size of this dataframe is determined, and it is found that there are total 4263 venues.

```
In [27]: print(toronto_venues.shape)
         toronto_venues.head()
```

```
(4263, 7)
```

Out[27]:

	Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
0	Parkwoods	43.753259	-79.329656	Allwyn's Bakery	43.759840	-79.324719	Caribbean Restaurant
1	Parkwoods	43.753259	-79.329656	Subway	43.760334	-79.326906	Sandwich Place
2	Parkwoods	43.753259	-79.329656	Allwyn's	43.761000	-79.325478	Caribbean Restaurant
3	Parkwoods	43.753259	-79.329656	Joey	43.753441	-79.321640	Burger Joint
4	Parkwoods	43.753259	-79.329656	A&W	43.760643	-79.326865	Fast Food Restaurant

Now, it is important to find out that how many unique categories can be curated from all the returned venues. There are 156 such categories, with most occurring venues as follows:

```
print('There are {} uniques categories.'.format(len(toronto_venues['Venue Category'].unique())))
toronto_venues.groupby('Venue Category')['Venue Category'].count().sort_values(ascending=False)
```

```
There are 156 uniques categories.

Venue Category
Coffee Shop                    568
Pizza Place                    260
Restaurant                     211
Café                           205
Bakery                         182
Fast Food Restaurant           175
Italian Restaurant             144
Chinese Restaurant             137
Sandwich Place                 121
Caribbean Restaurant            96
Indian Restaurant               87
Sushi Restaurant                84
Japanese Restaurant             80
Ice Cream Shop                  78
Grocery Store                   75
Asian Restaurant                68
Middle Eastern Restaurant       64
Breakfast Spot                  63
Burger Joint                    63
Thai Restaurant                 62
Dessert Shop                    62
Bubble Tea Shop                 61
American Restaurant             59
Fried Chicken Joint             53
```

Data Cleaning

It is crucial to understand that the point of interest in the project is to understand the cultural diversity of a neighborhood by clustering it categorically, using the venues’ categories. Thus, it is important to remove all the venues from the ‘dataframe’ which have generalized categories. Here, by generalized, it means that these categorized venues are common across different

cultures and food habits. Example of categories of this type of venues are Coffee Shop, Cafe, etc. So, firstly all the unique categories are fed into a python ‘list’.

```
['Dessert Shop',
 'Food',
 'Ice Cream Shop',
 'Fast Food Restaurant',
 'Donut Shop',
 'Caribbean Restaurant',
 'Bakery',
 'Sandwich Place',
 'Italian Restaurant',
 'Comfort Food Restaurant',
 'Fried Chicken Joint',
 'Deli / Bodega',
 'Food Truck',
 'Chinese Restaurant',
 'Pizza Place',
 'Southern / Soul Food Restaurant',
 'Halal Restaurant',
 'Asian Restaurant',
 'Bagel Shop',
 'American Restaurant',
 'Burger Joint',
 'Restaurant',
 'Mexican Restaurant',
 'Seafood Restaurant',
 'Frozen Yogurt Shop',
 'Spanish Restaurant',
```

Data Cleaning It is crucial to understand that the point of interest in the project is to understand the cultural diversity of a neighborhood by clustering it categorically, using the venues’ categories. Thus, it is important to remove all the venues from the ‘dataframe’ which have generalized categories. Here, by generalized, it means that these categorized venues are common across different cultures and food habits. Example of categories of this type of venues are Coffee Shop, Cafe, etc. So, firstly all the unique categories are fed into a python ‘list’. Then, manually the categories are determined to be ‘general’ (as explained above). This data pre-preparation totally depends upon the ‘Data Analyst’ discretion and can be modified as required. Following are the categories listed as ‘general’:

```
# manually create a list of generalized categories
general_categories = ['Dessert Shop','Food','Ice Cream Shop','Donut Shop','Bakery','Sandwich Place','Comfort Food R
                      'Deli / Bodega','Food Truck','Bagel Shop','Burger Joint','Restaurant','Frozen Yogurt Shop','Cof
                      'Diner','Wings Joint','Café','Juice Bar','Breakfast Spot','Grocery Store','Bar','Cupcake Shop',
                      'Pub','Fish & Chips Shop','Cafeteria','Other Nightlife','Arcade','Hot Dog Joint','Food Court',
                      'Health Food Store','Convenience Store','Food & Drink Shop','Cocktail Bar','Cheese Shop',
                      'Snack Place','Sports Bar','Lounge','Theme Restaurant','Buffet','Bubble Tea Shop','Building',
                      'Irish Pub','College Cafeteria','Tea Room','Supermarket','Hotpot Restaurant','Gastropub','Beer
                      'Fish Market','Beer Bar','Clothing Store','Music Venue','Bistro','Salad Place','Wine Bar','Gour
                      'Indie Movie Theater','Art Gallery','Gift Shop','Pie Shop','Fruit & Vegetable Store',
                      'Street Food Gathering','Dive Bar','Factory','Farmers Market','Mac & Cheese Joint','Creperie',
                      'Candy Store','Event Space','Skating Rink','Miscellaneous Shop','Gas Station','Organic Grocery'
                      'Pastry Shop','Club House','Flea Market','Hotel','Furniture / Home Store','Bookstore','Pet Café
                      'Gym / Fitness Center','Flower Shop','Financial or Legal Service','Hotel Bar','Hookah Bar','Pok
                      'Market','Gluten-free Restaurant','Smoothie Shop','Butcher','Food Stand','Beach Bar','Beach',
                      'Soup Place','Rock Club','Residential Building (Apartment / Condo)','Laundry Service',
                      'Government Building','Bowling Alley','Nightclub','Park','Moving Target']
```

A simple subtraction of two python ‘list’ i.e ‘unique_categories’ and ‘general_categories’ gives a ‘list’ of all the categories which are required for further analysis. Following image depicts the result of the above activity:

```
# fetch all the required food categories
food_categories =  list(set(unique_categories) - set(general_categories))
food_categories
```

```
['Paella Restaurant',
 'Brazilian Restaurant',
 'Steakhouse',
 'Mexican Restaurant',
 'Vegetarian / Vegan Restaurant',
 'Italian Restaurant',
 'Peruvian Restaurant',
 'Jewish Restaurant',
 'African Restaurant',
 'Ukrainian Restaurant',
 'Sri Lankan Restaurant',
 'Mediterranean Restaurant',
 'Caucasian Restaurant',
 'English Restaurant',
 'American Restaurant',
 'Korean Restaurant',
 'Vietnamese Restaurant',
 'Cantonese Restaurant',
 'Tapas Restaurant',
 'Hong Kong Restaurant',
 'Persian Restaurant',
 'Dosa Place',
 'Greek Restaurant',
 'Australian Restaurant',
 'Himalayan Restaurant',
 'Taiwanese Restaurant',
 'Seafood Restaurant',
 'Salvadoran Restaurant',
 'Japanese Restaurant',
 'Venezuelan Restaurant',
 'Ramen Restaurant',
```

A simple subtraction of two python ‘list’ i.e ‘unique_categories’ and ‘general_categories’ gives a ‘list’ of all the categories which are required for further analysis. Following image depicts the result of the above activity: The python ‘list’ curated above, is used to remove all the venues with categories not in ‘food_categories’, and the following dataframe is retrieved:

```
nyc_venues = nyc_venues[nyc_venues['Venue Category'].isin(food_categories)].reset_index()
nyc_venues.head(5)
```

	index	Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
0	3	Wakefield	40.894705	-73.847201	Burger King	40.895532	-73.856436	Fast Food Restaurant
1	5	Wakefield	40.894705	-73.847201	Cooler Runnings Jamaican Restaurant Inc	40.898276	-73.850381	Caribbean Restaurant
2	9	Wakefield	40.894705	-73.847201	McDonald's	40.902645	-73.849485	Fast Food Restaurant
3	10	Wakefield	40.894705	-73.847201	Ripe Kitchen & Bar	40.898152	-73.838875	Caribbean Restaurant
4	11	Wakefield	40.894705	-73.847201	Frank and Johnies	40.905019	-73.858392	Italian Restaurant

Again, the number of unique categories is examined, and it is found that there are only 92 of them, as compared to 156 earlier. That means, almost 50% of the data was a noise for the analysis. This essential step, data cleaning, helped to capture the data points of interest.

Feature Engineering

Now, each neighborhood is analyzed individually to understand the most common cuisine being served within its 500 meters of vicinity. The above process is taken forth by using ‘one

hot encoding' function of python 'pandas' library. One hot encoding converts the categorical variables (which are 'Venue Category') into a form that could be provided to ML algorithms to do a better job in prediction.

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

	Afghan Restaurant	African Restaurant	American Restaurant	Argentinian Restaurant	Asian Restaurant	BBQ Joint	Bangladeshi Restaurant	Belgian Restaurant	Bike Shop	Brazilian Restaurant	Brewery	Burmese Restaurant	Burrito Place	Cajun / Creole Restaurant
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
2	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
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Upon converting the categorical variables, as shown above, 'Neighbourhood' column is added back which results into the following:

In [54]:
```
# move neighborhood column to the first column
Neighborhood = toronto_onehot['Neighborhood']

toronto_onehot.drop(labels=['Neighborhood'], axis=1,inplace = True)
toronto_onehot.insert(0, 'Neighborhood', Neighborhood)

toronto_onehot.head()
```

Out[54]:

	Neighborhood	Afghan Restaurant	African Restaurant	American Restaurant	Argentinian Restaurant	Asian Restaurant	BBQ Joint	Bangladeshi Restaurant	Belgian Restaurant	Bike Shop	Brazilian Restaurant	Brewery	Burmese Restaurant	Burr Pla
0	Parkwoods	0	0	0	0	0	0	0	0	0	0	0	0	
1	Parkwoods	0	0	0	0	0	0	0	0	0	0	0	0	
2	Parkwoods	0	0	0	0	0	0	0	0	0	0	0	0	
3	Parkwoods	0	0	0	0	0	0	0	0	0	0	0	0	
4	Parkwoods	0	0	0	0	0	0	0	0	0	0	0	0	

Further, number of venues of each category in each neighborhood are counted.

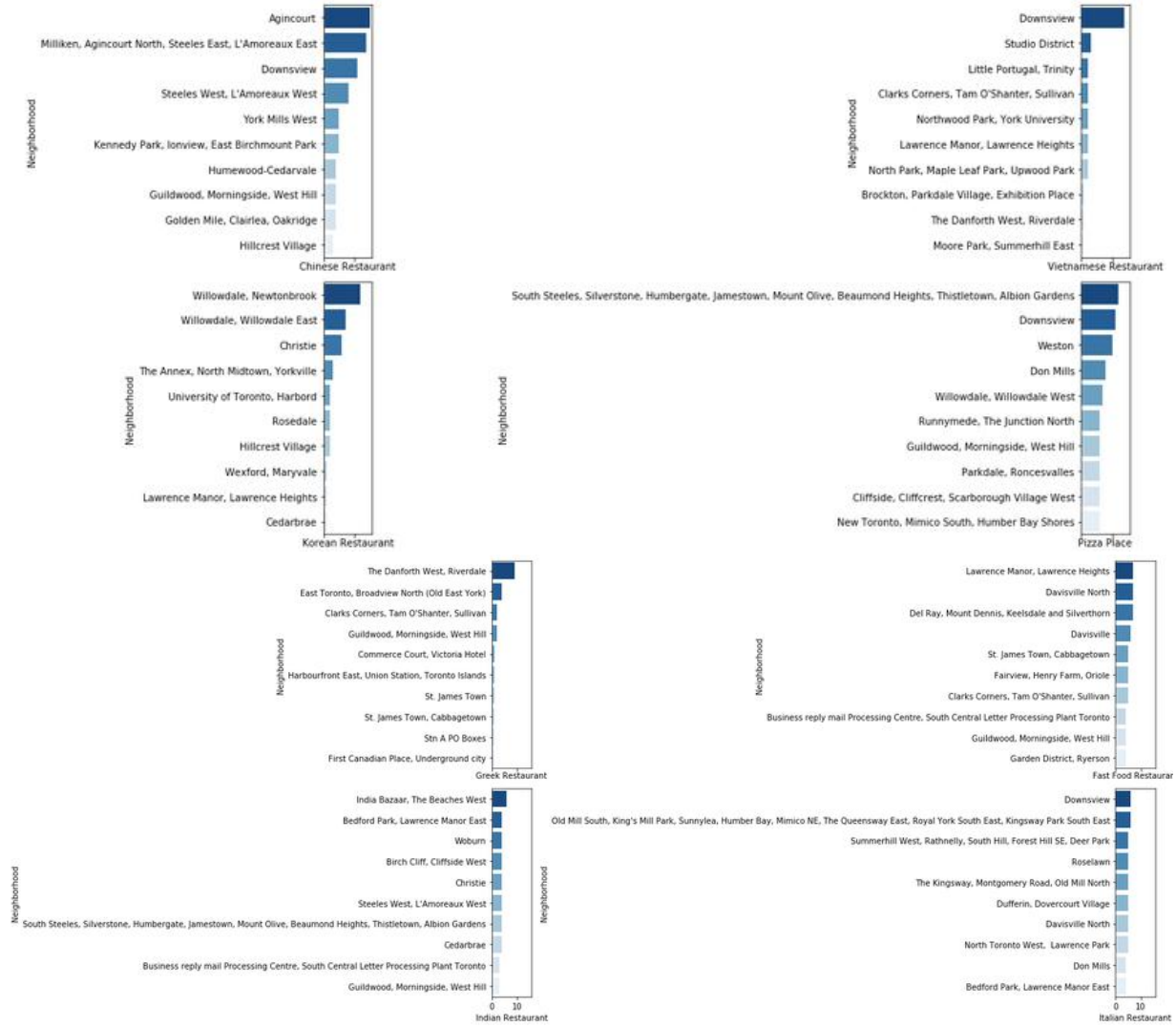
```
venue_counts = toronto_onehot.groupby('Neighborhood').sum()
venue_counts.head(5)
```

Neighborhood	Afghan Restaurant	African Restaurant	American Restaurant	Argentinian Restaurant	Asian Restaurant	BBQ Joint	Bangladeshi Restaurant	Belgian Restaurant	Bike Shop	Brazilian Restaurant	Brewery	Burmese Restaurant	Burrito Place
Agincourt	0	0	1	0	2	1	0	0	0	0	0	0	0
Alderwood, Long Branch	0	0	1	0	2	2	0	0	0	0	0	1	0
Bathurst Manor, Wilson Heights, Downsview North	0	0	1	0	0	0	0	0	0	0	0	0	0
Bayview Village	0	0	0	0	2	0	0	0	0	0	0	0	0
Bedford Park, Lawrence Manor East	0	0	1	0	0	0	0	0	0	0	0	0	0

Data visualization

These top 10 categories are further plotted individually on bar graph using python 'seaborn' library. The following code block creates the graph of top 10 neighborhoods for a category.

```
import seaborn as sns
import matplotlib.pyplot as plt

fig, axes =plt.subplots(5, 2, figsize=(20,20), sharex=True)
axes = axes.flatten()

for ax, category in zip(axes, venue_top10_list):
    data = venue_counts[[category]].sort_values([category], ascending=False)[0:10]
    pal = sns.color_palette("Blues", len(data))
    sns.barplot(x=category, y=data.index, data=data, ax=ax, palette=np.array(pal[::-1]))

plt.tight_layout()
plt.show();
```



Next, the rows of the neighborhood are grouped together and the frequency of occurrence of each category is calculated by taking the mean.

```
toronto_grouped = toronto_onehot.groupby('Neighborhood').mean().reset_index()
toronto_grouped.head()
```

	Neighborhood	Afghan Restaurant	African Restaurant	American Restaurant	Argentinian Restaurant	Asian Restaurant	BBQ Joint	Bangladeshi Restaurant	Belgian Restaurant	Bike Shop	Brazilian Restaurant	Brewery	Burmese Restaurant	E
0	Agincourt	0.0	0.0	0.031250	0.0	0.062500	0.031250	0.0	0.0	0.0	0.0	0.0	0.000000	
1	Alderwood, Long Branch	0.0	0.0	0.047619	0.0	0.095238	0.095238	0.0	0.0	0.0	0.0	0.0	0.047619	
2	Bathurst Manor, Wilson Heights, Downsview North	0.0	0.0	0.083333	0.0	0.000000	0.000000	0.0	0.0	0.0	0.0	0.0	0.000000	
3	Bayview Village	0.0	0.0	0.000000	0.0	0.222222	0.000000	0.0	0.0	0.0	0.0	0.0	0.000000	
4	Bedford Park, Lawrence Manor East	0.0	0.0	0.043478	0.0	0.000000	0.000000	0.0	0.0	0.0	0.0	0.0	0.000000	

As the limit is set to be 100, there will be many venues being returned by the Foursquare API. But a neighborhood food habit can be defined by the top 5 venues in its vicinity. Following 'for' loop creates a dataframe to record the above-mentioned data points:

```
num_top_venues = 5

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'] = toronto_grouped['Neighborhood']
```

Further, the above created dataframe is fed with the top 5 most common venues categories in the respective neighborhood.

	Neighborhood	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue
0	Agincourt	Chinese Restaurant	Caribbean Restaurant	Indian Restaurant	Asian Restaurant	Filipino Restaurant
1	Alderwood, Long Branch	Italian Restaurant	Pizza Place	Asian Restaurant	Thai Restaurant	BBQ Joint
2	Bathurst Manor, Wilson Heights, Downsview North	Pizza Place	Middle Eastern Restaurant	Sushi Restaurant	Japanese Restaurant	American Restaurant
3	Bayview Village	Japanese Restaurant	Asian Restaurant	Sushi Restaurant	Korean Restaurant	Chinese Restaurant
4	Bedford Park, Lawrence Manor East	Italian Restaurant	Indian Restaurant	Sushi Restaurant	Pizza Place	Fast Food Restaurant

Machine Learning

'k-means' is an unsupervised machine learning algorithm which creates clusters of data points aggregated together because of certain similarities. This algorithm will be used to count neighborhoods for each cluster label for variable cluster size. To implement this algorithm, it is very important to determine the optimal number of clusters (i.e. k). There are 2 most popular methods for the same, namely 'The Elbow Method' and 'The Silhouette Method'.

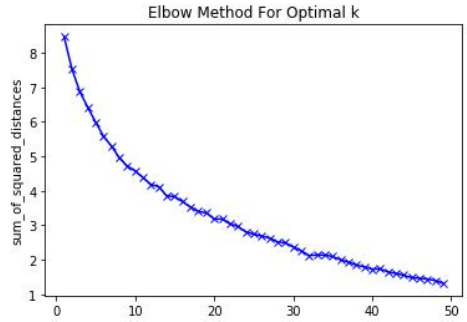
The Elbow Method:

The Elbow Method calculates the sum of squared distances of samples to their closest cluster center for different values of 'k'. The optimal number of clusters is the value after which there is no significant decrease in the sum of squared distances. Following is an implementation of this method (with varying number of clusters from 1 to 49):

```
sum_of_squared_distances = []
K = range(1,50)
for k in K:
    print(k, end=' ')
    kmeans = KMeans(n_clusters=k).fit(toronto_grouped_clustering)
    sum_of_squared_distances.append(kmeans.inertia_)
```

```
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41
42 43 44 45 46 47 48 49
```

```
plt.plot(K, sum_of_squared_distances, 'bx-')
plt.xlabel('k')
plt.ylabel('sum_of_squared_distances')
plt.title('Elbow Method For Optimal k');
```



Sometimes, Elbow method does not give the required result, which happened in this case. As, there is a gradual decrease in the sum of squared distances, optimal number of clusters can not be determined. To counter this, another method can be implemented, as discussed below.

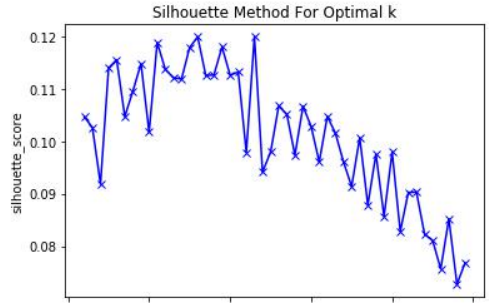
The Silhouette Method:

As quoted in Wikipedia – "The Silhouette Method measures how similar a point is to its own cluster (cohesion) compared to other clusters (separation)." Following is an implementation of this method. As it requires minimum 2 clusters to define dissimilarity number of clusters (i.e. 'k') will vary from 2 to 49:

```
from sklearn.metrics import silhouette_score

sil = []
K_sil = range(2,50)
# minimum 2 clusters required, to define dissimilarity
for k in K_sil:
    print(k, end=' ')
    kmeans = KMeans(n_clusters = k).fit(toronto_grouped_clustering)
    labels = kmeans.labels_
    sil.append(silhouette_score(toronto_grouped_clustering, labels, metric = 'euclidean'))
```

```
2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42
43 44 45 46 47 48 49
```

```
plt.plot(K_sil, sil, 'bx-')
plt.xlabel('k')
plt.ylabel('silhouette_score')
plt.title('Silhouette Method For Optimal k')
plt.show()
```



There is a peak at k = 2, k = 4 and k = 8. Two and four number of clusters will cluster the neighborhoods very broadly. Therefore, number of clusters (i.e. ‘k’) is chosen to be 8.

k-Means

Following code block runs the k-Means algorithm with number of clusters = 8 and prints the counts of neighborhoods assigned to different clusters:

```
# set number of clusters
kclusters = 8

# run k-means clustering
kmeans = KMeans(init="k-means++", n_clusters=kclusters, n_init=50).fit(toronto_grouped_clustering)

print(Counter(kmeans.labels_))
```

```
Counter({4: 29, 1: 23, 6: 19, 2: 11, 5: 8, 0: 5, 7: 2, 3: 1})
```

Further the cluster labels curated are added to the dataframe to get the desired results of segmenting the neighborhood based upon the most common venues in its vicinity:

```
# add clustering labels
try:
    neighborhoods_venues_sorted.drop('Cluster Labels', axis=1)
except:
    neighborhoods_venues_sorted.insert(0, 'Cluster Labels', kmeans.labels_)
```

```
neighborhoods_venues_sorted.head(5)
```

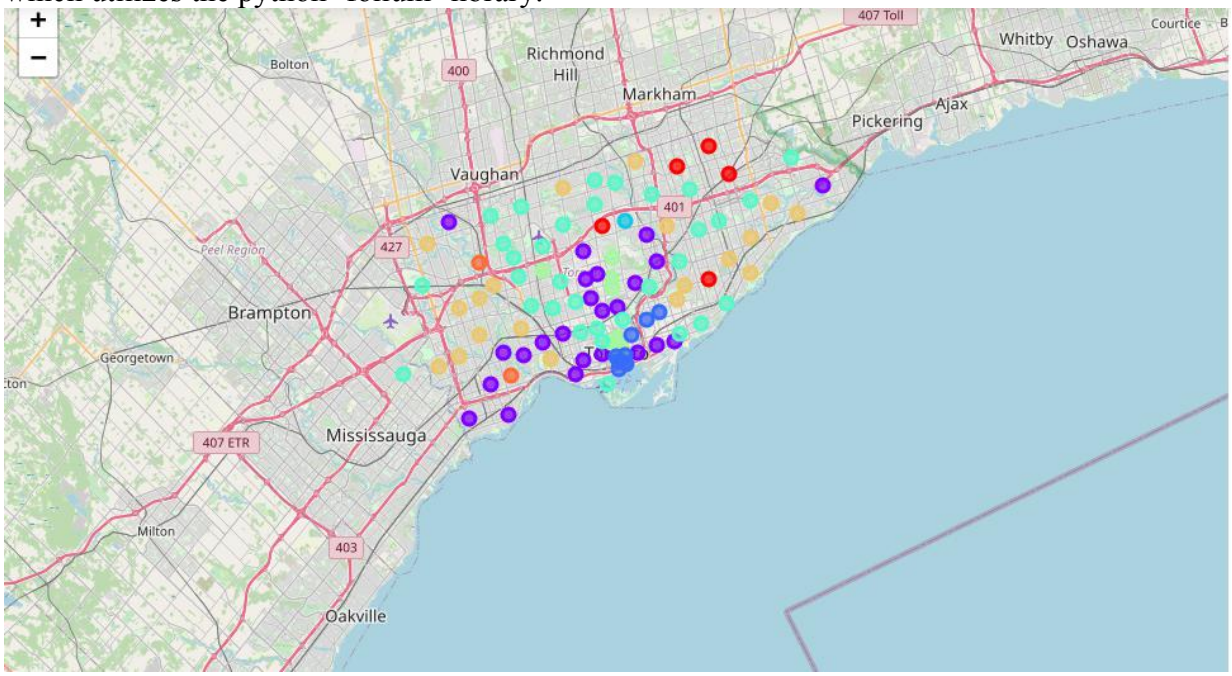
	Cluster Labels	Neighborhood	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue
0	0	Agincourt	Chinese Restaurant	Caribbean Restaurant	Indian Restaurant	Asian Restaurant	Filipino Restaurant
1	1	Alderwood, Long Branch	Italian Restaurant	Pizza Place	Asian Restaurant	Thai Restaurant	BBQ Joint
2	4	Bathurst Manor, Wilson Heights, Downsview North	Pizza Place	Middle Eastern Restaurant	Sushi Restaurant	Japanese Restaurant	American Restaurant
3	4	Bayview Village	Japanese Restaurant	Asian Restaurant	Sushi Restaurant	Korean Restaurant	Chinese Restaurant
4	1	Bedford Park, Lawrence Manor East	Italian Restaurant	Indian Restaurant	Sushi Restaurant	Pizza Place	Fast Food Restaurant

Now, ‘neighborhoods_venues_sorted’ is merged with ‘toronto_data’ to add the Borough, Latitude and Longitude for each neighborhood.

```
# merge neighborhoods_venues_sorted with nyc_data to add latitude/longitude for each neighborhood
toronto_merged = neighborhoods_venues_sorted.join(df2.set_index('Neighbourhood'), on='Neighborhood')
toronto_merged.head()
```

	Cluster Labels	Neighborhood	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	PostalCode	Borough	Latitude	Longitude
0	0	Agincourt	Chinese Restaurant	Caribbean Restaurant	Indian Restaurant	Asian Restaurant	Filipino Restaurant	M1S	Scarborough	43.794200	-79.262029
1	1	Alderwood, Long Branch	Italian Restaurant	Pizza Place	Asian Restaurant	Thai Restaurant	BBQ Joint	M8W	Etobicoke	43.602414	-79.543484
2	4	Bathurst Manor, Wilson Heights, Downsview North	Pizza Place	Middle Eastern Restaurant	Sushi Restaurant	Japanese Restaurant	American Restaurant	M3H	North York	43.754328	-79.442259
3	4	Bayview Village	Japanese Restaurant	Asian Restaurant	Sushi Restaurant	Korean Restaurant	Chinese Restaurant	M2K	North York	43.786947	-79.385975
4	1	Bedford Park, Lawrence Manor East	Italian Restaurant	Indian Restaurant	Sushi Restaurant	Pizza Place	Fast Food Restaurant	M5M	North York	43.733283	-79.419750

Again, the Toronto City’s neighborhoods are visualized by using the code block as shown, which utilizes the python ‘folium’ library.



Cluster 0

```
cluster_0 = toronto_merged.loc[toronto_merged['Cluster Labels'] == 0, toronto_merged.columns[1:12]]
cluster_0.head(5)
```

	Neighborhood	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	PostalCode	Borough	Latitude	Longitude
0	Agincourt	Chinese Restaurant	Caribbean Restaurant	Indian Restaurant	Asian Restaurant	Filipino Restaurant	M1S	Scarborough	43.794200	-79.262029
33	Golden Mile, Clairlea, Oakridge	Chinese Restaurant	Indian Restaurant	Pizza Place	Vegetarian / Vegan Restaurant	Falafel Restaurant	M1L	Scarborough	43.711112	-79.284577
51	Milliken, Agincourt North, Steeles East, L'Amo...	Chinese Restaurant	BBQ Joint	Noodle House	Asian Restaurant	Ramen Restaurant	M1V	Scarborough	43.815252	-79.284577
75	Steeles West, L'Amoreaux West	Chinese Restaurant	Indian Restaurant	BBQ Joint	Pizza Place	Fast Food Restaurant	M1W	Scarborough	43.799525	-79.318389
96	York Mills West	Chinese Restaurant	Japanese Restaurant	Burrito Place	Sushi Restaurant	Vietnamese Restaurant	M2P	North York	43.752758	-79.400049

Following are the results of the Cluster – 0 analysis:

```
for col in required_column:
    print(cluster_0[col].value_counts(ascending = False))
    print("-----------------------------------------------")
```

```
Chinese Restaurant    5
Name: 1st Most Common Venue, dtype: int64
-----------------------------------------------
Indian Restaurant       2
BBQ Joint               1
Caribbean Restaurant    1
Japanese Restaurant     1
Name: 2nd Most Common Venue, dtype: int64
-----------------------------------------------
M1V    1
M2P    1
M1L    1
M1S    1
M1W    1
Name: PostalCode, dtype: int64
-----------------------------------------------
```

Cluster 1

```
cluster_1 = toronto_merged.loc[toronto_merged['Cluster Labels'] == 1, toronto_merged.columns[1:12]]
cluster_1.head(5)
```

	Neighborhood	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	PostalCode	Borough	Latitude	Longitude
1	Alderwood, Long Branch	Italian Restaurant	Pizza Place	Asian Restaurant	Thai Restaurant	BBQ Joint	M8W	Etobicoke	43.602414	-79.543484
4	Bedford Park, Lawrence Manor East	Italian Restaurant	Indian Restaurant	Fast Food Restaurant	Pizza Place	Sushi Restaurant	M5M	North York	43.733283	-79.419750
22	Don Mills	Pizza Place	Japanese Restaurant	Asian Restaurant	Italian Restaurant	Thai Restaurant	M3B	North York	43.745906	-79.352188
22	Don Mills	Pizza Place	Japanese Restaurant	Asian Restaurant	Italian Restaurant	Thai Restaurant	M3C	North York	43.725900	-79.340923
25	Dufferin, Dovercourt Village	Italian Restaurant	Portuguese Restaurant	Pizza Place	Sushi Restaurant	Brazilian Restaurant	M6H	West Toronto	43.669005	-79.442259

Following are the results of the Cluster – 1 analysis:

```
for col in required_column:
    print(cluster_1[col].value_counts(ascending = False))
    print("-----------------------------------------------")
```

```
Italian Restaurant    14
Pizza Place            3
Sushi Restaurant       1
Seafood Restaurant     1
Name: 1st Most Common Venue, dtype: int64
-----------------------------------------------
Pizza Place                   5
Japanese Restaurant           3
Italian Restaurant            3
African Restaurant            1
Portuguese Restaurant         1
Vietnamese Restaurant         1
Indian Restaurant             1
Burrito Place                 1
Sushi Restaurant              1
Modern European Restaurant    1
Thai Restaurant               1
Name: 2nd Most Common Venue, dtype: int64
-----------------------------------------------
M3C    1
M5M    1
M5P    1
M4V    1
M8Z    1
M9M    1
M4R    1
M8W    1
M6J    1
M3B    1
M6H    1
```

Cluster 2

```
cluster_2 = toronto_merged.loc[toronto_merged['Cluster Labels'] == 2, toronto_merged.columns[1:12]]
cluster_2.head(5)
```

	Neighborhood	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	PostalCode	Borough	Latitude	Longitude
90	Wexford, Maryvale	Middle Eastern Restaurant	Pizza Place	African Restaurant	American Restaurant	Halal Restaurant	M1R	Scarborough	43.750072	-79.295849
97	York Mills, Silver Hills	Middle Eastern Restaurant	Mediterranean Restaurant	Xinjiang Restaurant	Greek Restaurant	Empanada Restaurant	M2L	North York	43.757490	-79.374714

Following are the results of the Cluster – 2 analysis:

```
for col in required_column:
    print(cluster_2[col].value_counts(ascending = False))
    print("-----------------------------------------------")
```

```
Middle Eastern Restaurant    2
Name: 1st Most Common Venue, dtype: int64
-----------------------------------------------
Pizza Place                 1
Mediterranean Restaurant    1
Name: 2nd Most Common Venue, dtype: int64
-----------------------------------------------
M2L    1
M1R    1
Name: PostalCode, dtype: int64
-----------------------------------------------
```

Cluster 3

```
cluster_3 = toronto_merged.loc[toronto_merged['Cluster Labels'] == 3, toronto_merged.columns[1:12]]
cluster_3.head(5)
```

	Neighborhood	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	PostalCode	Borough	Latitude	Longitude
13	Central Bay Street	Fast Food Restaurant	Italian Restaurant	Thai Restaurant	Japanese Restaurant	Mexican Restaurant	M5G	Downtown Toronto	43.657952	-79.387383
15	Church and Wellesley	Fast Food Restaurant	Italian Restaurant	Japanese Restaurant	Hawaiian Restaurant	Pizza Place	M4Y	Downtown Toronto	43.665860	-79.383160
19	Davisville	Fast Food Restaurant	Ramen Restaurant	Mexican Restaurant	Thai Restaurant	Taco Place	M4S	Central Toronto	43.704324	-79.388790
20	Davisville North	Fast Food Restaurant	Italian Restaurant	Ramen Restaurant	Pizza Place	Thai Restaurant	M4P	Central Toronto	43.712751	-79.390197
31	Garden District, Ryerson	Fast Food Restaurant	Japanese Restaurant	Italian Restaurant	American Restaurant	Thai Restaurant	M5B	Downtown Toronto	43.657162	-79.378937

Following are the results of the Cluster – 3 analysis:

```
for col in required_column:
    print(cluster_3[col].value_counts(ascending = False))
    print("-----------------------------------------------")
```

```
Fast Food Restaurant    7
Sushi Restaurant        1
Name: 1st Most Common Venue, dtype: int64
-----------------------------------------------
Italian Restaurant      5
Japanese Restaurant     1
Ramen Restaurant        1
Fast Food Restaurant    1
Name: 2nd Most Common Venue, dtype: int64
-----------------------------------------------
M4Y    1
M5B    1
M4N    1
M4S    1
M5G    1
M7A    1
M6A    1
M4P    1
Name: PostalCode, dtype: int64
```

Cluster 4

```
cluster_4 = toronto_merged.loc[toronto_merged['Cluster Labels'] == 4, toronto_merged.columns[1:12]]
cluster_4.head(5)
```

	Neighborhood	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	PostalCode	Borough	Latitude	Longitude
3	Bayview Village	Japanese Restaurant	Asian Restaurant	Sushi Restaurant	Korean Restaurant	Chinese Restaurant	M2K	North York	43.786947	-79.385975
14	Christie	Korean Restaurant	Indian Restaurant	Italian Restaurant	Mexican Restaurant	Sushi Restaurant	M6G	Downtown Toronto	43.669542	-79.422564
79	The Annex, North Midtown, Yorkville	Korean Restaurant	Middle Eastern Restaurant	Vegetarian / Vegan Restaurant	Mediterranean Restaurant	Falafel Restaurant	M5R	Central Toronto	43.672710	-79.405678
91	Willowdale, Newtonbrook	Korean Restaurant	Fried Chicken Joint	Pizza Place	BBQ Joint	Ramen Restaurant	M2M	North York	43.789053	-79.408493
92	Willowdale, Willowdale East	Korean Restaurant	Noodle House	Fried Chicken Joint	Sushi Restaurant	Italian Restaurant	M2N	North York	43.770120	-79.408493

Following are the results of the Cluster – 4 analysis:

```
for col in required_column:
    print(cluster_4[col].value_counts(ascending = False))
    print("-----------------------------------------------")
```

```
Korean Restaurant      4
Japanese Restaurant    1
Name: 1st Most Common Venue, dtype: int64
-----------------------------------------------
Noodle House                 1
Fried Chicken Joint          1
Indian Restaurant            1
Asian Restaurant             1
Middle Eastern Restaurant    1
Name: 2nd Most Common Venue, dtype: int64
-----------------------------------------------
M2K    1
M5R    1
M2M    1
M6G    1
M2N    1
Name: PostalCode, dtype: int64
-----------------------------------------------
```

Cluster 5

```
cluster_5 = toronto_merged.loc[toronto_merged['Cluster Labels'] == 5, toronto_merged.columns[1:12]]
cluster_5.head(5)
```

	Neighborhood	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	PostalCode	Borough	Latitude	Longitude
2	Bathurst Manor, Wilson Heights, Downsview North	Pizza Place	Middle Eastern Restaurant	Sushi Restaurant	Japanese Restaurant	American Restaurant	M3H	North York	43.754328	-79.442259
17	Cliffside, Cliffcrest, Scarborough Village West	Pizza Place	Fast Food Restaurant	Cajun / Creole Restaurant	American Restaurant	Filipino Restaurant	M1M	Scarborough	43.716316	-79.239476
27	Eringate, Bloordale Gardens, Old Burnhamthorpe...	Pizza Place	Fried Chicken Joint	Scandinavian Restaurant	Seafood Restaurant	Thai Restaurant	M9C	Etobicoke	43.643515	-79.577201
34	Guildwood, Morningside, West Hill	Pizza Place	Fast Food Restaurant	Chinese Restaurant	Indian Restaurant	Greek Restaurant	M1E	Scarborough	43.763573	-79.188711
36	High Park, The Junction South	Pizza Place	Mexican Restaurant	Sushi Restaurant	Indian Restaurant	Burrito Place	M6P	West Toronto	43.661608	-79.464763

Following are the results of the Cluster – 5 analysis:

```
for col in required_column:
    print(cluster_5[col].value_counts(ascending = False))
    print("-------------------------------------------")
```

```
Pizza Place             22
Indian Restaurant        1
Fast Food Restaurant     1
Name: 1st Most Common Venue, dtype: int64
-------------------------------------------
Chinese Restaurant           5
Fast Food Restaurant         5
Sushi Restaurant             3
Pizza Place                  2
Indian Restaurant            1
BBQ Joint                    1
Japanese Restaurant          1
Mexican Restaurant           1
Middle Eastern Restaurant    1
Italian Restaurant           1
Fried Chicken Joint          1
Caribbean Restaurant         1
Thai Restaurant              1
Name: 2nd Most Common Venue, dtype: int64
-------------------------------------------
M1M    1
M3H    1
M2H    1
M9A    1
M4T    1
M9C    1
M8V    1
M1G    1
M9N    1
M9V    1
M2R    1
M4B    1
M9B    1
M9R    1
```

Cluster 6

```
cluster_6 = toronto_merged.loc[toronto_merged['Cluster Labels'] == 6, toronto_merged.columns[1:12]]
cluster_6.head(5)
```

	Neighborhood	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	PostalCode	Borough	Latitude	Longitude
6	Birch Cliff, Cliffside West	Indian Restaurant	Thai Restaurant	Pizza Place	Asian Restaurant	Italian Restaurant	M1N	Scarborough	43.692657	-79.264848
7	Brockton, Parkdale Village, Exhibition Place	Pizza Place	Tibetan Restaurant	Indian Restaurant	Japanese Restaurant	Fast Food Restaurant	M6K	West Toronto	43.636847	-79.428191
8	Business reply mail Processing Centre, South C...	Pizza Place	Fast Food Restaurant	Indian Restaurant	Latin American Restaurant	Sushi Restaurant	M7Y	East Toronto	43.662744	-79.321558
10	Caledonia-Fairbanks	Pizza Place	Caribbean Restaurant	Middle Eastern Restaurant	Latin American Restaurant	Asian Restaurant	M6E	York	43.689026	-79.453512
11	Canada Post Gateway Processing Centre	Chinese Restaurant	Mediterranean Restaurant	Middle Eastern Restaurant	Sushi Restaurant	Indian Restaurant	M7R	Mississauga	43.636966	-79.615819

Following are the results of the Cluster – 6 analysis:

```
for col in required_column:
    print(cluster_6[col].value_counts(ascending = False))
    print("-------------------------------------------")
```

```
Pizza Place                 6
Vietnamese Restaurant       4
Fast Food Restaurant        4
Indian Restaurant           3
Chinese Restaurant          2
Greek Restaurant            2
African Restaurant          1
Latin American Restaurant   1
Burrito Place               1
Italian Restaurant          1
Caribbean Restaurant        1
Asian Restaurant            1
Thai Restaurant             1
Sushi Restaurant            1
Name: 1st Most Common Venue, dtype: int64
-------------------------------------------
Pizza Place                 7
Caribbean Restaurant        6
Fast Food Restaurant        4
Mediterranean Restaurant    2
American Restaurant         1
Latin American Restaurant   1
Chinese Restaurant          1
Korean Restaurant           1
Vietnamese Restaurant       1
Thai Restaurant             1
BBQ Joint                   1
Asian Restaurant            1
Tibetan Restaurant          1
Indian Restaurant           1
Name: 2nd Most Common Venue, dtype: int64
-------------------------------------------
M3L    1
```

Cluster 7

```
cluster_7 = toronto_merged.loc[toronto_merged['Cluster Labels'] == 7, toronto_merged.columns[1:12]]
cluster_7.head(5)
```

	Neighborhood	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	PostalCode	Borough	Latitude	Longitude
5	Berczy Park	Japanese Restaurant	Fast Food Restaurant	Italian Restaurant	Steakhouse	American Restaurant	M5E	Downtown Toronto	43.644771	-79.373306
9	CN Tower, King and Spadina, Railway Lands, Har...	American Restaurant	BBQ Joint	Japanese Restaurant	Tapas Restaurant	Chinese Restaurant	M5V	Downtown Toronto	43.628947	-79.394420
18	Commerce Court, Victoria Hotel	Japanese Restaurant	Pizza Place	Fast Food Restaurant	Brewery	Vegetarian / Vegan Restaurant	M5L	Downtown Toronto	43.648198	-79.379817
29	First Canadian Place, Underground city	Japanese Restaurant	Pizza Place	Thai Restaurant	Fast Food Restaurant	Greek Restaurant	M5X	Downtown Toronto	43.648429	-79.382280
35	Harbourfront East, Union Station, Toronto Islands	Italian Restaurant	Japanese Restaurant	Steakhouse	Pizza Place	Fried Chicken Joint	M5J	Downtown Toronto	43.640816	-79.381752

Following are the results of the Cluster – 7 analysis:

```
for col in required_column:
    print(cluster_7[col].value_counts(ascending = False))
    print("-----------------------------------------------")
```

```
Japanese Restaurant     4
American Restaurant     2
Fast Food Restaurant    2
Italian Restaurant      1
Thai Restaurant         1
Name: 1st Most Common Venue, dtype: int64
-----------------------------------------------
Japanese Restaurant     5
Pizza Place             3
Fast Food Restaurant    1
BBQ Joint               1
Name: 2nd Most Common Venue, dtype: int64
-----------------------------------------------
M5J    1
M5E    1
M5K    1
M5X    1
M5H    1
M4X    1
M5C    1
M5V    1
M5W    1
M5L    1
Name: PostalCode, dtype: int64
-----------------------------------------------
```

Conclusion

On application of Clustering Algorithm, k-Means or others, to a multi-dimensional dataset, a very inquisitive results can be curated which helps to understand and visualize the data. The neighborhoods of Toronto were very briefly segmented into eight clusters and upon analysis it was possible to rename them basis upon the categories of venues in and around that neighborhood. Along with the Canadian cuisine, Italian and Chinese are very dominant in Toronto and so is the diversity statistics. The results of this project can be improved and made more inquisitive by using a current Toronto’s dataset along with API platforms which are more interested in Food Venues (like Yelp, etc.) The scope of this project can be expanded further to understand the dynamics of each neighborhood and suggest a new vendor a profitable location to open his or her food place. Also, a government authority can utilize it to examine and study their city's culture diversity better.