Capstone Project - The Battle of the Neighborhoods (Week 2)

Applied Data Science Capstone by IBM/Coursera

Table of contents:

Introduction: Business problem

In this project we will try to find an optimal location for a restaurant. Specifically, this report will be targeted to stakeholders interested in opening a restaurant in Toronto, Canada.

I choosed the second proposition wich says: In a city of Toronto, if someone is looking to open a restaurant, where would you recommend that they open it? I've choosed this one beacause I think it is a good idea to apply both what we did in weeks 2 and 3. In week 2 we've learned how to use the Foursquare API which allows us to search and explore a given place or venue in order to get all the information we want about it, thus it is good start to know if this location will be beneficial for our client to open his restaurant. Secondly, in week 3 we have created our dataframe from the Wikipedia page about the different neighborhoods with their locations, as a result, we're going to use this information in our foursquare API for exploring neighborhoods.

We will use our data science powers to generate a few most promising neighborhoods based on these criteria. Advantages of each area will then be clearly expressed so that best possible final location can be chosen by stakeholders.

Data:

In week 3, we've worked on data about Toronto city and we've generated a dataframe with the names of neighborhoods and their coordinates. What we're going to do in this project is to use these coordinates to explore the different places in the city to find the best place for a restaurant. To do so, we need to use Foursquare API to generate JSON files that we are going to use later on in our python code. In week 2 we've seen how to use the Foursquare API to retrieve data from the Foursquare database and we used the API to explore a location and to get trending venues around a point of interest. When we were exploring Foursquare. com we were trying to find the popular spots around the Conrad Hotel we used the explore endpoint instead of the search endpoint, and we pass the latitude and the longitude coordinates of the Conrad Hotel along with our credentials, then we make the call to the database, and in return we get a JSON file of the trending venues that are nearby. In the JSON file, for each trending venue, we get mostly its name, unique ID, location, and category.

```
my_df = pd.read_csv('Neighberhoods in Toronto.csv')

my_df.head()
```

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

```
my_df = my_df[['PostalCode','Borough','Neighborhood','Latitude','Longitude']]
my_df.head()
```

	PostalCode	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

visualizing Toronto neighborhoods:

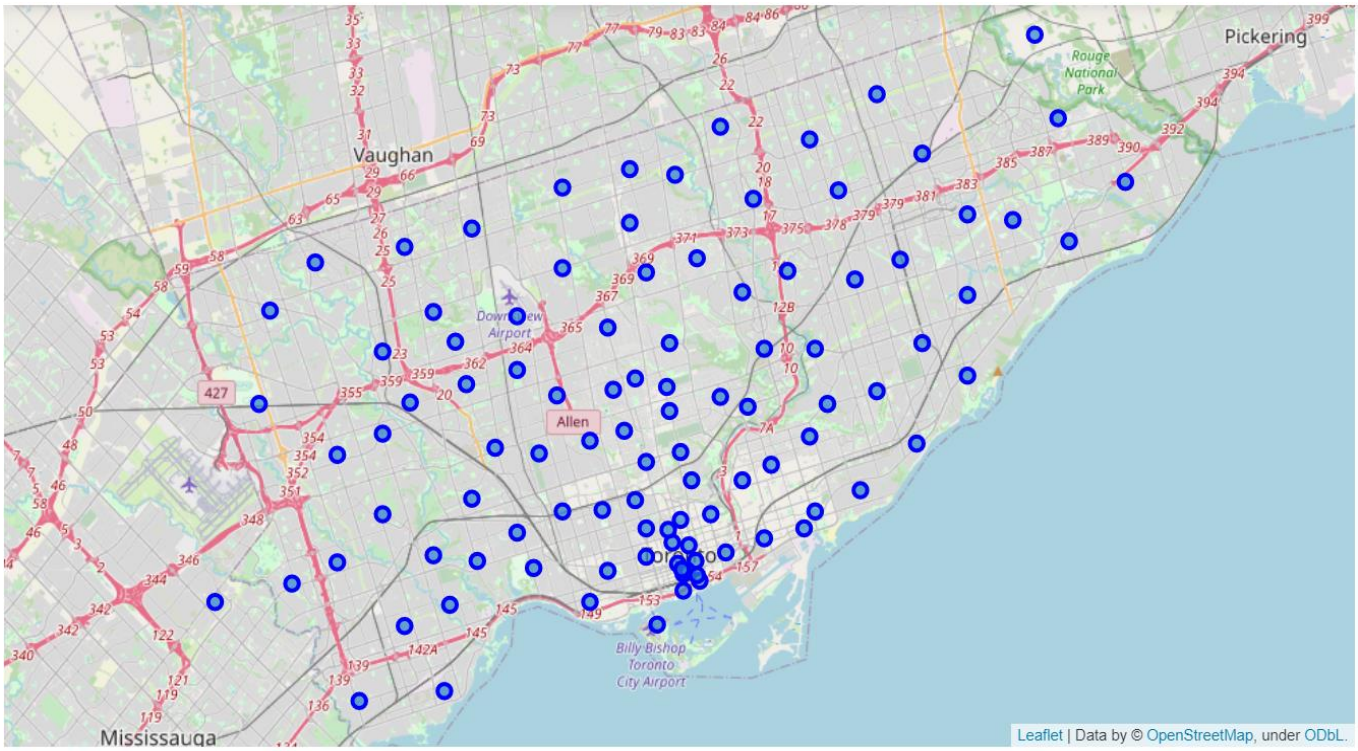
```
: #visualyzing Toronto neighborhoods

  #coordinates of Toronto city
  latitude=43.651070
  longitude=-79.347015

  # create map of Toronto using latitude and longitude values
  map_toronto = folium.Map(location=[latitude, longitude], zoom_start=11)

  # add markers to map
  for lat, lng, label in zip(my_df['Latitude'], my_df['Longitude'], my_df['Neighborhood']):
      label = folium.Popup(label, parse_html=True)
      folium.CircleMarker(
          [lat, lng],
          radius=5,
          popup=label,
          color='blue',
          fill=True,
          fill_color='#3186cc',
          fill_opacity=0.7,
          parse_html=False).add_to(map_toronto)

  map_toronto
```



Foursquare API:

We have to define foursquare credentials and version, then create an URL to generate a JSON file after that we clean the json and structure it into a pandas dataframe. To explore the neighborhoods in Toronto, we have to create a function that repeats the same process to all the neighborhoods.

Entrée [17]:

```
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={}'.form
            CLIENT_ID,
            CLIENT_SECRET,
            VERSION,
            lat,
            lng,
            radius,
            LIMIT)

        # 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']['location']['lat'],
            v['venue']['location']['lng'],
            v['venue']['categories'][0]['name']) for v in results])

    nearby_venues = pd.DataFrame([item for venue_list in venues_list for item in venue_list])
    nearby_venues.columns = ['Neighborhood',
                  'Neighborhood Latitude',
                  'Neighborhood Longitude',
                  'Venue',
                  'Venue Latitude',
                  'Venue Longitude',
                  'Venue Category']

    return(nearby_venues)
```

The resulting dataframe :

Out[20]:

	Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
0	Malvern, Rouge	43.806686	-79.194353	Wendy's	43.807448	-79.199056	Fast Food Restaurant
1	Rouge Hill, Port Union, Highland Creek	43.784535	-79.160497	Chris Effects Painting	43.784343	-79.163742	Construction & Landscaping
2	Rouge Hill, Port Union, Highland Creek	43.784535	-79.160497	Royal Canadian Legion	43.782533	-79.163085	Bar
3	Guildwood, Morningside, West Hill	43.763573	-79.188711	RBC Royal Bank	43.766790	-79.191151	Bank
4	Guildwood, Morningside, West Hill	43.763573	-79.188711	G & G Electronics	43.765309	-79.191537	Electronics Store

Methodology:

First, we have to check how many venues were returned for each neighborhood and find out how many unique categories can be curated from all the returned venues. Then, we analyze each neighborhood. Finally, we cluster the neighborhoods and visualize the resulting clusters.

Analysis:

We check how many venues were returned for each neighborhood:

Entrée [21]:
```
#Let's check how many venues were returned for each neighborhood
```

Entrée [22]:
```
toronto_venues.groupby('Neighborhood').count()
```

Out[22]:

Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
Agincourt	4	4	4	4	4	4
Alderwood, Long Branch	8	8	8	8	8	8
Bathurst Manor, Wilson Heights, Downsview North	22	22	22	22	22	22
Bayview Village	4	4	4	4	4	4
Bedford Park, Lawrence Manor East	21	21	21	21	21	21
...	...	...	...	...	...	...
Willowdale, Willowdale West	6	6	6	6	6	6
Woburn	4	4	4	4	4	4
Woodbine Heights	5	5	5	5	5	5
York Mills West	2	2	2	2	2	2
York Mills, Silver Hills	1	1	1	1	1	1

We group rows by neighborhood and by taking the mean of the frequency of occurrence of each category:

Entrée [28]:
```
#Let's group rows by neighborhood and by taking the mean of the frequency of occurrence of each category
```

Entrée [29]:
```
toronto_grouped = toronto_onehot.groupby('Neighborhood').mean().reset_index()
toronto_grouped
```

Out[29]:

	Neighborhood	Yoga Studio	Accessories Store	Afghan Restaurant	Airport	Airport Food Court	Airport Lounge	Airport Service	Airport Terminal	American Restaurant	...	Toy / Game Store	Trail	Train Station	Vegetarian / Vegan Restaurant	Video Game Store	Viet Res
0	Agincourt	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.000000	...	0.0	0.0	0.0	0.0	0.0	
1	Alderwood, Long Branch	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.000000	...	0.0	0.0	0.0	0.0	0.0	
2	Bathurst Manor, Wilson Heights, Downsview North	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.000000	...	0.0	0.0	0.0	0.0	0.0	
3	Bayview Village	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.000000	...	0.0	0.0	0.0	0.0	0.0	
4	Bedford Park, Lawrence Manor East	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.047619	...	0.0	0.0	0.0	0.0	0.0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...

We create the new dataframe and display the top 10 venues for each neighborhood:

:
```
#Now let's create the new dataframe and display the top 10 venues for each neighborhood.
```

:
```
import numpy as np

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'] = toronto_grouped['Neighborhood']

for ind in np.arange(toronto_grouped.shape[0]):
    neighborhoods_venues_sorted.iloc[ind, 1:] = return_most_common_venues(toronto_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
0	Agincourt	Latin American Restaurant	Lounge	Breakfast Spot	Skating Rink	Electronics Store	Eastern European Restaurant	Dumpling Restaurant	Drugstore	Escape Room	Donut Shop
1	Alderwood, Long Branch	Pizza Place	Gym	Coffee Shop	Pub	Skating Rink	Athletics & Sports	Sandwich Place	Diner	Department Store	Dessert Shop
2	Bathurst Manor, Wilson Heights, Downsview North	Coffee Shop	Bank	Sushi Restaurant	Pizza Place	Shopping Mall	Diner	Bridal Shop	Restaurant	Deli / Bodega	Intersection
3	Bayview Village	Café	Bank	Japanese Restaurant	Chinese Restaurant	Women's Store	Dessert Shop	Diner	Discount Store	Distribution Center	Dog Run
4	Bedford Park, Lawrence Manor East	Sandwich Place	Coffee Shop	Italian Restaurant	Thai Restaurant	Liquor Store	Juice Bar	Restaurant	Indian Restaurant	Pub	Butcher

Finally, we cluster the neighborhoods:

Entrée [38]:
```
#Cluster Neighborhoods
```

Entrée [39]:
```
from sklearn.cluster import KMeans

# set number of clusters
kclusters = 5

toronto_grouped_clustering = toronto_grouped.drop('Neighborhood', 1)

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

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

Out[39]:
```
array([1, 1, 1, 1, 1, 1, 1, 1, 1, 1])
```

Results and discussion:

Our analysis showed us that the label restaurant does not appear always in the column of the first venue name in each neighborhood although there are a lot of restaurants. We can deduce that we can find a lot of neighborhoods in Toronto and from the 5 tables of our clustering we can deduce that the neighborhood which will be the best for a restaurant will be “Scarborough”.

Conclusion:

Purpose of this project was to identify Toronto areas in order to aid stakeholders in narrowing down the search for optimal location for a new restaurant. First, we check how many venues were returned for each neighborhood then we clustered the neighborhoods and visualize the resulting clusters.