Applied Data Science Capstone project report (IBM, Coursera)

The Battle of Neighbourhoods

City of Toronto

Navpreet Sharma

Contents

1. Introduction

There are various ways of expanding a business, and companies can choose different growth strategies to reach their goal. But one the most common strategy is targeting new customers in new locations for the existing products. The following project takes into consideration the neighbourhoods of a city, and finds the similarity between them in order to develop insights for businesses to choose the right new market for their product.

1.1. Business Problem

To find a list of most apt neighbourhoods in the city of Toronto to open a new branch of a successful Chinese restaurant in Maple Leaf. This would be done by finding neighbourhoods that are most similar to Maple Leaf with respect to the venues distribution across these neighbourhoods and then using the demographic and income data of each neighbourhood to get the top ten list.

2. Data

In order to the analysis on the city of Toronto, the following data is needed:

- **Geospatial data** – this consists of neighbourhood names and their geographical coordinates.

  Source - https://open.toronto.ca/dataset/neighbourhoods/

- **Venues data** – this consists of a list of all the venues, such as parks and restaurants, present in each neighbourhood and information related to these venues, such as category, location, reviews and tips. This data will be sourced from the Foursquare API by making a query to it (using 'explore' endpoint on each neighbourhood) with our credentials. This data is in the form of a JSON object and the 'venue category' part of each venue is used to build the data for analysis.

- **Demographic data** - this consists of total population and the population of people with Chinese origin in each neighbourhood.

  Source - https://open.toronto.ca/dataset/wellbeing-toronto-demographics/

- **Income data** - this consists of median household income of each neighbourhood.

  Source - https://open.toronto.ca/dataset/wellbeing-toronto-demographics-nhs-indicators/

3. Methodology

3.1. Data pre-processing

In this stage, data is loaded from respective sources into data frames and prepared in the required format for analysis and use in machine learning process.

Using the folium library, the following map of Toronto city is visualised with neighbourhoods superimposed on it.

Out[5]:

Figure 1 - Map of Toronto city superimposed with neighbourhoods

By combining the geospatial, demographic and income data, the following data frame is generated.

In [3]:

```
print(toronto_data.shape)
toronto_data.head()
```

```
(140, 6)
```

Out[3]:

	Neighbourhood	Total Population	Chinese Origin	Median household income($)	Latitude	Longitude
0	West Humber-Clairville	34100	695	66241	43.716180	-79.596356
1	Mount Olive-Silverstone-Jamestown	32790	200	49934	43.746868	-79.587259
2	Thistletown-Beaumond Heights	10140	105	62042	43.737988	-79.563491
3	Rexdale-Kipling	10485	200	56545	43.723725	-79.566228
4	Elms-Old Rexdale	9550	120	50846	43.721519	-79.548983

Figure 2 – Toronto neighbourhoods data

By using a custom function and calling the "explore" endpoint to the Foursquare API, a dataset is created with top 100 venues within 500 meters of the centre of each neighbourhood.

```
In [10]: print(toronto_venues.shape)
         toronto_venues.head()
```

(2050, 7)

Out[10]:

	Neighbourhood	Neighbourhood Latitude	Neighbourhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
0	West Humber-Clairville	43.71618	-79.596356	Tim Hortons	43.714657	-79.593716	Coffee Shop
1	West Humber-Clairville	43.71618	-79.596356	Mandarin Buffet	43.720360	-79.594387	Chinese Restaurant
2	West Humber-Clairville	43.71618	-79.596356	Xawaash	43.715786	-79.593053	Mediterranean Restaurant
3	West Humber-Clairville	43.71618	-79.596356	Staples Rexdale	43.718539	-79.594570	Paper / Office Supplies Store
4	West Humber-Clairville	43.71618	-79.596356	Winners	43.719819	-79.594923	Department Store

Figure 3 – Toronto venues data

One-hot encoding is performed on the "Venue Category" column to create a new dataset as shown below.

```
In [13]: toronto_onehot = pd.get_dummies(toronto_venues[['Venue Category']], prefix="", prefix_sep="")

         # add neighborhood column back to dataframe
         toronto_onehot['Neighbourhood'] = toronto_venues['Neighbourhood']

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

Out[13]:

	Neighbourhood	African Restaurant	Airport Service	American Restaurant	Amphitheater	Animal Shelter	Antique Shop	Arcade	Argentinian Restaurant	Art Gallery	Arts & Crafts Store	Asian Restaurant	Athletics & Sports	De
0	West Humber-Clairville	0	0	0	0	0	0	0	0	0	0	0	0	0
1	West Humber-Clairville	0	0	0	0	0	0	0	0	0	0	0	0	0
2	West Humber-Clairville	0	0	0	0	0	0	0	0	0	0	0	0	0
3	West Humber-Clairville	0	0	0	0	0	0	0	0	0	0	0	0	0
4	West Humber-Clairville	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 4 – Toronto venues one-hot data frame

3.2. Exploratory Data Analysis

Venues dataset is evaluated to check the total number of venues and the number of venues categories.

```
In [14]: toronto_onehot.shape
```

Out[14]: (2050, 279)

Figure 5 - Total number of venues and venue categories

Next, rows are grouped by neighbourhood and the mean of the frequency of occurrence of each category is taken, to see the number of neighbourhoods returned by the Foursquare API with a venue.

In [16]:
```
toronto_grouped = toronto_onehot.groupby('Neighbourhood').mean().reset_index()
print(toronto_grouped.shape)
toronto_grouped
```

(138, 279)

Out[16]:

	Neighbourhood	African Restaurant	Airport Service	American Restaurant	Amphitheater	Animal Shelter	Antique Shop	Arcade	Argentinian Restaurant	Art Gallery	Arts & Crafts Store	Asian Restaurant
0	Agincourt North	0.0	0.0	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
1	Agincourt South-Malvern West	0.0	0.0	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.052632
2	Alderwood	0.0	0.0	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
3	Annex	0.0	0.0	0.038462	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
4	Banbury-Don Mills	0.0	0.0	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000

Figure 6 – Toronto venues data grouped by neighbourhood

Then a new data frame is created that displays top 10 venues for each neighbourhood.

Out[17]:

	Neighbourhood	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue	7th Most Common Venue	8th Most Common Venue	9th Most Common Venue	10th Most Common Venue
0	Agincourt North	Chinese Restaurant	Clothing Store	Fast Food Restaurant	Pizza Place	Fried Chicken Joint	Frozen Yogurt Shop	Liquor Store	Beer Store	Sandwich Place	Bank
1	Agincourt South-Malvern West	Chinese Restaurant	Pizza Place	BBQ Joint	Cantonese Restaurant	Mediterranean Restaurant	Filipino Restaurant	Pool Hall	Bank	Restaurant	Café
2	Alderwood	Pizza Place	Convenience Store	Pharmacy	Coffee Shop	Farm	Eastern European Restaurant	Egyptian Restaurant	Electronics Store	Ethiopian Restaurant	Event Space
3	Annex	Café	Sandwich Place	Pub	Coffee Shop	Convenience Store	Middle Eastern Restaurant	Metro Station	Pharmacy	Liquor Store	French Restaurant
4	Banbury-Don Mills	Coffee Shop	Women's Store	Pizza Place	Cantonese Restaurant	Kids Store	Sandwich Place	Liquor Store	Chocolate Shop	Bank	Italian Restaurant

Figure 7 - Top 10 venues

3.3. Clustering

k-means clustering method is used to build clusters with similar neighbourhoods based on the venues data received from calling the Foursquare API. Optimum value for k is found out by using the Elbow method and the Silhouette method.

Elbow method reveals that optimum value for k is 4.

```
In [18]: toronto_clustering = toronto_grouped.drop('Neighbourhood', 1)

         sse = []
         for i in range(1, 11):
             km = KMeans(
                 n_clusters=i, init='random',
                 n_init=10, max_iter=300,
                 tol=1e-04, random_state=0
             )
             km.fit(toronto_clustering)
             sse.append(km.inertia_)

         # plot
         plt.plot(range(1, 11), sse, marker='o')
         plt.xlabel('Number of clusters')
         plt.ylabel('SSE')
         plt.show()
```

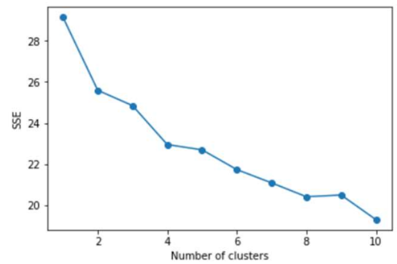

Figure 8 - Elbow method

Silhouette method is used in combination with Elbow method to validate the value of k.

```
In [19]: sil = []
         kmax = 10

         # dissimilarity would not be defined for a single cluster, thus, minimum number of clusters should be 2
         for k in range(2, kmax+1):
             kmeans = KMeans(n_clusters = k).fit(toronto_clustering)
             labels = kmeans.labels_
             sil.append(silhouette_score(toronto_clustering, labels, metric = 'euclidean'))


         # plot
         plt.plot(range(2, 11), sil, marker='o')
         plt.xlabel('Number of clusters')
         plt.ylabel('Silhouette score')
         plt.show()
```

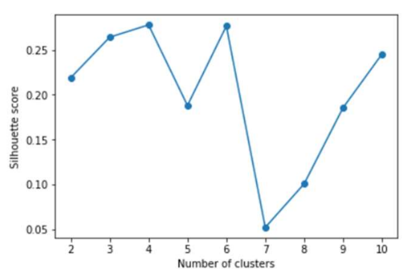

Figure 9 - Silhouette method

Clustering of the neighbourhoods is done using the value of 4 for k in k-mead clustering method.

```
In [21]: # set number of clusters
         kclusters = 4

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

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

Out[21]: array([0, 0, 0, 0, 0, 0, 0, 0, 1, 0], dtype=int32)
```

Figure 10 – Clustering

The following data frame is obtained by merging the Toronto data frame with sorted venues data frame containing cluster labels.

```
In [22]: # add clustering labels
         neighbourhoods_venues_sorted.insert(0, 'Cluster Labels', kmeans.labels_)

         toronto_merged = toronto_data

         # merge toronto_grouped with toronto_data to add latitude/longitude for each neighborhood
         toronto_merged = toronto_merged.join(neighbourhoods_venues_sorted.set_index('Neighbourhood'), on='Neighbourhood')

         # check the last columns!
         toronto_merged

Out[22]:
```

	Neighbourhood	Total Population	Chinese Origin	Median household income($)	Latitude	Longitude	Cluster Labels	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	
0	West Humber-Clairville	34100	695	66241	43.716180	-79.596356	0.0	Department Store	Hotel	Bank	Chinese Restaurant	P S S
1	Mount Olive-Silverstone-Jamestown	32790	200	49934	43.746868	-79.587259	1.0	Japanese Restaurant	Park	Coffee Shop	Zoo Exhibit	F R
2	Thistletown-Beaumond Heights	10140	105	62042	43.737988	-79.563491	0.0	Indian Restaurant	Caribbean Restaurant	Bank	Spa	T R
3	Rexdale-Kipling	10485	200	56545	43.723725	-79.566228	0.0	Flower Shop	Jewelry Store	Donut Shop	Eastern European Restaurant	E R
4	Elms-Old Rexdale	9550	120	50846	43.721519	-79.548983	0.0	African Restaurant	Pool	Business Service	Mobile Phone Shop	C & L

Figure 11 – Toronto neighbourhood and sorted venue data merged

Target cluster, the cluster containing the neighbourhood Maple Leaf, is identified

```
In [29]: target_cluster_df = toronto_merged.loc[toronto_merged['Neighbourhood']=='Maple Leaf']
         target_cluster_df.reset_index(inplace=True)
         target_cluster=target_cluster_df.loc[0].at['Cluster Labels']
         print('The target cluster is: {} '.format(target_cluster + 1))

         The target cluster is: 1
```

Figure 12 - Target cluster

Then normalised demographic and income data is added to the target cluster data and used to get the top 10 list of neighbourhoods in the target cluster by giving different weightage to these elements.

- Total population: 50%
- Population with Chinese origin: 30%
- Median household income: 20%

```
In [33]: possible_neighbourhoods = Cluster1.merge(df_norm[['Neighbourhood', 'Population Normalised', 'Chinese Normalised', 'Income Normal
ised']], on='Neighbourhood')
possible_neighbourhoods['Ranking'] = possible_neighbourhoods['Population Normalised'] * 0.5 + possible_neighbourhoods['Chinese N
ormalised'] * 0.3 + possible_neighbourhoods['Income Normalised'] * 0.2

recommended_neighbourhoods = possible_neighbourhoods.sort_values(by = 'Ranking', ascending = False).head(10)
recommended_neighbourhoods.reset_index(inplace = True, drop = True)
```

Figure 13 - Recommended neighbourhoods code

4. Results

The Maple Leaf neighbourhood was found in Cluster 1 containing 116 other similar neighbourhoods. But the following more narrowed down list was obtained by combining the demographic and income data, with different weightage to different elements, to the resulting cluster.

```
In [34]: recommended_neighbourhoods
```

Out[34]:

	Neighbourhood	Cluster Labels	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue	7th Most Common Venue	8th Most Common Venue	9th Most Common Venue	10th Mos Common Venu
0	L'Amoreaux	0	Chinese Restaurant	Pizza Place	Bank	Gym Pool	Sandwich Place	Thrift / Vintage Store	Bakery	Nail Salon	Camera Store	Coffee Shop
1	Willowdale East	0	Hotel	Dumpling Restaurant	Flea Market	Fish Market	Fish & Chips Shop	Filipino Restaurant	Field	Fast Food Restaurant	Food & Drink Shop	Farm
2	Woburn	0	Indian Restaurant	Bakery	American Restaurant	Farm	Eastern European Restaurant	Egyptian Restaurant	Electronics Store	Ethiopian Restaurant	Event Space	Falafel Restauran
3	Agincourt North	0	Chinese Restaurant	Clothing Store	Fast Food Restaurant	Pizza Place	Fried Chicken Joint	Frozen Yogurt Shop	Liquor Store	Beer Store	Sandwich Place	Bank
4	Steeles	0	Chinese Restaurant	Sushi Restaurant	Pizza Place	BBQ Joint	Shopping Mall	Vietnamese Restaurant	Korean Restaurant	Supermarket	Fast Food Restaurant	Pharmacy
5	Waterfront Communities-The Island	0	Boat or Ferry	Zoo Exhibit	Farmers Market	Egyptian Restaurant	Electronics Store	Ethiopian Restaurant	Event Space	Falafel Restaurant	Farm	Fast Food Restauran
6	Rouge	0	Zoo Exhibit	Zoo	Tram Station	Dessert Shop	Restaurant	Other Great Outdoors	Fast Food Restaurant	Gift Shop	Food & Drink Shop	Hardware Store
7	Malvern	0	Fast Food Restaurant	Sandwich Place	Bubble Tea Shop	Pizza Place	Pharmacy	Gym / Fitness Center	Event Space	Dumpling Restaurant	Eastern European Restaurant	Egyptian Restauran
8	Tam O'Shanter-Sullivan	0	Pizza Place	Pharmacy	Italian Restaurant	Rental Car Location	Chinese Restaurant	Noodle House	Bus Stop	Fast Food Restaurant	Convenience Store	Fried Chicken Joint
9	Islington-City Centre West	0	Sandwich Place	Fast Food Restaurant	Pizza Place	Turkish Restaurant	Café	Fried Chicken Joint	Garden Center	Thai Restaurant	Bank	Rental Cai Location

Figure 14 – Top 10 neighbourhoods

5. Discussion

The clustering of neighbourhoods using k-means based on venue data returned just 2 main clusters. The target cluster contained the maximum number of neighbourhoods, 117 of the total 140, making it difficult to choose a neighbourhood right away.

```
In [25]: Cluster1 = toronto_merged.loc[toronto_merged['Cluster Labels'] == 0, toronto_merged.columns[[0] + list(range(6, toronto_merged.s
hape[1]))]]
print(Cluster1.shape)
Cluster1

(117, 12)
Out[25]:
```

	Neighbourhood	Cluster Labels	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue	5th Most Common Venue	6th Most Common Venue	7th Most Common Venue	8th Most Common Venue	
0	West Humber-Clairville	0	Department Store	Hotel	Bank	Chinese Restaurant	Paper / Office Supplies Store	Swiss Restaurant	Coffee Shop	Gym / Fitness Center	Me Re
2	Thistletown-Beaumond Heights	0	Indian Restaurant	Caribbean Restaurant	Bank	Spa	Thai Restaurant	Supermarket	Asian Restaurant	Dance Studio	Co
3	Rexdale-Kipling	0	Flower Shop	Jewelry Store	Donut Shop	Eastern European Restaurant	Egyptian Restaurant	Electronics Store	Ethiopian Restaurant	Event Space	Fa Re
4	Elms-Old Rexdale	0	African Restaurant	Pool	Business Service	Mobile Phone Shop	Construction & Landscaping	History Museum	Hockey Arena	Filipino Restaurant	Fie

Figure 15 - Cluster 1

But the demographic and income data of the neighbourhoods proved to be very useful, and essential, to get a more reasonable list of recommended neighbourhoods for opening a Chinese restaurant.

Hence, it is recommended to try other clustering algorithms which generate more clusters with fewer neighbourhoods, if possible, and use more relevant data, such as reviews and tips from the Foursquare API for venues in order to understand the preferences of the customers, to get an even more precise outcome.

6. Conclusion

A reasonably good list of 10 neighbourhoods has been found to open a Chinese restaurant. To select one from this list, other criteria of choosing a location for a restaurant, such as available facilities/buildings in the neighbourhood, lease and rent terms, etc. can be used.

K-means clustering and Foursquare API are helpful tools to get a head start in such situations as discussed above, and saves a lot of time and resources for an organisation. It also becomes more visible in a clearer way by using visualisation tools, that how and why the choices made are relevant to the project.