Capstone Project - The Battle of Neighbourhoods (Week 2)

By Phiraphon Phankoson

Introduction

The travel agency would like to open a travel package to bring tourist into Toronto City so the travel agency have to understand in this city for creating travel package and finding area where able to be response to tourist by that area should have restaurants, cafe, museum and shopping mall.

For traveler that want to visit, travel or person who have to be in Toronto.

Data

To consider the objective, I listed the below data sources used for the analysis.

a) Toronto Neighborhood Data: The following Wikipedia page was scraped to pull out the necessary information: https://en.wikipedia.org/wiki/List_of_postal_codes_of_Canada:_M

The information obtained i.e. the table of postal codes was transformed into a pandas data frame for further analysis.

b) Coordinate data for each Neighborhood in Toronto: The following csv file gave us the geographical coordinates of each postal code: http://cocl.us/Geospatial_data

Methodology

1. Load Toronto Neighborhood Data and transformed into a pandas data frame for further analysis.

```
import pandas as pd
import numpy as np
import requests
from bs4 import BeautifulSoup
```

Load Toronto Neighborhood Data

```
url = "https://en.wikipedia.org/wiki/List_of_postal_codes_of_Canada:_M"
source = requests.get(url).text
Canada_data = BeautifulSoup(source, 'lxml')
```

Transform to pandas data frame

```
column_names = ['Postalcode','Borough','Neighborhood']
toronto = pd.DataFrame(columns = column_names)
```

```
content = Canada_data.find('div', class_='mw-parser-output')
table = content.table.tbody
postcode = 0
borough = 0
neighborhood = 0

for tr in table.find_all('tr'):
    i = 0
    for td in tr.find_all('td'):
        if i == 0:
            postcode = td.text
            i = i + 1
        elif i == 1:
            borough = td.text
            i = i + 1
        elif i == 2:
            neighborhood = td.text.strip('\n').replace(']','')
    toronto = toronto.append({'Postalcode': postcode,'Borough': borough,'Neighborhoo

toronto = toronto[toronto.Borough!='Not assigned']
toronto = toronto[toronto.Borough!= 0]
toronto.reset_index(drop = True, inplace = True)
i = 0
for i in range(0,toronto.shape[0]):
    if toronto.iloc[i][2] == 'Not assigned':
        toronto.iloc[i][2] = toronto.iloc[i][1]
        i = i+1

df = toronto.groupby(['Postalcode','Borough'])['Neighborhood'].apply(', '.join).rese
df.head()
```

	Postalcode	Borough	Neighborhood
0	M1B	Scarborough	Rouge, Malvern
1	M1C	Scarborough	Highland Creek, Rouge Hill, Port Union
2	M1E	Scarborough	Guildwood, Morningside, West Hill
3	M1G	Scarborough	Woburn
4	M1H	Scarborough	Cedarbrae

Methodology

2. Load Coordinate data for each Neighborhood in Toronto and put them to data frame.

```
file = 'http://cocl.us/Geospatial_data'
df_lo = pd.read_csv(file)
df_lo.head()
```

	Postal Code	Latitude	Longitude
0	M1B	43.806686	-79.194353
1	M1C	43.784535	-79.160497
2	M1E	43.763573	-79.188711
3	M1G	43.770992	-79.216917
4	M1H	43.773136	-79.239476

3. Merge 2 data frame together

```
df.rename(columns={'Postalcode':'Postal Code'}, inplace=True)
df_lo.set_index("Postal Code")
df.set_index("Postal Code")
toronto_data=pd.merge(df,df_lo)
toronto_data.head()
```

	Postal Code	Borough	Neighborhood	Latitude	Longitude
0	M1B	Scarborough	Rouge, Malvern	43.806686	-79.194353
1	M1C	Scarborough	Highland Creek, Rouge Hill, Port Union	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

Methodology

4. Explore and cluster the neighborhoods in Toronto

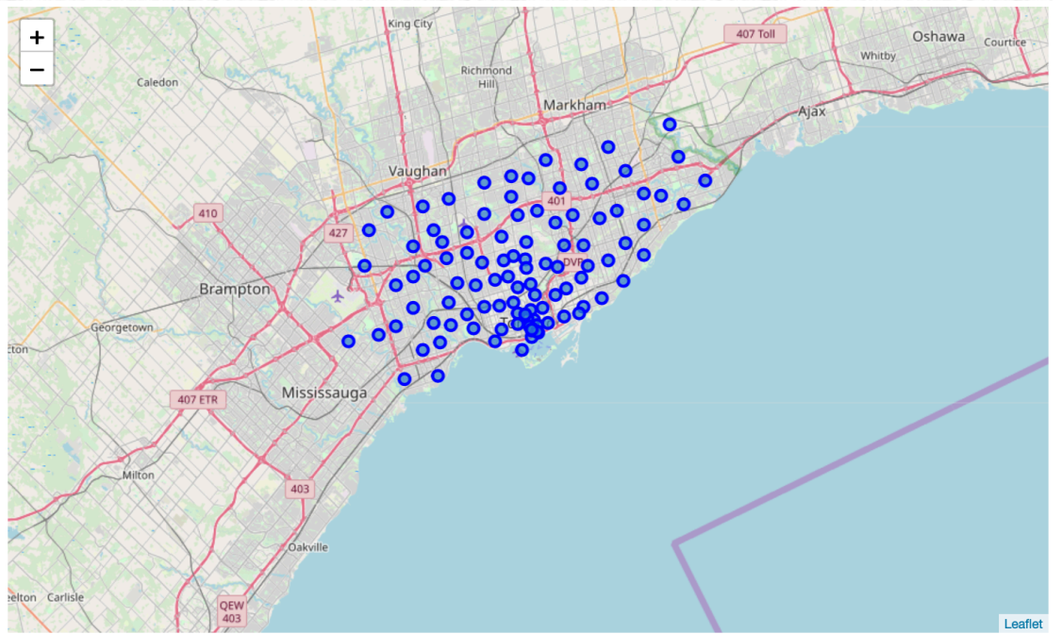
```
from geopy.geocoders import Nominatim
import matplotlib.cm as cm
import matplotlib.colors as colors
from sklearn.cluster import KMeans

!conda install -c conda-forge folium=0.5.0 --yes
import folium
```

```
Collecting package metadata (current_repodata.json): done
Solving environment: done

# All requested packages already installed.
```

```
address = 'Toronto, ON, Canada'

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

```
The geograpical coordinate of Toronto, ON, Canada are 43.653963, -79.387207.
```

```
# create map of Toronto using latitude and longitude values
map_toronto = folium.Map(location=[latitude, longitude], zoom_start=10)

# add markers to map
for lat, lng, borough, neighborhood in zip(toronto_data['Latitude'], toronto_data['Longitude'], toronto_dat
    label = '{}, {}'.format(neighborhood, borough)
    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
```



Methodology

5. Define Foursquare.

```
# defining radius and limit of venues to get
radius=500
LIMIT=100
```

```
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_se
            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 ve
    nearby_venues.columns = ['Neighborhood',
                  'Neighborhood Latitude',
                  'Neighborhood Longitude',
                  'Venue',
                  'Venue Latitude',
                  'Venue Longitude',
                  'Venue Category']

    return(nearby_venues)

```

```
toronto_venues = getNearbyVenues(names=toronto_data['Neighborhood'],
                                 latitudes=toronto_data['Latitude'],
                                 longitudes=toronto_data['Longitude']
                                )
```

Let's check the size of the resulting dataframe

```
print(toronto_venues.shape)
toronto_venues.head()
```

```
(2220, 7)
```

	Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
0	Rouge, Malvern	43.806686	-79.194353	Wendy's	43.807448	-79.199056	Fast Food Restaurant
1	Highland Creek, Rouge Hill, Port Union	43.784535	-79.160497	Royal Canadian Legion	43.782533	-79.163085	Bar
2	Highland Creek, Rouge Hill, Port Union	43.784535	-79.160497	Scarborough Historical Society	43.788755	-79.162438	History Museum
3	Guildwood, Morningside, West Hill	43.763573	-79.188711	Swiss Chalet Rotisserie & Grill	43.767697	-79.189914	Pizza Place
4	Guildwood, Morningside, West Hill	43.763573	-79.188711	G & G Electronics	43.765309	-79.191537	Electronics Store

Methodology

6. Analysing Each Neighbourhood

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

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

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

	Accessories Store	Afghan Restaurant	Airport	Airport Food Court	Airport Lounge	Airport Service	Airport Terminal	American Restaurant	Antique Shop	Aquariu
0	0	0	0	0	0	0	0	0	0	
1	0	0	0	0	0	0	0	0	0	
2	0	0	0	0	0	0	0	0	0	
3	0	0	0	0	0	0	0	0	0	
4	0	0	0	0	0	0	0	0	0	

And let's examine the new dataframe size.

```
toronto_onehot.shape
```

```
(2220, 268)
```

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

	Neighborhood	Accessories Store	Afghan Restaurant	Airport	Airport Food Court	Airport Lounge	Airport Service	Airport Terminal	American Restaurant	A
0	Adelaide, King, Richmond	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.02	
1	Agincourt	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.00	
2	Agincourt North, L'Amoreaux East, Milliken, St...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.00	
3	Albion Gardens, Beaumond Heights, Humbergate, ...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.00	
4	Alderwood, Long Branch	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.00	

Let's print each neighborhood along with the top 5 most common venues

```
num_top_venues = 5

for hood in toronto_grouped['Neighborhood']:
    print("----"+hood+"----")
    temp = toronto_grouped[toronto_grouped['Neighborhood'] == hood].T.reset_index()
    temp.columns = ['venue','freq']
    temp = temp.iloc[1:]
    temp['freq'] = temp['freq'].astype(float)
    temp = temp.round({'freq': 2})
    print(temp.sort_values('freq', ascending=False).reset_index(drop=True).head(num_
    print('\n')
```

```
----Adelaide, King, Richmond----
            venue  freq
0      Coffee Shop  0.06
1             Bar  0.04
2            Café  0.04
3      Steakhouse  0.04
4  Thai Restaurant  0.04
```

Methodology

7. Take result to data frame.

```python
def return_most_common_venues(row, num_top_venues):
    row_categories = row.iloc[1:]
    row_categories_sorted = row_categories.sort_values(ascending=False)

    return row_categories_sorted.index.values[0:num_top_venues]
```

```python
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_g

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	8 C
0	Adelaide, King, Richmond	Coffee Shop	Bar	Café	Thai Restaurant	Steakhouse	Restaurant	Bakery	Re
1	Agincourt	Clothing Store	Latin American Restaurant	Skating Rink	Lounge	Breakfast Spot	Falafel Restaurant	Event Space	
2	Agincourt North, L'Amoreaux East, Milliken, St...	Playground	Park	Yoga Studio	Dumpling Restaurant	Discount Store	Dog Run	Doner Restaurant	
3	Albion Gardens, Beaumond Heights, Humbergate, ...	Grocery Store	Pizza Place	Fried Chicken Joint	Coffee Shop	Beer Store	Pharmacy	Fast Food Restaurant	S
4	Alderwood, Long Branch	Pizza Place	Pub	Pharmacy	Gym	Sandwich Place	Coffee Shop	Skating Rink	

Methodology

8. Answer the question of this project.

Result

The travel agency can create the travel package from catogorie of 1st Most Common Venue

```
1 toronto_merged['1st Most Common Venue'].unique()
```

```
array(['Fast Food Restaurant', 'History Museum', 'Spa', 'Coffee Shop',
       'Caribbean Restaurant', 'Playground', 'Discount Store', 'Bakery',
       'American Restaurant', 'College Stadium', 'Indian Restaurant',
       'Clothing Store', 'Pizza Place', 'Dog Run', 'Café', 'Cafeteria',
       'Piano Bar', 'Ramen Restaurant', 'Park', 'Gym',
       'Miscellaneous Shop', 'Airport', 'Grocery Store', 'Baseball Field',
       'Liquor Store', 'Pub', 'Sporting Goods Shop', 'Greek Restaurant',
       'Pet Store', 'Dessert Shop', 'Summer Camp', 'Sandwich Place',
       'Garden', 'Mexican Restaurant', 'Bar', 'Airport Lounge',
       'Metro Station', 'Tennis Court', 'Gift Shop', 'Light Rail Station',
       'Furniture / Home Store', 'Rental Car Location'], dtype=object)
```

I assume that the travel package is travel Museum

```
1 toronto_merged.loc[toronto_merged['1st Most Common Venue'] == 'History Museum']
```

	Postal Code	Borough	Neighborhood	Latitude	Longitude	Cluster Labels	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue
1	M1C	Scarborough	Highland Creek, Rouge Hill, Port Union	43.784535	-79.160497	0	History Museum	Bar	Yoga Studio

Result

If travel package is travel Museum, the travel agency bring tourist to Scarborough and walk around the Highland Creek, Rouge Hill and Port Union.

This project show about if someone would like to use data to creating the travel package, you can do that!

Conclusion

Have to set travel package first and then will able to plan path for tourist

The accuracy of data depends purely depends on the data provided by FourSquare

Thank you.