This notebook will be mainly used for the capstone project.

Table of Contents

Part Ⅰ

In [1]:
```
import pandas as pd
import numpy as np
```

In [2]:
```
# creat an empty excel file, then copy and paste the table from the wiki.
a=pd.read_excel('raw_data.xlsx')
a.head(5)
```

Out[2]:

	Postcode	Borough	Neighbourhood
0	M5R	Central Toronto	Yorkville
1	M3J	North York	York University
2	M2P	North York	York Mills West
3	M2L	North York	York Mills
4	M4C	East York	Woodbine Heights

In [3]:
```
# drop the cells with 'Not assigned.'
b=a[a['Borough']!='Not assigned']
c=b[b['Neighbourhood']!='Not assigned']
c.shape
```

Out[3]: (209, 3)

In [4]:
```
# get those unique Postcodes and the according Borough
d=c.groupby('Postcode').first()
d.reset_index(inplace=True)
d.head()
```

Out[4]:

	Postcode	Borough	Neighbourhood
0	M1B	Scarborough	Rouge
1	M1C	Scarborough	Rouge Hill
2	M1E	Scarborough	West Hill
3	M1G	Scarborough	Woburn
4	M1H	Scarborough	Cedarbrae

In [5]:
```
m=a.shape[0]
n=d.shape[0]
```

In [6]:
```
#apply loop to find matched 'neighbourhood' from raw_data
for i in range(n):
    temp=[] # to collect each 'neighbourhood'
    temp1='' # to transfer temp into string
    for j in range(m):
        if a.iloc[j,0]==d.iloc[i,0] and a.iloc[j,1]==d.iloc[i,1]:
            temp.append(a.iloc[j,2])
    temp=pd.Series(temp)
    temp=temp.unique()
    for w in temp:
        temp1=temp1+w+','
    temp1=temp1.strip(',') # drop the last comma
    d.iloc[i,2]=temp1
```

In [7]:
```
d.head(7)
```

Out[7]:

	Postcode	Borough	Neighbourhood
0	M1B	Scarborough	Rouge,Malvern
1	M1C	Scarborough	Rouge Hill,Port Union,Highland Creek
2	M1E	Scarborough	West Hill,Morningside,Guildwood
3	M1G	Scarborough	Woburn
4	M1H	Scarborough	Cedarbrae
5	M1J	Scarborough	Scarborough Village
6	M1K	Scarborough	Kennedy Park,Ionview,East Birchmount Park

In [20]:
```
#number of rows
d.shape[0]
```

Out[20]: 102

Part II

In [9]:
```
geo=pd.read_csv('Geospatial_Coordinates.csv')
```

In [10]:
```
geo.head()
```

Out[10]:

	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

In [11]:
```
f=pd.merge(d,geo,left_on="Postcode",right_on="Postal Code")
f.head()
```

Out[11]:

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

In [12]:
```
f.drop(['Postal Code'],axis=1,inplace=True)
f.head()
```

Out[12]:

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

In [21]:
```
f.shape[0]
```

Out[21]: 102

Part Ⅲ

In [13]:
```
pd.set_option('display.max_columns', None)
pd.set_option('display.max_rows', None)
import json # library to handle JSON files
from geopy.geocoders import Nominatim # convert an address into latitude and longitude values
import requests # library to handle requests
from pandas.io.json import json_normalize # tranform JSON file into a pandas dataframe
# Matplotlib and associated plotting modules
import matplotlib.cm as cm
import matplotlib.colors as colors
# import k-means from clustering stage
from sklearn.cluster import KMeans
import folium # map rendering library
print('Libraries imported.')
```

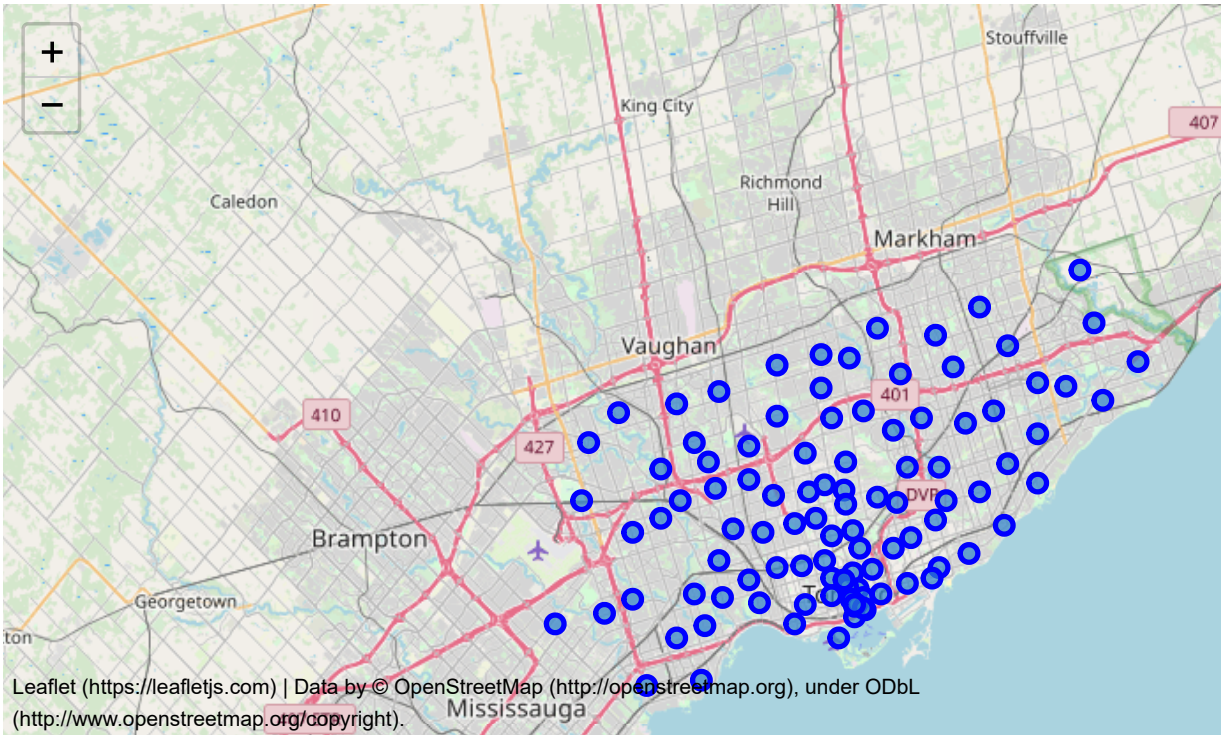
```
Libraries imported.
```

In [14]:
```
Latitude_avg=f['Latitude'].mean()
Longitude_avg=f['Longitude'].mean()
```

In [15]:
```
# create map using latitude and longitude values
map1 = folium.Map(location=[Latitude_avg, Longitude_avg], zoom_start=10)

# add markers to map
for lat, lng, postcode, borough in zip(f['Latitude'], f['Longitude'], f['Postcode'], f['Borough']):
    label = '{}, {}'.format(postcode, 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(map1)
map1 #somehow this map won't display in Github webpage. You may download this file and the raw_data.
```

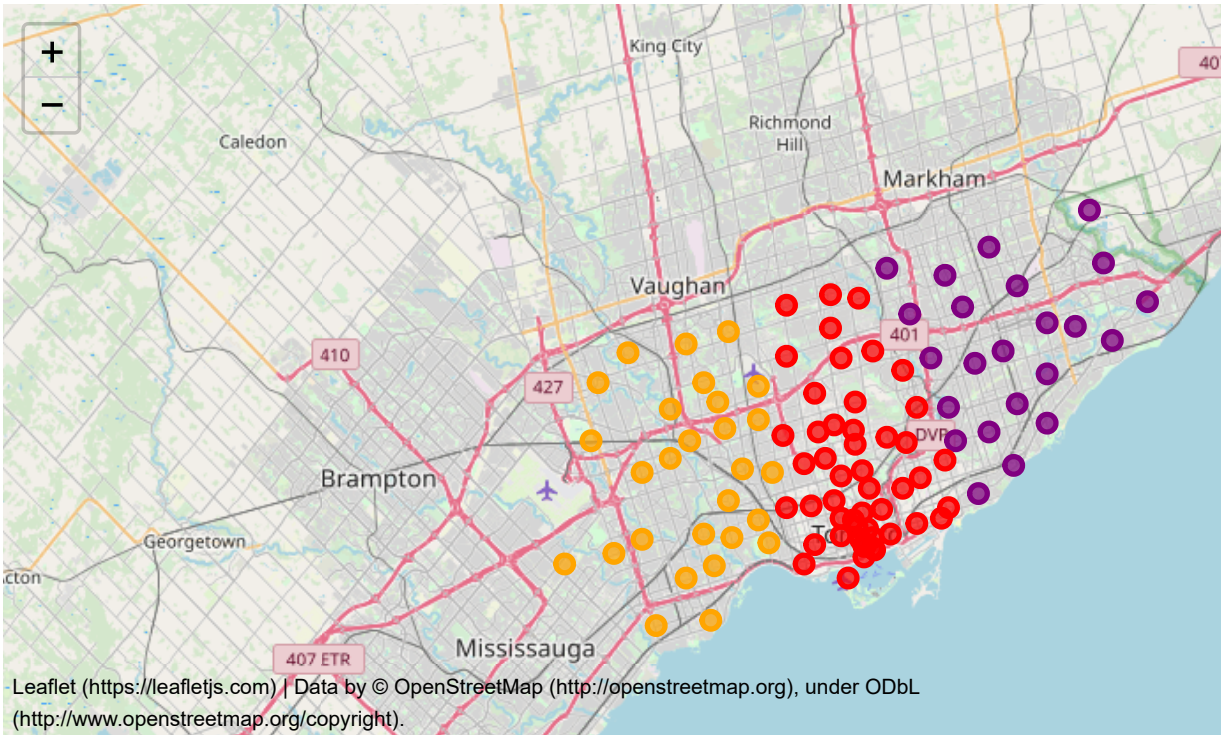
Out[15]:



In [16]:
```
#define a function to cluster those points into several groups according to their locations
def draw_map():
    g=f
    n=int(input('input an integer(1~8):'))
    k_means = KMeans(init = "k-means++", n_clusters = n, n_init = 12)
    k_means.fit(g[['Longitude','Latitude']])
    k_means_labels = k_means.labels_
    g['Label']=k_means_labels
    color_list=['purple','red','orange','blue','yellow','deeppink','brown','gray']
    map1 = folium.Map(location=[Latitude_avg, Longitude_avg], zoom_start=10)
    # add markers to map
    for lat, lng, postcode, borough,color_label in zip(g['Latitude'], g['Longitude'], g['Postcode'],
        label = '{}, {}'.format(postcode, borough)
        label = folium.Popup(label, parse_html=True)
        folium.CircleMarker(
            [lat, lng],
            radius=5,
            popup=label,
            color=color_list[color_label],
            fill=True,
            fill_opacity=0.7,
            parse_html=False).add_to(map1)
    display(map1)
```

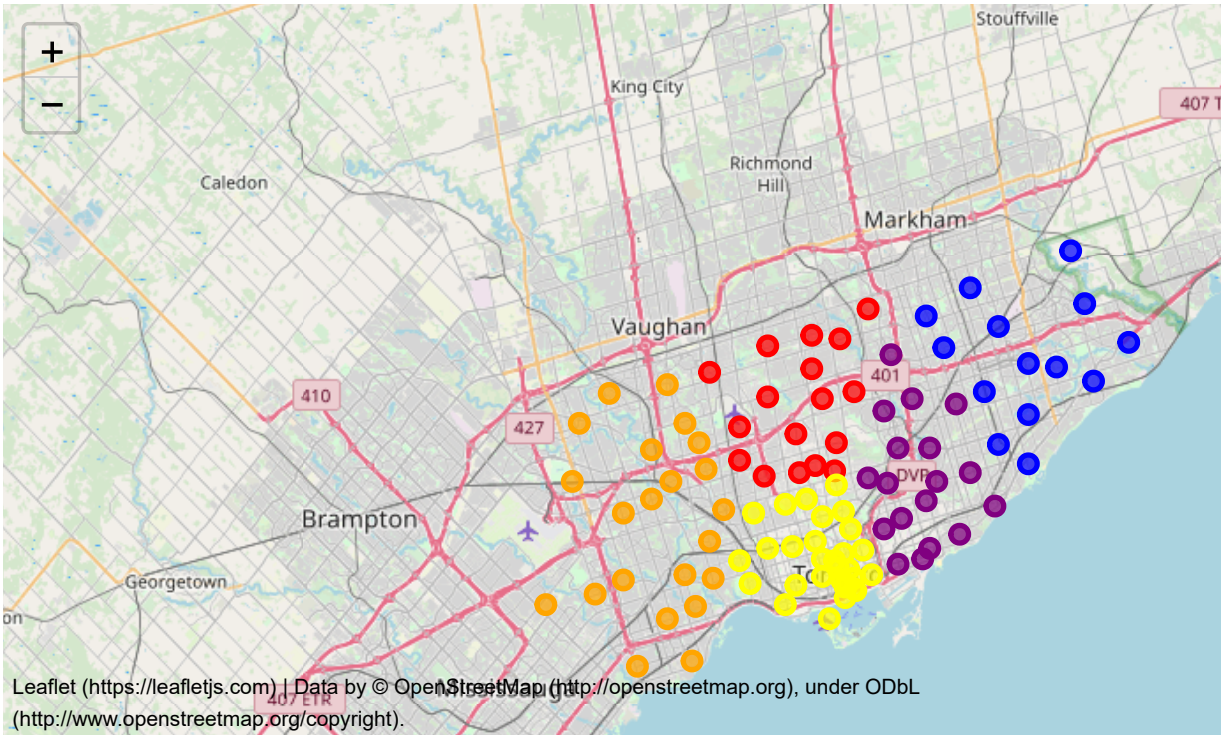
In [17]:
```
draw_map()#somehow this map won't display in Github webpage. You may download this file and the raw_
```

```
input an integer(1~8):3
```



In [18]:
```
draw_map() #somehow this map won't display in Github webpage. You may download this file and the raw
```

```
input an integer(1~8):5
```



In [19]:
```
draw_map() #somehow this map won't display in Github webpage. You may download this file and the raw
```

```
input an integer(1~8):8
```