Instulling Required Libraries

In [1]:
```
import sys
!{sys.executable} -m pip install geocoder
```

```
Requirement already satisfied: geocoder in /opt/conda/envs/Python36/lib/pytho
n3.6/site-packages (1.38.1)
Requirement already satisfied: click in /opt/conda/envs/Python36/lib/python3.
6/site-packages (from geocoder) (7.0)
Requirement already satisfied: requests in /opt/conda/envs/Python36/lib/pytho
n3.6/site-packages (from geocoder) (2.21.0)
Requirement already satisfied: future in /opt/conda/envs/Python36/lib/python
3.6/site-packages (from geocoder) (0.17.1)
Requirement already satisfied: six in /opt/conda/envs/Python36/lib/python3.6/
site-packages (from geocoder) (1.12.0)
Requirement already satisfied: ratelim in /opt/conda/envs/Python36/lib/python
3.6/site-packages (from geocoder) (0.1.6)
Requirement already satisfied: certifi>=2017.4.17 in /opt/conda/envs/Python3
6/lib/python3.6/site-packages (from requests->geocoder) (2019.11.28)
Requirement already satisfied: chardet<3.1.0,>=3.0.2 in /opt/conda/envs/Pytho
n36/lib/python3.6/site-packages (from requests->geocoder) (3.0.4)
Requirement already satisfied: idna<2.9,>=2.5 in /opt/conda/envs/Python36/li
b/python3.6/site-packages (from requests->geocoder) (2.8)
Requirement already satisfied: urllib3<1.25,>=1.21.1 in /opt/conda/envs/Pytho
n36/lib/python3.6/site-packages (from requests->geocoder) (1.24.1)
Requirement already satisfied: decorator in /opt/conda/envs/Python36/lib/pyth
on3.6/site-packages (from ratelim->geocoder) (4.3.2)
```

In [2]:
```
!conda install -c conda-forge folium=0.5.0 --yes
```

```
Solving environment: done

# All requested packages already installed.
```

Importing The Required Libraries

In [3]:
```
#importing the required library
import pandas as pd
import numpy as np
from sklearn.cluster import KMeans
import folium
from geopy.geocoders import Nominatim
import matplotlib.cm as cm
import matplotlib.colors as colors
import geocoder
```

Reading The Data From The HTML Page

In [4]:
```
#reading the data from the HTML page
data=pd.read_html("https://en.wikipedia.org/wiki/List_of_postal_codes_of_Canada:_M")
```

Extracting The First Table From The Dataset

In [5]:
```
#Extracting the first table from the dataset
df=data[0]
```

In [6]:
```
df.head()
```

Out[6]:

	Postcode	Borough	Neighbourhood
0	M1A	Not assigned	Not assigned
1	M2A	Not assigned	Not assigned
2	M3A	North York	Parkwoods
3	M4A	North York	Victoria Village
4	M5A	Downtown Toronto	Harbourfront

In [7]:
```
# reseting the index
df=df.reset_index()
```

In [8]:
```
df.head()
```

Out[8]:

	index	Postcode	Borough	Neighbourhood
0	0	M1A	Not assigned	Not assigned
1	1	M2A	Not assigned	Not assigned
2	2	M3A	North York	Parkwoods
3	3	M4A	North York	Victoria Village
4	4	M5A	Downtown Toronto	Harbourfront

In [9]:
```
# droping index column
df = df.drop("index", axis=1)
```

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

Out[10]:

	Postcode	Borough	Neighbourhood
0	M1A	Not assigned	Not assigned
1	M2A	Not assigned	Not assigned
2	M3A	North York	Parkwoods
3	M4A	North York	Victoria Village
4	M5A	Downtown Toronto	Harbourfront

Extracting All The Borough Except Not Assigned Value

In [11]:
```
#Extracting all the Borough except Not assigned value
df=df[df['Borough'] != 'Not assigned']
```

In [12]:
```
df.head()
```

Out[12]:

	Postcode	Borough	Neighbourhood
2	M3A	North York	Parkwoods
3	M4A	North York	Victoria Village
4	M5A	Downtown Toronto	Harbourfront
5	M6A	North York	Lawrence Heights
6	M6A	North York	Lawrence Manor

In [13]:
```
df.index = np.arange(1, len(df) + 1)
df.head()
```

Out[13]:

	Postcode	Borough	Neighbourhood
1	M3A	North York	Parkwoods
2	M4A	North York	Victoria Village
3	M5A	Downtown Toronto	Harbourfront
4	M6A	North York	Lawrence Heights
5	M6A	North York	Lawrence Manor

In [14]:
```
df['Neighbourhood'] = np.where(df['Neighbourhood']=='Not assigned',df['Borough'],df['Neighbourhood'])
df.head(10)
```

Out[14]:

	Postcode	Borough	Neighbourhood
1	M3A	North York	Parkwoods
2	M4A	North York	Victoria Village
3	M5A	Downtown Toronto	Harbourfront
4	M6A	North York	Lawrence Heights
5	M6A	North York	Lawrence Manor
6	M7A	Queen's Park	Queen's Park
7	M9A	Downtown Toronto	Queen's Park
8	M1B	Scarborough	Rouge
9	M1B	Scarborough	Malvern
10	M3B	North York	Don Mills North

In [15]:
```
df.head(10)
```

Out[15]:

	Postcode	Borough	Neighbourhood
1	M3A	North York	Parkwoods
2	M4A	North York	Victoria Village
3	M5A	Downtown Toronto	Harbourfront
4	M6A	North York	Lawrence Heights
5	M6A	North York	Lawrence Manor
6	M7A	Queen's Park	Queen's Park
7	M9A	Downtown Toronto	Queen's Park
8	M1B	Scarborough	Rouge
9	M1B	Scarborough	Malvern
10	M3B	North York	Don Mills North

Grouping The Postcode And Combining The Neighbourhood

In [16]:
```
# grouping the Postcode and combining the Neighbourhood
df["Neighbourhood"] = df.groupby("Postcode")["Neighbourhood"].transform(lambda neigh: ', '.join(neigh))
```

In [17]:
```
#remove duplicates
df = df.drop_duplicates()
```

In [18]:
```
df.head()
```

Out[18]:

	Postcode	Borough	Neighbourhood
1	M3A	North York	Parkwoods
2	M4A	North York	Victoria Village
3	M5A	Downtown Toronto	Harbourfront
4	M6A	North York	Lawrence Heights, Lawrence Manor
6	M7A	Queen's Park	Queen's Park

Replacing The Not Assigned of The Neighbourhood With The Borough Value

In [19]:
```
# replacing the Not assigned of the Neighbourhood with the Borough value
df['Neighbourhood'].replace("Not assigned", df["Borough"],inplace=True)
```

Finding The Shape Of The Dataset

In [20]:
```
# finding the shape of the dataset
df.shape
```

Out[20]:
```
(103, 3)
```

Getting The values Of Latitude And Longitude

In [21]:
```
def get_geos(postalCode):
    # initialize your variable to None
    lat_lng_coords = None
    # loop until you get the coordinates
    while(lat_lng_coords is None):
        g = geocoder.arcgis('{}, Toronto, Ontario'.format(postalCode.strip()))
        lat_lng_coords = g.latlng
        latitude = lat_lng_coords[0]
        longitude = lat_lng_coords[1]
    return latitude,longitude
```

In [22]:
```
df['Latitude'],df['Longitude'] = zip(*df['Postcode'].apply(get_geos))
df.head()
```

Out[22]:

	Postcode	Borough	Neighbourhood	Latitude	Longitude
1	M3A	North York	Parkwoods	43.752420	-79.329242
2	M4A	North York	Victoria Village	43.730600	-79.313265
3	M5A	Downtown Toronto	Harbourfront	43.650295	-79.359166
4	M6A	North York	Lawrence Heights, Lawrence Manor	43.723270	-79.451286
6	M7A	Queen's Park	Queen's Park	43.661150	-79.391715

Applying Clustring Algorithm For The Dataset

In [23]:
```
cluster_df=df.drop(['Postcode','Borough','Neighbourhood'], axis=1)
cluster_df.head()
```

Out[23]:

	Latitude	Longitude
1	43.752420	-79.329242
2	43.730600	-79.313265
3	43.650295	-79.359166
4	43.723270	-79.451286
6	43.661150	-79.391715

In [24]:
```
from sklearn.preprocessing import StandardScaler

X = cluster_df.values[:,0:]
X = np.nan_to_num(X)
cluster_dataset = StandardScaler().fit_transform(X)
cluster_dataset[0,:]
```

Out[24]:
```
array([0.914631  , 0.69584445])
```

In [25]:
```
num_clusters = 5

k_means = KMeans(init="k-means++", n_clusters=num_clusters, n_init=12)
k_means.fit(cluster_dataset)
labels = k_means.labels_

print(labels)
```

```
[3 3 0 2 0 4 1 3 0 0 2 4 1 3 0 0 0 4 1 0 0 2 1 0 0 0 1 3 3 0 0 0 1 3 2 0 0
 0 1 3 2 0 0 0 1 3 2 0 0 2 2 1 3 2 0 3 2 2 0 3 2 3 3 4 2 1 3 3 0 4 2 1 3 3
 0 4 0 2 1 0 0 4 1 0 0 1 0 0 4 2 1 0 0 4 2 1 0 0 4 0 0 4 4]
```

In [26]:
```
cluster_dataset=df
cluster_dataset['labels']=labels
cluster_dataset.head()
```

Out[26]:

	Postcode	Borough	Neighbourhood	Latitude	Longitude	labels
1	M3A	North York	Parkwoods	43.752420	-79.329242	3
2	M4A	North York	Victoria Village	43.730600	-79.313265	3
3	M5A	Downtown Toronto	Harbourfront	43.650295	-79.359166	0
4	M6A	North York	Lawrence Heights, Lawrence Manor	43.723270	-79.451286	2
6	M7A	Queen's Park	Queen's Park	43.661150	-79.391715	0

Drawing The Custring map

In [27]:
```
address = 'Toronto, Ontario'

geolocator = Nominatim(user_agent="toronto_ontario")
location = geolocator.geocode(address)
latitude = location.latitude
longitude = location.longitude
print('The geograpical coordinates of Toronto, Ontario are {}, {}.'.format(lat
itude, longitude))
```

```
The geograpical coordinates of Toronto, Ontario are 43.653963, -79.387207.
```

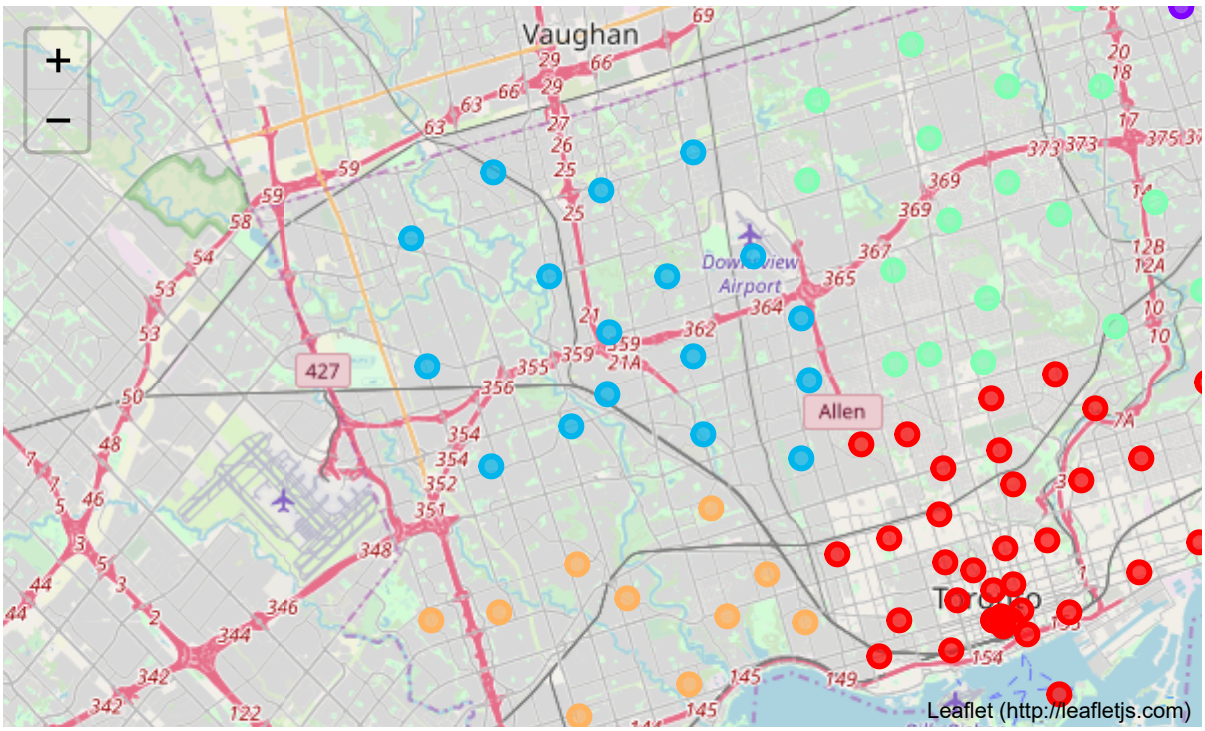
In [28]:
```
# create map
map_clusters = folium.Map(location=[latitude, longitude], zoom_start=11)

# set color scheme for the clusters
x = np.arange(num_clusters)
ys = [i + x + (i*x)**2 for i in range(num_clusters)]
colors_array = cm.rainbow(np.linspace(0, 1, len(ys)))
rainbow = [colors.rgb2hex(i) for i in colors_array]

# add markers to the map
markers_colors = []
for lat, lon, poi, cluster in zip(cluster_dataset['Latitude'], cluster_dataset
['Longitude'], cluster_dataset['Neighbourhood'], cluster_dataset['labels']):
    label = folium.Popup(str(poi) + ' Cluster ' + str(cluster), parse_html=Tru
e)
    folium.CircleMarker(
        [lat, lon],
        radius=5,
        popup=label,
        color=rainbow[cluster-1],
        fill=True,
        fill_color=rainbow[cluster-1],
        fill_opacity=0.7).add_to(map_clusters)

map_clusters
```

Out[28]:



In [ ]: