Coursera Capstone Project—the Battle of Neighbourhoods

PROBLEM & BACKGROUND

Toronto and New York are the famous places in the world. They are diverse in many ways. Both are multicultural as well as the financial hubs of their respective countries. We want to explore how much they are similar or dissimilar in aspects from a tourist point of view regarding food, accommodation, beautiful places, and many more.

Today Tourism is one of the pillars of the economy and the people most often visits those countries who are rich in heritage and developed enough from a foreign prospective, like friendly environment. Every city is unique in their own way and give something new. And now the information is so common regarding location of every place around the world on your fingertips which make it easier to explore. Therefore, tourists always eager to travel to different places on the basis of available information, and the comparison (the part of the information) between the two cities always assist to choose the specific places or according to their choice

DATA DESCRIPTION

For this problem, we will get the services of Foursquare API to explore the data of two cities, in terms of their neighborhoods. The data also include the information about the places around each neighborhood like restaurants, hotels, coffee shops, parks, theaters, art galleries, museums and many more. We selected one Borough from each city to analyze their neighborhoods. Manhattan from New York and Downtown Toronto from Toronto. We will use machine learning technique, “Clustering” to segment the neighborhoods with similar objects on the basis of each neighborhood data. These objects will be given priority on the basis of foot traffic (activity) in their respective neighborhoods. This will help to locate the tourist’s areas and hubs, and then we can judge the similarity or dissimilarity between two cities on that basis.

METHODOLOGY

As we have selected two cities Borough to explore their neighborhoods. The data exploration, analysis and visualization for both boroughs are done in the same way but separately.¶

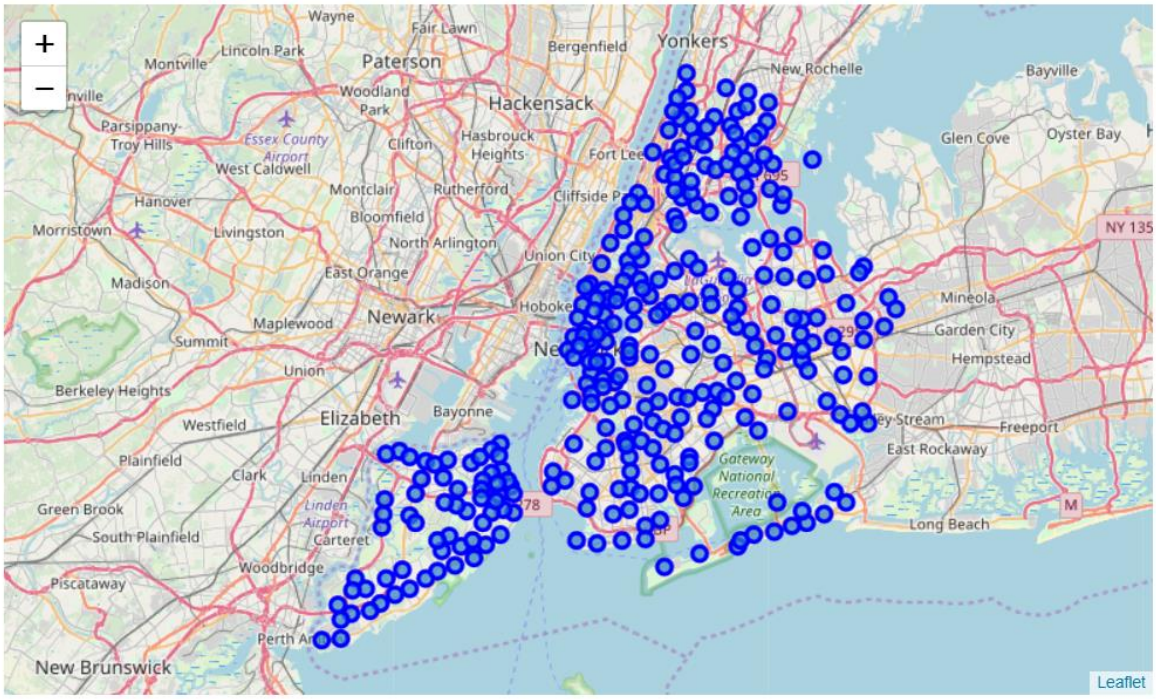
Let's load and explore the data about New York

```
!wget -q -O 'newyork_data.json' https://cocl.us/new_york_dataset
print('Data downloaded!')
with open('newyork_data.json') as json_data:
    newyork_data = json.load(json_data)

neighborhoods_data = newyork_data['features']

# define the dataframe columns
column_names = ['Borough', 'Neighborhood', 'Latitude', 'Longitude']

# instantiate the dataframe
neighborhoods = pd.DataFrame(columns=column_names)

for data in neighborhoods_data:
    borough = neighborhood_name = data['properties']['borough']
    neighborhood_name = data['properties']['name']

    neighborhood_latlon = data['geometry']['coordinates']
    neighborhood_lat = neighborhood_latlon[1]
    neighborhood_lon = neighborhood_latlon[0]

    neighborhoods = neighborhoods.append({'Borough': borough,
                                          'Neighborhood': neighborhood_name,
                                          'Latitude': neighborhood_lat,
                                          'Longitude': neighborhood_lon}, ignore_index=True)

print('The dataframe has {} boroughs and {} neighborhoods.'.format(
        len(neighborhoods['Borough'].unique()),
        neighborhoods.shape[0]
    )
)
```

```
Data downloaded!
The dataframe has 5 boroughs and 306 neighborhoods.
```

```
address = 'New York City, NY'

geolocator = Nominatim(user_agent="ny_explorer")
location = geolocator.geocode(address)
latitude = location.latitude
longitude = location.longitude
print('The geograpical coordinate of New York City are {}, {}.'.format(latitude, longitude))

# create map of New York using latitude and longitude values
map_newyork = folium.Map(location=[latitude, longitude], zoom_start=10)

# add markers to map
for lat, lng, borough, neighborhood in zip(neighborhoods['Latitude'], neighborhoods['Longitude'], neighbor
hoods['Borough'], neighborhoods['Neighborhood']):
    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_newyork)

map_newyork
```

```
The geograpical coordinate of New York City are 40.7127281, -74.0060152.
```



Let’s get Toronto data:

```
df_geo = pd.merge(df, g_data, on=['Postcode'])
df_geo = df_geo[df_geo['Borough'].str.contains('Toronto')]
```

```
lat_Toronto, long_Toronto = df_geo['Latitude'].mean(), df_geo['Longitude'].mean()
print('Toronto is at the coordinates (%s, %s).' % (lat_Toronto, long_Toronto))
map_toronto = folium.Map(location=[lat_Toronto, long_Toronto], zoom_start=12)

# add markers to map
for lat, lng, borough, neighborhood in zip(df_geo['Latitude'], df_geo['Longitude'], df_geo['Borough'], df_
geo['Neighbourhood']):
    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
```

```
Toronto is at the coordinates (43.66726218421053, -79.38988323421052).
```

ANALYSIS

We analyze both boroughs neighborhoods through one hot encoding (giving ‘1’ if a venue category is there, and ‘0’ in case of venue category is not there). On the basis of one hot encoding, we calculate mean of the frequency of occurrence of each category and picked top ten venues on that basis for each neighborhood. It means the top venues are showing the foot traffic or the more visited places.

New York:

In [118]:
```
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={}'.for
mat(
            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)
```

In [119]:
```
manhattan_venues = getNearbyVenues(names=manhattan_data['Neighborhood'],
                                   latitudes=manhattan_data['Latitude'],
                                   longitudes=manhattan_data['Longitude']
                                  )
```

In [119]:
```
manhattan_venues = getNearbyVenues(names=manhattan_data['Neighborhood'],
                                   latitudes=manhattan_data['Latitude'],
                                   longitudes=manhattan_data['Longitude']
                                  )
```

```
Marble Hill
Chinatown
Washington Heights
Inwood
Hamilton Heights
Manhattanville
Central Harlem
East Harlem
Upper East Side
Yorkville
Lenox Hill
Roosevelt Island
Upper West Side
Lincoln Square
Clinton
Midtown
Murray Hill
Chelsea
Greenwich Village
East Village
Lower East Side
Tribeca
Little Italy
Soho
West Village
Manhattan Valley
Morningside Heights
Gramercy
Battery Park City
Financial District
Carnegie Hill
Noho
Civic Center
Midtown South
Sutton Place
Turtle Bay
Tudor City
Stuyvesant Town
Flatiron
Hudson Yards
```

In [120]:
```
print(manhattan_venues.shape)
manhattan_venues.head()
```

```
(3317, 7)
```

Out[120]:

	Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
0	Marble Hill	40.876551	-73.91066	Arturo's	40.874412	-73.910271	Pizza Place
1	Marble Hill	40.876551	-73.91066	Bikram Yoga	40.876844	-73.906204	Yoga Studio
2	Marble Hill	40.876551	-73.91066	Tibbett Diner	40.880404	-73.908937	Diner
3	Marble Hill	40.876551	-73.91066	Dunkin'	40.877136	-73.906666	Donut Shop
4	Marble Hill	40.876551	-73.91066	Starbucks	40.877531	-73.905582	Coffee Shop

In [121]:
```
manhattan_venues.groupby('Neighborhood').count()
```

Out[121]:

Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
Battery Park City	100	100	100	100	100	100
Carnegie Hill	100	100	100	100	100	100
Central Harlem	44	44	44	44	44	44
Chelsea	100	100	100	100	100	100
Chinatown	100	100	100	100	100	100
Civic Center	100	100	100	100	100	100
Clinton	100	100	100	100	100	100
East Harlem	44	44	44	44	44	44
East Village	100	100	100	100	100	100
Financial District	100	100	100	100	100	100
Flatiron	100	100	100	100	100	100
Gramercy	100	100	100	100	100	100
Greenwich Village	100	100	100	100	100	100
Hamilton Heights	59	59	59	59	59	59
Hudson Yards	76	76	76	76	76	76
Inwood	57	57	57	57	57	57
Lenox Hill	100	100	100	100	100	100
Lincoln Square	100	100	100	100	100	100
Little Italy	100	100	100	100	100	100
Lower East Side	58	58	58	58	58	58
Manhattan Valley	59	59	59	59	59	59

```
num_top_venues = 5

for hood in manhattan_grouped['Neighborhood']:
    print("----"+hood+"----")
    temp = manhattan_grouped[manhattan_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_top_venues))
    print('\n')
```

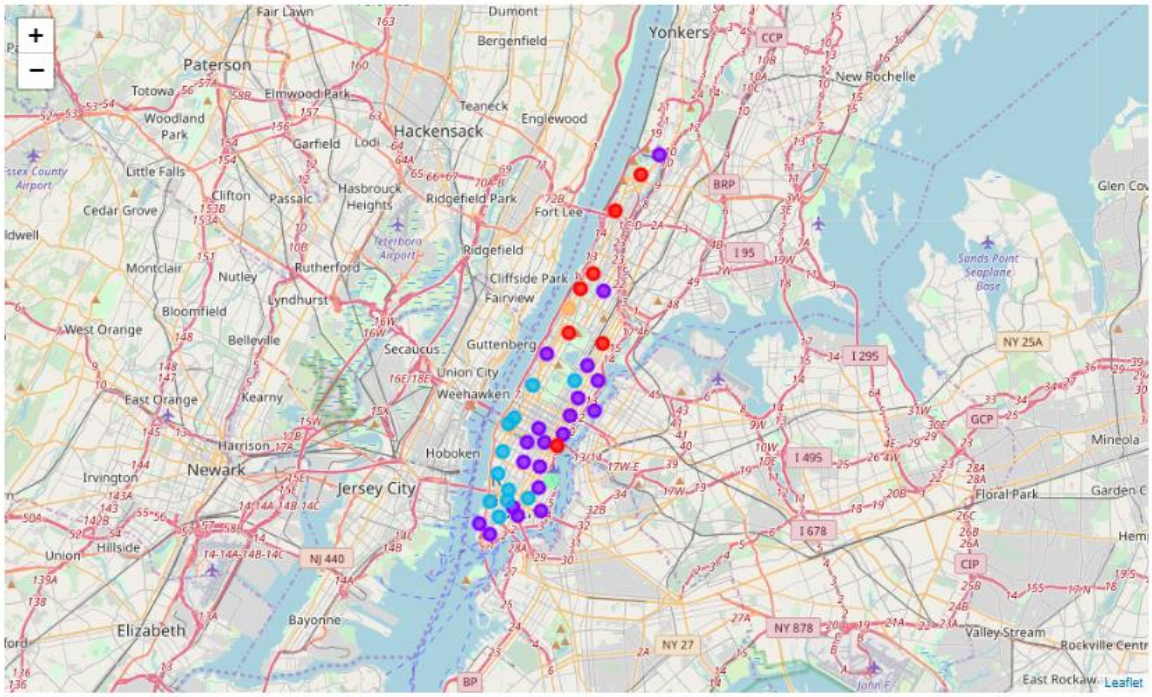
```
----Battery Park City----
           venue  freq
0           Park  0.08
1    Coffee Shop  0.07
2          Hotel  0.05
3            Gym  0.04
4  Memorial Site  0.04


----Carnegie Hill----
                venue  freq
0         Pizza Place  0.06
1         Coffee Shop  0.06
2                Café  0.04
3  Japanese Restaurant  0.03
4   French Restaurant  0.03


----Central Harlem----
                venue  freq
0   African Restaurant  0.07
1    French Restaurant  0.05
2       Cosmetics Shop  0.05
3  Gym / Fitness Center  0.05
4   Seafood Restaurant  0.05


----Chelsea----
                venue  freq
0         Coffee Shop  0.06
1  Italian Restaurant  0.05
2      Ice Cream Shop  0.05
3           Nightclub  0.04
4  American Restaurant  0.04


----Chinatown----
                venue  freq
0   Chinese Restaurant  0.10
1        Cocktail Bar  0.04
2  American Restaurant  0.04
3                 Spa  0.03
4  Dumpling Restaurant  0.03
```

In [126]:
```
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]
```

In [127]:
```
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'] = manhattan_grouped['Neighborhood']

for ind in np.arange(manhattan_grouped.shape[0]):
    neighborhoods_venues_sorted.iloc[ind, 1:] = return_most_common_venues(manhattan_grouped.iloc[ind, :], num_top_venues)

neighborhoods_venues_sorted.head()
```

Out[127]:

	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	Battery Park City	Park	Coffee Shop	Hotel	Gym	Memorial Site	Clothing Store	Wine Shop	Italian Restaurant	Men's Store	Shopping Mall
1	Carnegie Hill	Coffee Shop	Pizza Place	Café	Yoga Studio	Spa	Bar	Japanese Restaurant	Bookstore	Grocery Store	Gym
2	Central Harlem	African Restaurant	Gym / Fitness Center	American Restaurant	French Restaurant	Public Art	Fried Chicken Joint	Chinese Restaurant	Seafood Restaurant	Cosmetics Shop	Bookstore
3	Chelsea	Coffee Shop	Ice Cream Shop	Italian Restaurant	Bakery	American Restaurant	Nightclub	Theater	Hotel	Seafood Restaurant	Cupcake Shop
4	Chinatown	Chinese Restaurant	American Restaurant	Cocktail Bar	Bubble Tea Shop	Spa	Dumpling Restaurant	Vietnamese Restaurant	Malay Restaurant	Korean Restaurant	Jewelry Store




```
manhattan_merged.loc[manhattan_merged['Cluster Labels'] == 0, manhattan_merged.columns[[1] + list(range(5, manhattan_merged.shape[1]))]]
```

Toronto:

The dataframe has 4 boroughs and 38 neighborhoods.

```
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_secret={}&v={}&ll={},{}&radius={}&limit={}'.for
mat(
            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 = ['Neighbourhood',
                  'Neighbourhood Latitude',
                  'Neighbourhood Longitude',
                  'Venue',
                  'Venue Latitude',
                  'Venue Longitude',
                  'Venue Category']

    return(nearby_venues)
toronto_venues = getNearbyVenues(names=df_geo['Neighbourhood'],
                                 latitudes=df_geo['Latitude'],
                                 longitudes=df_geo['Longitude']
                                )
```

```
Harbourfront, Regent Park
Ryerson, Garden District
St. James Town
The Beaches
Berczy Park
Central Bay Street
Christie
Adelaide, King, Richmond
Dovercourt Village, Dufferin
Harbourfront East, Toronto Islands, Union Station
Little Portugal, Trinity
The Danforth West, Riverdale
Design Exchange, Toronto Dominion Centre
Brockton, Exhibition Place, Parkdale Village
The Beaches West, India Bazaar
Commerce Court, Victoria Hotel
Studio District
Lawrence Park
Roselawn
Davisville North
Forest Hill North, Forest Hill West
High Park, The Junction South
North Toronto West
The Annex, North Midtown, Yorkville
Parkdale, Roncesvalles
Davisville
Harbord, University of Toronto
Runnymede, Swansea
Moore Park, Summerhill East
Chinatown, Grange Park, Kensington Market
Deer Park, Forest Hill SE, Rathnelly, South Hill, Summerhill West
CN Tower, Bathurst Quay, Island airport, Harbourfront West, King and Spadina, Railway Lands, South Niagara
Rosedale
Stn A PO Boxes 25 The Esplanade
Cabbagetown, St. James Town
First Canadian Place, Underground city
Church and Wellesley
Business Reply Mail Processing Centre 969 Eastern
```

In [150]:
```
toronto_grouped = toronto_onehot.groupby('Neighbourhood').mean().reset_index()
toronto_grouped.head()
```

Out[150]:

	Neighbourhood	Adult Boutique	Afghan Restaurant	Airport	Airport Food Court	Airport Gate	Airport Lounge	Airport Service	Airport Terminal	American Restaurant	Antique Shop	Aquarium	Art Gallery	M
0	Adelaide, King, Richmond	0.0	0.0	0.000000	0.000000	0.000000	0.000000	0.0	0.000000	0.04	0.0	0.0	0.010000	0
1	Berczy Park	0.0	0.0	0.000000	0.000000	0.000000	0.000000	0.0	0.000000	0.00	0.0	0.0	0.018182	0
2	Brockton, Exhibition Place, Parkdale Village	0.0	0.0	0.000000	0.000000	0.000000	0.000000	0.0	0.000000	0.00	0.0	0.0	0.000000	0
3	Business Reply Mail Processing Centre 969 Eastern	0.0	0.0	0.000000	0.000000	0.000000	0.000000	0.0	0.000000	0.00	0.0	0.0	0.000000	0
4	CN Tower, Bathurst Quay, Island airport, Harbo...	0.0	0.0	0.066667	0.066667	0.066667	0.133333	0.2	0.133333	0.00	0.0	0.0	0.000000	0

In [151]:
```
num_top_venues = 5

for hood in toronto_grouped['Neighbourhood']:
    print("----"+hood+"----")
    temp = toronto_grouped[toronto_grouped['Neighbourhood'] == 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_top_venues))
    print('\n')
```

```
----Adelaide, King, Richmond----
              venue  freq
0       Coffee Shop  0.06
1              Café  0.05
2         Steakhouse  0.04
3               Bar  0.04
4  American Restaurant  0.04
```

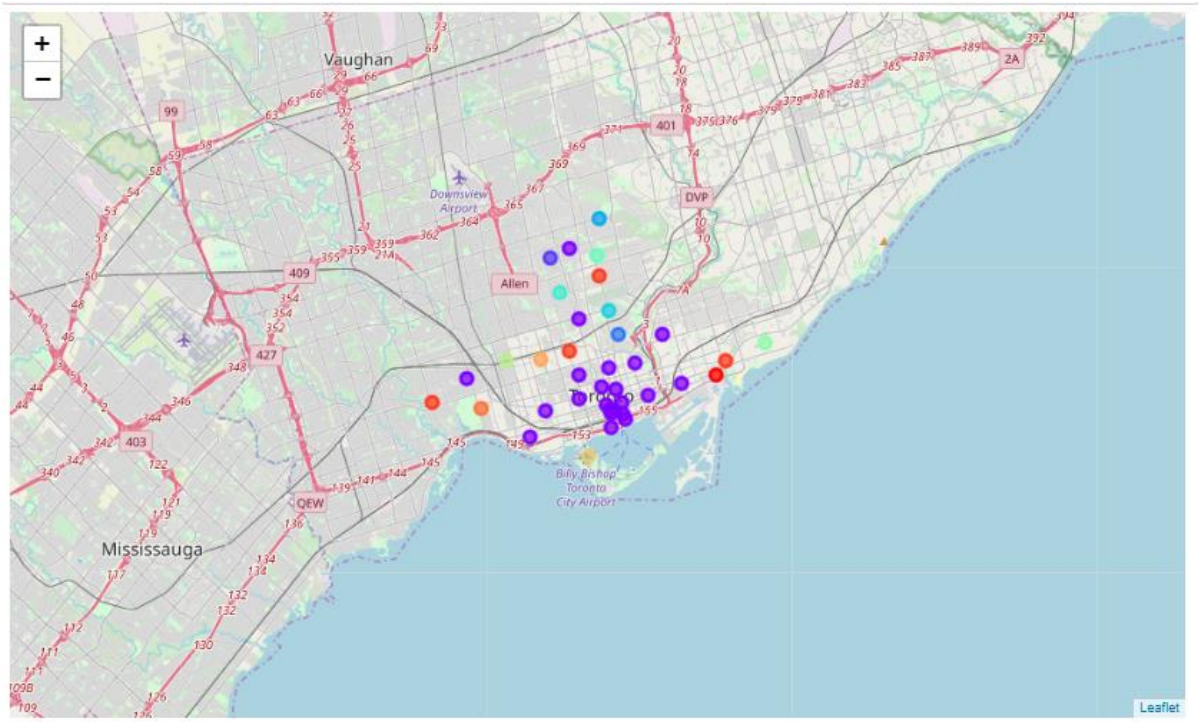
```
def return_most_common_venues(row, num_top_venues):
    row_categories = row.iloc[1:]
    row_categories_sorted = row_categories.sort_values(ascending=False)

    return row_categories_sorted.index.values[0:num_top_venues]
num_top_venues = 10

indicators = ['st', 'nd', 'rd']

# create columns according to number of top venues
columns = ['Neighbourhood']
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['Neighbourhood'] = toronto_grouped['Neighbourhood']

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(7)
```

	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	Adelaide, King, Richmond	Coffee Shop	Café	American Restaurant	Thai Restaurant	Steakhouse	Bar	Hotel	Burger Joint	Bakery	Cosmetics Shop
1	Berczy Park	Coffee Shop	Cocktail Bar	Seafood Restaurant	Café	Restaurant	Bakery	Steakhouse	Beer Bar	Italian Restaurant	Farmers Market
2	Brockton, Exhibition Place, Parkdale Village	Performing Arts Venue	Coffee Shop	Breakfast Spot	Café	Gym	Pet Store	Restaurant	Stadium	Furniture / Home Store	Convenience Store
3	Business Reply Mail Processing Centre 969 Eastern	Light Rail Station	Yoga Studio	Auto Workshop	Pizza Place	Recording Studio	Restaurant	Burrito Place	Brewery	Skate Park	Smoke Shop
4	CN Tower, Bathurst Quay, Island airport, Harbo...	Airport Service	Airport Lounge	Airport Terminal	Plane	Airport	Airport Food Court	Airport Gate	Boutique	Harbor / Marina	Boat or Ferry
5	Cabbagetown, St. James Town	Coffee Shop	Park	Restaurant	Café	Pizza Place	Italian Restaurant	Pub	Chinese Restaurant	Bakery	Jewelry Store
6	Central Bay Street	Coffee Shop	Italian Restaurant	Café	Burger Joint	Chinese Restaurant	Bakery	Indian Restaurant	Bubble Tea Shop	Salad Place	Sushi Restaurant

```
# import k-means from clustering stage
from sklearn.cluster import KMeans

# set number of clusters
kclusters = 14

toronto_grouped_clustering = toronto_grouped.drop('Neighbourhood', 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_[0:50]
```

```
array([ 1,  1,  1,  0, 10,  1,  1,  1, 11,  1,  1, 13,  7,  1,  1,  9,  1,
        6,  1,  1,  1,  1,  4,  1,  5,  1, 12,  3,  2, 13,  1,  1,  1,  1,
       13,  8, 13,  1], dtype=int32)
```

```
# add clustering labels
neighborhoods_venues_sorted.insert(0, 'Cluster Labels2', kmeans.labels_)

toronto_merged = df_geo

# merge toronto_grouped with toronto_data to add latitude/longitude for each neighborhood
toronto_merged = toronto_merged.join(neighborhoods_venues_sorted.set_index('Neighbourhood'), on='Neighbourhood')
toronto_merged.dropna(inplace=True)
# create map
map_clusters = folium.Map(location=[lat_Toronto, long_Toronto], zoom_start=11)

# set color scheme for the clusters
x = np.arange(kclusters)
ys = [i + x + (i*x)**2 for i in range(kclusters)]
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(toronto_merged['Latitude'], toronto_merged['Longitude'], toronto_merged['Neighbourhood'], toro
nto_merged['Cluster Labels2']):
    label = folium.Popup(str(poi) + ' Cluster ' + str(cluster), parse_html=True)
    folium.CircleMarker(
        [lat, lon],
        radius=5,
        popup=label,
        color=rainbow[int(cluster)-1],
        fill=True,
        fill_color=rainbow[int(cluster)-1],
        fill_opacity=0.7).add_to(map_clusters)

map_clusters
```




```
toronto_merged.loc[toronto_merged['Cluster Labels2'] == 0, toronto_merged.columns[[1] + list(range(5, toronto_merged.shape[1
]))]]
```

	Borough	Cluster Labels2	Cluster Labels1	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
100	East Toronto	0	0	Light Rail Station	Yoga Studio	Auto Workshop	Pizza Place	Recording Studio	Restaurant	Burrito Place	Brewery	Skate Park	Smoke Shop

RESULTS

After clustering the data of the respective neighborhoods, both cities (Boroughs) have venues which can be explored and attract the Tourists. The neighborhoods are much similar in features like Theaters, opera houses, food places, clubs, museums, parks etc. As far as concern to dissimilarity, it differs in terms of some unique places like historical places and monuments.

Observations & Recommendations

When we compare the tourist places, we observe that the historical place is only situated in Downtown Toronto and the Monument or landmark venue is in Manhattan neighborhoods. Similarly, Airport facility, Harbor, Sculpture garden and Boat or ferry services are also available i**n Downtown Toronto while venues like Nightlife, Climbing gym and Museums are present in Manhattan.

As far as concern to recommendations, we recommend Downtown Toronto Neighborhoods will be considered first to visit. The tourists have an easily travelling access due to Airport facility, which not only saves time but also helps to save money. This saved money can be utilized to explore more, the attracting venues.

Conclusion

The downtown Toronto and Manhattan neighborhoods have more like similar venues. As we know that every place is unique in its own way, so that’s argument is present in both neighborhoods. The dissimilarity exists in terms of some different venues and facilities but not on a larger extent.