Capstone project

Table of contents

- Introduction: Business Problem
- Data
- Methodology
- Analysis
- Results and Discussion
- Conclusion

Introduction: Businesss Problem

In this project, we suppose our client is working in New York and get two offres of jobs in Toronto and Paris. He want to choose the city that have more similar to New York to work. We will measure the similarity among New York, Toronto and Paris based on the clusters. First, for each city, we divide the set of neighborhoods into clusters, then assign each cluster with the name by using the top of venues. Seconde, we consider the clusters of each city to see which city Paris or Toronto is more similar to New York.

Data

We need data include:

- the list of neighborhoods New York city from https://cocl.us/new_york_dataset (https://cocl.us/new_york_dataset)
- the list of neighborhoods Toronto city from https://en.wikipedia.org/wiki/List_of_postal_codes_of_Canada:_M (https://en.wikipedia.org/wiki/List_of_postal_codes_of_Canada:_M)
- the list of neighborhoods Paris city that is given

For each city, we use the Google maps API to search the latitude and longitude of each neighborhoods.

Top of venues for each neighborhood is collected by using Foursquare API.

Methodology

```
In this project, we use the k-means cluster algorithm in scikit
learning library to cluster the set of neighborhoods for each city.
To do that, we collect the popular venues of every neighborhoods with
the radius 500m and limit 30. From we give the top of venues of every
neighborhoods and we use hot-coding to code the venues, then cluster
neighborhoods based on the information of the top venues. Then we
will assign the name of each cluster for the 1st common venues.
```

Analysis

Cluster neighborhood in Toronto

In [1]:

```
import pandas as pd #import the pandas library
# read the table of neighborhoods Toronto from url file
url="https://en.wikipedia.org/wiki/List_of_postal_codes_of_Canada:_M"
data=pd.read_html(url)
df=data[0]
df=pd.DataFrame({'Postcode':df.loc[1:,0],'Borough':df.loc[1:,1],'Neighl
df.head()
```

Out[1]:

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

In [2]:

```
# drop data with values be 'not assigned '
df.drop(df[df['Borough']=='Not assigned'].index,inplace=True)
for i in range(df.shape[0]):
    if df.iloc[i,2]=="Not assigned":
        df.iloc[i,2]=df.iloc[i,1]
df.head()
```

Out[2]:

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

In [3]:
```
# reset the index column
df.reset_index(inplace=True)
del df['index']
df.head()
```

Out[3]:

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

In [4]:
```
# group neighborhoods by borough
df=df.groupby(['Postcode','Borough'],sort=False).agg(','.join)
df.reset_index(inplace=True)
df.head()
```

Out[4]:

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

In [5]:
```
# collect the latitude and longitude from file
df_coor=pd.read_csv('Geospatial_Coordinates.csv')
df_coor.rename(index=str, columns={"Postal Code": "Postcode"},inplace=

df_coor.head()
```

Out[5]:

	Postcode	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 [6]:
```
Data=pd.merge(df, df_coor, on='Postcode')
Data.head()
```

Out[6]:

	Postcode	Borough	Neighbourhood	Latitude	Longitude
0	M3A	North York	Parkwoods	43.753259	-79.329656
1	M4A	North York	Victoria Village	43.725882	-79.315572
2	M5A	Downtown Toronto	Harbourfront,Regent Park	43.654260	-79.360636
3	M6A	North York	Lawrence Heights,Lawrence Manor	43.718518	-79.464763
4	M7A	Queen's Park	Queen's Park	43.662301	-79.389494

In [7]:
```
# Toronto data are ready after adding the latitude and longitude of ev
toronto_data = Data[Data['Borough'] == 'Downtown Toronto'].reset_index
toronto_data.head()
```

Out[7]:

	Postcode	Borough	Neighbourhood	Latitude	Longitude
0	M5A	Downtown Toronto	Harbourfront,Regent Park	43.654260	-79.360636
1	M5B	Downtown Toronto	Ryerson,Garden District	43.657162	-79.378937
2	M5C	Downtown Toronto	St. James Town	43.651494	-79.375418
3	M5E	Downtown Toronto	Berczy Park	43.644771	-79.373306
4	M5G	Downtown Toronto	Central Bay Street	43.657952	-79.387383

See the neighborhoods of Toronto by using map

In [9]:

```
#import library to get latitude and longitude
from geopy.geocoders import Nominatim
import folium # map rendering library

#
address = 'Toronto, CN'

geolocator = Nominatim(user_agent="ny_explorer")
location = geolocator.geocode(address)
latitude = location.latitude
longitude = location.longitude
print('The geograpical coordinate of Manhattan are {}, {}.'.format(lat
map_toronto = folium.Map(location=[latitude, longitude], zoom_start=11

# add markers to map
for lat, lng, label in zip(toronto_data['Latitude'], toronto_data['Lon
    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
```

```
The geograpical coordinate of Manhattan are 43.6425637, -79.38708718
32047.
```

Out[9]:

Input the information for using foursquare API

In [10]:
```
CLIENT_ID = 'YQYJICCBDPMQ5B3AV2QIBLIGHV2WP0AVBKDP2BGNTH41R1YB' # your
CLIENT_SECRET = 'IRPMEG1XPDMEUZFUJPH514KV32K3F1NFFKG13Y2OJ4FKQ5XR' # y
VERSION = '20180605' # Foursquare API version
LIMIT=30
print('Your credentails:')
print('CLIENT_ID: ' + CLIENT_ID)
print('CLIENT_SECRET:' + CLIENT_SECRET)
```

```
Your credentails:
CLIENT_ID: YQYJICCBDPMQ5B3AV2QIBLIGHV2WP0AVBKDP2BGNTH41R1YB
CLIENT_SECRET:IRPMEG1XPDMEUZFUJPH514KV32K3F1NFFKG13Y2OJ4FKQ5XR
```

In [12]:
```
import requests # import library to read file json


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_ID,
            CLIENT_SECRET,
            VERSION,
            lat,
            lng,
            radius,
            LIMIT)

        # make the GET request
        results = requests.get(url).json()["response"]['groups'][0]['i

        # 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 f
    nearby_venues.columns = ['Neighborhood',
                  'Neighborhood Latitude',
                  'Neighborhood Longitude',
                  'Venue',
                  'Venue Latitude',
```

```
                    'Venue Longitude',
                    'Venue Category']

    return(nearby_venues)
toronto_venues = getNearbyVenues(names=toronto_data['Neighbourhood'],
                                   latitudes=toronto_data['Latitude'],
                                   longitudes=toronto_data['Longitude'
                                  )
print(toronto_venues.shape)
toronto_venues.head()
```

```
Harbourfront,Regent Park
Ryerson,Garden District
St. James Town
Berczy Park
Central Bay Street
Christie
Adelaide,King,Richmond
Harbourfront East,Toronto Islands,Union Station
Design Exchange,Toronto Dominion Centre
Commerce Court,Victoria Hotel
Harbord,University of Toronto
Chinatown,Grange Park,Kensington Market
CN Tower,Bathurst Quay,Island airport,Harbourfront West,King and Spa
dina,Railway Lands,South Niagara
Rosedale
Stn A PO Boxes 25 The Esplanade
Cabbagetown,St. James Town
First Canadian Place,Underground city
Church and Wellesley
(488, 7)
```

Out[12]:

	Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venu Categoı
0	Harbourfront,Regent Park	43.65426	-79.360636	Roselle Desserts	43.653447	-79.362017	Bake
1	Harbourfront,Regent Park	43.65426	-79.360636	Tandem Coffee	43.653559	-79.361809	Coffe Shc
2	Harbourfront,Regent Park	43.65426	-79.360636	Toronto Cooper Koo Family Cherry St YMCA Centre	43.653191	-79.357947	Gym Fitnes Cent
3	Harbourfront,Regent Park	43.65426	-79.360636	Body Blitz Spa East	43.654735	-79.359874	Sp
4	Harbourfront,Regent Park	43.65426	-79.360636	Morning Glory Cafe	43.653947	-79.361149	Breakfa Spı

In [13]:
```
toronto_venues.groupby('Neighborhood').count()
print('There are {} uniques categories.'.format(len(toronto_venues['Ve
```

```
There are 147 uniques categories.
```

In [14]:
```
# one hot encoding
toronto_onehot = pd.get_dummies(toronto_venues[['Venue Category']], pr

# 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.col
toronto_onehot = toronto_onehot[fixed_columns]

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

Out[14]:

	Neighborhood	Yoga Studio	Airport	Airport Food Court	Airport Gate	Airport Lounge	Airport Service	Airport Terminal
0	Adelaide,King,Richmond	0.0	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
1	Berczy Park	0.0	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
2	CN Tower,Bathurst Quay,Island airport,Harbourf...	0.0	0.058824	0.058824	0.058824	0.117647	0.176471	0.117647
3	Cabbagetown,St. James Town	0.0	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
4	Central Bay Street	0.0	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000

5 rows × 147 columns

In [15]:
```
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 [16]:
```
num_top_venues = 10
import numpy as np
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, indicato
    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['Neighbo

for ind in np.arange(toronto_grouped.shape[0]):
    neighborhoods_venues_sorted.iloc[ind, 1:] = return_most_common_ven

neighborhoods_venues_sorted.head()
```

Out[16]:

	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
0	Adelaide,King,Richmond	Steakhouse	Hotel	Café	Coffee Shop	Pizza Place	Asian Restaurant
1	Berczy Park	Café	Cocktail Bar	Coffee Shop	Beer Bar	Seafood Restaurant	Farmers Market
2	CN Tower,Bathurst Quay,Island airport,Harbourf...	Airport Service	Airport Lounge	Airport Terminal	Plane	Coffee Shop	Boutique
3	Cabbagetown,St. James Town	Coffee Shop	Italian Restaurant	Restaurant	Café	Bakery	Japanese Restaurant
4	Central Bay Street	Coffee Shop	Spa	Italian Restaurant	Bubble Tea Shop	Sushi Restaurant	Bar

In [17]:
```
# set number of clusters
from sklearn.cluster import KMeans
kclusters = 5

toronto_grouped_clustering = toronto_grouped.drop('Neighborhood', 1)

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

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

Out[17]:
```
array([0, 1, 2, 0, 0, 1, 3, 1, 0, 0], dtype=int32)
```

In [18]:

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

toronto_merged = toronto_data

# merge toronto_grouped with toronto_data to add latitude/longitude fo
toronto_merged = toronto_merged.join(neighborhoods_venues_sorted.set_i

toronto_merged.head() # check the last columns!
```

Out[18]:

	Postcode	Borough	Neighbourhood	Latitude	Longitude	Cluster Labels	1st Most Common Venue	2nd M Comm Ve
0	M5A	Downtown Toronto	Harbourfront,Regent Park	43.654260	-79.360636	0	Coffee Shop	Bal
1	M5B	Downtown Toronto	Ryerson,Garden District	43.657162	-79.378937	1	Café	Clotl S
2	M5C	Downtown Toronto	St. James Town	43.651494	-79.375418	0	Coffee Shop	Gastro
3	M5E	Downtown Toronto	Berczy Park	43.644771	-79.373306	1	Café	Cocl
4	M5G	Downtown Toronto	Central Bay Street	43.657952	-79.387383	0	Coffee Shop	.

In [19]:
```
# create map
import matplotlib.cm as cm
import matplotlib.colors as colors
map_clusters = folium.Map(location=[latitude, longitude], zoom_start=1

# 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_
    label = folium.Popup(str(poi) + ' Cluster ' + str(cluster), parse_
    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[19]:

In [20]:
```
toronto_merged.loc[toronto_merged['Cluster Labels'] == 0, toronto_merg
```

Out[20]:

	Borough	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
0	Downtown Toronto	0	Coffee Shop	Bakery	Park	Gym / Fitness Center	Breakfast Spot	Mexican Restaurant
2	Downtown Toronto	0	Coffee Shop	Gastropub	Italian Restaurant	Restaurant	Hotel	Japanese Restaurant
4	Downtown Toronto	0	Coffee Shop	Spa	Italian Restaurant	Bubble Tea Shop	Sushi Restaurant	Bar
6	Downtown Toronto	0	Steakhouse	Hotel	Café	Coffee Shop	Pizza Place	Asian Restaurant
8	Downtown Toronto	0	Coffee Shop	Restaurant	Deli / Bodega	Café	Steakhouse	Hotel Bar
9	Downtown Toronto	0	Café	Coffee Shop	Restaurant	Gastropub	Deli / Bodega	Gym / Fitness Center
10	Downtown Toronto	0	Café	Bar	Bookstore	Bakery	Japanese Restaurant	Restaurant
15	Downtown Toronto	0	Coffee Shop	Italian Restaurant	Restaurant	Café	Bakery	Japanese Restaurant
16	Downtown Toronto	0	Café	Coffee Shop	Steakhouse	Restaurant	Deli / Bodega	Pizza Place

In [21]:
```
toronto_merged.loc[toronto_merged['Cluster Labels'] == 1, toronto_merg
```

Out[21]:

	Borough	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
1	Downtown Toronto	1	Café	Clothing Store	Steakhouse	Ramen Restaurant	Beer Bar	Japanese Restaurant
3	Downtown Toronto	1	Café	Cocktail Bar	Coffee Shop	Beer Bar	Seafood Restaurant	Farmers Market
7	Downtown Toronto	1	Café	Park	Hotel	Performing Arts Venue	Salad Place	Deli / Bodega
11	Downtown Toronto	1	Café	Vietnamese Restaurant	Vegetarian / Vegan Restaurant	Caribbean Restaurant	Mexican Restaurant	Bakery
14	Downtown Toronto	1	Café	Seafood Restaurant	Beer Bar	Farmers Market	Cocktail Bar	Hotel
17	Downtown Toronto	1	Gay Bar	Park	Juice Bar	Ramen Restaurant	Pub	Bookstore

In [22]:
```
toronto_merged.loc[toronto_merged['Cluster Labels'] == 2, toronto_merg
```

Out[22]:

	Borough	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 Mo Comm Ven
12	Downtown Toronto	2	Airport Service	Airport Lounge	Airport Terminal	Plane	Coffee Shop	Boutique	E

In [23]:
```
toronto_merged.loc[toronto_merged['Cluster Labels'] == 3, toronto_merg
```

Out[23]:

	Borough	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 M Comn Ve
5	Downtown Toronto	3	Grocery Store	Café	Park	Italian Restaurant	Coffee Shop	Nightclub	Restau

In [24]:
```
toronto_merged.loc[toronto_merged['Cluster Labels'] == 4, toronto_merg
```

Out[24]:

	Borough	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 Com V
13	Downtown Toronto	4	Park	Playground	Trail	Building	Wine Bar	Comfort Food Restaurant	Cre

In [25]:
```
toronto_cluster=pd.DataFrame({'label':range(5),'name': ['coffee shop',
```

In [26]:
```
toronto_cluster
```

Out[26]:

	label	name
0	0	coffee shop
1	1	cafe
2	2	Airport Service
3	3	Grocery Store
4	4	Park

Cluster New York

In [27]:
```
!wget -q -O 'newyork_data.json' https://cocl.us/new_york_dataset
print('Data downloaded!')
```

```
/bin/sh: wget: command not found
Data downloaded!
```

In [28]:
```
import requests
url="https://cocl.us/new_york_dataset"
data = requests.get(url).json()
```

In [29]:
```
data
```

Out[29]:

```
{'type': 'FeatureCollection',
 'totalFeatures': 306,
 'features': [{'type': 'Feature',
   'id': 'nyu_2451_34572.1',
   'geometry': {'type': 'Point',
    'coordinates': [-73.84720052054902, 40.89470517661]},
   'geometry_name': 'geom',
   'properties': {'name': 'Wakefield',
    'stacked': 1,
    'annoline1': 'Wakefield',
    'annoline2': None,
    'annoline3': None,
    'annoangle': 0.0,
    'borough': 'Bronx',
    'bbox': [-73.84720052054902,
     40.89470517661,
     -73.84720052054902,
     40.89470517661]}},
  {'type': 'Feature',
```

In [30]:
```
neighborhoods_data = data['features']
neighborhoods_data[0]
```

Out[30]:
```
{'type': 'Feature',
 'id': 'nyu_2451_34572.1',
 'geometry': {'type': 'Point',
  'coordinates': [-73.84720052054902, 40.89470517661]},
 'geometry_name': 'geom',
 'properties': {'name': 'Wakefield',
  'stacked': 1,
  'annoline1': 'Wakefield',
  'annoline2': None,
  'annoline3': None,
  'annoangle': 0.0,
  'borough': 'Bronx',
  'bbox': [-73.84720052054902,
   40.89470517661,
   -73.84720052054902,
   40.89470517661]}}
```

In [31]:
```
column_names = ['Borough', 'Neighborhood', 'Latitude', 'Longitude']

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

In [32]:
```
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_
                                          'Latitude': neighborhood_lat
                                          'Longitude': neighborhood_lo
```

In [33]:
```
neighborhoods.head()
```

Out[33]:

	Borough	Neighborhood	Latitude	Longitude
0	Bronx	Wakefield	40.894705	-73.847201
1	Bronx	Co-op City	40.874294	-73.829939
2	Bronx	Eastchester	40.887556	-73.827806
3	Bronx	Fieldston	40.895437	-73.905643
4	Bronx	Riverdale	40.890834	-73.912585

In [34]:
```
print('The dataframe has {} boroughs and {} neighborhoods.'.format(
        len(neighborhoods['Borough'].unique()),
        neighborhoods.shape[0]
    )
)
```

The dataframe has 5 boroughs and 306 neighborhoods.

In [35]:
```
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
```

The geograpical coordinate of New York City are 40.7127281, -74.0060
152.

In [36]:

```
# 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'], 
    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
```

Out[36]:

In [37]:
```
newyork_venues = getNearbyVenues(names=neighborhoods['Neighborhood'],
                                   latitudes=neighborhoods['Latitude']
                                   longitudes=neighborhoods['Longitude
                                  )
print(newyork_venues.shape)
newyork_venues.head()
```

```
Wakefield
Co-op City
Eastchester
Fieldston
Riverdale
Kingsbridge
Marble Hill
Woodlawn
Norwood
Williamsbridge
Baychester
Pelham Parkway
City Island
Bedford Park
University Heights
Morris Heights
Fordham
East Tremont
West Farms
```

In [38]:
```
print(newyork_venues.shape)
newyork_venues.head()
```

```
(6191, 7)
```

Out[38]:

	Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
0	Wakefield	40.894705	-73.847201	Lollipops Gelato	40.894123	-73.845892	Dessert Shop
1	Wakefield	40.894705	-73.847201	Rite Aid	40.896649	-73.844846	Pharmacy
2	Wakefield	40.894705	-73.847201	Carvel Ice Cream	40.890487	-73.848568	Ice Cream Shop
3	Wakefield	40.894705	-73.847201	Cooler Runnings Jamaican Restaurant Inc	40.898276	-73.850381	Caribbean Restaurant
4	Wakefield	40.894705	-73.847201	Dunkin'	40.890459	-73.849089	Donut Shop

In [39]:
```
# one hot encoding
newyork_onehot = pd.get_dummies(newyork_venues[['Venue Category']], pre

# add neighborhood column back to dataframe
newyork_onehot['Neighborhood'] = newyork_venues['Neighborhood']

# move neighborhood column to the first column
fixed_columns = [newyork_onehot.columns[-1]] + list(newyork_onehot.colu
newyork_onehot = newyork_onehot[fixed_columns]

newyork_onehot.head()
```

Out[39]:

	Yoga Studio	Accessories Store	Adult Boutique	Afghan Restaurant	African Restaurant	Airport Terminal	American Restaurant	Antique Shop	Arc
0	0	0	0	0	0	0	0	0	
1	0	0	0	0	0	0	0	0	
2	0	0	0	0	0	0	0	0	
3	0	0	0	0	0	0	0	0	
4	0	0	0	0	0	0	0	0	

5 rows × 373 columns

In [40]:
```
newyork_onehot.shape
```

Out[40]:
```
(6191, 373)
```

In [41]:
```
newyork_grouped = newyork_onehot.groupby('Neighborhood').mean().reset_
newyork_grouped
```

Out[41]:

	Neighborhood	Yoga Studio	Accessories Store	Adult Boutique	Afghan Restaurant	African Restaurant	Airport Terminal	Ameri Restau
0	Allerton	0.000000	0.000000	0.0	0.000000	0.000000	0.00	0.000
1	Annadale	0.000000	0.000000	0.0	0.000000	0.000000	0.00	0.090
2	Arden Heights	0.000000	0.000000	0.0	0.000000	0.000000	0.00	0.000
3	Arlington	0.000000	0.000000	0.0	0.000000	0.000000	0.00	0.000
4	Arrochar	0.000000	0.000000	0.0	0.000000	0.000000	0.00	0.000
5	Arverne	0.000000	0.000000	0.0	0.000000	0.000000	0.00	0.000
6	Astoria	0.000000	0.000000	0.0	0.000000	0.000000	0.00	0.000
7	Astoria Heights	0.000000	0.000000	0.0	0.000000	0.000000	0.00	0.000
8	Auburndale	0.000000	0.000000	0.0	0.000000	0.000000	0.00	0.055
9	Bath Beach	0.000000	0.000000	0.0	0.000000	0.000000	0.00	0.000

In [42]:
```
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, indicato
    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'] = newyork_grouped['Neighbo

for ind in np.arange(newyork_grouped.shape[0]):
    neighborhoods_venues_sorted.iloc[ind, 1:] = return_most_common_ven

neighborhoods_venues_sorted.head()
```

Out[42]:

	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 M Com Ve
0	Allerton	Pizza Place	Supermarket	Spa	Breakfast Spot	Fast Food Restaurant	Martial Arts Dojo	St
1	Annadale	Bakery	Diner	Sports Bar	Train Station	Sushi Restaurant	Restaurant	F F
2	Arden Heights	Deli / Bodega	Bus Stop	Pharmacy	Coffee Shop	Pizza Place	Home Service	I
3	Arlington	Intersection	Deli / Bodega	Boat or Ferry	Grocery Store	Bus Stop	Women's Store	Fina or L Se
4	Arrochar	Bus Stop	Bagel Shop	Deli / Bodega	Italian Restaurant	Middle Eastern Restaurant	Pizza Place	I 1

In [43]:
```
 #set number of clusters
kclusters = 5

newyork_grouped_clustering = newyork_grouped.drop('Neighborhood', 1)

# run k-means clustering
kmeans = KMeans(n_clusters=kclusters, random_state=0).fit(newyork_grou
kmeans.labels_[0:10]
```

Out[43]: array([3, 3, 0, 0, 0, 3, 3, 3, 3, 3], dtype=int32)

In [44]:
```
# add clustering labels
neighborhoods_venues_sorted.insert(0, 'Cluster Labels', kmeans.labels_

newyork_merged = neighborhoods

# merge toronto_grouped with toronto_data to add latitude/longitude fo
newyork_merged = newyork_merged.join(neighborhoods_venues_sorted.set_i
newyork_merged.head() # check the last columns!
```

Out[44]:

	Borough	Neighborhood	Latitude	Longitude	Cluster Labels	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue
0	Bronx	Wakefield	40.894705	-73.847201	3.0	Food Truck	Pharmacy	Caribbean Restaurant
1	Bronx	Co-op City	40.874294	-73.829939	3.0	Bus Station	Park	Restaurant
2	Bronx	Eastchester	40.887556	-73.827806	1.0	Caribbean Restaurant	Deli / Bodega	Diner
3	Bronx	Fieldston	40.895437	-73.905643	3.0	Bus Station	River	Plaza
4	Bronx	Riverdale	40.890834	-73.912585	3.0	Bus Station	Park	Plaza

In [45]:
```
newyork_merged.dtypes
```

Out[45]:
```
Borough                    object
Neighborhood               object
Latitude                  float64
Longitude                 float64
Cluster Labels            float64
1st Most Common Venue      object
2nd Most Common Venue      object
3rd Most Common Venue      object
4th Most Common Venue      object
5th Most Common Venue      object
6th Most Common Venue      object
7th Most Common Venue      object
8th Most Common Venue      object
9th Most Common Venue      object
10th Most Common Venue     object
dtype: object
```

In [46]:
```
newyork_merged.loc[207,:]
```

Out[46]:
```
Borough                  Staten Island
Neighborhood                Port Ivory
Latitude                       40.6397
Longitude                     -74.1746
Cluster Labels                     NaN
1st Most Common Venue              NaN
2nd Most Common Venue              NaN
3rd Most Common Venue              NaN
4th Most Common Venue              NaN
5th Most Common Venue              NaN
6th Most Common Venue              NaN
7th Most Common Venue              NaN
8th Most Common Venue              NaN
9th Most Common Venue              NaN
10th Most Common Venue             NaN
Name: 207, dtype: object
```

In [47]:
```
for i in range(306):
    if pd.isna(newyork_merged.loc[i,'Cluster Labels']):
        newyork_merged.loc[i,'Cluster Labels']=0
newyork_merged[['Cluster Labels']]=newyork_merged[['Cluster Labels']].
```

In [48]:
```
newyork_merged.dtypes
```

Out[48]:
```
Borough                    object
Neighborhood               object
Latitude                  float64
Longitude                 float64
Cluster Labels              int64
1st Most Common Venue      object
2nd Most Common Venue      object
3rd Most Common Venue      object
4th Most Common Venue      object
5th Most Common Venue      object
6th Most Common Venue      object
7th Most Common Venue      object
8th Most Common Venue      object
9th Most Common Venue      object
10th Most Common Venue     object
dtype: object
```

In [49]:
```
# create map
map_clusters = folium.Map(location=[latitude, longitude], zoom_start=1

# 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(newyork_merged['Latitude'], newyork_m
    label = folium.Popup(str(poi) + ' Cluster ' + str(cluster), parse_h
    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[49]:

In [50]:
```
cluster=newyork_merged.loc[newyork_merged['Cluster Labels'] == 0, newy
name_cluster=cluster['1st Most Common Venue'].value_counts()
name_cluster=pd.DataFrame(name_cluster)
name_cluster.reset_index(inplace=True)
name_0=name_cluster.iloc[0,0]
name_0
```

Out[50]: 'Bus Stop'

In [51]:
```
cluster=newyork_merged.loc[newyork_merged['Cluster Labels'] == 1, newy
name_cluster=cluster['1st Most Common Venue'].value_counts()
name_cluster=pd.DataFrame(name_cluster)
name_cluster.reset_index(inplace=True)
name_1=name_cluster.iloc[0,0]
name_1
```

Out[51]: 'Deli / Bodega'

In [52]:
```
cluster=newyork_merged.loc[newyork_merged['Cluster Labels'] == 2, newy
name_cluster=cluster['1st Most Common Venue'].value_counts()
name_cluster=pd.DataFrame(name_cluster)
name_cluster.reset_index(inplace=True)
name_2=name_cluster.iloc[0,0]
name_2
```

Out[52]: 'Supermarket'

In [53]:
```
cluster=newyork_merged.loc[newyork_merged['Cluster Labels'] == 3, newy
name_cluster=cluster['1st Most Common Venue'].value_counts()
name_cluster=pd.DataFrame(name_cluster)
name_cluster.reset_index(inplace=True)
name_3=name_cluster.iloc[0,0]
name_3
```

Out[53]: 'Pizza Place'

In [54]:
```
cluster=newyork_merged.loc[newyork_merged['Cluster Labels'] == 4, newy
name_cluster=cluster['1st Most Common Venue'].value_counts()
name_cluster=pd.DataFrame(name_cluster)
name_cluster.reset_index(inplace=True)
name_4=name_cluster.iloc[0,0]
name_4
```

Out[54]: 'Park'

In [55]:
```
newyork_cluster=pd.DataFrame({'label':range(5),'name': [name_0,name_1,
```

In [56]:

```
newyork_cluster
```

Out[56]:

	label	name
0	0	Bus Stop
1	1	Deli / Bodega
2	2	Supermarket
3	3	Pizza Place
4	4	Park

In [57]:

```
ls=['Louvre','Bourse','Temple','Hôtel-de-Ville','Panthéon','Luxembourg
   'Palais-Bourbon','Élysée','Opéra','Entrepôt','Popincourt','Reuilly'
   'Gobelins','Observatoire','Vaugirard','Passy','Batignolles-Monceau'
   'Butte-Montmartre','Buttes-Chaumont','Ménilmontant']
paris_data=pd.DataFrame({'Neighborhood':ls})
paris_data.head()
```

Out[57]:

	Neighborhood
0	Louvre
1	Bourse
2	Temple
3	Hôtel-de-Ville
4	Panthéon

In [59]:
```
from geopy.distance import geodesic
ls_latitude=[]
ls_longitude=[]
for i in range(paris_data.shape[0]):
    print(i,end='')
    address = paris_data['Neighborhood'][i]+ ',Paris'

    geolocator = Nominatim(user_agent="ny_explorer")
    location = geolocator.geocode(address)
    Nei_latitude = location.latitude
    Nei_longitude = location.longitude
    ls_latitude.append(Nei_latitude)
    ls_longitude.append(Nei_longitude)
paris_data['Latitude']=ls_latitude
paris_data['Longitude']=ls_longitude
paris_data.head()
```

```
012345678910111213141516171819
```

Out[59]:

	Neighborhood	Latitude	Longitude
0	Louvre	48.861147	2.338028
1	Bourse	48.867687	2.343122
2	Temple	48.862683	2.358681
3	Hôtel-de-Ville	48.856426	2.352528
4	Panthéon	48.846191	2.346079

In [94]:

```
address = 'Paris,FR'

geolocator = Nominatim(user_agent="ny_explorer")
location = geolocator.geocode(address)
latitude = location.latitude
longitude = location.longitude
print('The geograpical coordinate of Paris are {}, {}.'.format(latitud
map_paris = folium.Map(location=[latitude, longitude], zoom_start=11)

# add markers to map
for lat, lng, label in zip(paris_data['Latitude'], paris_data['Longitu
    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_paris)

map_paris
```

```
The geograpical coordinate of Paris are 48.8566101, 2.3514992.
```

Out[94]:

In [62]:
```
paris_venues = getNearbyVenues(names=paris_data['Neighborhood'],
                                   latitudes=paris_data['Latitude'],
                                   longitudes=paris_data['Longitude']
                                  )
print(toronto_venues.shape)
toronto_venues.head()
```

```
Louvre
Bourse
Temple
Hôtel-de-Ville
Panthéon
Luxembourg
Palais-Bourbon
Élysée
Opéra
Entrepôt
Popincourt
Reuilly
Gobelins
Observatoire
Vaugirard
Passy
Batignolles-Monceau
Butte-Montmartre
Buttes-Chaumont
Ménilmontant
(488, 7)
```

Out[62]:

	Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venu Categoi
0	Harbourfront,Regent Park	43.65426	-79.360636	Roselle Desserts	43.653447	-79.362017	Bake
1	Harbourfront,Regent Park	43.65426	-79.360636	Tandem Coffee	43.653559	-79.361809	Coffe Shc
2	Harbourfront,Regent Park	43.65426	-79.360636	Toronto Cooper Koo Family Cherry St YMCA Centre	43.653191	-79.357947	Gym Fitnes Cent
3	Harbourfront,Regent Park	43.65426	-79.360636	Body Blitz Spa East	43.654735	-79.359874	Sp
4	Harbourfront,Regent Park	43.65426	-79.360636	Morning Glory Cafe	43.653947	-79.361149	Breakfa Sp

In [63]:
```
paris_venues.groupby('Neighborhood').count()
print('There are {} uniques categories.'.format(len(paris_venues['Venu
```

There are 152 uniques categories.

In [64]:
```
 #one hot encoding
paris_onehot = pd.get_dummies(paris_venues[['Venue Category']], prefix

# add neighborhood column back to dataframe
paris_onehot['Neighborhood'] = paris_venues['Neighborhood']

# move neighborhood column to the first column
fixed_columns = [paris_onehot.columns[-1]] + list(paris_onehot.columns
paris_onehot = paris_onehot[fixed_columns]

paris_grouped = paris_onehot.groupby('Neighborhood').mean().reset_inde
paris_grouped
```

Out[64]:

	Neighborhood	Afghan Restaurant	African Restaurant	Alsatian Restaurant	Art Gallery	Art Museum	Arts & Crafts Store	As Restau
0	Batignolles-Monceau	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
1	Bourse	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
2	Butte-Montmartre	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
3	Buttes-Chaumont	0.000000	0.033333	0.000000	0.000000	0.000000	0.000000	0.000
4	Entrepôt	0.000000	0.066667	0.000000	0.000000	0.000000	0.000000	0.000
5	Gobelins	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.166
6	Hôtel-de-Ville	0.000000	0.000000	0.033333	0.066667	0.000000	0.000000	0.000
7	Louvre	0.000000	0.000000	0.000000	0.000000	0.033333	0.000000	0.000
8	Luxembourg	0.000000	0.000000	0.000000	0.033333	0.033333	0.000000	0.000
9	Ménilmontant	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
10	Observatoire	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
11	Opéra	0.000000	0.000000	0.000000	0.033333	0.000000	0.000000	0.000
12	Palais-Bourbon	0.000000	0.000000	0.000000	0.000000	0.033333	0.000000	0.000
13	Panthéon	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
14	Passy	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
15	Popincourt	0.033333	0.033333	0.000000	0.000000	0.033333	0.000000	0.000
16	Reuilly	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000
17	Temple	0.000000	0.000000	0.000000	0.066667	0.000000	0.000000	0.000

18	Vaugirard	0.000000	0.000000	0.000000	0.000000	0.000000	0.033333	0.000
19	Élysée	0.000000	0.000000	0.000000	0.000000	0.033333	0.000000	0.000

20 rows × 153 columns

In [65]:
```
num_top_venues = 10
import numpy as np
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, indicato
    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'] = paris_grouped['Neighborh

for ind in np.arange(paris_grouped.shape[0]):
    neighborhoods_venues_sorted.iloc[ind, 1:] = return_most_common_ven

neighborhoods_venues_sorted.head()
```

Out[65]:

	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
0	Batignolles-Monceau	French Restaurant	Pastry Shop	Italian Restaurant	Bar	Restaurant	Gym / Fitness Center
1	Bourse	Cocktail Bar	French Restaurant	Women's Store	Bistro	Italian Restaurant	Souvlaki Shop
2	Butte-Montmartre	French Restaurant	Bar	Vietnamese Restaurant	Restaurant	Gastropub	Mediterranean Restaurant
3	Buttes-Chaumont	French Restaurant	Bar	Beer Bar	Restaurant	Bistro	Italian Restaurant
4	Entrepôt	French Restaurant	Coffee Shop	Bistro	African Restaurant	Mediterranean Restaurant	Café

In [66]:
```
from sklearn.cluster import KMeans
kclusters = 5

paris_grouped_clustering = paris_grouped.drop('Neighborhood', 1)

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

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

Out[66]:
```
array([2, 1, 2, 2, 2, 4, 1, 1, 1, 2], dtype=int32)
```

In [67]:
```
# add clustering labels
neighborhoods_venues_sorted.insert(0, 'Cluster Labels', kmeans.labels_

paris_merged = paris_data

# merge toronto_grouped with toronto_data to add latitude/longitude fo
paris_merged = paris_merged.join(neighborhoods_venues_sorted.set_index

paris_merged.head() # check the last columns!
```

Out[67]:

	Neighborhood	Latitude	Longitude	Cluster Labels	1st Most Common Venue	2nd Most Common Venue	3rd Most Common Venue	4th Most Common Venue
0	Louvre	48.861147	2.338028	1	Plaza	Exhibit	French Restaurant	Café
1	Bourse	48.867687	2.343122	1	Cocktail Bar	French Restaurant	Women's Store	Bistro
2	Temple	48.862683	2.358681	1	Sandwich Place	Burger Joint	Moroccan Restaurant	Hotel
3	Hôtel-de-Ville	48.856426	2.352528	1	Gay Bar	Art Gallery	French Restaurant	Ice Cream Shop
4	Panthéon	48.846191	2.346079	1	Italian Restaurant	Pub	French Restaurant	Plaza

In [68]:

```
import matplotlib.cm as cm
import matplotlib.colors as colors
map_clusters = folium.Map(location=[latitude, longitude], zoom_start=1

# 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(paris_merged['Latitude'], paris_merg
    label = folium.Popup(str(poi) + ' Cluster ' + str(cluster), parse_
    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[68]:

In [71]:
```
cluster=paris_merged.loc[paris_merged['Cluster Labels'] == 0, paris_me
name_cluster=cluster['1st Most Common Venue'].value_counts()
name_cluster=pd.DataFrame(name_cluster)
name_cluster.reset_index(inplace=True)
name_0=name_cluster.iloc[0,0]
name_0
```

Out[71]: 'French Restaurant'

In [72]:
```
cluster
```

Out[72]:

	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 Mos Commo Venu
6	Palais-Bourbon	French Restaurant	Plaza	Hotel	Italian Restaurant	Pedestrian Plaza	Food Truck	Bake
13	Observatoire	French Restaurant	Hotel	Café	Bistro	Sushi Restaurant	Bus Stop	Fast Foc Restaura

In [83]:
```
cluster=paris_merged.loc[paris_merged['Cluster Labels'] == 1, paris_me
name_cluster=cluster['1st Most Common Venue'].value_counts()
name_cluster=pd.DataFrame(name_cluster)
name_cluster.reset_index(inplace=True)
name_1=name_cluster.iloc[0,0]
name_1
```

Out[83]: 'Italian Restaurant'

In [89]:
```
cluster=paris_merged.loc[paris_merged['Cluster Labels'] == 2, paris_me
name_cluster=cluster['1st Most Common Venue'].value_counts()
name_cluster=pd.DataFrame(name_cluster)
name_cluster.reset_index(inplace=True)
name_2=name_cluster.iloc[0,0]
name_2
```

Out[89]: 'French Restaurant'

In [90]:
```
cluster
```

Out[90]:

	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
7	Élysée	Hotel	French Restaurant	Bar	Cocktail Bar	Pedestrian Plaza	Italian Restaurant
9	Entrepôt	French Restaurant	Coffee Shop	Bistro	African Restaurant	Mediterranean Restaurant	Café
10	Popincourt	Wine Bar	Restaurant	French Restaurant	Bar	Afghan Restaurant	Gastropub
16	Batignolles-Monceau	French Restaurant	Pastry Shop	Italian Restaurant	Bar	Restaurant	Gym / Fitness Center
17	Butte-Montmartre	French Restaurant	Bar	Vietnamese Restaurant	Restaurant	Gastropub	Mediterranean Restaurant
18	Buttes-Chaumont	French Restaurant	Bar	Beer Bar	Restaurant	Bistro	Italian Restaurant
19	Ménilmontant	French Restaurant	Sushi Restaurant	Bistro	Bakery	Bar	Bookstore

In [92]:
```
cluster=paris_merged.loc[paris_merged['Cluster Labels'] == 3, paris_me
name_cluster=cluster['1st Most Common Venue'].value_counts()
name_cluster=pd.DataFrame(name_cluster)
name_cluster.reset_index(inplace=True)
name_3=name_cluster.iloc[0,0]
name_3
```

Out[92]: 'Basketball Court'

In [93]:
```
cluster
```

Out[93]:

	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
15	Passy	Basketball Court	Circus	Bike Rental / Bike Share	Lake	Pool	Women's Store	Electronics Store

In [86]:
```
cluster=paris_merged.loc[paris_merged['Cluster Labels'] == 4, paris_me
name_cluster=cluster['1st Most Common Venue'].value_counts()
name_cluster=pd.DataFrame(name_cluster)
name_cluster.reset_index(inplace=True)
name_4=name_cluster.iloc[0,0]
name_4
```

Out[86]: 'Vietnamese Restaurant'

In [87]:
```
paris_cluster=pd.DataFrame({'label':range(5),'name': [name_0,name_1,nar
```

Result and discussion

In [88]:
```
paris_cluster
```

Out[88]:

	label	name
0	0	French Restaurant
1	1	Italian Restaurant
2	2	French Restaurant
3	3	Basketball Court
4	4	Vietnamese Restaurant

In [79]:
```
newyork_cluster
```

Out[79]:

	label	name
0	0	Bus Stop
1	1	Deli / Bodega
2	2	Supermarket
3	3	Pizza Place
4	4	Park

In [80]:
```
toronto_cluster
```

Out[80]:

	label	name
0	0	coffee shop
1	1	cafe
2	2	Airport Service
3	3	Grocery Store
4	4	Park

We can see there are four clusters of restaurants in Paris while only two clusters for restaurants in New York and Toronto. Further, both of NewYork and Toronto have the 'Park' cluster and one cluster for buying food: supermarket and grocery store, also one cluster for transport: airport service and bus stop, and two cluster for favourite food: Pizza place, Deli/Bodega and coffee,cafe. That means New York city is similar to Toronto. Our client should choose Toronto to work.

Conclusion

This project is to cluster the neighborhoods of cities based on the popular venues. By using foursquare and google maps API, we determined the popular venues of each neighborhoods. Clustering is based on the information of the popular venues. The name of each cluster is assigned by the most common venues. Then we had the table of cluster with name for three cities: Paris, New York, Toronto. Then by comparing the groups of 5 clusters of every cities, we verify that which city is more similarity to New York.