Capstone Project - The Battle of the Neighborhoods (Week 4 & 5)

IBM Data Science Course on Coursera

Opening a new Hotel in Toronto or New York? A report for hotel chains for entering in a new market.

Table of contents

1. Introduction & Business Problem

In this project I am goint to explore the City of Toronto and New York to find the best place for opening a new Hotel. Therefore, the different neighborhoods of both cities have to be analyzed and criteria for the optimal place have to be found in this project.

Of course there are already many hotels in Toronto and New York. Thus, the key question is to find a district where is a low density of hotels and it should be near the city centre or near tourist attractions.

This project and analysis is intended for entrepreneurs of hotels who a searching for a location for their new hotel. The analysis is made with the power of data science to examine the neighborhoods of the city. The findings will be shown in a report and also summarized in a presentation.

2. Data

For solving the problem and analyzing the neighborhoods in both cities data about the following factors are needed:

- the neighborhoods from both cities
- the coordinates from theses neighborhoods (centre)
- number of hotels in each neighborhood

These data will be aquired from the following platforms:

- Wikipedia (Neighborhoods)
- Geolocator library (coordinates)
- Json Datafile from New York
- Foursquare (information about hotels in each city)

2.1 Data Sources and import

First the neccesary libraries will be imported.

```
from bs4 import BeautifulSoup

import numpy as np

import pandas as pd
pd.set_option('display.max_columns', None)
pd.set_option('display.max_rows', None)

import json

from geopy.geocoders import Nominatim

import requests
from pandas.io.json import json_normalize


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

print('Libraries imported.')
```

2.2 Toronto

Then the URL is defined from which the table will be downloaded. For this project the wikipedia page with the postal codes of Toronto is used.

```
# Define the page and download
data_source = requests.get('https://en.wikipedia.org/wiki/List_of_postal_codes_of_Canada:_M').text
soup = BeautifulSoup(data_source, 'html.parser')

# Find the table on the page
table = soup.find('table',{'class':'wikitable sortable'})
table_rows = table.find_all('tr')

# Fill the table
data=[]
for row in table_rows:
    data.append([t.text.strip() for t in row.find_all('td')])

# Build the dataframe
df_TorPostCodes=pd.DataFrame(data, columns=['PostalCode', 'Borough', 'Neighborhood'])

df_TorPostCodes.head(10)
```

Then the table is transformed into a dataframe with the use of pandas.

	PostalCode	Borough	Neighborhood
0	None	None	None
1	M1A	Not assigned	
2	M2A	Not assigned	
3	M3A	North York	Parkwoods
4	M4A	North York	Victoria Village
5	M5A	Downtown Toronto	Regent Park / Harbourfront
6	M6A	North York	Lawrence Manor / Lawrence Heights
7	M7A	Downtown Toronto	Queen's Park / Ontario Provincial Government
8	M8A	Not assigned	
9	M9A	Etobicoke	Islington Avenue

Then the coordinates are imported.

```
df_TorCoordinat = pd.read_csv('https://cocl.us/Geospatial_data/Geospatial_Coordinates.csv')
df_TorCoordinat=df_TorCoordinat.rename(columns={'Postal Code':'PostalCode'})
df_Toronto=pd.merge(df_TorPostCodes,df_TorCoordinat, on='PostalCode')
df_Toronto.head(12)
```

]:

	PostalCode	Borough	Neighborhood	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	Regent Park / Harbourfront	43.654260	-79.360636
3	M6A	North York	Lawrence Manor / Lawrence Heights	43.718518	-79.464763
4	M7A	Downtown Toronto	Queen's Park / Ontario Provincial Government	43.662301	-79.389494
5	M9A	Etobicoke	Islington Avenue	43.667856	-79.532242
6	M1B	Scarborough	Malvern / Rouge	43.806686	-79.194353
7	M3B	North York	Don Mills	43.745906	-79.352188
8	M4B	East York	Parkview Hill / Woodbine Gardens	43.706397	-79.309937
9	M5B	Downtown Toronto	Garden District / Ryerson	43.657162	-79.378937
10	M6B	North York	Glencairn	43.709577	-79.445073
11	M9B	Etobicoke	West Deane Park / Princess Gardens / Martin Gr...	43.650943	-79.554724

Data wrangling and cleaning

Before using the data it has to be cleaned and to be brought in the right style. Rows with not defined data (e.g. "None", "Not assigned") are deleted, rows with the same postal codes are summed together and the format is adapted (e.g. from "/" to ",").

```
df_Toronto=df_Toronto.dropna(0)
df_Toronto=df_Toronto[df_Toronto['Borough'] != "Not assigned"]
df_Toronto['Neighborhood']= df_Toronto['Neighborhood'].str.replace(' /',',')

df_Toronto.head(10)
```

5]:

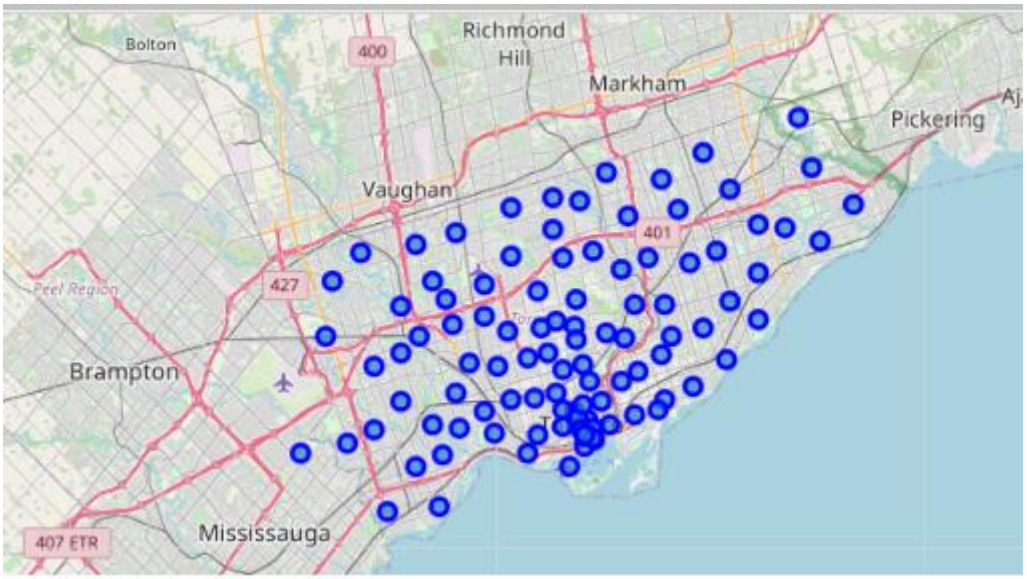
	PostalCode	Borough	Neighborhood	Latitude	Longitude
0	M3A	North York	Parkwoods	43.753259	-79.329656
1	M4A	North York	Victoria Village	43.725882	-79.315572
2	M5A	Downtown Toronto	Regent Park, Harbourfront	43.654260	-79.360636
3	M6A	North York	Lawrence Manor, Lawrence Heights	43.718518	-79.464763
4	M7A	Downtown Toronto	Queen's Park, Ontario Provincial Government	43.662301	-79.389494
5	M9A	Etobicoke	Islington Avenue	43.667856	-79.532242
6	M1B	Scarborough	Malvern, Rouge	43.806686	-79.194353
7	M3B	North York	Don Mills	43.745906	-79.352188
8	M4B	East York	Parkview Hill, Woodbine Gardens	43.706397	-79.309937
9	M5B	Downtown Toronto	Garden District, Ryerson	43.657162	-79.378937

Here are the neighborhoods form Toronto visualized in a map:

```
address_TO = 'Toronto, CA'

geolocator = Nominatim(user_agent="TO_explorer")
location_TO = geolocator.geocode(address_TO)
latitude_TO = location_TO.latitude
longitude_TO = location_TO.longitude
print('The geograpical coordinate of Toronto are {}, {}.'.format(latitude_TO, longitude_TO))
```

```
The geograpical coordinate of Toronto are 43.6534817, -79.3839347.
```

```
map_toronto = folium.Map(location=[latitude_TO, longitude_TO], zoom_start=10)

for lat, lng, borough, neighborhood in zip(df_Toronto['Latitude'], df_Toronto['Longitude'], df_Toronto['Borough'], df_Toronto
    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
```



2.3 New York

Data import from a json Datafile:

New York

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

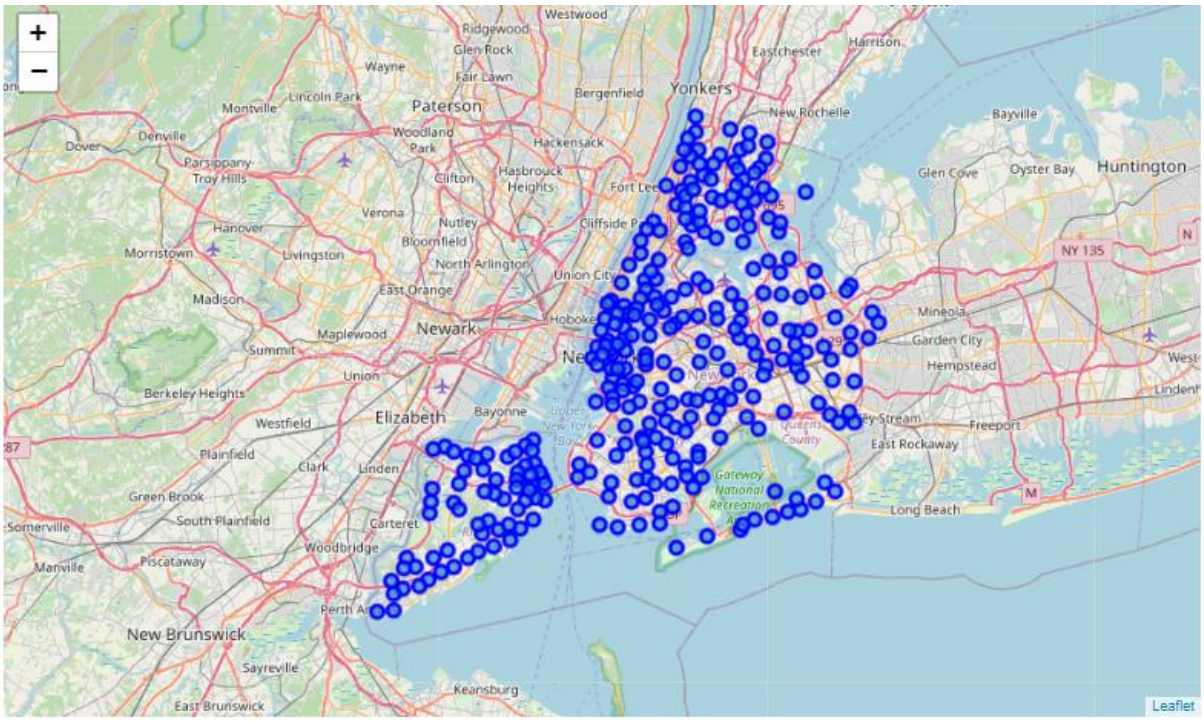
```
Data downloaded!
```

```
with open('newyork_data.json') as json_data:
    newyork_data = json.load(json_data)

neighborhoods_data = newyork_data['features']

column_names = ['Borough', 'Neighborhood', 'Latitude', 'Longitude']

df_NewYork = 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]

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

df_NewYork.head(10)
```

	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
5	Bronx	Kingsbridge	40.881687	-73.902818
6	Manhattan	Marble Hill	40.876551	-73.910660
7	Bronx	Woodlawn	40.898273	-73.867315
8	Bronx	Norwood	40.877224	-73.879391
9	Bronx	Williamsbridge	40.881039	-73.857446

Map the data from NewYork and the neighborhoods:

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

geolocator = Nominatim(user_agent="NY_explorer")
location_NY = geolocator.geocode(address_NY)
latitude_NY = location_NY.latitude
longitude_NY = location_NY.longitude
print('The geograpical coordinate of New York City are {}, {}.'.format(latitude_NY, longitude_NY))

map_newyork = folium.Map(location=[latitude_NY, longitude_NY], zoom_start=10)

for lat, lng, borough, neighborhood in zip(df_NewYork['Latitude'], df_NewYork['Longitude'], df_NewYork['Borough'], df_NewYork
    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
```


Westwood
Ridgewood
Glen Rock
Wayne
Fair Lawn
Eastchester
Harrison
Yonkers
Bergenfield
Lincoln Park
Montville
Paterson
New Rochelle
Bayville
Denville
Dover
Woodland Park
Hackensack
Parsippany-Troy Hills
Clifton
Hasbrouck Heights
Glen Cove
Oyster Bay
Huntington
Verona
Nutley
Cliffside P
Hanover
Bloomfield
Morristown
Livingston
North Arlington
NY 135
Union City
East Orange
Madison
Hoboke
Mineola
Maplewood
Newark
Garden City
Summit
Hempstead
Union
Berkeley Heights
Elizabeth
Bayonne
Westfield
Freeport
Plainfield
Clark
Linden
East Rockaway
Green Brook
Long Beach
Somerville
South Plainfield
Carteret
Woodbridge
Manville
Piscataway
New Brunswick
Sayreville
Keansburg
East Brunswick
Leaflet

3. Methodology

In this project the density of hotels in the neighborhoods of Toronto and NewYork is going to be explored. The results should show the number of hotel of each neighborhood and therefore the hotel owner should have a basic for decision making and can decide in which neighborhood the hotel should been opened.

In the first step the data from all neighborhoods in NewYork and Toronto are collected. Here we use the postal code, name, and the coordinates (latitude and longitude. Of course these datasets have to be wrangled and cleaned to be able to start with the analysis.
All the neighborhoods will be shown in a visualization to get a deeper knowledge of the cities.

In the second step the platform foursquare will be used to get data of each neighborhood. Therefore queries are made to the platform and the result with the query-question “HOTEL”. So you get all hotels in each neighborhood.

The final step is to analyse the density of each neighborhood. This will be done with one-hot-encoding to cluster each region depending on the number of hotels in it. So we get an result with the numbers of hotels in each region and the hotel owner can choose the neighborhood with the lowest number of hotels.

4. Analysis & Results

To get all hotels from the neighborhood foursquare is used with the query question “Hotel”. This is done twice to get results for Toronto and New York.

Toronto

```
def getNearbyVenuesTO(names_TO, latitudes_TO, longitudes_TO, query='Hotels', radius=500):

    venues_list=[]
    for name, lat, lng in zip(names_TO, latitudes_TO, longitudes_TO):

        # create the API request URL

        url = 'https://api.foursquare.com/v2/venues/explore?&client_id={}&client_secret={}&v={}&ll={},{}&query={}&radius={}&l
            CLIENT_ID,
            CLIENT_SECRET,
            VERSION,
            lat,
            lng,
            query,
            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)

toronto_venues = getNearbyVenuesTO(names_TO=df_Toronto['Neighborhood'],
                                   latitudes_TO=df_Toronto['Latitude'],
                                   longitudes_TO=df_Toronto['Longitude']
                                   )
toronto_venues.head()
```

	Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
0	Regent Park, Harbourfront	43.654260	-79.360636	Red Lion Inn	43.652245	-79.363126	Hotel
1	Regent Park, Harbourfront	43.654260	-79.360636	Residence & Conference Centre	43.653040	-79.357040	Hotel
2	Regent Park, Harbourfront	43.654260	-79.360636	Mill St. Brew Pub	43.650353	-79.358489	Pub
3	Queen's Park, Ontario Provincial Government	43.662301	-79.389494	Shangri-La Hotel Toronto	43.659494	-79.390224	Hotel
4	Queen's Park, Ontario Provincial Government	43.662301	-79.389494	Canada Suites Toronto	43.659121	-79.385565	Hotel

New York

```
def getNearbyVenuesNY(names_NY, latitudes_NY, longitudes_NY, query='Hotels', radius=500):

    venues_list=[]
    for name, lat, lng in zip(names_NY, latitudes_NY, longitudes_NY):

        # create the API request URL

        url = 'https://api.foursquare.com/v2/venues/explore?&client_id={}&client_secret={}&v={}&ll={},{}&query={}&radius={}&l
            CLIENT_ID,
            CLIENT_SECRET,
            VERSION,
            lat,
            lng,
            query,
            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)

newyork_venues = getNearbyVenuesNY(names_NY=df_NewYork['Neighborhood'],
                                   latitudes_NY=df_NewYork['Latitude'],
                                   longitudes_NY=df_NewYork['Longitude']
                                   )
newyork_venues.head()
```

	Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
0	Kingsbridge	40.881687	-73.902818	Deegan Motel	40.882749	-73.898860	Hotel
1	Marble Hill	40.876551	-73.910660	Boston Harbor Hotel	40.877790	-73.905810	Hotel
2	Marble Hill	40.876551	-73.910660	Nyinns – Extended Stay Hotels Manhattan New York	40.872799	-73.912089	Hotel
3	Woodlawn	40.898273	-73.867315	River Road Motor Inn	40.896439	-73.863477	Motel
4	Fordham	40.860997	-73.896427	Grand Concourse Hotel	40.858195	-73.899070	Motel

With “one-hot-encoding” we get the sum of hotels and can sort to see the neighborhoods with the lowest density of hotels:

Toronto

```
Toronto=toronto_venues.groupby('Neighborhood').count()
Toronto.sort_values('Venue', ascending=True)
```

14]:

Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
Bathurst Manor, Wilson Heights, Downsview North	1	1	1	1	1	1
New Toronto, Mimico South, Humber Bay Shores	1	1	1	1	1	1
India Bazaar, The Beaches West	1	1	1	1	1	1
Dufferin, Dovercourt Village	1	1	1	1	1	1
Downsview	1	1	1	1	1	1
High Park, The Junction South	1	1	1	1	1	1
Christie	1	1	1	1	1	1
Cliffside, Cliffcrest, Scarborough Village West	2	2	2	2	2	2
University of Toronto, Harbord	2	2	2	2	2	2

New York

```
NewYork=newyork_venues.groupby('Neighborhood').count()
NewYork=NewYork.sort_values('Venue', ascending=True)
```

Neighborhood	Neighborhood Latitude	Neighborhood Longitude	Venue	Venue Latitude	Venue Longitude	Venue Category
Allerton	1	1	1	1	1	1
Lefrak City	1	1	1	1	1	1
Kingsbridge Heights	1	1	1	1	1	1
Kingsbridge	1	1	1	1	1	1
Kew Gardens	1	1	1	1	1	1
Kensington	1	1	1	1	1	1
Jamaica Hills	1	1	1	1	1	1
Jamaica Center	1	1	1	1	1	1
Lindenwood	1	1	1	1	1	1
Woodlawn	1	1	1	1	1	1
Homecrest	1	1	1	1	1	1
Gowanus	1	1	1	1	1	1
Georgetown	1	1	1	1	1	1
Fresh Meadows	1	1	1	1	1	1
Fox Hills	1	1	1	1	1	1
Fort Greene	1	1	1	1	1	1
Flatbush	1	1	1	1	1	1
Howard Beach	1	1	1	1	1	1
Mariner's Harbor	1	1	1	1	1	1
Melrose	1	1	1	1	1	1
Midwood	1	1	1	1	1	1
Westerleigh	1	1	1	1	1	1
Washington Heights	1	1	1	1	1	1
Tompkinsville	1	1	1	1	1	1
Sunnyside Gardens	1	1	1	1	1	1
Sheepshead Bay	1	1	1	1	1	1
Rossville	1	1	1	1	1	1
Roosevelt Island	1	1	1	1	1	1
Ravenswood	1	1	1	1	1	1

Result

As you can see in the list above the hotel owner can choose between many neighborhoods. In Toronto are 7 neighborhoods with just one hotel in it. In New York are many more regions with just one hotel in it.

5. Discussion

The results show that there are many neighborhoods for opening a new hotel. In these neighborhoods the density of hotels is very low and so there is no great competition. So this is a good basis for starting up a hotel.

In further research there can be analyzed which neighborhood is more touristic what is also a very important part for opening a new hotel

6. Conclusion

The purpose of this project was to help a hotel entrepreneur where to open a new hotel in the city of Toronto or New York. Therefore data from both cities were collected, cleaned and structured. This datasets were used to make queries from foursquare which brought the information about hotels. After clustering the dataset we get the result for this project and now can choose between the possible neighborhoods.