# PizzaCapestoneProject

July 30, 2020

[1]:
```
import numpy as np # library to handle data in a vectorized manner

import pandas as pd # library for data analsysis
pd.set_option('display.max_columns', None)
pd.set_option('display.max_rows', None)
import requests # library to handle requests
from pandas.io.json import json_normalize # tranform JSON file into a pandas␣
 ↪dataframe
import folium # map rendering library

print('Libraries imported.')
```

```
Libraries imported.
```

[2]:
```
CLIENT_ID = 'R01LINGO2WC45KLRLKT3ZHU2QENAO2IPRK2N2ELOHRNK4P3K' # your␣
 ↪Foursquare ID
CLIENT_SECRET = '4JT1TWRMXMPLX5IOKNBAFU3L3ARXK4D5JJDPFK1CLRZM2ZVW' # your␣
 ↪Foursquare Secret
VERSION = '20180605' # Foursquare API version

print('Your credentails:')
print('CLIENT_ID: ' + CLIENT_ID)
print('CLIENT_SECRET:' + CLIENT_SECRET)
```

```
Your credentails:
CLIENT_ID: R01LINGO2WC45KLRLKT3ZHU2QENAO2IPRK2N2ELOHRNK4P3K
CLIENT_SECRET:4JT1TWRMXMPLX5IOKNBAFU3L3ARXK4D5JJDPFK1CLRZM2ZVW
```

[18]:
```
LIMIT = 500 # Maximum is 100
cities = ["Chennai, India", 'Mumbai, India', 'Delhi, India', 'Bangalore,␣
 ↪India','Pune, India']
results = {}
for city in cities:
    url = 'https://api.foursquare.com/v2/venues/explore?
 ↪&client_id={}&client_secret={}&v={}&near={}&limit={}&categoryId={}'.format(
        CLIENT_ID,
        CLIENT_SECRET,
```

```
        VERSION,
        city,
        LIMIT,
        "4bf58dd8d48988d1ca941735") # PIZZA PLACE CATEGORY ID
    results[city] = requests.get(url).json()
```

[19]:
```
df_venues={}
for city in cities:
    venues = json_normalize(results[city]['response']['groups'][0]['items'])
    df_venues[city] = venues[['venue.name', 'venue.location.address', 'venue.
 ↪location.lat', 'venue.location.lng']]
    df_venues[city].columns = ['Name', 'Address', 'Lat', 'Lng']
```

```
/home/jupyterlab/conda/envs/python/lib/python3.6/site-
packages/ipykernel_launcher.py:3: FutureWarning: pandas.io.json.json_normalize
is deprecated, use pandas.json_normalize instead
  This is separate from the ipykernel package so we can avoid doing imports
until
```

[20]:
```
maps = {}
for city in cities:
    city_lat = np.
 ↪mean([results[city]['response']['geocode']['geometry']['bounds']['ne']['lat'],
                          ␣
 ↪results[city]['response']['geocode']['geometry']['bounds']['sw']['lat']])
    city_lng = np.
 ↪mean([results[city]['response']['geocode']['geometry']['bounds']['ne']['lng'],
                          ␣
 ↪results[city]['response']['geocode']['geometry']['bounds']['sw']['lng']])
    maps[city] = folium.Map(location=[city_lat, city_lng], zoom_start=11)

    # add markers to map
    for lat, lng, label in zip(df_venues[city]['Lat'], df_venues[city]['Lng'],␣
 ↪df_venues[city]['Name']):
        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(maps[city])
    print(f"Total number of pizza places in {city} = ",␣
 ↪results[city]['response']['totalResults'])
    print("Showing Top 100")
```

```
Total number of pizza places in Chennai, India =  67
Showing Top 100
Total number of pizza places in Mumbai, India =  124
Showing Top 100
Total number of pizza places in Delhi, India =  35
Showing Top 100
Total number of pizza places in Bangalore, India =  118
Showing Top 100
Total number of pizza places in Pune, India =  85
Showing Top 100
```

[8]:
```
maps[cities[0]]
```

[8]:
```
<folium.folium.Map at 0x7f7dd25abac8>
```

[9]:
```
maps[cities[1]]
```

[9]:
```
<folium.folium.Map at 0x7f7dd25abda0>
```

[10]:
```
maps[cities[2]]
```

[10]:
```
<folium.folium.Map at 0x7f7dd2479438>
```

[11]:
```
maps[cities[3]]
```

[11]:
```
<folium.folium.Map at 0x7f7dd2373748>
```

[21]:
```
maps[cities[4]]
```

[21]:
```
<folium.folium.Map at 0x7f7dcbc5b128>
```

[22]:
```
maps = {}
for city in cities:
    city_lat = np.
↪mean([results[city]['response']['geocode']['geometry']['bounds']['ne']['lat'],
                        ␣
↪results[city]['response']['geocode']['geometry']['bounds']['sw']['lat']])
    city_lng = np.
↪mean([results[city]['response']['geocode']['geometry']['bounds']['ne']['lng'],
                        ␣
↪results[city]['response']['geocode']['geometry']['bounds']['sw']['lng']])
    maps[city] = folium.Map(location=[city_lat, city_lng], zoom_start=11)
    venues_mean_coor = [df_venues[city]['Lat'].mean(), df_venues[city]['Lng'].
↪mean()]
    # add markers to map
    for lat, lng, label in zip(df_venues[city]['Lat'], df_venues[city]['Lng'],␣
↪df_venues[city]['Name']):
```

```
        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(maps[city])
        folium.PolyLine([venues_mean_coor, [lat, lng]], color="green", weight=1.
↪5, opacity=0.5).add_to(maps[city])

    label = folium.Popup("Mean Co-ordinate", parse_html=True)
    folium.CircleMarker(
        venues_mean_coor,
        radius=10,
        popup=label,
        color='green',
        fill=True,
        fill_color='#3186cc',
        fill_opacity=0.7,
        parse_html=False).add_to(maps[city])

    print(city)
    print("Mean Distance from Mean coordinates")
    print(np.mean(np.apply_along_axis(lambda x: np.linalg.norm(x -␣
↪venues_mean_coor),1,df_venues[city][['Lat','Lng']].values)))
```

```
Chennai, India
Mean Distance from Mean coordinates
0.05542567149127943
Mumbai, India
Mean Distance from Mean coordinates
0.0824020507226477
Delhi, India
Mean Distance from Mean coordinates
0.08947422146104712
Bangalore, India
Mean Distance from Mean coordinates
0.0561448029555672
Pune, India
Mean Distance from Mean coordinates
0.05624537605431966
```

[14]:
```
maps[cities[0]]
```

[14]:
```
<folium.folium.Map at 0x7f7dd07d9278>
```

[15]:
```
maps[cities[1]]
```

[15]:
```
<folium.folium.Map at 0x7f7dd06d8f60>
```

[16]:
```
maps[cities[2]]
```

[16]:
```
<folium.folium.Map at 0x7f7dd0531a58>
```

[17]:
```
maps[cities[3]]
```

[17]:
```
<folium.folium.Map at 0x7f7dd04d21d0>
```

[23]:
```
maps[cities[4]]
```

[23]:
```
<folium.folium.Map at 0x7f7dcb543b38>
```