Importing the necessary libraries

In [113]:

```
import json, requests
import pandas as pd
import numpy as np
import folium
import geopy.distance
from bs4 import BeautifulSoup
import lxml.html as lh
```

Using the Foursquare API to find coffee shops in Bengaluru

In [17]:

```
url = 'https://api.foursquare.com/v2/venues/explore'

params = dict(
  client_id='I30QFOWBDCH5BSYJEBQGHON3RMAOQWO2HJPD1BKP4RRA1ATH',
  client_secret='S5B3U0GBMGGVPZ4VV3JHXEPY1Y4MGKBYG0C0ZHP1KVATPTZS',
  v='20180323',
  near='Bangalore, Karnataka',
  query='coffee',
  limit=200
)
resp = requests.get(url=url, params=params)
data = json.loads(resp.text)
```

Storing the cafe names and ID in a dataframe

In [ ]:

```
df = pd.DataFrame(columns=["Cafe Name", "ID", "Likes", "Latitude", "Longitude",
"Pincode"])
for i in range(100):
    x = data['response']['groups'][0]['items'][i]['venue']['name']
    y = data['response']['groups'][0]['items'][i]['venue']['id']
    lat = data['response']['groups'][0]['items'][i]['venue']['location']['lat']
    lng = data['response']['groups'][0]['items'][i]['venue']['location']['lng']
    df = df.append({"Cafe Name":x, "ID":y, "Latitude":lat, "Longitude":lng}, ign
ore_index=True)
df.head()
```

Getting pincode of each cafe

In [ ]:

```
for i in range(100):
    try:
        temp= data["response"]["groups"][0]["items"][i]["venue"]["location"]["po
stalCode"]
        df["Pincode"][i] = temp
    except KeyError:
        df["Pincode"][i] = "NaN"
print(df.head())
```

Getting details about each venue

In [ ]:

```
for i in range(100):
    VENUE_ID = df["ID"][i]
    url = 'https://api.foursquare.com/v2/venues/{0}/likes'.format(VENUE_ID)
    params = dict(
      client_id='I30QFOWBDCH5BSYJEBQGHON3RMAOQWO2HJPD1BKP4RRA1ATH',
      client_secret='S5B3U0GBMGGVPZ4VV3JHXEPY1Y4MGKBYG0C0ZHP1KVATPTZS',
      v='20180323'
    )
    resp = requests.get(url=url, params=params)
    data = json.loads(resp.text)
    likes = data['response']['likes']['count']
    df["Likes"][i] = likes
df.head()
```

There are 40 places with no pincode in the API

In [ ]:

```
df[np.isnan(df["Pincode"])].shape
```

For every cafe with no pincode, I will try to enter values manually

In [ ]:

```
df.to_excel("missingPincodes.xlsx")
```

Reading the pincodes after the manual entry

In [22]:

```
df = pd.read_excel("missingPincodes.xlsx")
df.head()
```

Out[22]:

	Unnamed: 0	Cafe Name	ID	Likes	Latitude	Longitude	Pincode
0	0	Brahmins Coffee Bar	4bf61de3d4cdb713d04984fe	169	12.953983	77.568862	560004
1	1	Truffles Ice & Spice	4c11f49ca9420f47c3a07d51	415	12.933443	77.614265	560095
2	2	Infinitea	4b7965c1f964a52085f72ee3	125	12.987157	77.594835	560052
3	3	The Leela Palace	4ad0af16f964a52012d920e3	215	12.960928	77.648556	560008
4	4	Yogisthan	550acfdd498e25775d60b7e7	46	12.981037	77.638348	560038

Grouping the dataframe by Pincode

In [29]:

```
df = df.drop("Unnamed: 0", axis=1)
```

In [39]:

```
df_grouped = df.groupby(["Pincode"]).sum()["Likes"]
```

In [41]:

```
df_grouped.head()
```

Out[41]:

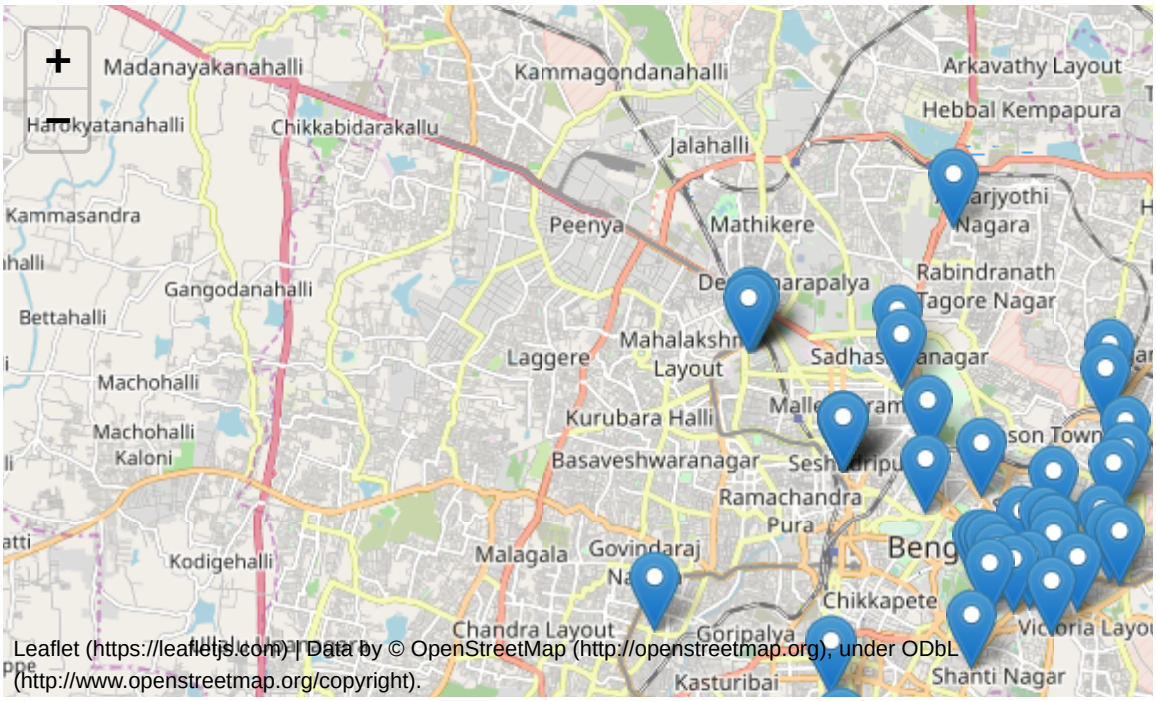
```
Pincode
500011      7
560001    927
560003     75
560004    199
560005     10
Name: Likes, dtype: int64
```

Visualizing the locations

In [176]:

```
m = folium.Map(location=[12.9716, 77.5946], zoom_start=12)
for i in range(100):
    lat = df["Latitude"][i]
    lng = df["Longitude"][i]
    folium.Marker(location=[lat, lng]).add_to(m)
m
```

Out[176]:



In [82]:

```
df_grouped = df_grouped.to_frame(name="None").reset_index()
```

In [90]:

```
df_grouped = df_grouped.rename(columns={"None":"Likes"})
```

In [104]:

```
df_grouped = df_grouped.astype({"Pincode":"str"})
```

In [105]:

```
df_grouped.dtypes
```

Out[105]:

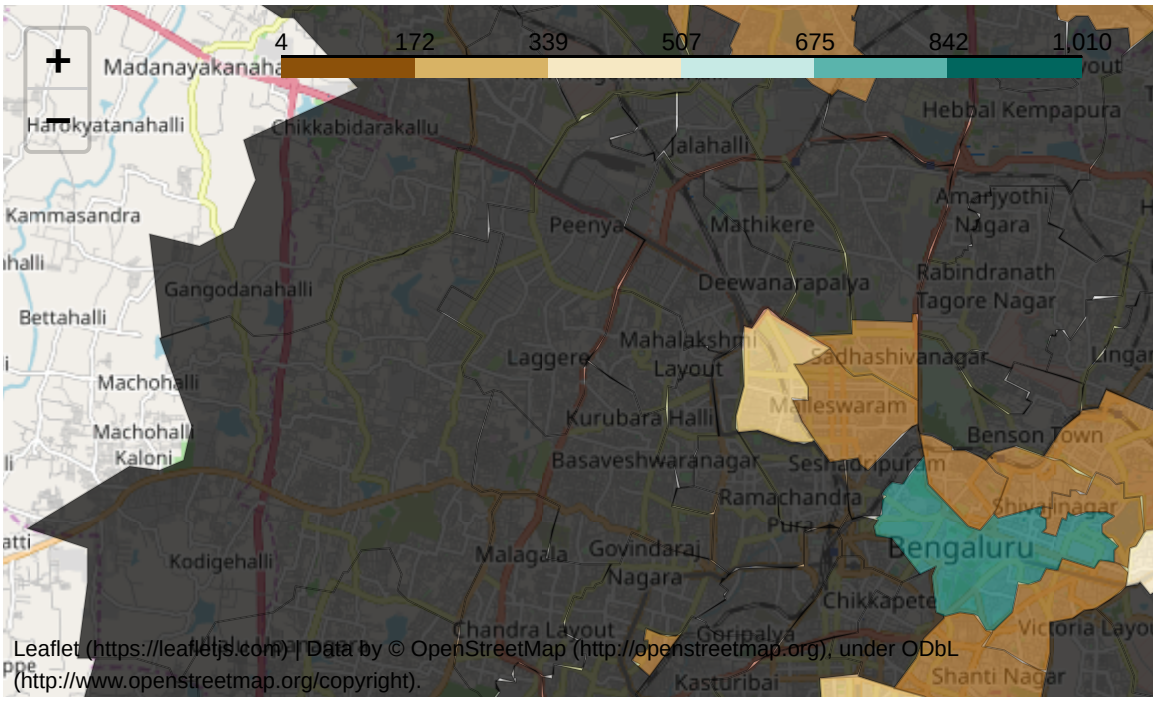
```
Pincode    object
Likes       int64
dtype: object
```

A choropleth map showing the popular coffee areas in Bengaluru

In [111]:

```
state_geo = r"bangalore_pincode.json"
m = folium.Map(location=[12.9716, 77.5946], zoom_start=12)
folium.Choropleth(geo_data=state_geo, name="choropleth",
                  key_on="properties.pincode",
                  data = df_grouped,
                  columns=["Pincode", "Likes"],
                  fill_color="BrBG",
                  fill_opacity=0.7,
                  line_opacity=0.2).add_to(m)
m
```

Out[111]:



Now we have to get the property rates in Bengaluru through web scraping

In [179]:

```
frames = []
for i in range(1, 85):
    url = "https://www.makaan.com/price-trends/property-rates-for-buy-in-bangalo
re?page={}".format(i)
    res = requests.get(url)
    soup = BeautifulSoup(res.content, "html.parser")
    table = soup.find_all("table")
    df = pd.read_html(str(table))[0]
    df.columns = ["Locality", "A", "Average Price", "B", "C", "D"]
    df.drop(["A", "B", "C", "D"], axis=1, inplace=True)
    frames.append(df)
result = pd.concat(frames)
result.head()
```

Out[179]:

	Locality	Average Price
0	HSR Layout	17,829.02 / sqft
1	Koramangala	19,123.45 / sqft
2	Jigani	4,873.24 / sqft
3	Mahadevapura	10,788.91 / sqft
4	Whitefield Hope Farm Junction	11,230.53 / sqft

In [186]:

```
result.reset_index(inplace=True, drop=True)
```

In [191]:

```
result.drop("index", axis=1, inplace=True)
```

In [199]:

```
result
```

Out[199]:

	Locality	Average Price
0	HSR Layout	17,829.02 / sqft
1	Koramangala	19,123.45 / sqft
2	Jigani	4,873.24 / sqft
3	Mahadevapura	10,788.91 / sqft
4	Whitefield Hope Farm Junction	11,230.53 / sqft
...	...	...
4988	Yeshwanthpur Industrial Area	-
4989	Yeshwanthpur Industrial Suburb	-
4990	Yesvantpur Industrial Suburb	-
4991	YK Layout	-
4992	Yogesh Nagar	-

4993 rows × 2 columns

Removing rows with no value for Average Price

In [204]:

```
result = result[result["Average Price"] != "-"]
```

In [206]:

```
result.reset_index(drop=True, inplace=True)
```

In [207]:

```
result
```

Out[207]:

	Locality	Average Price
0	HSR Layout	17,829.02 / sqft
1	Koramangala	19,123.45 / sqft
2	Jigani	4,873.24 / sqft
3	Mahadevapura	10,788.91 / sqft
4	Whitefield Hope Farm Junction	11,230.53 / sqft
...	...	...
862	Varthur Main Road Number 2	5,882.35 / sqft
863	Venkataswamappa Layout Doddabommasandra	5,416.67 / sqft
864	VV Giri Colony	8,785.94 / sqft
865	Yadava Upanagara	8,491.51 / sqft
866	Yelachanayakanapura	3,271.54 / sqft

867 rows × 2 columns

Creating a new row

In [ ]:

```
result["Pincode"] = ""
```

In [223]:

```
result
```

Out[223]:

	Locality	Average Price	Pincode
0	HSR Layout	17,829.02 / sqft	
1	Koramangala	19,123.45 / sqft	
2	Jigani	4,873.24 / sqft	
3	Mahadevapura	10,788.91 / sqft	
4	Whitefield Hope Farm Junction	11,230.53 / sqft	
...	...	...	...
862	Varthur Main Road Number 2	5,882.35 / sqft	
863	Venkataswamappa Layout Doddabommasandra	5,416.67 / sqft	
864	VV Giri Colony	8,785.94 / sqft	
865	Yadava Upanagara	8,491.51 / sqft	
866	Yelachanayakanapura	3,271.54 / sqft	

867 rows × 3 columns

Using the Indian Postal Code API to find Pincode

In [ ]:

```
for i in range(0, 867):
    try:
        url = "https://api.postalpincode.in/postoffice/{}".format(result["Locali
ty"][i])
        res = requests.get(url)
        text = res.json()
        pin = text[0]["PostOffice"][0]["Pincode"]
        result["Pincode"][i] = pin
        print(pin)
    except:
        result["Pincode"][i] = "NaN"
        print("Exception")
```

Removing rows with no Pincode

In [234]:

```
result = result[result["Pincode"] != "NaN"]
```

In [280]:

```
result.reset_index(inplace=True)
```

In [ ]:

```
result["Pincode"] = result["Pincode"].astype(int)
```

In [ ]:

```
result.drop(labels=["index"], axis=1, inplace=True)
```

Removing rows which do not have pincode from Bangalore i.e starting from 56

In [300]:

```
result = result[result["Pincode"] > 560000]
```

In [305]:

```
result = result[result["Pincode"] <= 570000]
```

In [333]:

```
result.reset_index(inplace=True, drop=True)
```

Cleaning the Average Price column

In [ ]:

```
for i in range(result.shape[0]):
    result["Average Price"][i] = result["Average Price"][i][:-7]
```

In [ ]:

```
result["Average Price"][0] = '17,829.02'
result["Average Price"][1] = "19,123.45"
result["Average Price"][2] = "4873.24"
result["Average Price"][3] = "23858.23"
result["Average Price"][4] = "8934.25"
result["Average Price"][5] = "10405.02"
```

In [ ]:

```
for i in range(result.shape[0]):
    result["Average Price"][i] = result["Average Price"][i].replace(",", "")
```

In [ ]:

```
result["Average Price"] = result["Average Price"].astype(float)
```

Grouping the table by Pincode

In [371]:

```
result = result.groupby("Pincode").mean()
```

In [390]:

```
result.reset_index(inplace=True)
```

In [397]:

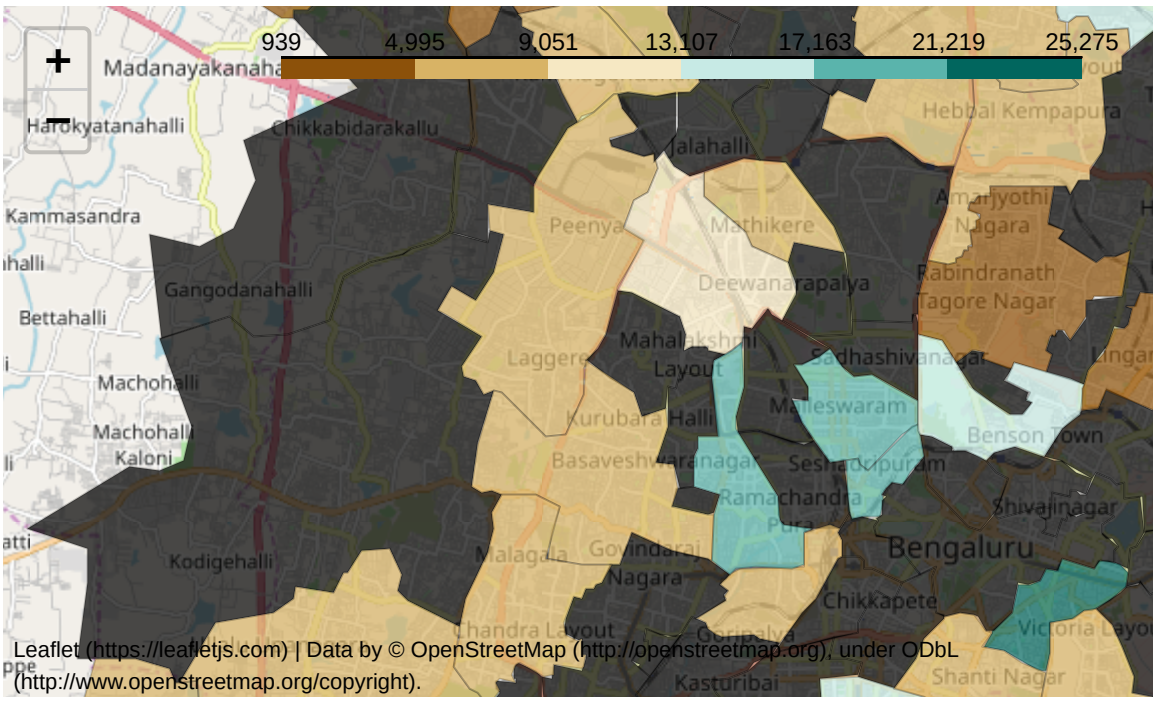
```
result["Pincode"] = result["Pincode"].astype(str)
```

Visualizing the property rates per square feet in Bengaluru

In [398]:

```
state_geo = r"bangalore_pincode.json"
m = folium.Map(location=[12.9716, 77.5946], zoom_start=12)
folium.Choropleth(geo_data=state_geo, name="choropleth",
                  key_on="properties.pincode",
                  data = result,
                  columns=["Pincode", "Average Price"],
                  fill_color="BrBG",
                  fill_opacity=0.7,
                  line_opacity=0.2).add_to(m)
m
```

Out[398]:



In [402]:

```
print(df_grouped.head())
print(result.head())
```

```
  Pincode  Likes
0  500011      7
1  560001    927
2  560003     75
3  560004    199
4  560005     10
  Pincode  Average Price
0  560003       21115.60
1  560004       14581.95
2  560010       17448.02
3  560014        5180.14
4  560015        6383.76
```

Merging the two tables to combine Likes and Average Price

In [405]:

```
finalTable = pd.merge(df_grouped, result, how="inner", on=["Pincode"])
```

In [407]:

```
finalTable.head()
```

Out[407]:

	Pincode	Likes	Average Price
0	560003	75	21115.60
1	560004	199	14581.95
2	560025	104	25131.31
3	560027	16	7809.81
4	560034	158	19123.45

Estimating the average profit in an area using the "Likes" column as a popularity metric

In [411]:

```
finalTable["Average Profit"] = finalTable["Likes"] / finalTable["Average Price"]
```

In [414]:

```
finalTable
```

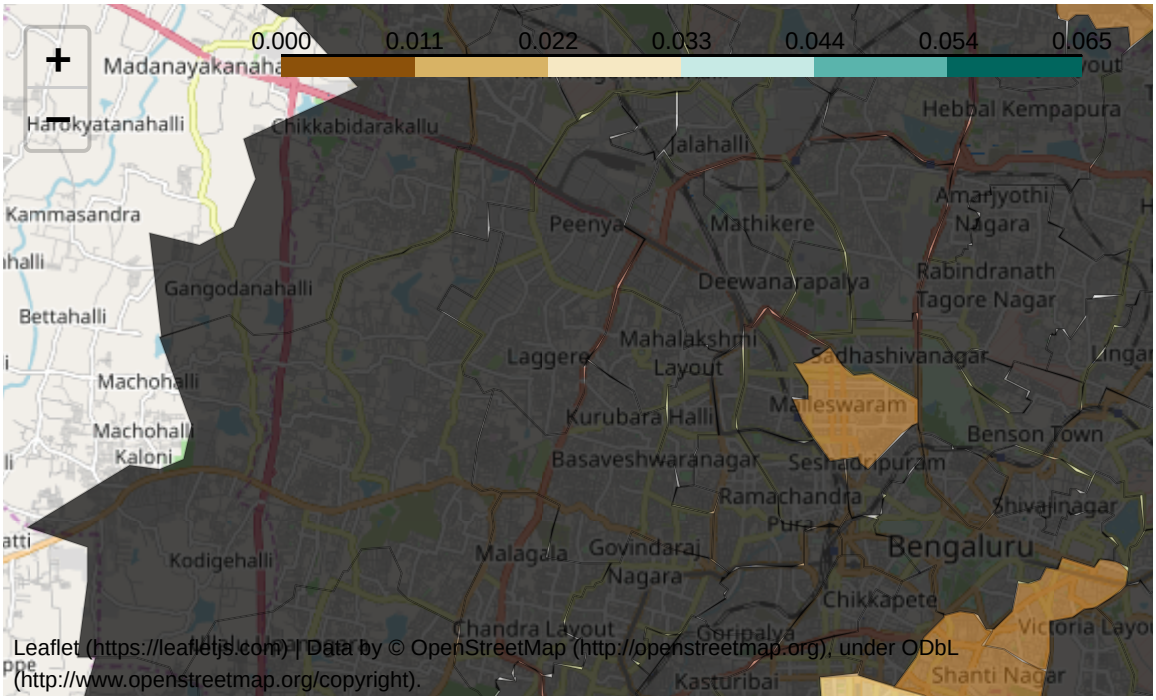
Out[414]:

	Pincode	Likes	Average Price	Average Profit
0	560003	75	21115.600000	0.003552
1	560004	199	14581.950000	0.013647
2	560025	104	25131.310000	0.004138
3	560027	16	7809.810000	0.002049
4	560034	158	19123.450000	0.008262
5	560043	6	25274.635000	0.000237
6	560048	68	1042.190000	0.065247
7	560064	14	14548.600000	0.000962
8	560068	16	6156.380000	0.002599
9	560071	43	939.300000	0.045779
10	560075	10	5304.550000	0.001885
11	560076	17	6407.373333	0.002653
12	560078	104	7448.263333	0.013963
13	560085	6	6343.870000	0.000946
14	560102	22	13024.785000	0.001689
15	560103	9	18134.115000	0.000496

In [415]:

```
state_geo = r"bangalore_pincode.json"
m = folium.Map(location=[12.9716, 77.5946], zoom_start=12)
folium.Choropleth(geo_data=state_geo, name="choropleth",
                  key_on="properties.pincode",
                  data = finalTable,
                  columns=["Pincode", "Average Profit"],
                  fill_color="BrBG",
                  fill_opacity=0.7,
                  line_opacity=0.2).add_to(m)
m
```

Out[415]:



In [421]:

```
topPincodes = finalTable.sort_values(by="Average Profit", ascending=False).head(5)["Pincode"]
```

Following are the top 5 places to open a cafe in Bengaluru

In [423]:

```
topPincodes
```

Out[423]:

```
6     560048
9     560071
12    560078
1     560004
4     560034
Name: Pincode, dtype: object
```