hackathon2

April 19, 2025

[1]:
```
from IPython.display import display, HTML
```

[5]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from geopy.geocoders import Nominatim
from geopy.exc import GeocoderTimedOut
from IPython.display import display, HTML

import folium
```

[6]:
```
data1 = pd.read_csv('zomato_data.csv')
data2 = pd.read_csv('Geographical Coordinates.csv')
data1.head(), data2.head()
```

[6]:
```
(  online_order book_table   rate  votes            rest_type  \
 0          Yes        Yes  4.1/5    775        Casual Dining
 1          Yes         No  4.1/5    787        Casual Dining
 2          Yes         No  3.8/5    918  Cafe, Casual Dining
 3           No         No  3.7/5     88          Quick Bites
 4           No         No  3.8/5    166        Casual Dining

                                        dish_liked  \
 0  Pasta, Lunch Buffet, Masala Papad, Paneer Laja…
 1  Momos, Lunch Buffet, Chocolate Nirvana, Thai G…
 2  Churros, Cannelloni, Minestrone Soup, Hot Choc…
 3                                    Masala Dosa
 4                         Panipuri, Gol Gappe

                        cuisines approx_costfor_two_people listed_intype  \
 0  North Indian, Mughlai, Chinese                      800        Buffet
 1     Chinese, North Indian, Thai                      800        Buffet
 2          Cafe, Mexican, Italian                      800        Buffet
 3      South Indian, North Indian                      300        Buffet
 4        North Indian, Rajasthani                      600        Buffet
```

```
  listed_incity
0  Banashankari
1  Banashankari
2  Banashankari
3  Banashankari
4  Banashankari  ,
       listed_incity   Latitude  Longitude
0       Banashankari  12.939333  77.553982
1  Bannerghatta Road  12.952660  77.605048
2       Basavanagudi  12.941726  77.575502
3          Bellandur  12.925352  77.675941
4       Brigade Road  12.967358  77.606435)
```

[8]:
```
data1['rate'] = data1['rate'].replace('-', np.nan)
data1['rate'] = data1['rate'].str.replace('/5', '').apply(pd.to_numeric,␣
 ↪errors='coerce')
data1['rate'] = data1['rate'].astype(float)

data1['rate'].fillna(data1['rate'].median(), inplace=True)
print(data1['rate'].describe())
print(data1['rate'].isnull().sum())
```

```
count    51717.000000
mean         3.700362
std          0.395391
min          1.800000
25%          3.500000
50%          3.700000
75%          3.900000
max          4.900000
Name: rate, dtype: float64
0
```

[9]:
```
data1['approx_costfor_two_people'] = data1['approx_costfor_two_people'].
 ↪replace({'\,': ''}, regex=True)
data1['approx_costfor_two_people'] = pd.
 ↪to_numeric(data1['approx_costfor_two_people'], errors='coerce')
data1['approx_costfor_two_people'].fillna(data1['approx_costfor_two_people'].
 ↪median(), inplace=True)
```

[10]:
```
data1['dish_liked'].fillna('Not Available', inplace=True)
data1['cuisines'].fillna('Other', inplace=True)
data1['rest_type'].fillna('Unknown', inplace=True)
```

[11]:
```
data1['votes'].fillna(data1['votes'].median(), inplace=True)
```

[12]:
```
data1['online_order'] = data1['online_order'].map({'Yes': 1, 'No': 0})
data1['book_table'] = data1['book_table'].map({'Yes': 1, 'No': 0})
```

[13]:
```
data1['rate'] = data1['rate'].astype(float)
data1['votes'] = data1['votes'].astype(int)
data1['approx_costfor_two_people'] = data1['approx_costfor_two_people'].
 ↪astype(int)
```

[14]:
```
data1.info()

data1.describe()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 51717 entries, 0 to 51716
Data columns (total 10 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   online_order                 51717 non-null  int64
 1   book_table                   51717 non-null  int64
 2   rate                         51717 non-null  float64
 3   votes                        51717 non-null  int32
 4   rest_type                    51717 non-null  object
 5   dish_liked                   51717 non-null  object
 6   cuisines                     51717 non-null  object
 7   approx_costfor_two_people    51717 non-null  int32
 8   listed_intype                51717 non-null  object
 9   listed_incity                51717 non-null  object
dtypes: float64(1), int32(2), int64(2), object(5)
memory usage: 3.6+ MB
```

[14]:

	online_order	book_table	rate	votes	\
count	51717.000000	51717.000000	51717.000000	51717.000000	
mean	0.588665	0.124698	3.700362	283.697527	
std	0.492080	0.330379	0.395391	803.838853	
min	0.000000	0.000000	1.800000	0.000000	
25%	0.000000	0.000000	3.500000	7.000000	
50%	1.000000	0.000000	3.700000	41.000000	
75%	1.000000	0.000000	3.900000	198.000000	
max	1.000000	1.000000	4.900000	16832.000000	

	approx_costfor_two_people
count	51717.000000
mean	554.391689
std	437.563723
min	40.000000
25%	300.000000
50%	400.000000

```
75%                   650.000000
max                  6000.000000
```

[15]:
```
merged_df = pd.merge(data1, data2, on='listed_incity', how='left')

merged_df.head()
```

[15]:
```
   online_order  book_table  rate  votes           rest_type  \
0             1           1   4.1    775      Casual Dining
1             1           0   4.1    787      Casual Dining
2             1           0   3.8    918  Cafe, Casual Dining
3             0           0   3.7     88        Quick Bites
4             0           0   3.8    166      Casual Dining

                                        dish_liked  \
0  Pasta, Lunch Buffet, Masala Papad, Paneer Laja…
1  Momos, Lunch Buffet, Chocolate Nirvana, Thai G…
2  Churros, Cannelloni, Minestrone Soup, Hot Choc…
3                                     Masala Dosa
4                            Panipuri, Gol Gappe

                        cuisines  approx_costfor_two_people listed_intype  \
0  North Indian, Mughlai, Chinese                        800        Buffet
1     Chinese, North Indian, Thai                        800        Buffet
2          Cafe, Mexican, Italian                        800        Buffet
3      South Indian, North Indian                        300        Buffet
4        North Indian, Rajasthani                        600        Buffet

  listed_incity   Latitude  Longitude
0  Banashankari  12.939333  77.553982
1  Banashankari  12.939333  77.553982
2  Banashankari  12.939333  77.553982
3  Banashankari  12.939333  77.553982
4  Banashankari  12.939333  77.553982
```

[17]:
```
merged_df = merged_df.dropna(subset=['Latitude', 'Longitude'])

for idx, row in merged_df.iterrows():
    folium.CircleMarker(
        location=[row['Latitude'], row['Longitude']],
        radius=5,
        color='blue',
        fill=True,
        fill_color='blue',
        fill_opacity=0.6
    ).add_to(bangalore_map)
bangalore_map
```

[17]:
```
<folium.folium.Map at 0x2728cc0e490>
```

[22]:
```
italian_restaurants = merged_df[merged_df['cuisines'].str.contains('Italian',␣
↪case=False, na=False)]

italian_map = folium.Map(location=[12.9716, 77.5946], zoom_start=12)

for idx, row in italian_restaurants.iterrows():
    folium.Marker(
        location=[row['Latitude'], row['Longitude']],
        popup=row['rest_type'] + ' - ' + row['dish_liked'],
        icon=folium.Icon(color='green', icon='cloud')
    ).add_to(italian_map)

italian_map
```

[22]:
```
<folium.folium.Map at 0x272a8130dd0>
```

[ ]: