project

April 19, 2025

[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 time
import folium
```

[7]:
```
# Load datasets
zomato = pd.read_csv("zomato_data.csv")
geo = pd.read_csv("Geographical Coordinates.csv")

# Preview
zomato.head()
geo.head()
```

[7]:
```
        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
```

[15]:
```
# Step-by-step safe rate cleaning
zomato['rate'] = zomato['rate'].replace(['-', 'NEW'], np.nan)  # Step 1:␣
 ↪Replace non-numeric indicators with NaN

zomato['rate'] = zomato['rate'].astype(str)  # Step 2: Force string type to␣
 ↪safely use str.replace

zomato['rate'] = zomato['rate'].str.replace('/5', '', regex=False)  # Step 3:␣
 ↪Remove '/5'

zomato['rate'] = pd.to_numeric(zomato['rate'], errors='coerce')  # Step 4:␣
 ↪Convert to float safely
```

```
zomato['rate'] = zomato['rate'].fillna(zomato['rate'].median())  # Step 5: Fill␣
  ↪missing with median
```

[19]:
```
zomato['approx_costfor_two_people'] = zomato['approx_costfor_two_people'].
  ↪astype(str).str.replace(',', '')
zomato['approx_costfor_two_people'] = pd.
  ↪to_numeric(zomato['approx_costfor_two_people'], errors='coerce')
#zomato['approx_costfor_two_people'].fillna(zomato['approx_costfor_two_people'].
  ↪median(), inplace=True)
zomato['approx_costfor_two_people'] = zomato['approx_costfor_two_people'].
  ↪fillna(zomato['approx_costfor_two_people'].median())
```

[23]:
```
zomato['dish_liked'] = zomato['dish_liked'].fillna("Not Available")
zomato['cuisines'] = zomato['cuisines'].fillna("Other")
zomato['rest_type'] = zomato['rest_type'].fillna("Unknown")
```

[27]:
```
zomato['votes'] = zomato['votes'].fillna(zomato['votes'].median())
```

[29]:
```
zomato['online_order'] = zomato['online_order'].map({'Yes': 1, 'No': 0})
zomato['book_table'] = zomato['book_table'].map({'Yes': 1, 'No': 0})
```

[31]:
```
zomato['votes'] = zomato['votes'].astype(int)
zomato['approx_costfor_two_people'] = zomato['approx_costfor_two_people'].
  ↪astype(int)
zomato['rate'] = zomato['rate'].astype(float)
```

[33]:
```
zomato.info()
zomato.isnull().sum()
zomato.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
```

```
[33]:
       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
```

```
[35]: merged_df = pd.merge(zomato, geo, on='listed_incity', how='left')
      merged_df.head()
```

```
[35]:
   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
```

[39]:

```python
# Base Map
bangalore_map = folium.Map(location=[12.9716, 77.5946], zoom_start=11)

# Add restaurant locations
for index, row in merged_df.iterrows():
    if not np.isnan(row['Latitude']) and not np.isnan(row['Longitude']):
        folium.CircleMarker(
            location=[row['Latitude'], row['Longitude']],
            radius=1,
            color='blue',
            fill=True,
            fill_opacity=0.5
        ).add_to(bangalore_map)

bangalore_map
```

[39]:

```
<folium.folium.Map at 0x25217677830>
```

[38]:

```python
# Filter Italian cuisine
italian_df = merged_df[merged_df['cuisines'].str.contains('Italian',␣
 ↪case=False, na=False)]

# Italian restaurant map
italian_map = folium.Map(location=[12.9716, 77.5946], zoom_start=11)

for index, row in italian_df.iterrows():
    if not np.isnan(row['Latitude']) and not np.isnan(row['Longitude']):
        folium.Marker(
            location=[row['Latitude'], row['Longitude']],
            popup=row['name'] if 'name' in row else '',
            icon=folium.Icon(color='green', icon='cutlery', prefix='fa')
        ).add_to(italian_map)

italian_map
```

[38]:

```
<folium.folium.Map at 0x25217675d90>
```

[ ]:

```python

```