cognifyz2lv1

February 19, 2025

[ ]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

[ ]:
```
data=pd.read_csv('/content/Dataset  (1).csv')
```

[ ]:
```
data.head()
```

[ ]:
```
   Restaurant ID         Restaurant Name  Country Code              City  \
0        6317637        Le Petit Souffle           162       Makati City
1        6304287        Izakaya Kikufuji           162       Makati City
2        6300002  Heat - Edsa Shangri-La           162  Mandaluyong City
3        6318506                    Ooma           162  Mandaluyong City
4        6314302             Sambo Kojin           162  Mandaluyong City

                                             Address  \
0  Third Floor, Century City Mall, Kalayaan Avenu…
1  Little Tokyo, 2277 Chino Roces Avenue, Legaspi…
2  Edsa Shangri-La, 1 Garden Way, Ortigas, Mandal…
3  Third Floor, Mega Fashion Hall, SM Megamall, O…
4  Third Floor, Mega Atrium, SM Megamall, Ortigas…

                                    Locality  \
0   Century City Mall, Poblacion, Makati City
1  Little Tokyo, Legaspi Village, Makati City
2  Edsa Shangri-La, Ortigas, Mandaluyong City
3      SM Megamall, Ortigas, Mandaluyong City
4      SM Megamall, Ortigas, Mandaluyong City

                                    Locality Verbose   Longitude   Latitude  \
0  Century City Mall, Poblacion, Makati City, Mak…  121.027535  14.565443
1  Little Tokyo, Legaspi Village, Makati City, Ma…  121.014101  14.553708
2  Edsa Shangri-La, Ortigas, Mandaluyong City, Ma…  121.056831  14.581404
3  SM Megamall, Ortigas, Mandaluyong City, Mandal…  121.056475  14.585318
4  SM Megamall, Ortigas, Mandaluyong City, Mandal…  121.057508  14.584450
```

```
                          Cuisines  …         Currency Has Table booking  \
0       French, Japanese, Desserts  …  Botswana Pula(P)               Yes
1                         Japanese  …  Botswana Pula(P)               Yes
2  Seafood, Asian, Filipino, Indian  …  Botswana Pula(P)               Yes
3                  Japanese, Sushi  …  Botswana Pula(P)                No
4                 Japanese, Korean  …  Botswana Pula(P)               Yes

  Has Online delivery Is delivering now Switch to order menu Price range  \
0                  No                No                   No           3
1                  No                No                   No           3
2                  No                No                   No           4
3                  No                No                   No           4
4                  No                No                   No           4

   Aggregate rating  Rating color Rating text Votes
0               4.8    Dark Green   Excellent   314
1               4.5    Dark Green   Excellent   591
2               4.4         Green   Very Good   270
3               4.9    Dark Green   Excellent   365
4               4.8    Dark Green   Excellent   229

[5 rows x 21 columns]
```

[ ]:
```python
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9551 entries, 0 to 9550
Data columns (total 21 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   Restaurant ID         9551 non-null   int64
 1   Restaurant Name       9551 non-null   object
 2   Country Code          9551 non-null   int64
 3   City                  9551 non-null   object
 4   Address               9551 non-null   object
 5   Locality              9551 non-null   object
 6   Locality Verbose      9551 non-null   object
 7   Longitude             9551 non-null   float64
 8   Latitude              9551 non-null   float64
 9   Cuisines              9542 non-null   object
 10  Average Cost for two  9551 non-null   int64
 11  Currency              9551 non-null   object
 12  Has Table booking     9551 non-null   object
 13  Has Online delivery   9551 non-null   object
 14  Is delivering now     9551 non-null   object
 15  Switch to order menu  9551 non-null   object
 16  Price range           9551 non-null   int64
```

```
 17  Aggregate rating      9551 non-null   float64
 18  Rating color          9551 non-null   object
 19  Rating text           9551 non-null   object
 20  Votes                 9551 non-null   int64
dtypes: float64(3), int64(5), object(13)
memory usage: 1.5+ MB
```

Task: Data Exploration and Preprocessing

Explore the dataset and identify the number of rows and columns.

[ ]:
```python
print(f"No of Rows and Columns: {data.shape}")
```

```
No of Rows and Columns: (9551, 21)
```

Check for missing values in each column and handle them accordingly.

[ ]:
```python
data.isnull().sum()
```

[ ]:
```
Restaurant ID           0
Restaurant Name         0
Country Code            0
City                    0
Address                 0
Locality                0
Locality Verbose        0
Longitude               0
Latitude                0
Cuisines                9
Average Cost for two    0
Currency                0
Has Table booking       0
Has Online delivery     0
Is delivering now       0
Switch to order menu    0
Price range             0
Aggregate rating        0
Rating color            0
Rating text             0
Votes                   0
dtype: int64
```

[ ]:
```python
data["Cuisines"]=data["Cuisines"].fillna(data['Cuisines'].mode()[0])
```

Perform data type conversion if necessary.

[ ]:
```python
data.dtypes
```

[ ]:
```
Restaurant ID              int64
Restaurant Name           object
Country Code               int64
City                      object
Address                   object
Locality                  object
Locality Verbose          object
Longitude                float64
Latitude                 float64
Cuisines                  object
Average Cost for two       int64
Currency                  object
Has Table booking         object
Has Online delivery       object
Is delivering now         object
Switch to order menu      object
Price range                int64
Aggregate rating         float64
Rating color              object
Rating text               object
Votes                      int64
dtype: object
```

[ ]:
```
data["Has Table booking"] = data["Has Table booking"].map({"Yes": 1, "No": 0})
data["Has Online delivery"] = data["Has Online delivery"].map({"Yes": 1, "No":
 ↪0})
data["Is delivering now"] = data["Is delivering now"].map({"Yes": 1, "No": 0})
data["Switch to order menu"] = data["Switch to order menu"].map({"Yes": 1, "No":
 ↪ 0})
```

Analyze the distribution of the target variable ("Aggregate rating") and identify any class imbalances.

[ ]:
```
data["Aggregate rating"].value_counts()
```

[ ]:
```
Aggregate rating
0.0    2148
3.2     522
3.1     519
3.4     498
3.3     483
3.5     480
3.0     468
3.6     458
3.7     427
3.8     400
2.9     381
```

```
3.9     335
2.8     315
4.1     274
4.0     266
2.7     250
4.2     221
2.6     191
4.3     174
4.4     144
2.5     110
4.5      95
2.4      87
4.6      78
4.9      61
2.3      47
4.7      42
2.2      27
4.8      25
2.1      15
2.0       7
1.9       2
1.8       1
Name: count, dtype: int64
```

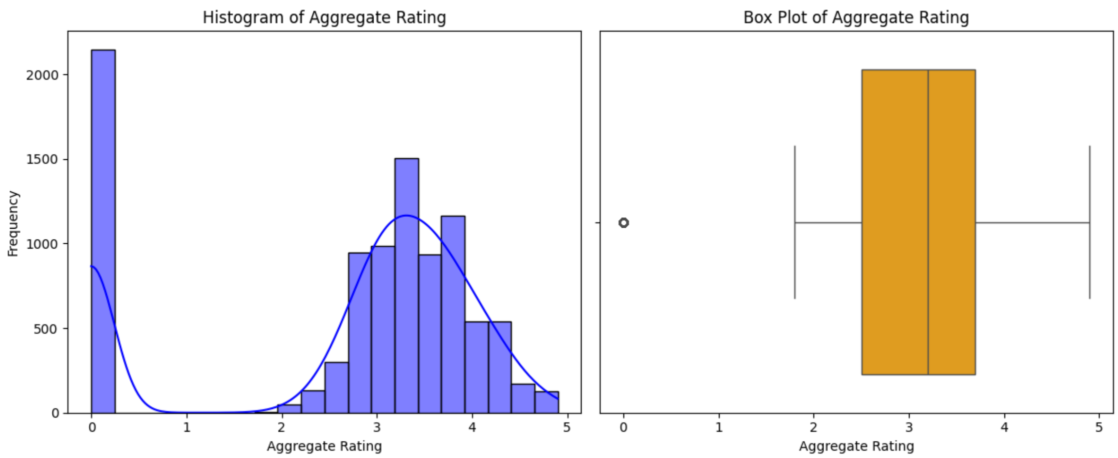
[ ]:

```
plt.figure(figsize=(12, 5))

# Histogram
plt.subplot(1, 2, 1)
sns.histplot(data["Aggregate rating"], bins=20, kde=True, color="blue")
plt.xlabel("Aggregate Rating")
plt.ylabel("Frequency")
plt.title("Histogram of Aggregate Rating")

# Box Plot
plt.subplot(1, 2, 2)
sns.boxplot(x=data["Aggregate rating"], color="orange")
plt.xlabel("Aggregate Rating")
plt.title("Box Plot of Aggregate Rating")

plt.tight_layout()
plt.show()
```



Majority of Ratings are Zero (2148 entries)

A significant number of restaurants have a rating of 0.0, which could mean: They are unrated. The rating data is missing or not available. Most Ratings Fall Between 2.5 and 4.0

The majority of ratings are in the 2.5 to 4.0 range, with the most frequent ratings being around 3.0 to 3.7. This suggests that most restaurants have average ratings, with fewer highly rated ones. Fewer High Ratings (4.5 and above)

Ratings above 4.5 are rare, which indicates that only a few restaurants are rated as excellent. The highest rating (4.9) has just 61 entries, and only 25 restaurants have a 4.8 rating. Very Few Low Ratings (Below 2.0)

Ratings below 2.0 are extremely rare, suggesting that poorly rated restaurants are either uncommon or not rated at all.

Task: Descriptive Analysis

Calculate basic statistical measures (mean,median, standard deviation, etc.) for numericalcolumns.

[ ]:
```
data.describe()
```

[ ]:
```
       Restaurant ID  Country Code   Longitude    Latitude  \
count   9.551000e+03   9551.000000 9551.000000 9551.000000
mean    9.051128e+06     18.365616   64.126574   25.854381
std     8.791521e+06     56.750546   41.467058   11.007935
min     5.300000e+01      1.000000 -157.948486  -41.330428
25%     3.019625e+05      1.000000   77.081343   28.478713
50%     6.004089e+06      1.000000   77.191964   28.570469
75%     1.835229e+07      1.000000   77.282006   28.642758
max     1.850065e+07    216.000000  174.832089   55.976980

       Average Cost for two  Has Table booking  Has Online delivery  \
count           9551.000000        9551.000000          9551.000000
```

```
mean            1199.210763          0.121244          0.256622
std            16121.183073          0.326428          0.436792
min                0.000000          0.000000          0.000000
25%              250.000000          0.000000          0.000000
50%              400.000000          0.000000          0.000000
75%              700.000000          0.000000          1.000000
max           800000.000000          1.000000          1.000000

       Is delivering now  Switch to order menu  Price range  Aggregate rating  \
count        9551.000000                9551.0  9551.000000       9551.000000
mean            0.003560                   0.0     1.804837          2.666370
std             0.059561                   0.0     0.905609          1.516378
min             0.000000                   0.0     1.000000          0.000000
25%             0.000000                   0.0     1.000000          2.500000
50%             0.000000                   0.0     2.000000          3.200000
75%             0.000000                   0.0     2.000000          3.700000
max             1.000000                   0.0     4.000000          4.900000

              Votes
count   9551.000000
mean     156.909748
std      430.169145
min        0.000000
25%        5.000000
50%       31.000000
75%      131.000000
max    10934.000000
```

Explore the distribution of categoricalvariables like "Country Code," "City," and"Cuisines."

[ ]:
```python
categorical_cols = ["Country Code", "City", "Cuisines"]

for col in categorical_cols:
    print(f"\nTop 10 Most Frequent Values in {col}:\n", data[col].
 ↪value_counts().head(10))
```

```
Top 10 Most Frequent Values in Country Code:
 Country Code
1      8652
216     434
215      80
30       60
214      60
189      60
148      40
208      34
14       24
```

```
162      22
Name: count, dtype: int64

Top 10 Most Frequent Values in City:
 City
New Delhi       5473
Gurgaon         1118
Noida           1080
Faridabad        251
Ghaziabad         25
Bhubaneshwar      21
Amritsar          21
Ahmedabad         21
Lucknow           21
Guwahati          21
Name: count, dtype: int64

Top 10 Most Frequent Values in Cuisines:
 Cuisines
North Indian                        945
North Indian, Chinese               511
Chinese                             354
Fast Food                           354
North Indian, Mughlai               334
Cafe                                299
Bakery                              218
North Indian, Mughlai, Chinese      197
Bakery, Desserts                    170
Street Food                         149
Name: count, dtype: int64
```

Identify the top cuisines and cities with thehighest number of restaurants.

[ ]:
```
cuisines_split = data['Cuisines'].dropna().str.split(',').explode().str.
 ↪strip()# because if we are using cuicine combinations given in data, we␣
 ↪willl get top cuisine combinations, not cuisines

# Get the top cuisines
top_cuisines = cuisines_split.value_counts().head()
print(top_cuisines)
```

```
Cuisines
North Indian    3969
Chinese         2735
Fast Food       1986
Mughlai          995
Italian          764
Name: count, dtype: int64
```

[ ]:
```
print(data["City"].value_counts().head())
```

```
City
New Delhi    5473
Gurgaon      1118
Noida        1080
Faridabad     251
Ghaziabad      25
Name: count, dtype: int64
```

Task: Geospatial Analysis

Visualize the locations of restaurants on amap using latitude and longitudeinformation.

[ ]:
```
import folium
from folium.plugins import HeatMap

# Create a base map centered around an approximate middle location
map_center = [data["Latitude"].mean(), data["Longitude"].mean()]
restaurant_map = folium.Map(location=map_center, zoom_start=5)

# Add restaurant locations to the map
for _, row in data.iterrows():
    folium.CircleMarker(
        location=[row["Latitude"], row["Longitude"]],
        radius=3,
        color="blue",
        fill=True,
        fill_color="blue",
        fill_opacity=0.6,
    ).add_to(restaurant_map)

# Display the map
restaurant_map
```

[ ]: <folium.folium.Map at 0x7d540e9a0dd0>

Analyze the distribution of restaurantsacross different cities or countries.

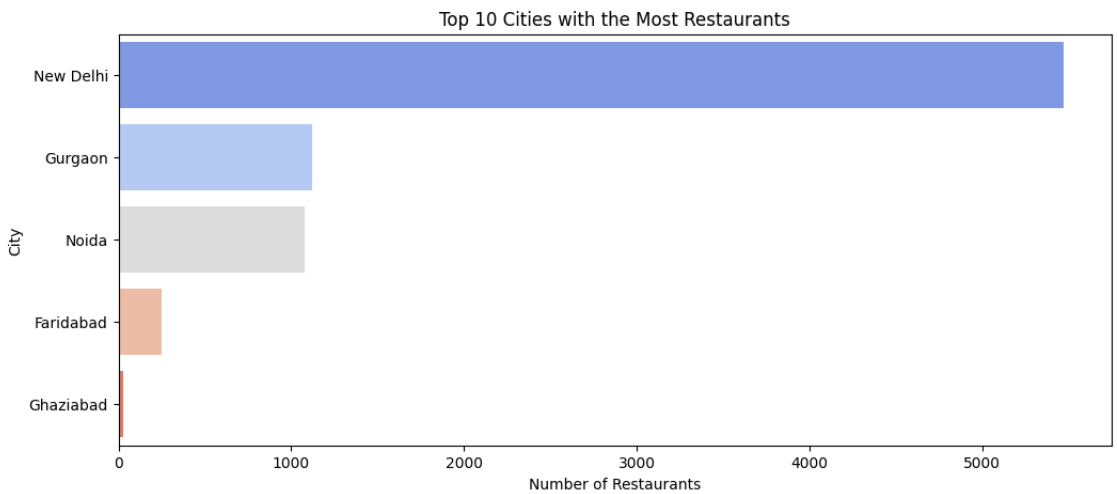
[ ]:
```
top_cities = data["City"].value_counts().head()

plt.figure(figsize=(12, 5))
sns.barplot(x=top_cities.values, y=top_cities.index, palette="coolwarm")
plt.xlabel("Number of Restaurants")
plt.ylabel("City")
plt.title("Top 10 Cities with the Most Restaurants")
plt.show()
```

```
<ipython-input-37-872a1b4705f4>:4: FutureWarning:
```

```
Passing `palette` without assigning `hue` is deprecated and will be removed in
v0.14.0. Assign the `y` variable to `hue` and set `legend=False` for the same
effect.

  sns.barplot(x=top_cities.values, y=top_cities.index, palette="coolwarm")
```



Determine if there is any correlationbetween the restaurant's location and itsrating.

[ ]:
```
restaurant_heatmap = folium.Map(location=map_center, zoom_start=5)
heat_data = data[["Latitude", "Longitude", "Aggregate rating"]].dropna().values.
  ↪tolist()

HeatMap(heat_data, radius=10).add_to(restaurant_heatmap)

restaurant_heatmap
```

[ ]:
```
<folium.folium.Map at 0x7d540c12cbd0>
```

Colors with more intensity represents locations with more highly rated restraunts. They are more prevalent in regions with many restraunts.

[ ]:
```
from sklearn.cluster import DBSCAN
import numpy as np

# Select features for clustering (latitude, longitude, and aggregate rating)
clustering_data = data[['Latitude', 'Longitude', 'Aggregate rating']].dropna()

# Standardize the ratings to avoid bias towards location
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
```

```
clustering_data[['Latitude', 'Longitude']] = scaler.
↪fit_transform(clustering_data[['Latitude', 'Longitude']])

# Apply DBSCAN clustering
db = DBSCAN(eps=0.3, min_samples=10).fit(clustering_data[['Latitude',␣
↪'Longitude', 'Aggregate rating']])

# Add the cluster labels to the data
clustering_data['Cluster'] = db.labels_

# Visualizing the clusters on a map
import folium
from folium.plugins import MarkerCluster

# Map center based on the mean latitude and longitude
map_center = [data["Latitude"].mean(), data["Longitude"].mean()]
restaurant_map = folium.Map(location=map_center, zoom_start=5)

# Create MarkerCluster to group markers based on proximity
marker_cluster = MarkerCluster().add_to(restaurant_map)

# Add clustered restaurants to the map
for idx, row in clustering_data.iterrows():
    folium.Marker([data.iloc[idx]["Latitude"], data.iloc[idx]["Longitude"]],
                  popup=f"Rating: {data.iloc[idx]['Aggregate rating']}, Cluster:
↪ {row['Cluster']}").add_to(marker_cluster)

# Display the map
restaurant_map
```

[ ]: <folium.folium.Map at 0x7d5403e12ed0>

[ ]:
```
# Visualizing clusters with rating color coding
restaurant_map = folium.Map(location=map_center, zoom_start=5)

# Add clusters to the map, color-coded by average rating
for idx, row in clustering_data.iterrows():
    color = 'green' if row['Aggregate rating'] >= 4 else 'orange' if␣
↪row['Aggregate rating'] >= 3 else 'red'
    folium.Marker(
        [data.iloc[idx]["Latitude"], data.iloc[idx]["Longitude"]],
        popup=f"Rating: {data.iloc[idx]['Aggregate rating']}, Cluster:␣
↪{row['Cluster']}",
        icon=folium.Icon(color=color)
    ).add_to(restaurant_map)

# Display the map
```

```
restaurant_map
```

```
[ ]: <folium.folium.Map at 0x7d5401b29ad0>
```