cognify-l1-t2

January 20, 2024

[1]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

[2]:
```
df = pd.read_csv("./L1T2_Dataset.csv")
df.head()
```

[2]:

	Restaurant Name	Country Code	City	Address	Longitude	Latitude	Cuisines	Average Cost for two	Currency	Has Table booking	Has Online delivery	Is delivering now	Price range	Aggregate rating	Rating color	Rating text	Votes
0	Le Petit Souffle	162	Makati City	Third Floor, Century City Mall, Kalayaan Avenu…	121.027535	14.565443	French, Japanese, Desserts	1100	Botswana Pula(P)	1	0	0	3	4.8	Dark Green	Excellent	314
1	Izakaya Kikufuji	162	Makati City	Little Tokyo, 2277 Chino Roces Avenue, Legaspi…	121.014101	14.553708	Japanese	1200	Botswana Pula(P)	1	0	0	3	4.5	Dark Green	Excellent	591
2	Heat - Edsa Shangri-La	162	Mandaluyong City	Edsa Shangri-La, 1 Garden Way, Ortigas, Mandal…	121.056831	14.581404	Seafood, Asian, Filipino, Indian	4000	Botswana Pula(P)	1	0	0	4	4.4	Green	Very Good	270
3	Ooma	162	Mandaluyong City	Third Floor, Mega Fashion Hall, SM Megamall, O…	121.056475	14.585318	Japanese, Sushi	1500	Botswana Pula(P)	0	0	0	4	4.9	Dark Green	Excellent	365
4	Sambo Kojin	162	Mandaluyong City	Third Floor, Mega Atrium, SM Megamall, Ortigas…	121.057508	14.584450	Japanese, Korean	1500	Botswana Pula(P)	1	0	0	4	4.8	Dark Green	Excellent	365

```
0              4.8             0             1    314
1              4.5             0             1    591
2              4.4             1             5    270
3              4.9             0             1    365
4              4.8             0             1    229
```

[12]:
```
country_codes = df["Country Code"].unique()
country_codes
```

[12]:
```
array([162,  30, 216,  14,  37, 184, 214,   1,  94, 148, 215, 166, 189,
       191, 208])
```

[115]:
```
country_city_dict = dict()

for country_code in country_codes:
    city_list = list(set(df[df['Country Code'] == country_code]['City']))
    country_city_dict[country_code] = city_list
```

[24]:
```
for country_code, city_name in sorted(country_city_dict.items()):
  print(country_code, city_name)
```

```
1 ['Secunderabad', 'Ghaziabad', 'Faridabad', 'Mohali', 'Goa', 'Mangalore',
'Puducherry', 'Panchkula', 'Mysore', 'Ahmedabad', 'Coimbatore', 'Indore',
'Kochi', 'Chandigarh', 'Vizag', 'Bhubaneshwar', 'Aurangabad', 'Kolkata', 'Pune',
'Jaipur', 'Guwahati', 'Nagpur', 'Patna', 'Ranchi', 'Chennai', 'Varanasi', 'New
Delhi', 'Bangalore', 'Hyderabad', 'Nashik', 'Vadodara', 'Bhopal', 'Ludhiana',
'Kanpur', 'Amritsar', 'Noida', 'Allahabad', 'Agra', 'Surat', 'Dehradun',
'Lucknow', 'Mumbai', 'Gurgaon']
14 ['Paynesville', 'Flaxton', 'Balingup', 'Mayfield', 'Hepburn Springs', 'Lorn',
'Tanunda', 'Montville', 'Huskisson', 'Penola', 'Macedon', 'Phillip Island',
'Middleton Beach', 'Trentham East', 'Inverloch', 'Palm Cove', 'Beechworth',
'Victor Harbor', 'Dicky Beach', 'Armidale', 'Lakes Entrance', 'Forrest', 'East
Ballina']
30 ['Bras _lia', 'Rio de Janeiro', 'S o Paulo']
37 ['Chatham-Kent', 'Yorkton', 'Vineland Station', 'Consort']
94 ['Bogor', 'Tangerang', 'Bandung', 'Jakarta']
148 ['Auckland', 'Wellington City']
162 ['Pasig City', 'Pasay City', 'San Juan City', 'Makati City', 'Quezon City',
'Mandaluyong City', 'Tagaytay City', 'Taguig City', 'Santa Rosa']
166 ['Doha']
184 ['Singapore']
189 ['Randburg', 'Cape Town', 'Sandton', 'Inner City', 'Johannesburg',
'Pretoria']
191 ['Colombo']
208 [' stanbul', 'Ankara']
214 ['Abu Dhabi', 'Dubai', 'Sharjah']
215 ['London', 'Manchester', 'Edinburgh', 'Birmingham']
```

```
216 ['Boise', 'Princeton', 'Ojo Caliente', 'Gainesville', 'Dubuque', 'Mc
Millan', 'Davenport', 'Augusta', 'Sioux City', 'Winchester Bay', 'Orlando',
'Albany', 'Valdosta', 'Pocatello', 'Lincoln', 'Cochrane', 'Tampa Bay', 'Dalton',
'Weirton', 'Fernley', 'Savannah', 'Rest of Hawaii', 'Cedar Rapids/Iowa City',
'Clatskanie', 'Des Moines', 'Columbus', 'Lakeview', 'Macon', 'Potrero',
'Athens', 'Waterloo', 'Pensacola', 'Monroe', 'Vernonia']
```

[25]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9542 entries, 0 to 9541
Data columns (total 17 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   Restaurant Name       9542 non-null   object
 1   Country Code          9542 non-null   int64
 2   City                  9542 non-null   object
 3   Address               9542 non-null   object
 4   Longitude             9542 non-null   float64
 5   Latitude              9542 non-null   float64
 6   Cuisines              9542 non-null   object
 7   Average Cost for two  9542 non-null   int64
 8   Currency              9542 non-null   object
 9   Has Table booking     9542 non-null   int64
 10  Has Online delivery   9542 non-null   int64
 11  Is delivering now     9542 non-null   int64
 12  Price range           9542 non-null   int64
 13  Aggregate rating      9542 non-null   float64
 14  Rating color          9542 non-null   int64
 15  Rating text           9542 non-null   int64
 16  Votes                 9542 non-null   int64
dtypes: float64(3), int64(9), object(5)
memory usage: 1.2+ MB
```

[48]:
```
print("Price Range")
print(df['Price range'].describe(), '\n')

print("Aggregate rating")
print(df['Aggregate rating'].describe(), '\n')

print("Rating color")
print(df['Rating color'].describe(), '\n')

print("Rating text")
print(df['Rating text'].describe(), '\n')

print("Votes")
```

```
print(df['Votes'].describe())
```

```
Price Range
count    9542.000000
mean        1.804968
std         0.905563
min         1.000000
25%         1.000000
50%         2.000000
75%         2.000000
max         4.000000
Name: Price range, dtype: float64

Aggregate rating
count    9542.000000
mean        2.665238
std         1.516588
min         0.000000
25%         2.500000
50%         3.200000
75%         3.700000
max         4.900000
Name: Aggregate rating, dtype: float64

Rating color
count    9542.000000
mean        2.952840
std         1.492629
min         0.000000
25%         2.000000
50%         2.000000
75%         4.000000
max         5.000000
Name: Rating color, dtype: float64

Rating text
count    9542.000000
mean        1.788933
std         1.694795
min         0.000000
25%         0.000000
50%         2.000000
75%         3.000000
max         5.000000
Name: Rating text, dtype: float64

Votes
```

```
count     9542.000000
mean       156.772060
std        430.203324
min          0.000000
25%          5.000000
50%         31.000000
75%        130.000000
max      10934.000000
Name: Votes, dtype: float64
```

[42]:
```
for country_code in sorted(country_codes):
  filtered_df = df[df['Country Code'] == country_code]
  most_ordered_cuisine = filtered_df['Cuisines'].mode()

  if not most_ordered_cuisine.empty:
      print(f"The most ordered cuisine in {country_code} is:␣
 ↪{most_ordered_cuisine.iloc[0]}")
  else:
      print(f"No most ordered cuisine found for {country_code}")
```

```
The most ordered cuisine in 1 is: North Indian
The most ordered cuisine in 14 is: Breakfast, Coffee and Tea
The most ordered cuisine in 30 is: Brazilian
The most ordered cuisine in 37 is: Asian
The most ordered cuisine in 94 is: Sunda, Indonesian
The most ordered cuisine in 148 is: Cafe
The most ordered cuisine in 162 is: Filipino
The most ordered cuisine in 166 is: Indian
The most ordered cuisine in 184 is: French
The most ordered cuisine in 189 is: Mexican
The most ordered cuisine in 191 is: American, Chinese, North Indian
The most ordered cuisine in 208 is: Cafe
The most ordered cuisine in 214 is: Indian
The most ordered cuisine in 215 is: Italian
The most ordered cuisine in 216 is: Mexican
```

[51]:
```
for country_code in sorted(country_codes):
  filtered_df = df[df['Country Code'] == country_code]
  max_restaurant_city = filtered_df['City'].mode()

  if not most_ordered_cuisine.empty:
      print(f"City with most restaurants in {country_code} is:␣
 ↪{max_restaurant_city.iloc[0]}")
  else:
      print(f"No city with restaurant {country_code}")
```

```
City with most restaurants in 1 is: New Delhi
City with most restaurants in 14 is: Hepburn Springs
```

```
City with most restaurants in 30 is: Bras _lia
City with most restaurants in 37 is: Chatham-Kent
City with most restaurants in 94 is: Jakarta
City with most restaurants in 148 is: Auckland
City with most restaurants in 162 is: Mandaluyong City
City with most restaurants in 166 is: Doha
City with most restaurants in 184 is: Singapore
City with most restaurants in 189 is: Cape Town
City with most restaurants in 191 is: Colombo
City with most restaurants in 208 is: Ankara
City with most restaurants in 214 is: Abu Dhabi
City with most restaurants in 215 is: Birmingham
City with most restaurants in 216 is: Athens
```

[64]:
```
df.head()
```

[64]:

	Restaurant Name	Country Code	City	Address	Longitude	Latitude	Cuisines	Average Cost for two	Currency	Has Table booking	Has Online delivery	Is delivering now	Price range	Aggregate rating	Rating color	Rating text	Votes
0	Le Petit Souffle	162	Makati City	Third Floor, Century City Mall, Kalayaan Avenu…	121.027535	14.565443	French, Japanese, Desserts	1100	Botswana Pula(P)	1	0	0	3	4.8	0	1	314
1	Izakaya Kikufuji	162	Makati City	Little Tokyo, 2277 Chino Roces Avenue, Legaspi…	121.014101	14.553708	Japanese	1200	Botswana Pula(P)	1	0	0	3	4.5	0	1	591
2	Heat - Edsa Shangri-La	162	Mandaluyong City	Edsa Shangri-La, 1 Garden Way, Ortigas, Mandal…	121.056831	14.581404	Seafood, Asian, Filipino, Indian	4000	Botswana Pula(P)	1	0	0	4				
3	Ooma	162	Mandaluyong City	Third Floor, Mega Fashion Hall, SM Megamall, O…	121.056475	14.585318	Japanese, Sushi	1500	Botswana Pula(P)	0	0	0	4				
4	Sambo Kojin	162	Mandaluyong City	Third Floor, Mega Atrium, SM Megamall, Ortigas…	121.057508	14.584450	Japanese, Korean	1500	Botswana Pula(P)	1	0	0	4				

```
2               4.4             1               5     270
3               4.9             0               1     365
4               4.8             0               1     229
```

[77]:
```
most_visited_restaurant = df['Restaurant Name'].mode()[0]
most_visited_restaurant
```

[77]:
```
'Cafe Coffee Day'
```

[81]:
```
count_most_visited = (df['Restaurant Name']=='Cafe Coffee Day').sum()
count_most_visited
```

[81]:
```
83
```

[107]:
```
most_visited_restaurant_country = set()  # Initialize an empty set

for i in range(len(df)):
    if df['Restaurant Name'].iloc[i] == most_visited_restaurant:
        most_visited_restaurant_country.add(str(df['Country Code'].iloc[i]))  #␣
 ↪Convert to string using str()
```

```
Country codes for Cafe Coffee Day: 1
```

[114]:
```
print("Most visited restaurant is '{0}', in Country Code '{1}', \nNo.of␣
 ↪instance for"
      "'{0}' is {2}".format(most_visited_restaurant,
                            list(most_visited_restaurant_country)[0],
                            count_most_visited))
```

```
Most visited restaurant is 'Cafe Coffee Day', in Country Code '1',
No.of instance for'Cafe Coffee Day' is 83
```

[135]:
```
from statistics import mode

most_frequent_city = df['City'].mode()[0]

# Create a set containing unique restaurant names for the most frequent city
restaurant_list_most_frequent_city = list(df[df['City'] ==␣
 ↪most_frequent_city]['Restaurant Name'])
most_visited_restaurant = mode(restaurant_list_most_frequent_city)
most_visited_restaurant
```

[135]:
```
'Cafe Coffee Day'
```

[137]:
```
print("Most visited city is {}.".format(df['City'].mode()[0]))
print("Most visited restaurant in '{0}' is '{1}'.".format(df['City'].mode()[0],␣
 ↪most_visited_restaurant))
```

```
Most visited city is New Delhi.
Most visited restaurant in 'New Delhi' is 'Cafe Coffee Day'.
```

[138]:
```python
most_ordered_cuisine = filtered_df['Cuisines'].mode()

if not most_ordered_cuisine.empty:
    print(f"The most ordered cuisine is: {most_ordered_cuisine.iloc[0]}")
else:
    pass
```

```
The most ordered cuisine is: Mexican
```