

RESTAURANT ANALYSIS

```
In [1]: import warnings
warnings.filterwarnings('ignore')
```

```
In [2]: import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

```
In [3]: df = pd.read_csv(r'D:\NAYAN_DS\DATA_SCIENCE\INTERNSHIP\Cognifyz\Dataset.csv')
```

Data Understanding (EDA) + Data Preprocessing (Data Cleaning)

```
In [4]: df.head(2)
```

Out[4]:

	Restaurant ID	Restaurant Name	Country Code	City	Address	Locality	Locality Verbose	Longitude	Latitude	Cuisines	...
0	6317637	Le Petit Souffle	162	Makati City	Third Floor, Century City Mall, Kalayaan Avenu...	Century City Mall, Poblacion, Makati City	Century City Mall, Poblacion, Makati City, Mak...	121.027535	14.565443	French, Japanese, Desserts	...
1	6304287	Izakaya Kikufuji	162	Makati City	Little Tokyo, 2277 Chino Roces Avenue, Legaspi...	Little Tokyo, Legaspi Village, Makati City	Little Tokyo, Legaspi Village, Makati City, Ma...	121.014101	14.553708	Japanese	...

2 rows × 21 columns

```
In [5]: df.tail(2)
```

Out[5]:

	Restaurant ID	Restaurant Name	Country Code	City	Address	Locality	Locality Verbose	Longitude	Latitude	Cuisines	...
9549	5916112	Ak Kahve	208	Istanbul	Kurucaesme Mahallesi, Muallim Naci Caddesi, N...	Kurucaesme	Kurucaesme, Istanbul	29.036019	41.075911	Turkish, Mediterranean	...
9550	5927402	Walter's Coffee Roastery	208	Istanbul	Cafea Mahallesi, Bademalti Sokak, No 21/B, ...	Moda	Moda, Istanbul	29.026016	40.981511	Coffee, Pastries	...

2 rows × 21 columns

```
In [6]: df.columns
```

```
Out[6]: Index(['Restaurant ID', 'Restaurant Name', 'Country Code', 'City', 'Address',
              'Locality', 'Locality Verbose', 'Longitude', 'Latitude', 'Cuisines',
              'Average Cost for two', 'Currency', 'Has Table booking',
              'Has Online delivery', 'Is delivering now', 'Switch to order menu',
              'Price range', 'Aggregate rating', 'Rating color', 'Rating text',
              'Votes'],
              dtype='object')
```

```
In [7]: df.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
```

```
In [8]: df.shape
```

```
Out[8]: (9551, 21)
```

```
In [9]: continuous = ['Restaurant ID', 'Country Code', 'Longitude', 'Latitude',
                     'Average Cost for two', 'Aggregate rating', 'Votes']

discrete_categorical = ['Restaurant Name', 'City', 'Address', 'Locality',
                       'Locality Verbose', 'Cuisines', 'Currency',
                       'Has Table booking', 'Has Online delivery',
                       'Is delivering now', 'Switch to order menu',
                       'Rating color', 'Rating text']

discrete_count = ['Price range']
```

```
In [10]: df['Price range'].unique()
```

```
Out[10]: array([3, 4, 2, 1], dtype=int64)
```

```
In [11]: df.duplicated().sum()
```

```
Out[11]: 0
```

```
In [12]: df['Restaurant Name'].unique()
```

```
Out[12]: array(['Le Petit Souffle', 'Izakaya Kikufuji', 'Heat - Edsa Shangri-La',
               ..., 'Huqqa', 'A☐☐☐k Kahve', "Walter's Coffee Roastery"],
               dtype=object)
```

```
In [13]: df['Restaurant Name'] = df['Restaurant Name'].str.replace('☐', '')
df['Restaurant Name']
```

```
Out[13]: 0          Le Petit Souffle
        1          Izakaya Kikufuji
        2          Heat - Edsa Shangri-La
        3              Ooma
        4          Sambo Kojin
        ...
        9546          Nam1\ Gurme
        9547          Ceviz Aac\
        9548              Huqqa
        9549              Ak Kahve
        9550      Walter's Coffee Roastery
        Name: Restaurant Name, Length: 9551, dtype: object
```

```
In [14]: df['Restaurant Name'].unique()
```

```
Out[14]: array(['Le Petit Souffle', 'Izakaya Kikufuji', 'Heat - Edsa Shangri-La',
        ..., 'Huqqa', 'Ak Kahve', "Walter's Coffee Roastery"], dtype=object)
```

```
In [15]: df['City'].unique()
```

```
Out[15]: array(['Makati City', 'Mandaluyong City', 'Pasay City', 'Pasig City',
        'Quezon City', 'San Juan City', 'Santa Rosa', 'Tagaytay City',
        'Taguig City', 'Bras_ia', 'Rio de Janeiro', 'S_ao Paulo',
        'Albany', 'Armidale', 'Athens', 'Augusta', 'Balingup',
        'Beechworth', 'Boise', 'Cedar Rapids/Iowa City', 'Chatham-Kent',
        'Clatskanie', 'Cochrane', 'Columbus', 'Consort', 'Dalton',
        'Davenport', 'Des Moines', 'Dicky Beach', 'Dubuque',
        'East Ballina', 'Fernley', 'Flaxton', 'Forrest', 'Gainesville',
        'Hepburn Springs', 'Huskisson', 'Inverloch', 'Lakes Entrance',
        'Lakeview', 'Lincoln', 'Lorn', 'Macedon', 'Macon', 'Mayfield',
        'Mc Millan', 'Middleton Beach', 'Miller', 'Monroe', 'Montville',
        'Ojo Caliente', 'Orlando', 'Palm Cove', 'Paynesville', 'Penola',
        'Pensacola', 'Phillip Island', 'Pocatello', 'Potrero', 'Princeton',
        'Rest of Hawaii', 'Savannah', 'Singapore', 'Sioux City',
        'Tampa Bay', 'Tanunda', 'Trentham East', 'Valdosta', 'Vernonia',
        'Victor Harbor', 'Vineland Station', 'Waterloo', 'Weirton',
        'Winchester Bay', 'Yorkton', 'Abu Dhabi', 'Dubai', 'Sharjah',
        'Agra', 'Ahmedabad', 'Allahabad', 'Amritsar', 'Aurangabad',
        'Bangalore', 'Bhopal', 'Bhubaneshwar', 'Chandigarh', 'Chennai',
        'Coimbatore', 'Dehradun', 'Faridabad', 'Ghaziabad', 'Goa',
        'Gurgaon', 'Guwahati', 'Hyderabad', 'Indore', 'Jaipur', 'Kanpur',
        'Kochi', 'Kolkata', 'Lucknow', 'Ludhiana', 'Mangalore', 'Mohali',
        'Mumbai', 'Mysore', 'Nagpur', 'Nashik', 'New Delhi', 'Noida',
        'Panchkula', 'Patna', 'Puducherry', 'Pune', 'Ranchi',
        'Secunderabad', 'Surat', 'Vadodara', 'Varanasi', 'Vizag',
        'Bandung', 'Bogor', 'Jakarta', 'Tangerang', 'Auckland',
        'Wellington City', 'Birmingham', 'Edinburgh', 'London',
        'Manchester', 'Doha', 'Cape Town', 'Inner City', 'Johannesburg',
        'Pretoria', 'Randburg', 'Sandton', 'Colombo', 'Ankara',
        '_istanbul'], dtype=object)
```

```
In [16]: df['City'] = df['City'].str.replace('_', '')
        df['City']
```

```
Out[16]: 0          Makati City
        1          Makati City
        2      Mandaluyong City
        3      Mandaluyong City
        4      Mandaluyong City
        ...
        9546          stanbul
        9547          stanbul
        9548          stanbul
        9549          stanbul
        9550          stanbul
        Name: City, Length: 9551, dtype: object
```

```
In [17]: df['City'].unique()
```

```
Out[17]: array(['Makati City', 'Mandaluyong City', 'Pasay City', 'Pasig City',
               'Quezon City', 'San Juan City', 'Santa Rosa', 'Tagaytay City',
               'Taguig City', 'Bras_lia', 'Rio de Janeiro', 'So Paulo', 'Albany',
               'Armidale', 'Athens', 'Augusta', 'Balingup', 'Beechworth', 'Boise',
               'Cedar Rapids/Iowa City', 'Chatham-Kent', 'Clatskanie', 'Cochrane',
               'Columbus', 'Consort', 'Dalton', 'Davenport', 'Des Moines',
               'Dicky Beach', 'Dubuque', 'East Ballina', 'Fernley', 'Flaxton',
               'Forrest', 'Gainesville', 'Hepburn Springs', 'Huskisson',
               'Inverloch', 'Lakes Entrance', 'Lakeview', 'Lincoln', 'Lorn',
               'Macedon', 'Macon', 'Mayfield', 'Mc Millan', 'Middleton Beach',
               'Miller', 'Monroe', 'Montville', 'Ojo Caliente', 'Orlando',
               'Palm Cove', 'Paynesville', 'Penola', 'Pensacola',
               'Phillip Island', 'Pocatello', 'Potrero', 'Princeton',
               'Rest of Hawaii', 'Savannah', 'Singapore', 'Sioux City',
               'Tampa Bay', 'Tanunda', 'Trentham East', 'Valdosta', 'Vernonia',
               'Victor Harbor', 'Vineland Station', 'Waterloo', 'Weirton',
               'Winchester Bay', 'Yorkton', 'Abu Dhabi', 'Dubai', 'Sharjah',
               'Agra', 'Ahmedabad', 'Allahabad', 'Amritsar', 'Aurangabad',
               'Bangalore', 'Bhopal', 'Bhubaneshwar', 'Chandigarh', 'Chennai',
               'Coimbatore', 'Dehradun', 'Faridabad', 'Ghaziabad', 'Goa',
               'Gurgaon', 'Guwahati', 'Hyderabad', 'Indore', 'Jaipur', 'Kanpur',
               'Kochi', 'Kolkata', 'Lucknow', 'Ludhiana', 'Mangalore', 'Mohali',
               'Mumbai', 'Mysore', 'Nagpur', 'Nashik', 'New Delhi', 'Noida',
               'Panchkula', 'Patna', 'Puducherry', 'Pune', 'Ranchi',
               'Secunderabad', 'Surat', 'Vadodara', 'Varanasi', 'Vizag',
               'Bandung', 'Bogor', 'Jakarta', 'Tangerang', 'Auckland',
               'Wellington City', 'Birmingham', 'Edinburgh', 'London',
               'Manchester', 'Doha', 'Cape Town', 'Inner City', 'Johannesburg',
               'Pretoria', 'Randburg', 'Sandton', 'Colombo', 'Ankara', 'istanbul'],
              dtype=object)
```

```
In [18]: df['Address'].unique()
```

```
Out[18]: array(['Third Floor, Century City Mall, Kalayaan Avenue, Poblacion, Makati City',
               'Little Tokyo, 2277 Chino Roces Avenue, Legaspi Village, Makati City',
               'Edsa Shangri-La, 1 Garden Way, Ortigas, Mandaluyong City', ...,
               'Kuru_e_eme Mahallesi, Muallim Naci Caddesi, No 56, Beikta, istanbul',
               'Kuru_e_eme Mahallesi, Muallim Naci Caddesi, No 64/B, Beikta, istanbul',
               'Cafeaa Mahallesi, Bademalt Sokak, No 21/B, Kad'k_y, istanbul'],
              dtype=object)
```

```
In [19]: df['Address'] = df['Address'].str.replace(' ', '')
         df['Address']
```

```
Out[19]: 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...
         ...
         9546     Kemanke Karamustafa Paa Mahallesi, R'ht'm Cadd...
         9547     Kouyolu Mahallesi, Muhittin st_nda Caddesi, No...
         9548     Kuru_eme Mahallesi, Muallim Naci Caddesi, No 5...
         9549     Kuru_eme Mahallesi, Muallim Naci Caddesi, No 6...
         9550     Cafeaa Mahallesi, Bademalt Sokak, No 21/B, Ka...
         Name: Address, Length: 9551, dtype: object
```

```
In [20]: df['Address'].unique()
```

```
Out[20]: array(['Third Floor, Century City Mall, Kalayaan Avenue, Poblacion, Makati City',
               'Little Tokyo, 2277 Chino Roces Avenue, Legaspi Village, Makati City',
               'Edsa Shangri-La, 1 Garden Way, Ortigas, Mandaluyong City', ...,
               'Kuru_eme Mahallesi, Muallim Naci Caddesi, No 56, Beikta, stanbul',
               'Kuru_eme Mahallesi, Muallim Naci Caddesi, No 64/B, Beikta, stanbul',
               'Cafeaa Mahallesi, Bademalt Sokak, No 21/B, Kad'k_y, stanbul'],
              dtype=object)
```

```
In [21]: df['Locality'].unique()
```

```
Out[21]: array(['Century City Mall, Poblacion, Makati City',
               'Little Tokyo, Legaspi Village, Makati City',
               'Edsa Shangri-La, Ortigas, Mandaluyong City', ..., 'Ko_uyolu',
               'Kuru_e_eme', 'Moda'], dtype=object)
```

```
In [22]: df['Locality'] = df['Locality'].str.replace('◆', '')
df['Locality']
```

```
Out[22]: 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
...
9546      Karak_y
9547      Kouyolu
9548      Kuru_eme
9549      Kuru_eme
9550      Moda
Name: Locality, Length: 9551, dtype: object
```

```
In [23]: df['Locality'].unique()
```

```
Out[23]: array(['Century City Mall, Poblacion, Makati City',
                'Little Tokyo, Legaspi Village, Makati City',
                'Edsa Shangri-La, Ortigas, Mandaluyong City', ..., 'Kouyolu',
                'Kuru_eme', 'Moda'], dtype=object)
```

```
In [24]: df['Locality Verbose'].unique()
```

```
Out[24]: array(['Century City Mall, Poblacion, Makati City, Makati City',
                'Little Tokyo, Legaspi Village, Makati City, Makati City',
                'Edsa Shangri-La, Ortigas, Mandaluyong City, Mandaluyong City',
                ..., 'Ko◆uyolu, ◆◆stanbul', 'Kuru◆_e◆◆me, ◆◆stanbul',
                'Moda, ◆◆stanbul'], dtype=object)
```

```
In [25]: df['Locality Verbose'] = df['Locality Verbose'].str.replace('◆', '')
df['Locality Verbose']
```

```
Out[25]: 0      Century City Mall, Poblacion, Makati City, Mak...
1      Little Tokyo, Legaspi Village, Makati City, Ma...
2      Edsa Shangri-La, Ortigas, Mandaluyong City, Ma...
3      SM Megamall, Ortigas, Mandaluyong City, Mandal...
4      SM Megamall, Ortigas, Mandaluyong City, Mandal...
...
9546      Karak_y, stanbul
9547      Kouyolu, stanbul
9548      Kuru_eme, stanbul
9549      Kuru_eme, stanbul
9550      Moda, stanbul
Name: Locality Verbose, Length: 9551, dtype: object
```

```
In [26]: df['Locality Verbose'].unique()
```

```
Out[26]: array(['Century City Mall, Poblacion, Makati City, Makati City',
                'Little Tokyo, Legaspi Village, Makati City, Makati City',
                'Edsa Shangri-La, Ortigas, Mandaluyong City, Mandaluyong City',
                ..., 'Kouyolu, stanbul', 'Kuru_eme, stanbul', 'Moda, stanbul'],
                dtype=object)
```

```
In [27]: df['Cuisines'].unique()
```

```
Out[27]: array(['French, Japanese, Desserts', 'Japanese',
                'Seafood, Asian, Filipino, Indian', ..., 'Burger, Izgara',
                'World Cuisine, Patisserie, Cafe', 'Italian, World Cuisine'],
                dtype=object)
```

```
In [28]: df['Cuisines'] = df['Cuisines'].str.replace('◆', '')
df['Cuisines']
```

```
Out[28]: 0      French, Japanese, Desserts
        1      Japanese
        2      Seafood, Asian, Filipino, Indian
        3      Japanese, Sushi
        4      Japanese, Korean
        ...
        9546      Turkish
        9547      World Cuisine, Patisserie, Cafe
        9548      Italian, World Cuisine
        9549      Restaurant Cafe
        9550      Cafe
        Name: Cuisines, Length: 9551, dtype: object
```

```
In [29]: df['Cuisines'].unique()
```

```
Out[29]: array(['French, Japanese, Desserts', 'Japanese',
        'Seafood, Asian, Filipino, Indian', ..., 'Burger, Izgara',
        'World Cuisine, Patisserie, Cafe', 'Italian, World Cuisine'],
        dtype=object)
```

Drop unimportant columns

```
In [30]: df.drop(['Country Code', 'Address', 'Address', 'Locality',
        'Locality Verbose', 'Currency', 'Is delivering now', 'Average Cost for two', 'Switch to order men
```

```
In [31]: df.head(2)
```

```
Out[31]:
```

	Restaurant ID	Restaurant Name	City	Longitude	Latitude	Cuisines	Has Table booking	Has Online delivery	Price range	Aggregate rating	Rating text
0	6317637	Le Petit Souffle	Makati City	121.027535	14.565443	French, Japanese, Desserts	Yes	No	3	4.8	Excellent
1	6304287	Izakaya Kikufuji	Makati City	121.014101	14.553708	Japanese	Yes	No	3	4.5	Excellent

Replace columns name

```
In [32]: df.rename(columns = {'Has Online delivery': 'Online_delivery', 'Has Table booking': 'Table_booking'}, inplace=True)
```

After making changes, again understand the data

```
In [33]: df.head(2)
```

```
Out[33]:
```

	Restaurant ID	Restaurant Name	City	Longitude	Latitude	Cuisines	Table_booking	Online_delivery	Price range	Aggregate rating	Rating text
0	6317637	Le Petit Souffle	Makati City	121.027535	14.565443	French, Japanese, Desserts	Yes	No	3	4.8	Excellent
1	6304287	Izakaya Kikufuji	Makati City	121.014101	14.553708	Japanese	Yes	No	3	4.5	Excellent

```
In [34]: df.tail(2)
```

	Restaurant ID	Restaurant Name	City	Longitude	Latitude	Cuisines	Table_booking	Online_delivery	Price range	A
Out[34]:	9549	5916112	Ak Kahve	stanbul	29.036019	41.057979	Restaurant Cafe	No	No	4
	9550	5927402	Walter's Coffee Roastery	stanbul	29.026016	40.984776	Cafe	No	No	2

In [35]: `df.columns`

Out[35]: Index(['Restaurant ID', 'Restaurant Name', 'City', 'Longitude', 'Latitude', 'Cuisines', 'Table_booking', 'Online_delivery', 'Price range', 'Aggregate rating', 'Rating text', 'Votes'], dtype='object')

In [36]: `df.info()`

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9551 entries, 0 to 9550
Data columns (total 12 columns):
#   Column                Non-Null Count  Dtype
---  -
0   Restaurant ID         9551 non-null  int64
1   Restaurant Name       9551 non-null  object
2   City                  9551 non-null  object
3   Longitude              9551 non-null  float64
4   Latitude               9551 non-null  float64
5   Cuisines               9542 non-null  object
6   Table_booking         9551 non-null  object
7   Online_delivery       9551 non-null  object
8   Price range           9551 non-null  int64
9   Aggregate rating      9551 non-null  float64
10  Rating text           9551 non-null  object
11  Votes                 9551 non-null  int64
dtypes: float64(3), int64(3), object(6)
memory usage: 895.5+ KB
```

In [37]: `df.shape`

Out[37]: (9551, 12)

```
In [38]: continuous = ['Restaurant ID', 'Longitude', 'Latitude', 'Aggregate rating', 'Votes']

discrete_categorical = ['Restaurant Name', 'City', 'Cuisines',
                        'Table_booking', 'Online_delivery',
                        'Rating text']

discrete_count = ['Price range']
```

In [39]: `df[continuous].describe()`

	Restaurant ID	Longitude	Latitude	Aggregate rating	Votes
count	9.551000e+03	9551.000000	9551.000000	9551.000000	9551.000000
mean	9.051128e+06	64.126574	25.854381	2.666370	156.909748
std	8.791521e+06	41.467058	11.007935	1.516378	430.169145
min	5.300000e+01	-157.948486	-41.330428	0.000000	0.000000
25%	3.019625e+05	77.081343	28.478713	2.500000	5.000000
50%	6.004089e+06	77.191964	28.570469	3.200000	31.000000
75%	1.835229e+07	77.282006	28.642758	3.700000	131.000000
max	1.850065e+07	174.832089	55.976980	4.900000	10934.000000

In [40]: `df[discrete_categorical].describe()`

Out[40]:

	Restaurant Name	City	Cuisines	Table_booking	Online_delivery	Rating text
count	9551	9551	9542	9551	9551	9551
unique	7446	141	1825	2	2	6
top	Cafe Coffee Day	New Delhi	North Indian	No	No	Average
freq	83	5473	936	8393	7100	3737

In [41]: `from skimp import skim`
`skim(df)`

skimpy summary

Data Summary

dataframe	Values
Number of rows	9551
Number of columns	12

Data Types

Column Type	Count
string	6
int32	3
float64	3

number

column_name	NA	NA %	mean	sd	p0	p25	p50	p75	p100
Restaurant ID	0	0	9100000	8800000	53	300000	6000000	18000000	190000
Longitude	0	0	64	41	-160	77	77	77	1
Latitude	0	0	26	11	-41	28	29	29	
Price range	0	0	1.8	0.91	1	1	2	2	
Aggregate rating	0	0	2.7	1.5	0	2.5	3.2	3.7	4
Votes	0	0	160	430	0	5	31	130	110

string

column_name	NA	NA %	words per row	total word
Restaurant Name	0	0		2.6
City	0	0		1.6
Cuisines	9	0.09		2.9
Table_booking	0	0		1
Online_delivery	0	0		1
Rating text	0	0		1.3

End

Analysis for Business Problem Understanding

Level 1 = Task 1 Task: Top Cuisines

Determine the top three most common cuisines in the dataset.

In [42]: `df.head(1)`

	Restaurant ID	Restaurant Name	City	Longitude	Latitude	Cuisines	Table_booking	Online_delivery	Price range	Aggregate rating
0	6317637	Le Petit Souffle	Makati City	121.027535	14.565443	French, Japanese, Desserts	Yes	No	3	

In [43]: `df['Cuisines'].head()`


```
Out[43]: 0      French, Japanese, Desserts
        1      Japanese
        2      Seafood, Asian, Filipino, Indian
        3      Japanese, Sushi
        4      Japanese, Korean
        Name: Cuisines, dtype: object
```

```
In [44]: df['Cuisines'].unique()
```

```
Out[44]: array(['French, Japanese, Desserts', 'Japanese',
        'Seafood, Asian, Filipino, Indian', ..., 'Burger, Izgara',
        'World Cuisine, Patisserie, Cafe', 'Italian, World Cuisine'],
        dtype=object)
```

```
In [45]: Count_Cuisines = df['Cuisines'].value_counts()
        Count_Cuisines
```

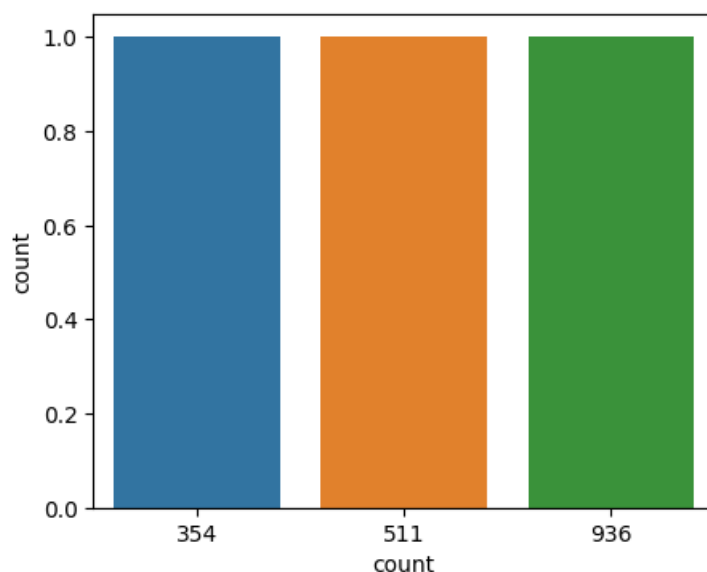
```
Out[45]: Cuisines
        North Indian      936
        North Indian, Chinese    511
        Chinese      354
        Fast Food      354
        North Indian, Mughlai    334
        ...
        Bengali, Fast Food      1
        North Indian, Rajasthani, Asian    1
        Chinese, Thai, Malaysian, Indonesian    1
        Bakery, Desserts, North Indian, Bengali, South Indian    1
        Italian, World Cuisine      1
        Name: count, Length: 1825, dtype: int64
```

```
In [46]: top_three_cuisines = Count_Cuisines.head(3)
        print("The most three Cuisines are:"\n\n' ,top_three_cuisines)
```

"The most three Cuisines are:"

```
        Cuisines
        North Indian      936
        North Indian, Chinese    511
        Chinese      354
        Name: count, dtype: int64
```

```
In [47]: plt.figure(figsize=(5, 4))
        sns.countplot(x=top_three_cuisines)
        plt.show()
```



As we can see based on our value_counts the most common three cuisines are **North Indian, Chinese, and Chinese**

Calculate the percentage of restaurants that serve each of the top cuisines.

```
In [48]: df['Restaurant ID'].unique()
```

Out[48]: 9551

```
In [49]: number_of_Restaurant = len(df['Restaurant ID'])  
number_of_Restaurant
```

Out[49]: 9551

```
In [50]: percentage_Restaurant = (top_three_cuisines/number_of_Restaurant) * 100  
print("The percentage of the Restaurant is:"\n\n',percentage_Restaurant)
```

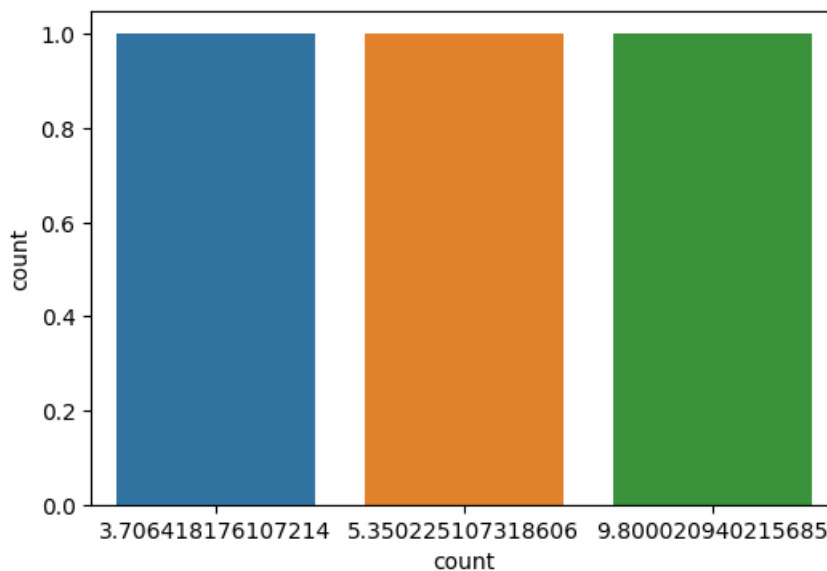
"The percentage of the Restaurant is:"

```
Cuisines  
North Indian          9.800021  
North Indian, Chinese  5.350225  
Chinese               3.706418  
Name: count, dtype: float64
```

```
In [51]: percentage_Restaurant
```

```
Out[51]: Cuisines  
North Indian          9.800021  
North Indian, Chinese  5.350225  
Chinese               3.706418  
Name: count, dtype: float64
```

```
In [52]: plt.figure(figsize=(6, 4))  
sns.countplot(x=percentage_Restaurant)  
plt.show()
```



Level 1 = Task 2 Task: City Analysis

Identify the city with the highest number of restaurants in the dataset.

```
In [53]: count_city = df['City'].value_counts()  
count_city
```

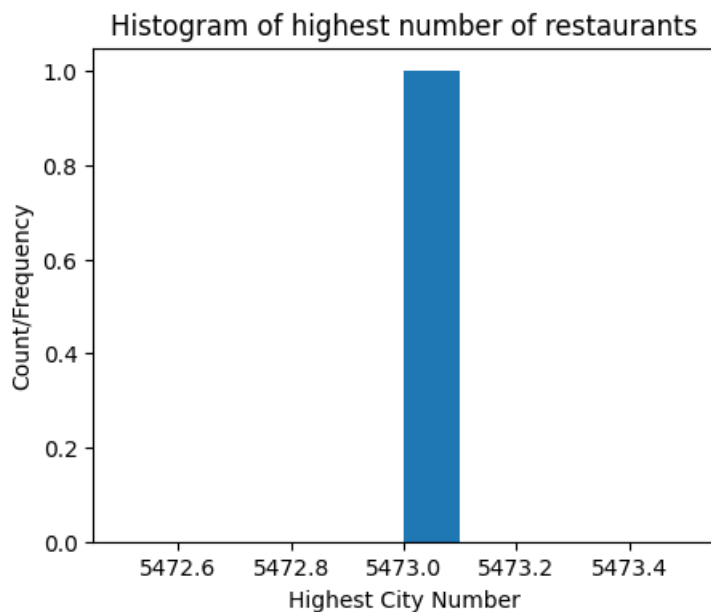
```
Out[53]: City
New Delhi      5473
Gurgaon        1118
Noida          1080
Faridabad      251
Ghaziabad      25
...
Panchkula      1
Mc Millan      1
Mayfield       1
Macedon        1
Vineland Station 1
Name: count, Length: 141, dtype: int64
```

```
In [54]: highest_Restaurant = count_city.head(1)
print("The city with highest number of Restaurant is:"\n\n',highest_Restaurant)
```

"The city with highest number of Restaurant is:"

```
City
New Delhi      5473
Name: count, dtype: int64
```

```
In [55]: plt.figure(figsize=(5, 4))
plt.hist(highest_Restaurant)
plt.xlabel('Highest City Number')
plt.ylabel('Count/Frequency')
plt.title('Histogram of highest number of restaurants')
plt.show()
```



Calculate the average rating for restaurants in each city.

```
In [56]: df['City']
```

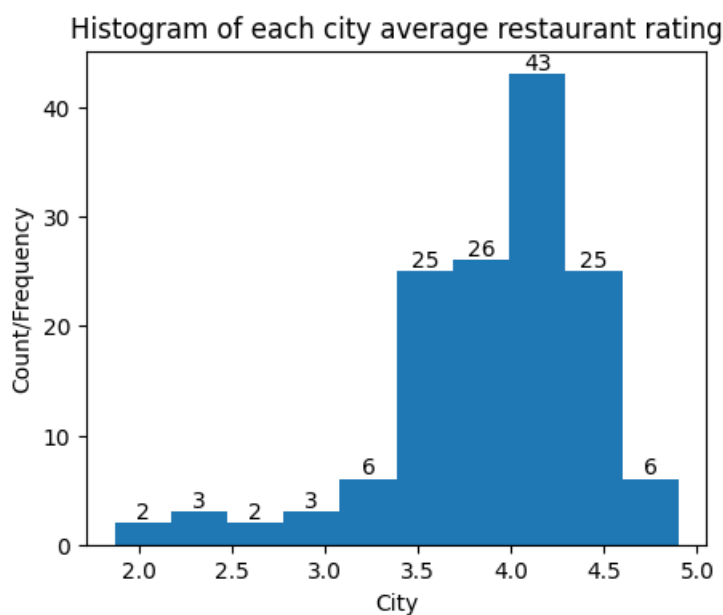
```
Out[56]: 0      Makati City
1      Makati City
2      Mandaluyong City
3      Mandaluyong City
4      Mandaluyong City
...
9546    stanbul
9547    stanbul
9548    stanbul
9549    stanbul
9550    stanbul
Name: City, Length: 9551, dtype: object
```

```
In [57]: avg_rating_Restaurant = df.groupby('City')['Aggregate rating'].mean()
print("The average rating for each city Restaurant is:"\n\n',avg_rating_Restaurant)
```

"The average rating for each city Restaurant is:"

```
City
Abu Dhabi      4.300000
Agra           3.965000
Ahmedabad      4.161905
Albany         3.555000
Allahabad      3.395000
...
Weirton        3.900000
Wellington City 4.250000
Winchester Bay 3.200000
Yorkton        3.300000
istanbul       4.292857
Name: Aggregate rating, Length: 141, dtype: float64
```

```
In [58]: plt.figure(figsize=(5, 4))
n, bins, patches = plt.hist(avg_rating_Restaurant, bins=10)
plt.bar_label(patches)
plt.xlabel('City')
plt.ylabel('Count/Frequency')
plt.title('Histogram of each city average restaurant rating')
plt.show()
```



Determine the city with the highest average rating.

```
In [59]: highest_rating = avg_rating_Restaurant.max()
print("The city with highest average rating is:\n\n", highest_rating)
```

"The city with highest average rating is:"

4.9

Level 1 = Task 3 Task: Price Range Distribution

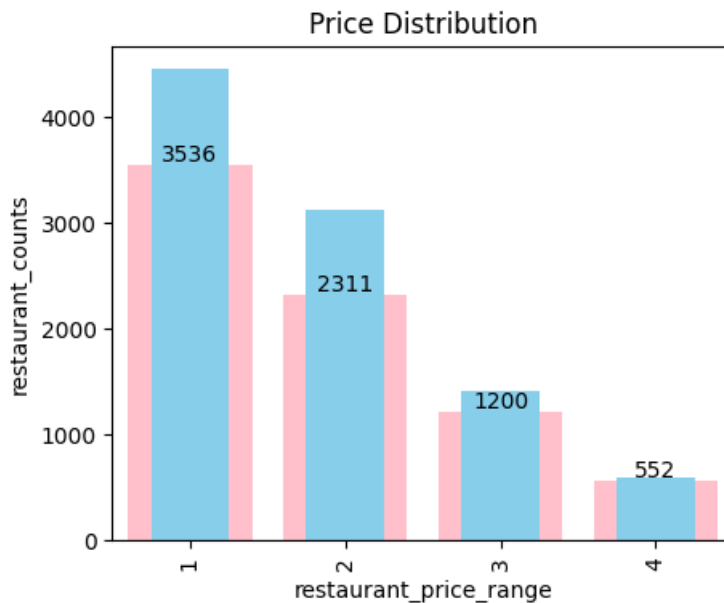
Create a histogram or bar chart to visualize the distribution of price ranges among the restaurants.

```
In [60]: a = df[df['Price range']==1]['Restaurant Name'].nunique()
b = df[df['Price range']==2]['Restaurant Name'].nunique()
c = df[df['Price range']==3]['Restaurant Name'].nunique()
d = df[df['Price range']==4]['Restaurant Name'].nunique()
print(a, ", ", b, ", ", c, ", ", d)

restaurant_prices = ['1', '2', '3', '4']
restaurant_counts = [3536, 2311, 1200, 552]
plt.figure(figsize=(5, 4))
plt.bar(restaurant_prices, restaurant_counts, color='pink')
```

```
ax = df['Price range'].value_counts().sort_index().plot(kind='bar',color='skyblue')
ax.bar_label(ax.containers[0])
plt.title('Price Distribution')
plt.xlabel('restaurant_price_range')
plt.ylabel('restaurant_counts')
plt.show()
```

3536 , 2311 , 1200 , 552



Calculate the percentage of restaurants in each price range category

```
In [61]: price_range = df['Price range'].value_counts()
price_range
```

```
Out[61]: Price range
1      4444
2      3113
3      1408
4       586
Name: count, dtype: int64
```

```
In [62]: price_range = df['Price range'].value_counts(normalize=True) * 100
price_range
```

```
Out[62]: Price range
1      46.529159
2      32.593446
3      14.741912
4       6.135483
Name: proportion, dtype: float64
```

```
In [63]: number_of_Restaurant = len(df['Restaurant ID'])
number_of_Restaurant
```

```
Out[63]: 9551
```

```
In [64]: percentage_of_Restaurant = (price_range/number_of_Restaurant) * 100
print("The percetage of restaurant in each price range :"\n\n' ,percentage_Restaurant)
```

"The percentage of restaurant in each price range :"

```
Cuisines
North Indian      9.800021
North Indian, Chinese  5.350225
Chinese           3.706418
Name: count, dtype: float64
```

Level 1 = Task 4 Task: Online Delivery

Determine the percentage of restaurants that offer online delivery.

```
In [65]: online_delivery = df['Online_delivery'].value_counts()
online_delivery
```

```
Out[65]: Online_delivery
No      7100
Yes     2451
Name: count, dtype: int64
```

```
In [66]: number_of_Restaurant = len(df['Restaurant ID'])
number_of_Restaurant
```

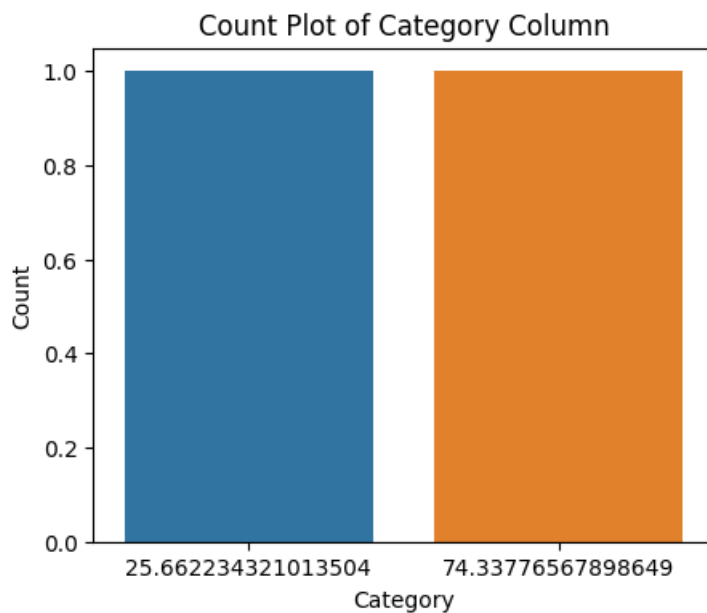
```
Out[66]: 9551
```

```
In [67]: percentage_Restaurant_online_deliver = (online_delivery/number_of_Restaurant) * 100
print('The percentage of restaurant that deliver food online or not:\n\n',percentage_Restaurant_onlin
```

"The percentage of restaurant that deliver food online or not:"

```
Online_delivery
No      74.337766
Yes     25.662234
Name: count, dtype: float64
```

```
In [68]: plt.figure(figsize=(5, 4))
sns.countplot(data=df, x=percentage_Restaurant_online_deliver)
plt.xlabel('Category')
plt.ylabel('Count')
plt.title('Count Plot of Category Column')
plt.show()
```



Compare the average ratings of restaurants with and without online delivery.

```
In [69]: df['Online_delivery'].value_counts()
```

```
Out[69]: Online_delivery
No      7100
Yes     2451
Name: count, dtype: int64
```

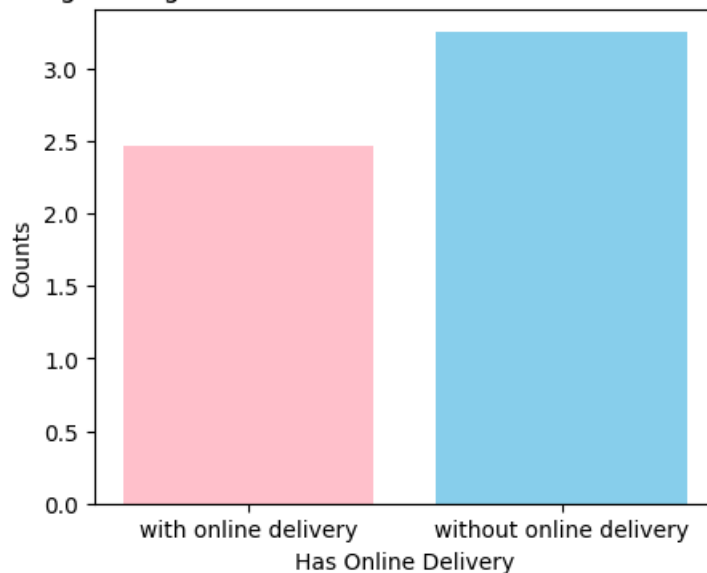
```
In [70]: average_rating_of_restaurant = df.groupby('Online_delivery')['Aggregate rating'].mean()
print('The average rating for the restaurant is:\n\n',average_rating_of_restaurant)
```

"The average rating for the restaurant is:"

```
Online_delivery  
No      2.465296  
Yes     3.248837  
Name: Aggregate rating, dtype: float64
```

```
In [71]: x_axis = ['with online delivery', 'without online delivery']  
plt.figure(figsize=(5, 4))  
plt.bar(x_axis, average_rating_of_restaurant, color=['pink', 'skyblue'])  
plt.xlabel('Has Online Delivery')  
plt.ylabel('Counts')  
plt.title('average ratings of restaurants with and without online delivery')  
plt.show()
```

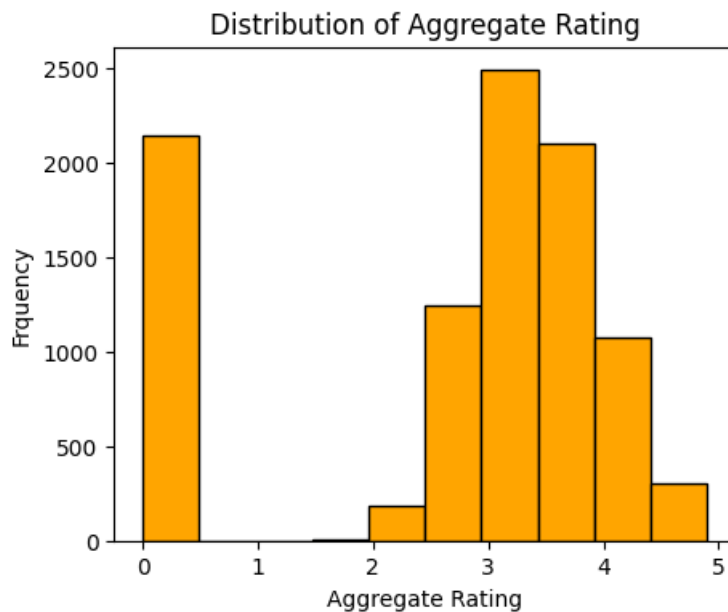
average ratings of restaurants with and without online delivery



Level 2 = Task 1 Task: Restaurant Ratings

Analyze the distribution of aggregate ratings and determine the most common rating range.

```
In [72]: plt.figure(figsize=(5, 4))  
plt.hist(df['Aggregate rating'], bins=10, color='orange', edgecolor='black')  
plt.xlabel('Aggregate Rating')  
plt.ylabel('Frquency')  
plt.title('Distribution of Aggregate Rating')  
plt.show()
```



The most common rating ranges between 3 and 4

Calculate the average number of votes received by restaurants.

```
In [73]: average_Vote = df['Votes'].mean()
print('The average number of votes received by restaurant is:', average_Vote)
```

The average number of votes received by restaurant is: 156.909747670401

Level 2 = Task 2 Task: Cuisine Combination

Identify the most common combinations of cuisines in the dataset.

```
In [74]: cuisine_counts = df['Cuisines'].str.split(' ', expand=True)
cuisine_counts
```

```
Out[74]:
```

	0	1	2	3	4	5	6	7
0	French	Japanese	Desserts	None	None	None	None	None
1	Japanese	None	None	None	None	None	None	None
2	Seafood	Asian	Filipino	Indian	None	None	None	None
3	Japanese	Sushi	None	None	None	None	None	None
4	Japanese	Korean	None	None	None	None	None	None
...
9546	Turkish	None	None	None	None	None	None	None
9547	World Cuisine	Patisserie	Cafe	None	None	None	None	None
9548	Italian	World Cuisine	None	None	None	None	None	None
9549	Restaurant Cafe	None	None	None	None	None	None	None
9550	Cafe	None	None	None	None	None	None	None

9551 rows × 8 columns

```
In [75]: cuisine_counts = df['Cuisines'].str.split(' ', expand=True).stack().value_counts()
print('The most common combination of cuisines in the dataset is:\n\n', cuisine_counts.head(1))
```

"The most common combination of cuisines in the dataset is:"

```
North Indian    3960
Name: count, dtype: int64
```


Determine if certain cuisine combinations tend to have higher ratings.

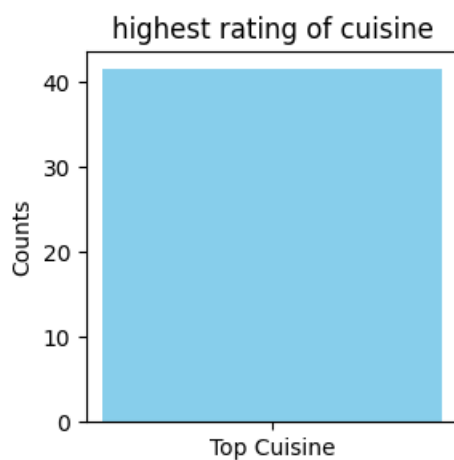
```
In [76]: cuisnaine_higher_ratings = cuisine_counts/len(df['Aggregate rating']) * 100
cuisnaine_higher_ratings
print("The higher ratings of the cuisine combinations are:"\n\n' ,cuisnaine_higher_ratings.head())
```

"The higher ratings of the cuisine combinations are:"

```
North Indian    41.461627
Chinese         28.635745
Fast Food       20.793634
Mughlai         10.417757
Italian         7.999162
```

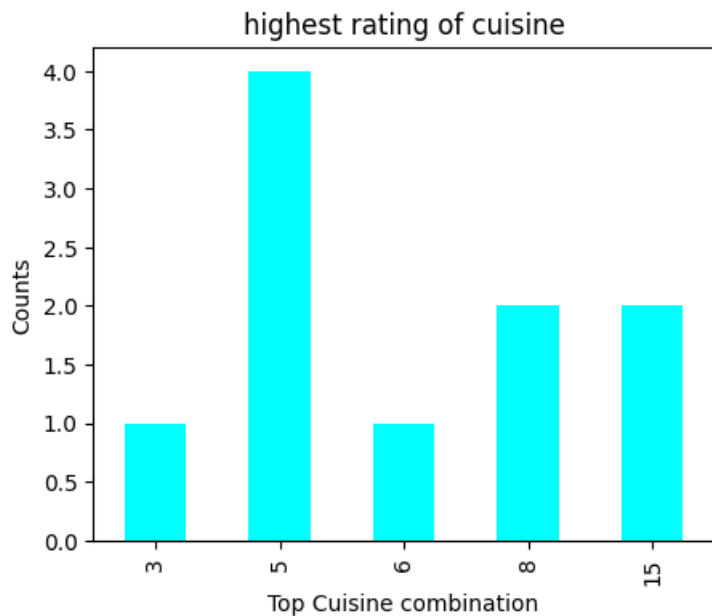
Name: count, dtype: float64

```
In [77]: x_axis = ['Top Cuisine']
plt.figure(figsize=(3, 3))
plt.bar(x_axis, cuisnaine_higher_ratings, color=['skyblue'])
plt.ylabel('Counts')
plt.title('highest rating of cuisine')
plt.show()
```



```
In [78]: cuisine = cuisnaine_higher_ratings.value_counts().head(10)
print(cuisine)
plt.figure(figsize=(5, 4))
ax = cuisine.value_counts().sort_index().plot(kind='bar',color='aqua')
plt.xlabel('Top Cuisine combination')
plt.ylabel('Counts')
plt.title('highest rating of cuisine')
plt.show()
```

```
count
0.010470    15
0.020940    15
0.041880     8
0.104701     8
0.031410     6
0.115171     5
0.062821     5
0.219872     5
0.083761     5
0.303633     3
Name: count, dtype: int64
```



Level 2 = Task 3 Task: Geographic Analysis

Plot the locations of restaurants on a map using longitude and latitude coordinates.

In [79]: `df.head(2)`

Out[79]:

	Restaurant ID	Restaurant Name	City	Longitude	Latitude	Cuisines	Table_booking	Online_delivery	Price range	Aggre ra
0	6317637	Le Petit Souffle	Makati City	121.027535	14.565443	French, Japanese, Desserts	Yes	No	3	
1	6304287	Izakaya Kikufuji	Makati City	121.014101	14.553708	Japanese	Yes	No	3	

```
In [80]: from shapely.geometry import Point
import geopandas as gpd
import matplotlib.pyplot as plt

# Create Point geometry from latitude and longitude using shapely
geometry = [Point(xy) for xy in zip(df['Longitude'], df['Latitude'])]
gdf = gpd.GeoDataFrame(df, geometry=geometry)

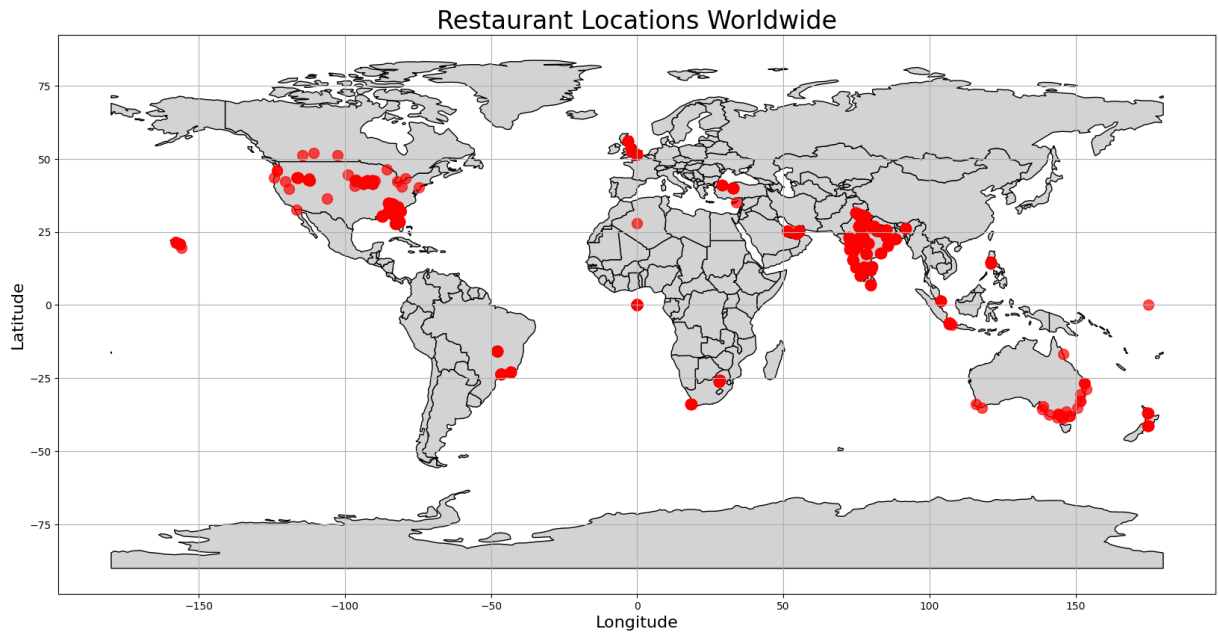
# Read world map data
world = gpd.read_file(gpd.datasets.get_path('naturalearth_lowres'))

# Plot the world map
ax = world.plot(figsize=(20, 12), color='lightgray', edgecolor='black')

# Plot the restaurants on top of the world map
gdf.plot(ax=ax, marker='o', color='red', markersize=100, alpha=0.7)

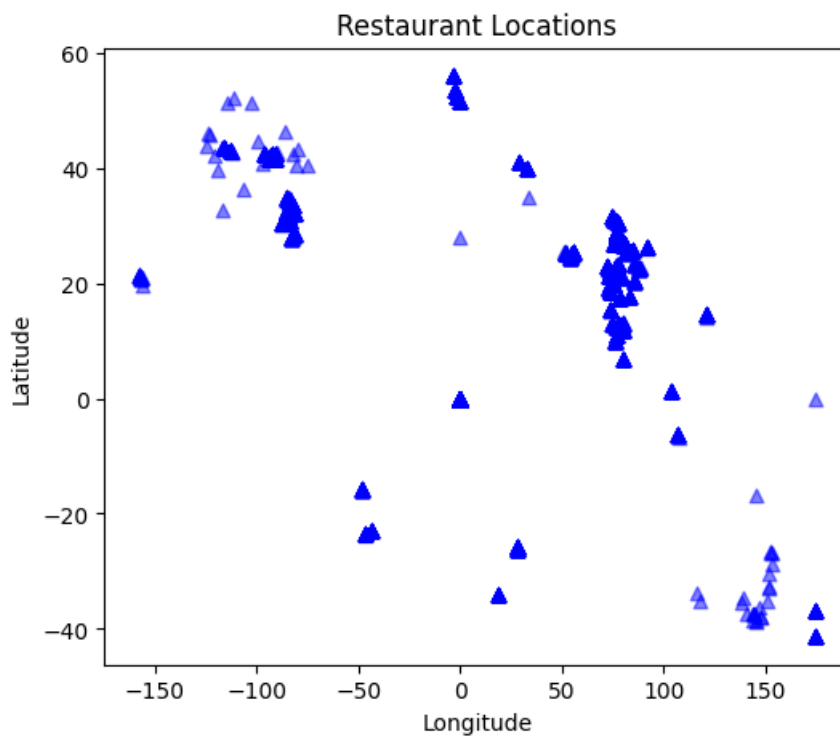
# Add additional features to the map (you can customize this based on your needs)
ax.set_title("Restaurant Locations Worldwide", fontsize=24)
ax.set_xlabel("Longitude", fontsize=16)
ax.set_ylabel("Latitude", fontsize=16)
plt.grid(True)

# Show the map
plt.show()
```



Identify any patterns or clusters of restaurants in specific areas.

```
In [81]: plt.figure(figsize=(6, 5))
plt.scatter(df['Longitude'], df['Latitude'], marker='^', alpha=0.5, c='b', cmap='viridis')
plt.title('Restaurant Locations')
plt.xlabel('Longitude')
plt.ylabel('Latitude')
plt.show()
```



Level 2 = Task 4 Task: Restaurant Chains

Identify if there are any restaurant chains present in the dataset.

```
In [82]: restaurant_chain = df['Restaurant Name'].value_counts()
print('The restaurant chain in the dataset', restaurant_chain[restaurant_chain > 1])
```

```

The restaurant chain in the dataset Restaurant Name
Cafe Coffee Day      83
Domino's Pizza       79
Subway               63
Green Chick Chop     51
McDonald's          48
..
Town Hall            2
Halki Aanch          2
Snack Junction       2
Delhi Biryani Hut    2
Beliram Degchiwala   2
Name: count, Length: 734, dtype: int64

```

```
In [83]: restaurant_chain.value_counts()
```

```

Out[83]: count
1      6712
2       468
3       108
4        46
5        28
6        18
7         13
8          7
9          6
19         4
18         4
14         4
11         3
22         3
13         3
12         3
20         2
16         2
10         1
83         1
79         1
26         1
28         1
29         1
30         1
34         1
48         1
51         1
63         1
15         1
Name: count, dtype: int64

```

Analyze the ratings and popularity of different restaurant chains.

```

In [84]: df = df.dropna(subset=['Aggregate rating', 'Votes'])
chain_stats = df.groupby('Restaurant Name').agg({'Aggregate rating': 'mean', 'Votes': 'sum'})

sorted_chain_stats_by_rating = chain_stats.sort_values(by='Aggregate rating', ascending=False)
sorted_chain_stats_by_votes = chain_stats.sort_values(by='Votes', ascending=False)
print(sorted_chain_stats_by_rating.head())
print(sorted_chain_stats_by_votes.head())

```

	Aggregate rating	Votes
Restaurant Name		
Ingleside Village Pizza	4.9	478
Ministry of Crab	4.9	203
Oakwood Cafe	4.9	249
Marukame Udon	4.9	602
Flat Iron	4.9	309

	Aggregate rating	Votes
Restaurant Name		
Barbeque Nation	4.353846	28142
AB's - Absolute Barbecues	4.825000	13400
Toit	4.800000	10934
Big Chill	4.475000	10853
Farzi Cafe	4.366667	10098

```
In [85]: chain_groups = df.groupby('Restaurant Name')
average_ratings = chain_groups['Aggregate rating'].mean()
total_votes = chain_groups['Votes'].sum()

rank_by_ratings = average_ratings.sort_values(ascending=False)
print(rank_by_ratings.head())
print()
rank_by_votes = total_votes.sort_values(ascending=False)
print(rank_by_votes.head())
```

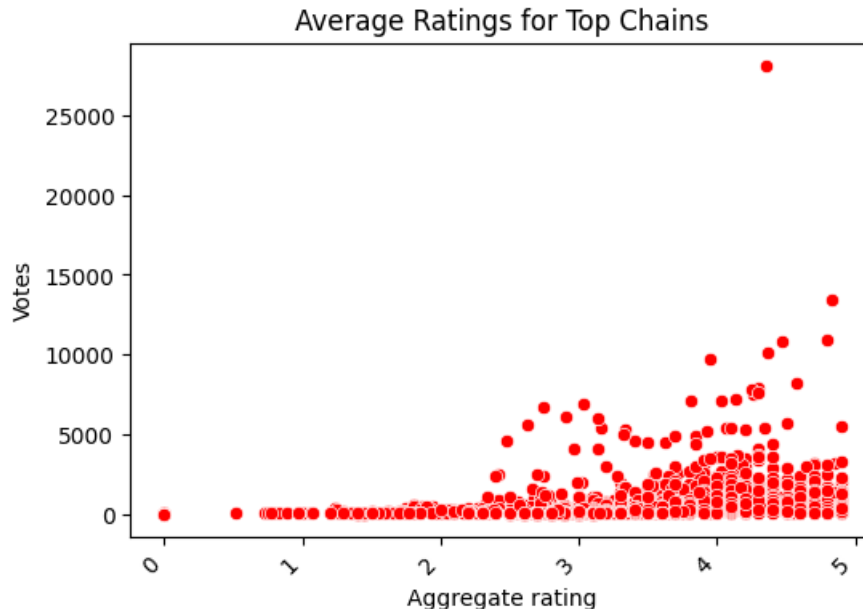
```
Restaurant Name
Ingleside Village Pizza    4.9
Ministry of Crab           4.9
Oakwood Cafe              4.9
Marukame Udon             4.9
Flat Iron                 4.9
Name: Aggregate rating, dtype: float64
```

```
Restaurant Name
Barbeque Nation           28142
AB's - Absolute Barbecues 13400
Toit                     10934
Big Chill                10853
Farzi Cafe               10098
Name: Votes, dtype: int64
```

```
In [86]: print(len(rank_by_ratings))
print(len(rank_by_votes))
```

```
7446
7446
```

```
In [87]: plt.figure(figsize=(6, 4))
sns.scatterplot(x=rank_by_ratings, y=rank_by_votes, color='r')
plt.xticks(rotation=45, ha='right')
plt.title('Average Ratings for Top Chains')
plt.show()
```



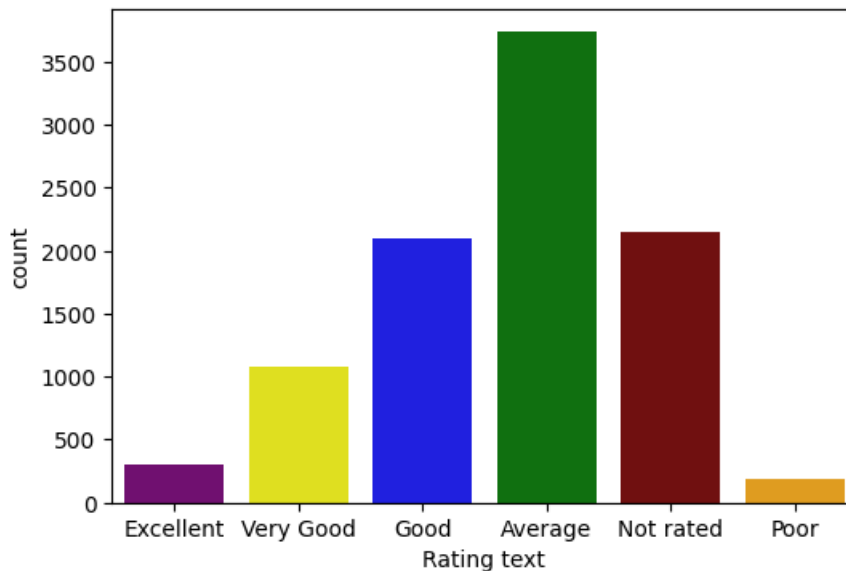
Level 3 = Task 1 Task: Restaurant Reviews

Analyze the text reviews to identify the most common positive and negative keywords.

```
In [88]: df['Rating text'].value_counts()
```

```
Out[88]: Rating text
Average      3737
Not rated    2148
Good         2100
Very Good    1079
Excellent     301
Poor         186
Name: count, dtype: int64
```

```
In [89]: plt.figure(figsize=(6, 4))
colors = ["purple", "yellow", "blue", "green", 'maroon', 'orange']
sns.countplot(x=df['Rating text'], palette=colors)
plt.show()
```



Calculate the average length of reviews and explore if there is a relationship between review length and rating.

```
In [90]: a = df['Aggregate rating'].value_counts().mean()
a
```

```
Out[90]: 289.42424242424244
```

```
In [91]: df.columns
```

```
Out[91]: Index(['Restaurant ID', 'Restaurant Name', 'City', 'Longitude', 'Latitude',
              'Cuisines', 'Table_booking', 'Online_delivery', 'Price range',
              'Aggregate rating', 'Rating text', 'Votes'],
              dtype='object')
```

```
In [92]: import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Step 1: Calculate Review Length (using 'Cuisines' as a proxy)
df['Review Length'] = df['Cuisines'].apply(lambda x: len(str(x)))

# Step 2: Explore Review Length Distribution
plt.figure(figsize=(7, 5))
sns.histplot(data=df, x='Review Length', bins=30, kde=True)
plt.title('Distribution of Review Lengths')
plt.xlabel('Review Length')
plt.ylabel('Frequency')
plt.show()

# Step 3: Calculate Average Review Length
average_review_length = df['Review Length'].mean()
print(f'Average Review Length: {average_review_length:.2f} characters')

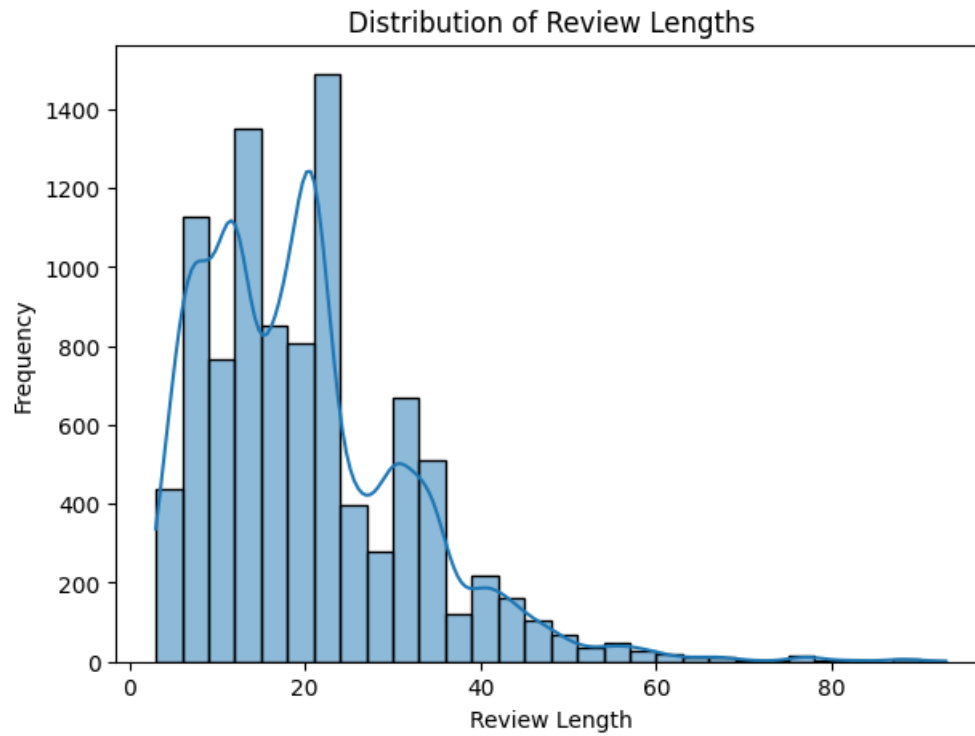
# Step 4: Explore Relationship with Ratings
plt.figure(figsize=(7, 5))
sns.scatterplot(data=df, x='Aggregate rating', y='Review Length')
```

```
plt.title('Relationship between Review Length and Ratings')
plt.xlabel('Aggregate Rating')
plt.ylabel('Review Length')
plt.show()

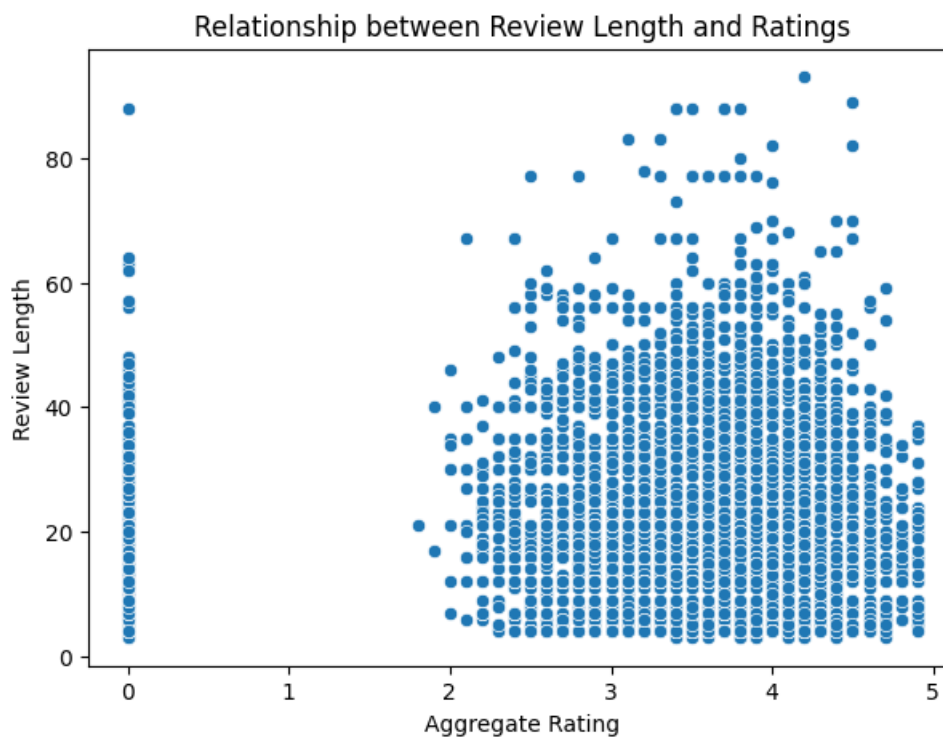
# Step 5: Correlation Analysis
correlation_coefficient = df[['Aggregate rating', 'Review Length']].corr().iloc[0, 1]
print(f'Correlation Coefficient: {correlation_coefficient:.2f}')

# Step 6: Statistical Analysis (Optional)
from scipy.stats import pearsonr

correlation, p_value = pearsonr(df['Aggregate rating'], df['Review Length'])
print(f'Correlation: {correlation:.2f}, p-value: {p_value:.4f}')
```



Average Review Length: 19.91 characters



Correlation Coefficient: 0.19

Correlation: 0.19, p-value: 0.0000

Level 3 = Task 2 Task: Votes Analysis

Identify the restaurants with the highest and lowest number of votes.

```
In [93]: Restaurant_votes = df.groupby(['Restaurant Name'])['Votes'].mean()
print(Restaurant_votes.max())
print(Restaurant_votes.min())
```

```
10934.0
0.0
```

```
In [94]: Restaurant_highest_votes = df.loc[df['Votes'].idxmax()]
print('Restaurant with the highest vote number is:')
print(Restaurant_highest_votes[['Restaurant Name', 'Votes']])
print('\n')
Restaurant_lowest_votes = df.loc[df['Votes'].idxmin()]
print('Restaurant with the lowest vote number is:')
print(Restaurant_lowest_votes[['Restaurant Name', 'Votes']])
```

```
Restaurant with the highest vote number is:
Restaurant Name    Toit
Votes              10934
Name: 728, dtype: object
```

```
Restaurant with the lowest vote number is:
Restaurant Name    Cantinho da Gula
Votes              0
Name: 69, dtype: object
```

Analyze if there is a correlation between the number of votes and the rating of a restaurant.

```
In [95]: df[['Votes', 'Aggregate rating']].head()
```

```
Out[95]:
```

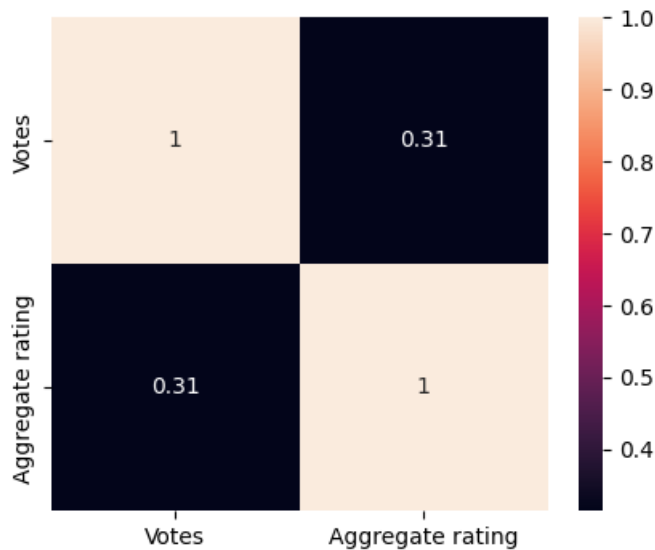
	Votes	Aggregate rating
0	314	4.8
1	591	4.5
2	270	4.4
3	365	4.9
4	229	4.8

```
In [96]: correlation = df[['Votes', 'Aggregate rating']].corr()
correlation
```

```
Out[96]:
```

	Votes	Aggregate rating
Votes	1.000000	0.313691
Aggregate rating	0.313691	1.000000

```
In [97]: plt.figure(figsize=(5, 4))
sns.heatmap(correlation, annot=True)
plt.show()
```

Level 3 = Task 3 Task: Price Range vs. Online Delivery and Table Booking

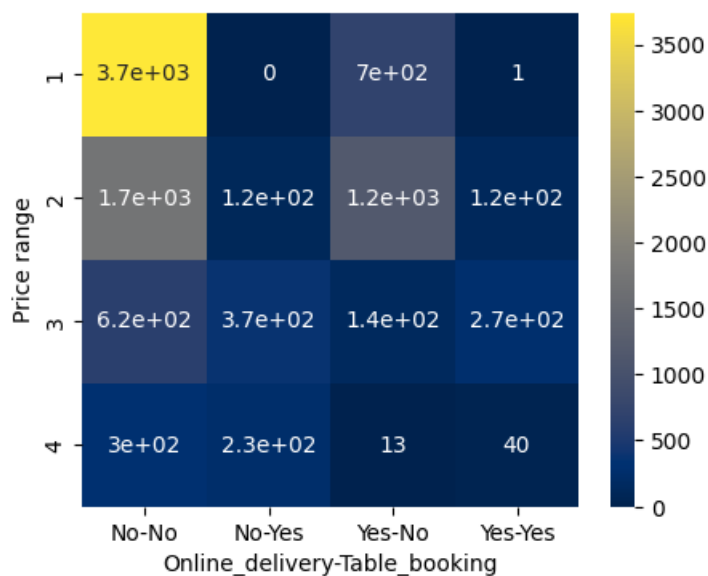
Analyze if there is a relationship between the price range and the availability of online delivery and table booking.

```
In [98]: cross_tab = pd.crosstab(index=df['Price range'], columns=[df['Online_delivery'], df['Table_booking']])
cross_tab
```

```
Out[98]:
```

Price range	Online_delivery		Table_booking	
	No	Yes	No	Yes
	1	3743	0	700
2	1711	116	1163	123
3	624	373	140	271
4	299	234	13	40

```
In [99]: plt.figure(figsize=(5, 4))
sns.heatmap(cross_tab, annot=True, cmap='cividis')
plt.show()
```



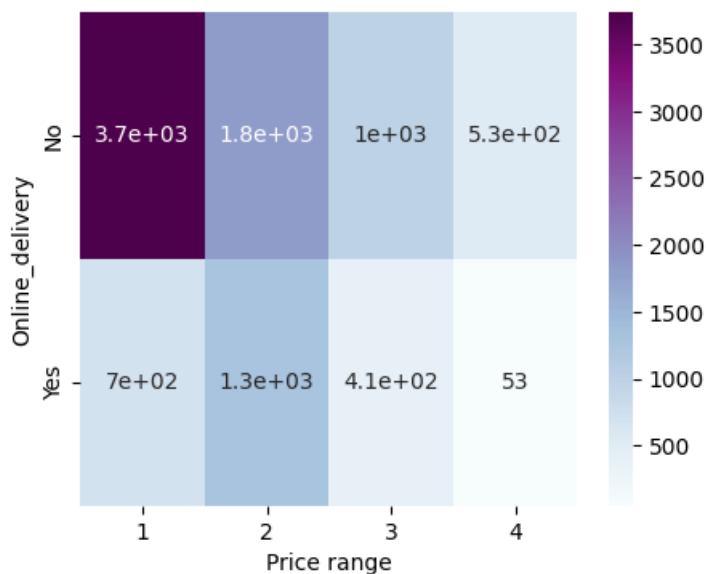
```
In [100]: df['Table_booking'].unique()
```

```
Out[100]: array(['Yes', 'No'], dtype=object)
```

```
In [101]: df.groupby('Online_delivery')['Price range'].mean()
```

```
Out[101]: Online_delivery  
No      1.763380  
Yes     1.924929  
Name: Price range, dtype: float64
```

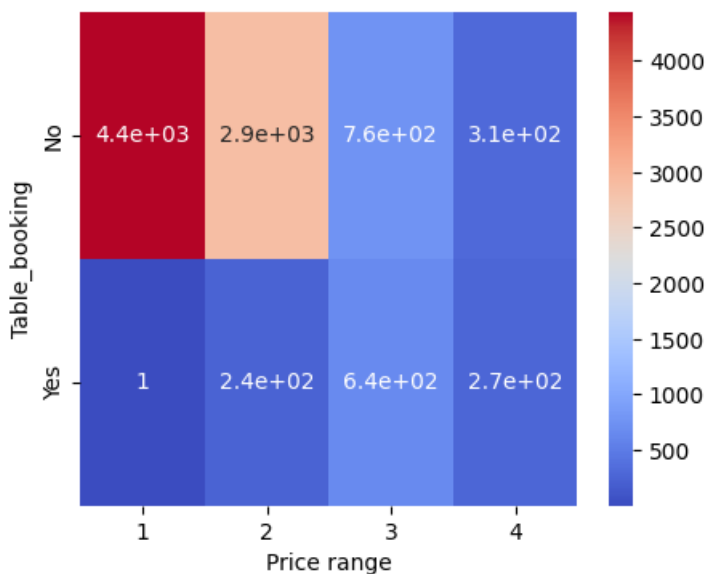
```
In [102]: crosstab = pd.crosstab(df['Online_delivery'], df['Price range'])  
plt.figure(figsize=(5, 4))  
sns.heatmap(crosstab, annot=True, cmap='BuPu')  
plt.show()
```



```
In [103]: df.groupby('Table_booking')['Price range'].mean()
```

```
Out[103]: Table_booking  
No      1.636006  
Yes     3.028497  
Name: Price range, dtype: float64
```

```
In [104]: crosstab = pd.crosstab(df['Table_booking'], df['Price range'])  
plt.figure(figsize=(5, 4))  
sns.heatmap(crosstab, annot=True, cmap='coolwarm')  
plt.show()
```



Determine if higher-priced restaurants are more likely to offer these services.

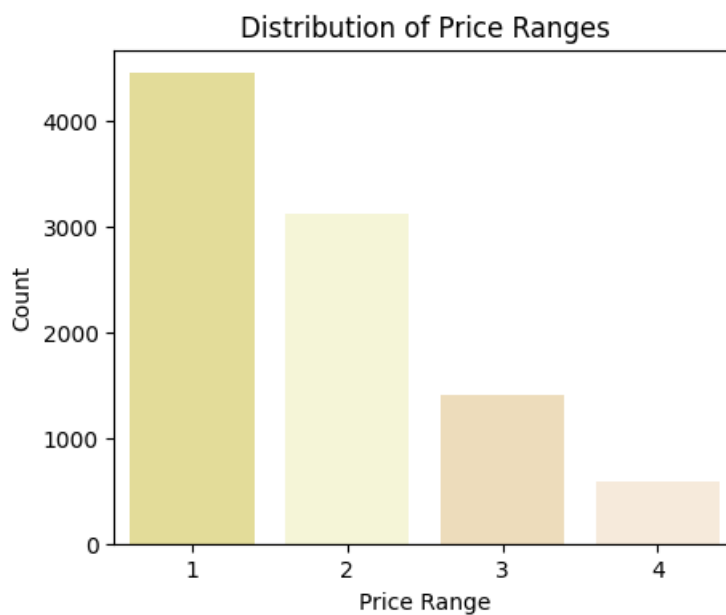
```
In [105... df['Price range'].max()
```

```
Out[105]: 4
```

```
In [106... df['Online_delivery'].value_counts()
```

```
Out[106]: Online_delivery
No      7100
Yes     2451
Name: count, dtype: int64
```

```
In [107... plt.figure(figsize=(5, 4))
custom_palette = ["khaki", "lightgoldenrodyellow", "wheat", "antiquewhite"]
sns.countplot(x='Price range', data=df, palette=custom_palette)
plt.title('Distribution of Price Ranges')
plt.xlabel('Price Range')
plt.ylabel('Count')
plt.show()
```



```
In [108... table_booking_proportion = df['Table_booking'].value_counts(normalize=True)
online_delivery_proportion = df['Online_delivery'].value_counts(normalize=True)

print("Proportion of Restaurants Offering Table Booking:")
print(table_booking_proportion)

print("\nProportion of Restaurants Offering Online Delivery:")
print(online_delivery_proportion)
```

```
Proportion of Restaurants Offering Table Booking:
Table_booking
No      0.878756
Yes     0.121244
Name: proportion, dtype: float64
```

```
Proportion of Restaurants Offering Online Delivery:
Online_delivery
No      0.743378
Yes     0.256622
Name: proportion, dtype: float64
```

```
In [109... plt.figure(figsize=(12, 4))

plt.subplot(1, 2, 1)
custom_palette = ["gold", "maroon"]
sns.countplot(x='Price range', hue='Table_booking', data=df, palette=custom_palette)
plt.title('Relationship between Price Range and Table Booking')

plt.subplot(1, 2, 2)
```

```
custom_palette = ["gold", "maroon"]
sns.countplot(x='Price range', hue='Online_delivery', data=df, palette=custom_palette)
plt.title('Relationship between Price Range and Online Delivery')

plt.show()
```

