```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

```
df = pd.read_excel('/content/drive/MyDrive/Data Analysis Project - 1/data.xlsx')
```

```
df.head()
```

	Restaurant ID	Restaurant Name	Country Code	City	Address	Locality	Locality Verbose	Longitude	Latitude	Cuisines	Average Cost for two	Currency	T boo
0	7402935	Skye	94	Jakarta	Menara BCA, Lantai 56, Jl. MH. Thamrin, Thamri...	Grand Indonesia Mall, Thamrin	Grand Indonesia Mall, Thamrin, Jakarta	106.821999	-6.196778	Italian, Continental	800000	Indonesian Rupiah(IDR)	
1	7410290	Satoo - Hotel Shangri-La	94	Jakarta	Hotel Shangri-La, Jl. Jend. Sudirman	Hotel Shangri-La, Sudirman	Hotel Shangri-La, Sudirman, Jakarta	106.818961	-6.203292	Asian, Indonesian, Western	800000	Indonesian Rupiah(IDR)	
2	7420899	Sushi Masa	94	Jakarta	Jl. Tuna Raya No. 5, Penjaringan	Penjaringan	Penjaringan, Jakarta	106.800144	-6.101298	Sushi, Japanese	500000	Indonesian Rupiah(IDR)	
3	7421967	3 Wise Monkeys	94	Jakarta	Jl. Suryo No. 26, Senopati, Jakarta	Senopati	Senopati, Jakarta	106.813400	-6.235241	Japanese	450000	Indonesian Rupiah(IDR)	
4	7422489	Avec Moi Restaurant and Bar	94	Jakarta	Gedung PIC, Jl. Teluk Betung 43, Thamrin, Jakarta	Thamrin	Thamrin, Jakarta	106.821023	-6.196270	French, Western	350000	Indonesian Rupiah(IDR)	

Perform preliminary data inspection and report the findings as the structure of the data, missing values, duplicates, etc.

```
structure = df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9551 entries, 0 to 9550
Data columns (total 19 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   Restaurant ID         9551 non-null   int64
 1   Restaurant Name       9550 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  Price range           9551 non-null   int64
 15  Aggregate rating      9551 non-null   float64
 16  Rating color          9551 non-null   object
 17  Rating text           9551 non-null   object
 18  Votes                 9551 non-null   int64
dtypes: float64(3), int64(5), object(11)
memory usage: 1.4+ MB
```

```
df['Has Table booking'].head()
```

	Has Table booking
0	No
1	No
2	No
3	No
4	No

dtype: object

```
df['Has Online delivery'].head()
```

	Has Online delivery
0	No
1	No
2	No
3	No
4	No

dtype: object

```
duplicates = df.duplicated().sum()
```

```
missing_values = df.isnull().sum()
```

convert " " to "_" and in lower case of all column name

```
df.columns = df.columns.str.lower().str.replace(' ', '_')
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9551 entries, 0 to 9550
Data columns (total 19 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   restaurant_id         9551 non-null   int64
 1   restaurant_name       9550 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  price_range           9551 non-null   int64
 15  aggregate_rating      9551 non-null   float64
 16  rating_color          9551 non-null   object
 17  rating_text           9551 non-null   object
 18  votes                 9551 non-null   int64
dtypes: float64(3), int64(5), object(11)
memory usage: 1.4+ MB
```

2. Based on the findings from the previous questions, identify duplicates and remove them

```
df_cleaned = df.dropna(subset= ['restaurant_name'])
```

```
# Summary after cleaning
```

```
summary_after_cleaning = {
```

```
    'missing_values_after_cleaning': df_cleaned.isnull().sum(),
    'duplicates_after_cleaning': df_cleaned.duplicated().sum(),
    'structure_after_cleaning': df_cleaned.info(),
    'unique_has_table_booking': df_cleaned['has_table_booking'].unique(),
    'unique_has_online_delivery': df_cleaned['has_online_delivery'].unique()
}
print(structure)
print('missing_values_before_cleaning:', missing_values)
print('duplicates_before_cleaning:',duplicates)
print('variable_names_after_cleaining:', df_cleaned.columns)
print('summary_after_cleaining:', summary_after_cleaning)
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 9550 entries, 0 to 9550
Data columns (total 19 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   restaurant_id         9550 non-null   int64
 1   restaurant_name       9550 non-null   object
 2   country_code          9550 non-null   int64
 3   city                  9550 non-null   object
 4   address               9550 non-null   object
 5   locality              9550 non-null   object
 6   locality_verbose      9550 non-null   object
 7   longitude             9550 non-null   float64
 8   latitude              9550 non-null   float64
 9   cuisines              9541 non-null   object
 10  average_cost_for_two  9550 non-null   int64
 11  currency              9550 non-null   object
 12  has_table_booking     9550 non-null   object
 13  has_online_delivery   9550 non-null   object
 14  price_range           9550 non-null   int64
 15  aggregate_rating      9550 non-null   float64
 16  rating_color          9550 non-null   object
 17  rating_text           9550 non-null   object
 18  votes                 9550 non-null   int64
dtypes: float64(3), int64(5), object(11)
memory usage: 1.5+ MB
None
missing_values_before_cleaning: Restaurant ID           0
Restaurant Name         1
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
Price range             0
Aggregate rating        0
Rating color            0
Rating text             0
Votes                   0
dtype: int64
duplicates_before_cleaning: 0
variable_names_after_cleaining: 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', 'price_range', 'aggregate_rating',
       'rating_color', 'rating_text', 'votes'],
      dtype='object')
summary_after_cleaining: {'missing_values_after_cleaning': restaurant_id           0
restaurant_name         0
country_code            0
city                    0
```

```
df_country =pd.read_excel('/content/drive/MyDrive/Data Analysis Project - 1/Country-Code.xlsx')
```

```
df_country.head()
```

	Country Code	Country
0	1	India
1	14	Australia
2	30	Brazil
3	37	Canada
4	94	Indonesia

```
df_country.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 15 entries, 0 to 14
Data columns (total 2 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   Country Code  15 non-null     int64
 1   Country       15 non-null     object
dtypes: int64(1), object(1)
memory usage: 372.0+ bytes
```

```
df_country.duplicated().sum()
```

```
np.int64(0)
```

Performing EDA

1-Explore the geographical distribution of the restaurants and identify the cities with the maximum and minimum number of restaurants

```
city_counts = df['city'].value_counts()
city_counts.head()
```

city	count
New Delhi	5473
Gurgaon	1118
Noida	1080
Faridabad	251
Ghaziabad	25

dtype: int64

```
max_restaurants_city = city_counts.idxmax()
max_restaurants_counts = city_counts.max()
min_restaurants_city = city_counts.idxmin()
min_restaurants_counts = city_counts.min()

print(f"The city with the maximum number of restaurants is: {max_restaurants_city} ({max_restaurants_counts} restaurants)")
print(f'The city with the minimum number of restaurants is: {min_restaurants_city}({min_restaurants_counts} restaurants)')
```

```
The city with the maximum number of restaurants is: New Delhi (5473 restaurants)
The city with the minimum number of restaurants is: Bandung(1 restaurants)
```

2- Restaurant franchising is a thriving venture. So, it is very important to explore the franchise with most national presence

```
# Count the number of cities in which our restaurants presence.
franchise_presence = df_cleaned.groupby('restaurant_name')['city'].nunique().sort_values(ascending=False)

# Count the restaurants with most national presence.
top_franchise = franchise_presence.idxmax()
top_franchise_count = franchise_presence.max()

print(f"The restaurant with the most national presence is: {top_franchise} (present in {top_franchise_count} cities)")
```

```
The restaurant with the most national presence is: Barbeque Nation (present in 22 cities)
```

3- Find out the ratio between restaurants that allow table booking vs. those that do not allow table booking

```
# Convert upper case into lower case
df_cleaned['has_table_booking']= df_cleaned['has_table_booking'].str.strip().str.lower()
# Number of table booking 'Yes' and 'No'
table_booking= df_cleaned['has_table_booking'].value_counts()
# Cities that allow table booking.
table_booking_count_yes = table_booking.get('yes',0)

# cities that don't allow table booking.
table_booking_count_no = table_booking.get('no',0)

table_booking_ratio = (table_booking_count_yes / table_booking_count_no )*100
print(f"The ratio between restaurants that allow table booking vs. those that do not allow table booking is: {table_booking_ratio:.2f}%")
```

```
The ratio between restaurants that allow table booking vs. those that do not allow table booking is: 13.80%
```

4- Find out the percentage of restaurants providing online delivery

```
df_cleaned['has_online_delivery']=df_cleaned['has_online_delivery'].str.strip().str.lower()
# Number of 'has online delivery'
delivery_count = df_cleaned['has_online_delivery'].value_counts()
# Number of 'has online delivery' - Yes
online_delivery_yes = delivery_count.get('yes',0)
# Number of 'has online delivery' - No
online_delivery_no = delivery_count.get('no',0)

# % of online delivery 'Yes'
percent_online_delivery_yes = (online_delivery_yes / (online_delivery_yes + online_delivery_no))*100
print(f"The percentage of restaurants providing online delivery is: {percent_online_delivery_yes:.2f}%")
```

```
The percentage of restaurants providing online delivery is: 25.66%
```

5- Calculate the difference in number of votes for the restaurants that deliver and the restaurants that do not deliver

```
online_delivery_votes_yes= df_cleaned[df_cleaned['has_online_delivery'] =='yes']['votes']
online_delivery_votes_no= df_cleaned[df_cleaned['has_online_delivery'] =='no']['votes']

average_votes_yes = online_delivery_votes_yes.mean()
average_votes_no = online_delivery_votes_no.mean()

print(f"The number of votes for the restaurants that deliver is: {average_votes_yes: 2f}")
print(f"The number of votes for the restaurants that do not deliver is: {average_votes_no: 2f}")

delivery_votes_difference = average_votes_yes - average_votes_no
print(f"The difference in number of votes for the restaurants that deliver and the restaurants that do not deliver is: {delivery_votes_diffe
```

```
The number of votes for the restaurants that deliver is:  211.307222
The number of votes for the restaurants that do not deliver is:  138.042259
The difference in number of votes for the restaurants that deliver and the restaurants that do not deliver is:  73.264962
```

```
df_cleaned.columns
```

```
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', 'price_range', 'aggregate_rating',
       'rating_color', 'rating_text', 'votes'],
      dtype='object')
```

Week 3

1- What are the top 10 cuisines served across cities?

```
# Split by ',' and place into columns for each item and remove whitespace
cuisines_series = df['cuisines'].str.split(',').explode().str.strip()

# Count the occurance of each cuisines.
top_10_cuisines = cuisines_series.value_counts().head(10)
```

```
print(f'top 10 cuisiness:{top_10_cuisines}')
```

```
top 10 cuisiness:cuisines
North Indian    3960
Chinese         2735
Fast Food       1986
Mughlai          995
Italian          764
Bakery           745
Continental      736
Cafe             703
Desserts         653
South Indian     636
Name: count, dtype: int64
```

2- What is the maximum and minimum number of cuisines that a restaurant serves? Also, which is the most served cuisine across the restaurant for each city?

```
# Count number of cuisines per restaurant.
df['count_cuisines'] = df['cuisines'].fillna('').str.split(',').apply(len)
max_cuisines = df['count_cuisines'].max()
min_cuisines = df['count_cuisines'].min()
print(f"The maximum number of cuisines that a restaurant serves is: {max_cuisines}")
print(f"The minimum number of cuisines that a restaurant serves is: {min_cuisines}")
```

```
The maximum number of cuisines that a restaurant serves is: 8
The minimum number of cuisines that a restaurant serves is: 1
```

```
# Step 1: Copy original DataFrame
df1 = df.copy()

# Step 2: Handle missing values before splitting
df1['cuisines'] = df1['cuisines'].fillna('')

# Step 3: Split, explode, strip and create a new DataFrame
df1_exploded = df1.assign(cuisines=df1['cuisines'].str.split(',')).explode('cuisines')
df1_exploded['cuisines'] = df1_exploded['cuisines'].str.strip()

# Step 4: Remove empty cuisine values (optional but safe)
df1_exploded = df1_exploded[df1_exploded['cuisines'] != '']

# Step 5: Group by city and cuisines
cuisines_counts = df1_exploded.groupby(['city', 'cuisines']).size().reset_index(name='count')

# Step 6: Get the most served cuisine per city
most_served_cuisines_city = cuisines_counts.loc[cuisines_counts.groupby('city')['count'].idxmax()]

# Step 7: Display
print(most_served_cuisines_city)
```

```
                  city      cuisines  count
11          Abu Dhabi        Indian      7
37               Agra  North Indian     15
48          Ahmedabad   Continental     12
62             Albany      American      4
91          Allahabad  North Indian     12
...               ...           ...    ...
1815          Weirton        Burger      1
1820  Wellington City          Cafe      9
1835   Winchester Bay        Burger      1
1838          Yorkton         Asian      1
1842        €¡stanbul          Cafe      4

[140 rows x 3 columns]
```

3- What is the distribution cost across the restaurants?

```
df_cleaned.columns
```

```
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', 'price_range', 'aggregate_rating',
       'rating_color', 'rating_text', 'votes'],
      dtype='object')
```

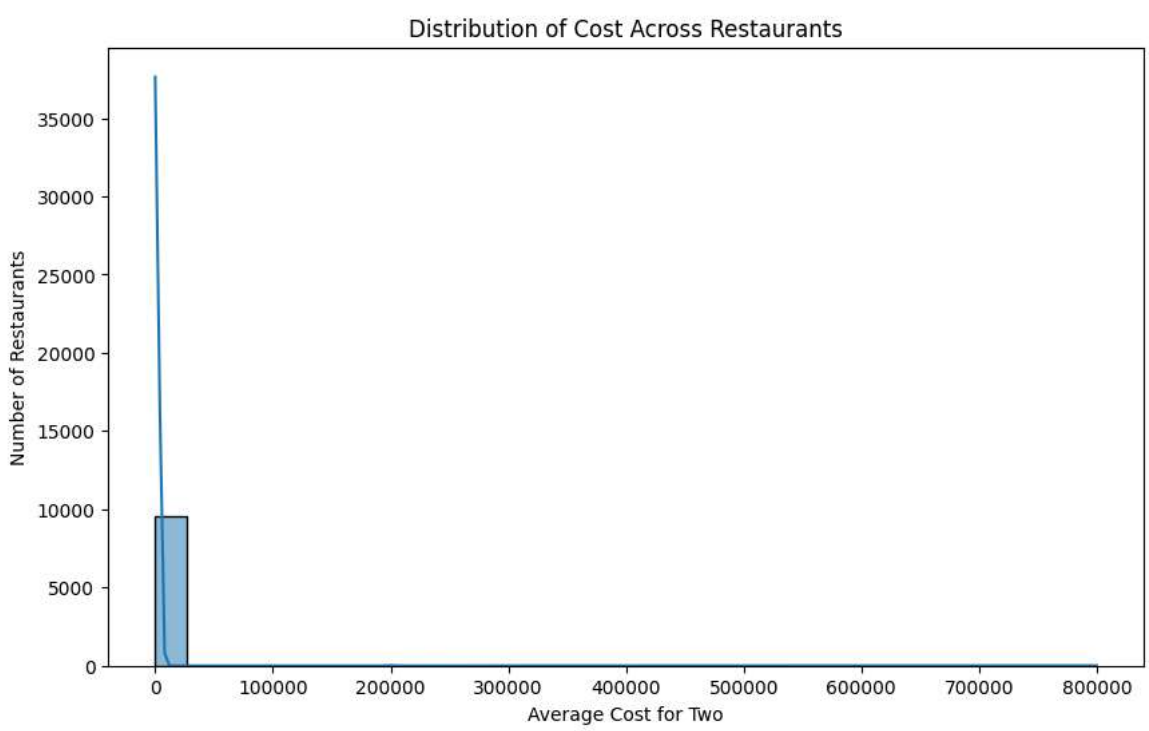
```
print(df_cleaned['average_cost_for_two'].describe())
```

```
count      9550.000000
mean       1199.252565
std       16122.026663
min           0.000000
25%         250.000000
50%         400.000000
75%         700.000000
max      800000.000000
Name: average_cost_for_two, dtype: float64
```

```
cost_distribution = df_cleaned['average_cost_for_two'].sort_index()
print(cost_distribution)
```

```
0       800000
1       800000
2       500000
3       450000
4       350000
         ...
9546         0
9547         0
9548         0
9549         0
9550         0
Name: average_cost_for_two, Length: 9550, dtype: int64
```

```
import matplotlib.pyplot as plt
import seaborn as sns

plt.figure(figsize=(10, 6))
sns.histplot(df_cleaned['average_cost_for_two'], bins=30, kde=True)
plt.title('Distribution of Cost Across Restaurants')
plt.xlabel('Average Cost for Two')
plt.ylabel('Number of Restaurants')
plt.show()
```



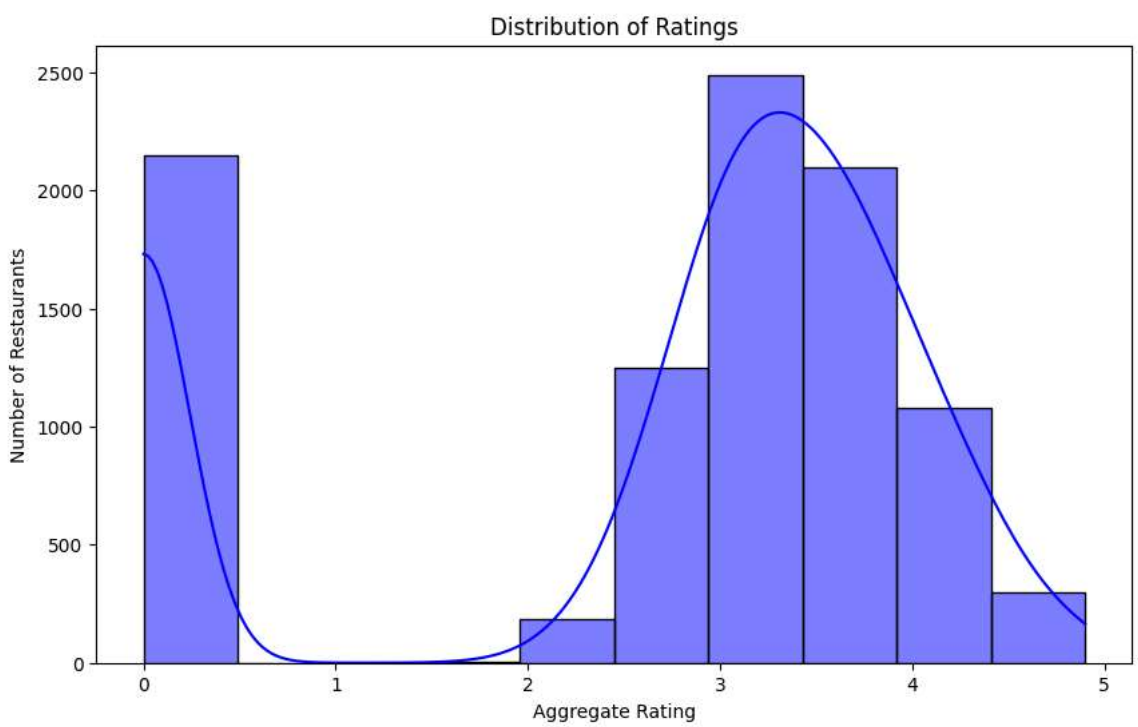
Double-click (or enter) to edit

4- How ratings are distributed among the various factors?

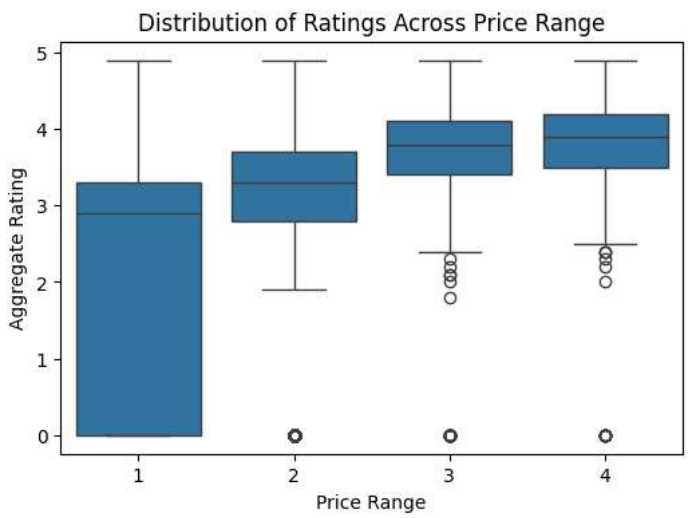
```
# Overall distribution of ratings.
overall_rating_distribution = df_cleaned['aggregate_rating'].value_counts().sort_index()
print(overall_rating_distribution)
```

```
aggregate_rating
0.0    2148
1.8       1
1.9       2
2.0       7
2.1      15
2.2      27
2.3      47
2.4      87
2.5     110
2.6     191
2.7     250
2.8     315
2.9     381
3.0     468
3.1     519
3.2     522
3.3     483
3.4     498
3.5     480
3.6     458
3.7     427
3.8     400
3.9     335
4.0     266
4.1     273
4.2     221
4.3     174
4.4     144
4.5      95
4.6      78
4.7      42
4.8      25
4.9      61
Name: count, dtype: int64
```

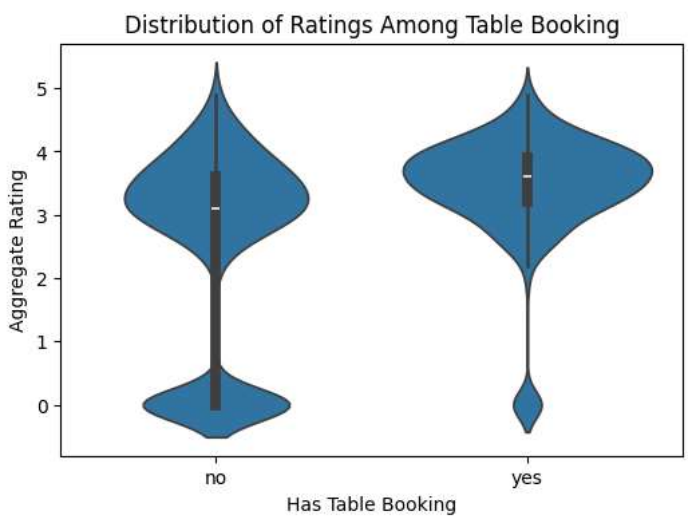
```
# overall distribution of ratings.
plt.figure(figsize=(10, 6))
ax = sns.histplot(df_cleaned['aggregate_rating'], bins = 10, kde = True, color = 'blue')
plt.title('Distribution of Ratings')
plt.xlabel('Aggregate Rating')
plt.ylabel('Number of Restaurants')
plt.show()
```



```
# overall distribution of rating among price_range
plt.figure(figsize = (6,4))
sns.boxplot(x = 'price_range', y = 'aggregate_rating', data = df_cleaned)
plt.title('Distribution of Ratings Across Price Range')
plt.xlabel('Price Range')
plt.ylabel('Aggregate Rating')
plt.show()
```

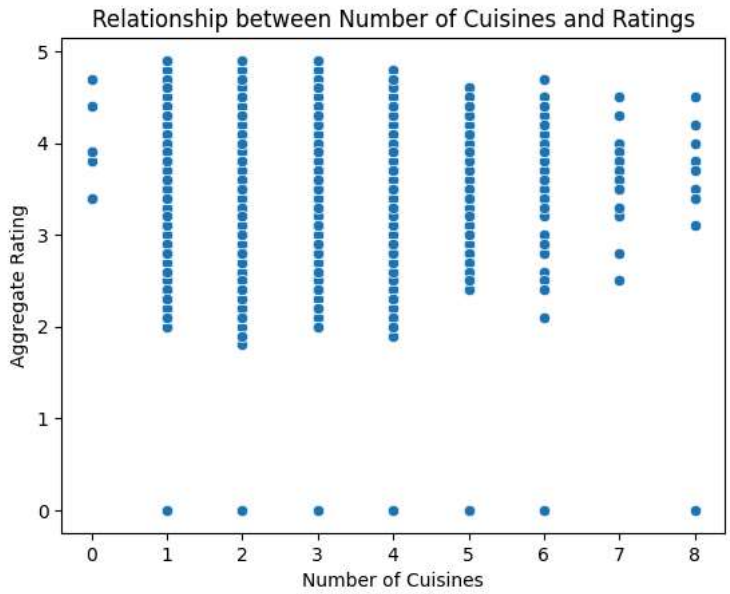



```
# Compare ratings among 'has_table_booking'
plt.figure(figsize = (6,4))
sns.violinplot(x = 'has_table_booking', y = 'aggregate_rating', data = df_cleaned)
plt.title('Distribution of Ratings Among Table Booking')
plt.xlabel('Has Table Booking')
plt.ylabel('Aggregate Rating')
plt.show()
```



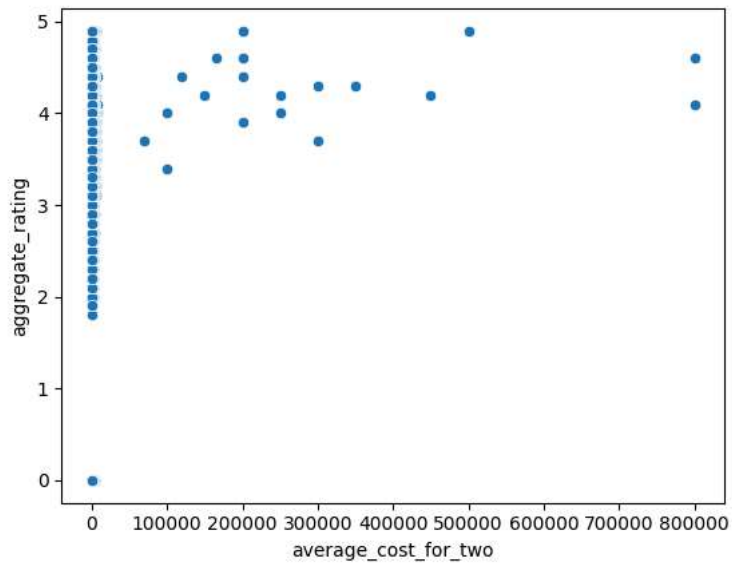
5- Explain the factors in the data that may have an effect on ratings. For example, number of cuisines, cost, delivery option, etc.

```
df_cleaned['num_cuisines'] = df_cleaned['cuisines'].fillna('').apply(lambda x: len(x.split(',')) if x != '' else 0)

import seaborn as sns
import matplotlib.pyplot as plt

sns.scatterplot(x='num_cuisines', y='aggregate_rating', data=df_cleaned)
plt.title('Relationship between Number of Cuisines and Ratings')
plt.xlabel('Number of Cuisines')
plt.ylabel('Aggregate Rating')
plt.show()
```




```
sns.scatterplot(x='average_cost_for_two', y='aggregate_rating', data=df_cleaned)
```

```
<Axes: xlabel='average_cost_for_two', ylabel='aggregate_rating'>
```




```
sns.boxplot(x='has_online_delivery', y='aggregate_rating', data=df_cleaned)
sns.boxplot(x='has_table_booking', y='aggregate_rating', data=df_cleaned)
```

```
<Axes: xlabel='has_online_delivery', ylabel='aggregate_rating'>
```