DATA ANALYSIS TASK LIST

Level 1

Task 1

Task: Top Cuisines

Determine the top three most common cuisines in the dataset.

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

In [3]:
```python
import pandas as pd
import matplotlib.pyplot as plt
import plotly.express as px
```

In [4]:
```python
df = pd.read_csv(r"C:\Users\devan\OneDrive\Desktop\project.py\project.py\Dataset .csv")
df.columns = df.columns.str.strip()
```

In [5]:
```python
df.head()
```

Out[5]:

	Restaurant ID	Restaurant Name	Country Code	City	Address	Locality	Locality Verbose	Longitude	Latitude	Cuisines	...	Currency	Has Table booking	Has Online delivery	Is delivering now	Switch to order menu	Price range	Aggregate rating	Rating color	Rating text	Votes
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	...	Botswana Pula(P)	Yes	No	No	No	3	4.8	Dark Green	Excellent	314
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	...	Botswana Pula(P)	Yes	No	No	No	3	4.5	Dark Green	Excellent	591
2	6300002	Heat - Edsa Shangri-La	162	Mandaluyong City	Edsa Shangri-La, 1 Garden Way, Ortigas, Mandal...	Edsa Shangri-La, Ortigas, Mandaluyong City	Edsa Shangri-La, Ortigas, Mandaluyong City, Ma...	121.056831	14.581404	Seafood, Asian, Filipino, Indian	...	Botswana Pula(P)	Yes	No	No	No	4	4.4	Green	Very Good	270
3	6318506	Ooma	162	Mandaluyong City	Third Floor, Mega Fashion Hall, SM Megamall, O...	SM Megamall, Ortigas, Mandaluyong City	SM Megamall, Ortigas, Mandaluyong City, Mandal...	121.056475	14.585318	Japanese, Sushi	...	Botswana Pula(P)	No	No	No	No	4	4.9	Dark Green	Excellent	365
4	6314302	Sambo Kojin	162	Mandaluyong City	Third Floor, Mega Atrium, SM Megamall, Ortigas...	SM Megamall, Ortigas, Mandaluyong City	SM Megamall, Ortigas, Mandaluyong City, Mandal...	121.057508	14.584450	Japanese, Korean	...	Botswana Pula(P)	Yes	No	No	No	4	4.8	Dark Green	Excellent	229

5 rows × 21 columns

In [6]:
```python
print(df['Cuisines'].value_counts().head(3))
```

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

In [7]:
```python
top_cuisines = df['Cuisines'].value_counts().head(3)
total = len(df)
percentages = (top_cuisines / total) * 100
print(percentages)
```

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

Level 1

Task 2

Task: City Analysis

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

Calculate the average rating for restaurants in each city.

Determine the city with the highest average rating.

In [8]:
```python
city_counts = df['City'].value_counts()
print("City with highest number of restaurants:\n", city_counts.head(1))
```

```
City with highest number of restaurants:
 City
New Delhi    5473
Name: count, dtype: int64
```

In [9]:
```python
avg_rating_per_city = df.groupby('City')['Aggregate rating'].mean()
print("\nAverage rating per city:\n", avg_rating_per_city)
```

```
Average rating per city:
 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
İstanbul           4.292857
Name: Aggregate rating, Length: 141, dtype: float64
```

In [10]:
```python
top_avg_city = avg_rating_per_city.sort_values(ascending=False)
print("\nCity with highest average rating:\n", top_avg_city.head(1))
```

```
City with highest average rating:
 City
Inner City    4.9
Name: Aggregate rating, dtype: float64
```

Level Task 3 Task: Price Range Distribution

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

Calculate the percentage of restaurants in each price range category.

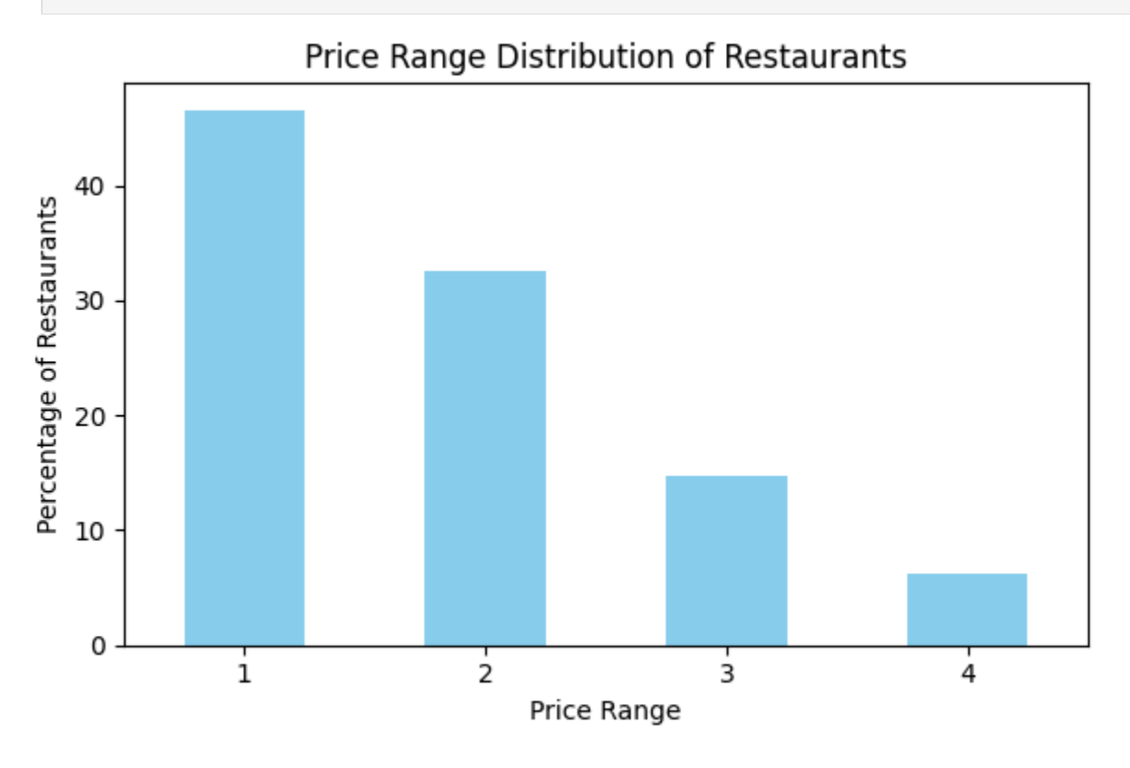
In [11]:
```python
price_counts = df['Price range'].value_counts().sort_index()
print("Restaurant count by price range:\n", price_counts)
```

```
Restaurant count by price range:
 Price range
1    4444
2    3113
3    1408
4     586
Name: count, dtype: int64
```

In [12]:
```python
total = len(df)
price_percent = (price_counts / total) * 100
print("\nPercentage by price range:\n", price_percent)
```

```
Percentage by price range:
 Price range
1    46.529159
2    32.593446
3    14.741912
4     6.135483
Name: count, dtype: float64
```

In [13]:
```python
plt.figure(figsize=(6,4))
price_percent.plot(kind='bar', color='skyblue')
plt.title('Price Range Distribution of Restaurants')
plt.xlabel('Price Range')
plt.ylabel('Percentage of Restaurants')
plt.xticks(rotation=0)
plt.tight_layout()
plt.show()
```



Level 1 Task 4

Task: Online Delivery

Determine the percentage of restaurants that offer online delivery.

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

In [14]:
```python
online_counts = df['Has Online delivery'].value_counts()
print("Online delivery counts:\n", online_counts)
```

```
Online delivery counts:
 Has Online delivery
No     7100
Yes    2451
Name: count, dtype: int64
```

In [15]:
```python
total = len(df)
online_percentage = (online_counts / total) * 100
print("\nPercentage:\n", online_percentage)
```

```
Percentage:
 Has Online delivery
No     74.337766
Yes    25.662234
Name: count, dtype: float64
```

In [16]:
```python
avg_with_delivery = df[df['Has Online delivery'] == 'Yes']['Aggregate rating'].mean()
avg_without_delivery = df[df['Has Online delivery'] == 'No']['Aggregate rating'].mean()
```

In [17]:
```python
print("\nAverage rating with online delivery:", round(avg_with_delivery, 2))
print("Average rating without online delivery:", round(avg_without_delivery, 2))
```

```
Average rating with online delivery: 3.25
Average rating without online delivery: 2.47
```

Level 2 Task 1

Task: Restaurant Ratings

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

Calculate the average number of votes received by restaurants.

In [18]:
```python
rating_counts = df['Aggregate rating'].value_counts().sort_index()
print("Aggregate rating distribution:\n", rating_counts)
```

```
Aggregate 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     274
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
```

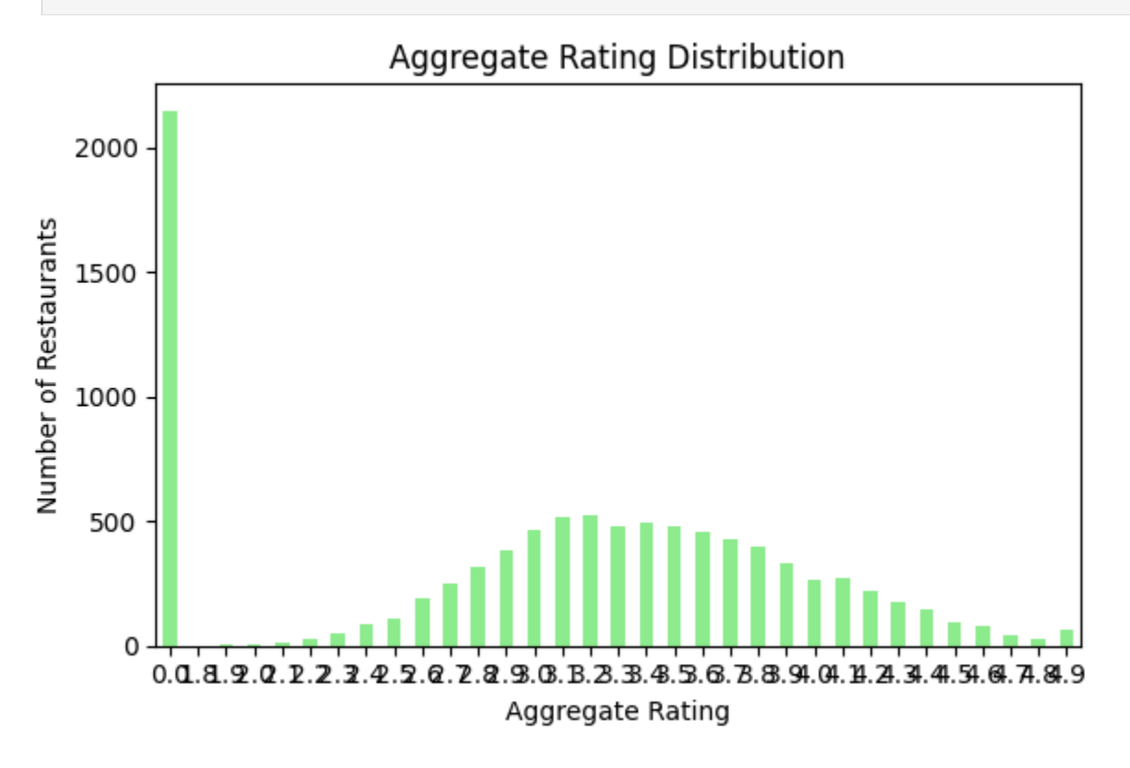
In [19]:
```python
most_common_rating = rating_counts.idxmax()
print("\nMost common rating is:", most_common_rating)
```

```
Most common rating is: 0.0
```

In [20]:
```python
average_votes = df['Votes'].mean()
print("\nAverage number of votes per restaurant:", round(average_votes, 2))
```

```
Average number of votes per restaurant: 156.91
```

In [21]:
```python
plt.figure(figsize=(6,4))
rating_counts.plot(kind='bar', color='lightgreen')
plt.title("Aggregate Rating Distribution")
plt.xlabel("Aggregate Rating")
plt.ylabel("Number of Restaurants")
plt.xticks(rotation=0)
plt.tight_layout()
plt.show()
```



Level 2 Task 2 Task: Cuisine Combination

Identify the most common combinations of cuisines in the dataset.

Determine if certain cuisine combinations tend to have higher ratings.

In [22]:
```python
top_combos = df['Cuisines'].value_counts().head(10)
print("Top Cuisine Combinations:\n", top_combos)
```

```
Top Cuisine Combinations:
 Cuisines
North Indian                      936
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
```

In [23]:
```python
combo_rating = df.groupby('Cuisines')['Aggregate rating'].mean().sort_values(ascending=False)
print("\nCuisine Combinations with Highest Average Ratings:\n", combo_rating.head(10))
```

```
Cuisine Combinations with Highest Average Ratings:
 Cuisines
Italian, Deli               4.9
Hawaiian, Seafood           4.9
American, Sandwich, Tea     4.9
Continental, Indian         4.9
European, Asian, Indian     4.9
European, Contemporary      4.9
European, German            4.9
BBQ, Breakfast, Southern    4.9
American, Coffee and Tea    4.9
Sunda, Indonesian           4.9
Name: Aggregate rating, dtype: float64
```

Level Task 3 Task: Geographic Analysis

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

Identify any patterns or clusters of restaurants in specific areas.

In [24]:
```python
df = df[(df['Latitude'].notnull()) & (df['Longitude'].notnull())]
```

In [25]:
```python
fig = px.scatter_mapbox(
    df,
    lat="Latitude",
    lon="Longitude",
    hover_name="Restaurant Name",
    hover_data=["City", "Aggregate rating"],
    color="Aggregate rating",
    color_continuous_scale="Turbo",
    zoom=3,
    height=500
)

fig.update_layout(mapbox_style="open-street-map")
fig.update_layout(title="Restaurant Locations")
fig.show()
```

```
C:\Users\devan\AppData\Local\Temp\ipykernel_13604\2879649869.py:1: DeprecationWarning: *scatter_mapbox* is deprecated! Use *scatter_map* instead. Learn more at: https://plotly.com/python/mapbox-to-maplibre/
  fig = px.scatter_mapbox(
```

Level 2 Task 4 Task: Restaurant Chains

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

Analyze the ratings and popularity of different restaurant chains.

In [26]:
```python
chain_counts = df['Restaurant Name'].value_counts()
restaurant_chains = chain_counts[chain_counts > 1]
print("Top restaurant chains:\n", restaurant_chains.head(10))
```

```
Top restaurant chains:
 Restaurant Name
Cafe Coffee Day     83
Domino's Pizza      79
Subway              63
Green Chick Chop    51
McDonald's          48
Keventers           34
Pizza Hut           30
Giani               29
Baskin Robbins      28
Barbeque Nation     26
Name: count, dtype: int64
```

In [27]:
```python
chains_df = df[df['Restaurant Name'].isin(restaurant_chains.index)]
```

In [28]:
```python
chain_analysis = chains_df.groupby('Restaurant Name').agg({
    'Aggregate rating': 'mean',
    'Votes': 'mean',
    'City': 'count'
}).rename(columns={
    'Aggregate rating': 'Average Rating',
    'Votes': 'Average Votes',
    'City': 'Branch Count'
}).sort_values(by='Branch Count', ascending=False)

print("\nTop Restaurant Chains (with ratings & votes):\n")
print(chain_analysis.head(10))
```

```
Top Restaurant Chains (with ratings & votes):

                  Average Rating  Average Votes  Branch Count
Restaurant Name
Cafe Coffee Day         2.419277      29.253012            83
Domino's Pizza          2.740506      84.088608            79
Subway                  2.907937      97.206349            63
Green Chick Chop        2.672549      18.901961            51
McDonald's              3.339583     110.229167            48
Keventers               2.870588      37.147059            34
Pizza Hut               3.320000     165.366667            30
Giani                   2.689655      29.448276            29
Baskin Robbins          1.860714      15.285714            28
Barbeque Nation         4.353846    1082.384615            26
```

Level 3 Task 1 Task: Restaurant Reviews

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

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

In [29]:
```python
import pandas as pd
import nltk
import matplotlib.pyplot as plt
import seaborn as sns
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
```

In [30]:
```python
rating_counts = df['Rating text'].value_counts()
print("Number of Restaurants in each Rating Category:\n", rating_counts)
```

```
Number of Restaurants in each Rating Category:
 Rating text
Average      3737
Not rated    2148
Good         2100
Very Good    1079
Excellent     301
Poor          186
Name: count, dtype: int64
```

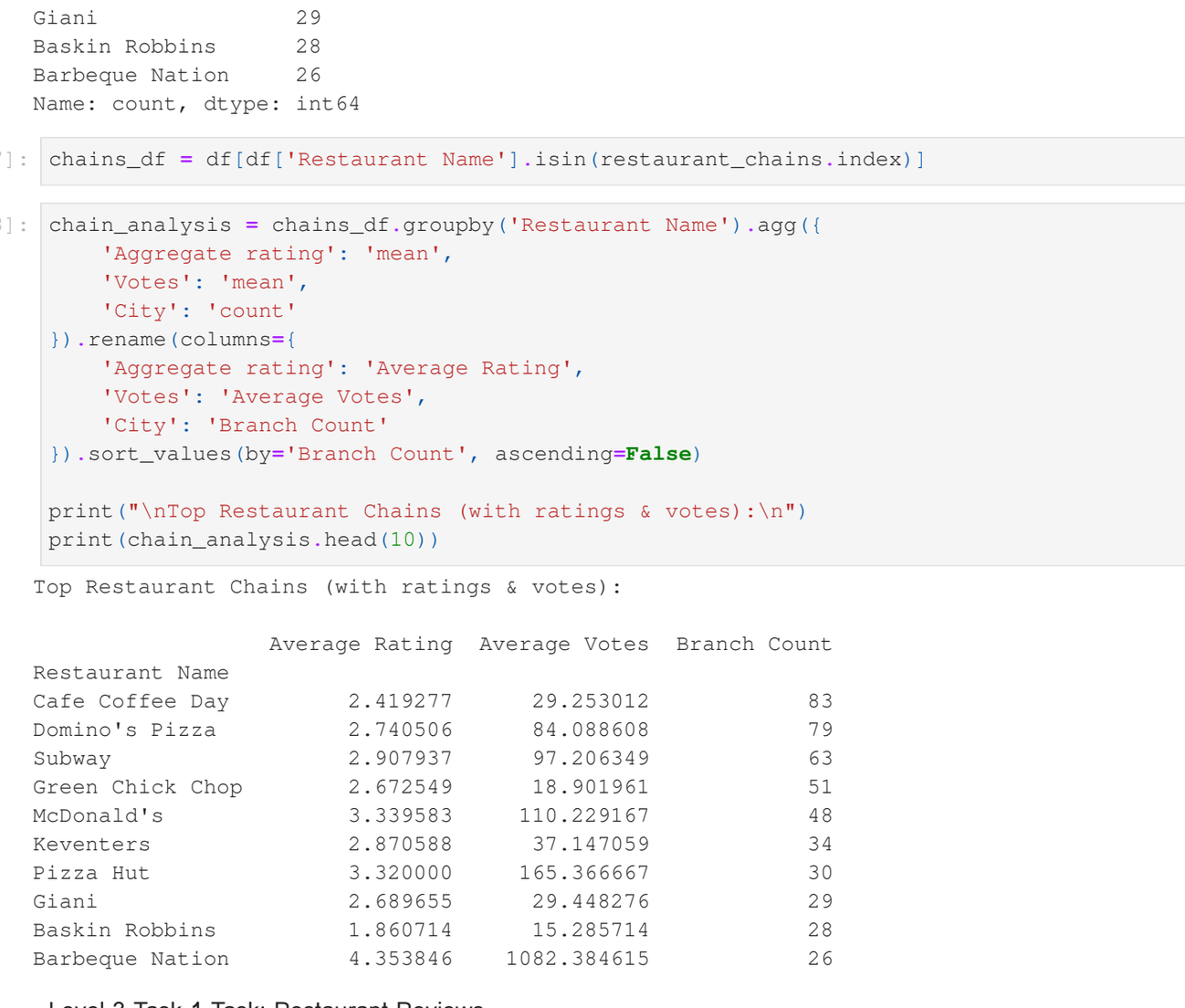
In [31]:
```python
avg_votes = df.groupby('Rating text')['Votes'].mean().sort_values(ascending=False)
print("\nAverage Votes per Rating Category:\n", avg_votes)
```

```
Average Votes per Rating Category:
 Rating text
Excellent    851.770764
Very Good    520.458758
Good         229.351429
Poor          90.715054
Average       48.249130
Not rated      0.870112
Name: Votes, dtype: float64
```

In [32]:
```python
plt.figure(figsize=(7,5))
sns.barplot(x=avg_votes.index, y=avg_votes.values, palette="viridis")
plt.title("Average Votes by Rating Category")
plt.xlabel("Rating Category")
plt.ylabel("Average Number of Votes")
plt.xticks(rotation=30)
plt.tight_layout()
plt.show()
```

```
C:\Users\devan\AppData\Local\Temp\ipykernel_13604\2008967702.py:2: FutureWarning:


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



Level 3 Task 2 Task: Votes Analysis

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

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

In [33]:
```python
top_votes = df.sort_values(by='Votes', ascending=False).head(5)
low_votes = df[df['Votes'] > 0].sort_values(by='Votes', ascending=True).head(5)
print("🔼 Top 5 Most Voted Restaurants:\n", top_votes[['Restaurant Name', 'Votes', 'Aggregate rating']])
print("\n🔽 Bottom 5 (non-zero votes):\n", low_votes[['Restaurant Name', 'Votes', 'Aggregate rating']])
```

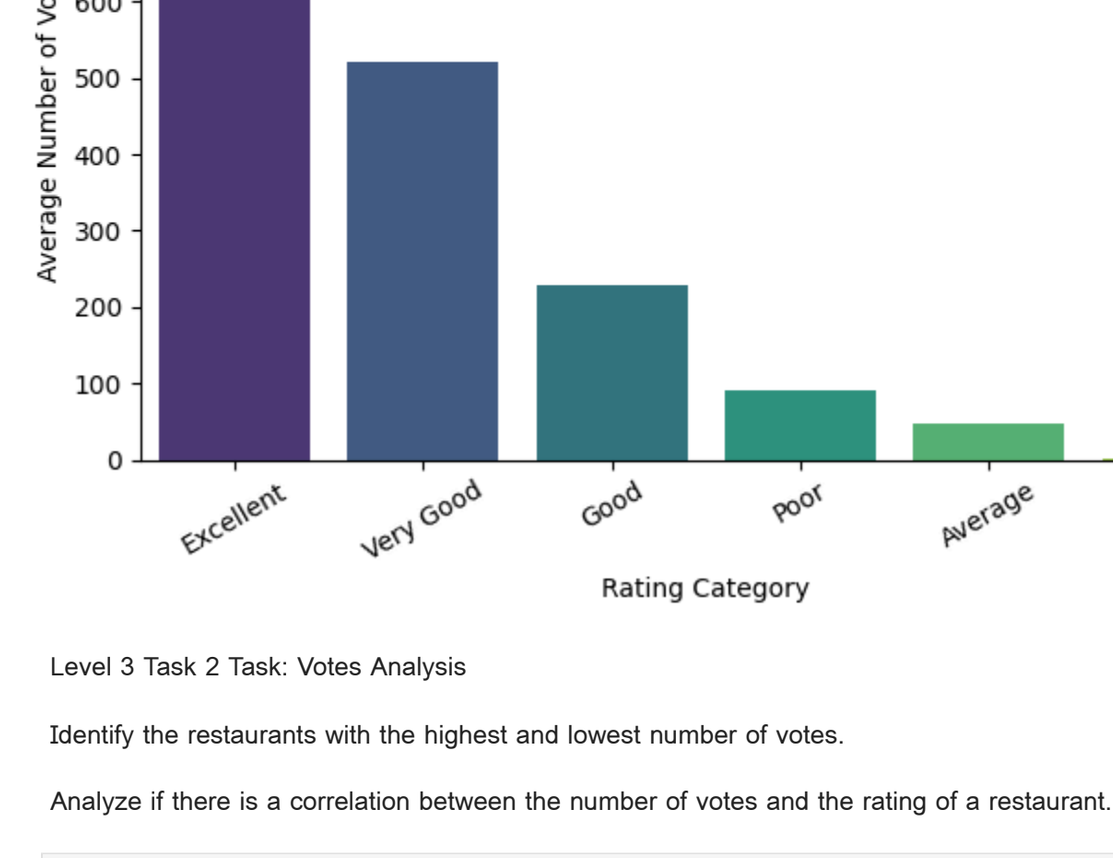
```
🔼 Top 5 Most Voted Restaurants:
                 Restaurant Name  Votes  Aggregate rating
728                        Toit  10934               4.8
735                    Truffles   9667               4.7
3994          Hauz Khas Social   7931               4.3
2412                 Peter Cat   7574               4.3
739   AB's - Absolute Barbecues   6907               4.6

🔽 Bottom 5 (non-zero votes):
          Restaurant Name  Votes  Aggregate rating
1014               Ahata      1               0.0
5653     The Cake Basket      1               0.0
5647  Namaste Restaurant      1               0.0
5646      KT's Shik-Shack      1               0.0
5636      Aggarwal Sweets      1               0.0
```

In [34]:
```python
correlation = df[['Votes', 'Aggregate rating']].corr().iloc[0,1]
print(f"\n📊 Correlation between Votes and Rating: {round(correlation, 2)}")
```

```
📊 Correlation between Votes and Rating: 0.31
```

In [35]:
```python
plt.figure(figsize=(6,4))
sns.scatterplot(data=df, x='Votes', y='Aggregate rating', alpha=0.6)
plt.title("Votes vs Aggregate Rating")
plt.xlabel("Number of Votes")
plt.ylabel("Aggregate Rating")
plt.tight_layout()
plt.show()
```



Level 3 Level 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.

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

In [36]:
```python
df['Has Online delivery'] = df['Has Online delivery'].map({'Yes': 1, 'No': 0})
df['Has Table booking'] = df['Has Table booking'].map({'Yes': 1, 'No': 0})
```

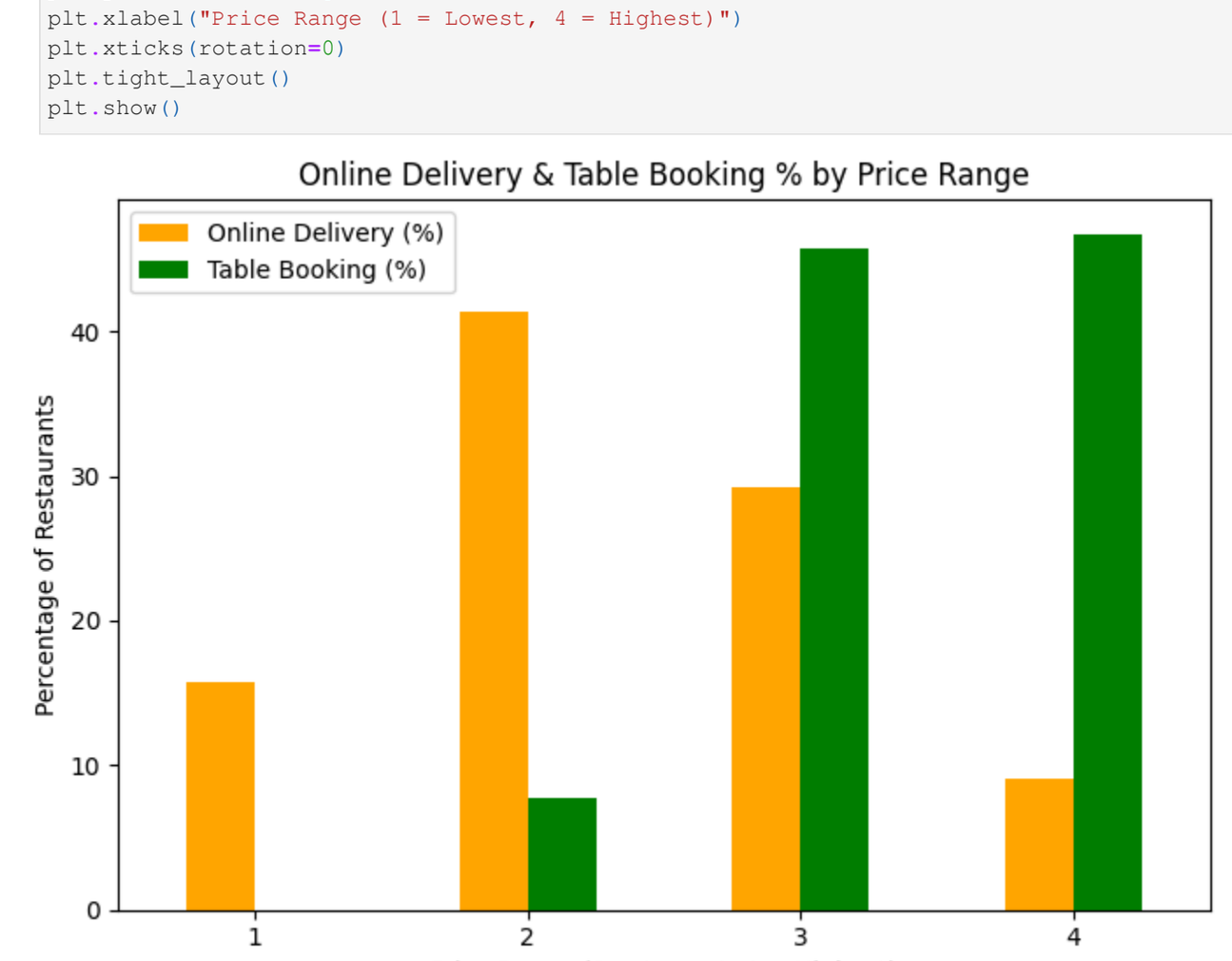
In [37]:
```python
grouped = df.groupby('Price range')[['Has Online delivery', 'Has Table booking']].mean() * 100
grouped = grouped.rename(columns={
    'Has Online delivery': 'Online Delivery (%)',
    'Has Table booking': 'Table Booking (%)'
})
```

In [38]:
```python
print("📊 Service Availability by Price Range:\n")
print(grouped)
```

```
📊 Service Availability by Price Range:

             Online Delivery (%)  Table Booking (%)
Price range
1                      15.774077           0.022502
2                      41.310633           7.677482
3                      29.190341          45.738636
4                       9.044369          46.757679
```

In [39]:
```python
grouped.plot(kind='bar', figsize=(7,5), color=['orange', 'green'])
plt.title("Online Delivery & Table Booking % by Price Range")
plt.ylabel("Percentage of Restaurants")
plt.xlabel("Price Range (1 = Lowest, 4 = Highest)")
plt.xticks(rotation=0)
plt.tight_layout()
plt.show()
```



While completing these data analysis tasks, I explored real-world restaurant data, performed data cleaning, visualization, and gained hands-on experience with tools like Python (pandas, matplotlib) and Excel.

This project helped me build confidence in solving real datasets and improved my overall understanding of data analytics.