level2 task1 : Table Booking and Online Delivery 📋 🍽️

Objective🎯 :

- The goal of this task is to analyze restaurant features related to table booking and online delivery. Specifically, we aim to:

1. Determine the percentage of restaurants that offer table booking and online delivery.
2. Compare the average ratings of restaurants with table booking versus those without.
3. Examine the availability of online delivery across different price ranges.
4. Analyze the correlation between average cost and ratings.

In [4]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [5]:
```
file_path = ('D:/cognifyz/Dataset .csv')
df = pd.read_csv(file_path)
```

In [6]:
```
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
```

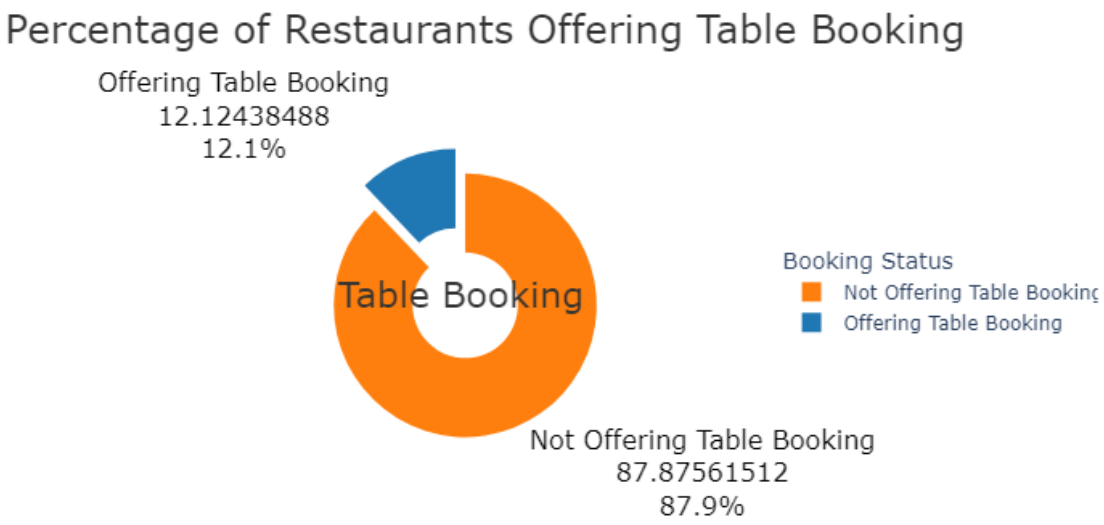
Analysis 🔍

1. Table Booking: Calculate the percentage of restaurants offering table booking using the Has Table booking column.
2. Online Delivery: Calculate the percentage of restaurants offering online delivery using the Has Online delivery column.

In [7]:
```
table_booking_percentage = (df['Has Table booking'].value_counts(normalize=True) *
no_table_booking_percentage = 100 - table_booking_percentage
print(f"Percentage of restaurants offering table booking: {table_booking_percentage
print(f"Percentage of restaurants not offering table booking: {no_table_booking_per
import plotly.graph_objects as go
table_booking_data = {
    'Offering Table Booking': table_booking_percentage,
    'Not Offering Table Booking': no_table_booking_percentage
}
colors = ['#1f77b4', '#ff7f0e']
fig = go.Figure(data=[go.Pie(
    labels=list(table_booking_data.keys()),
    values=list(table_booking_data.values()),
    hole=0.4,
    marker=dict(colors=colors),
    textinfo='label+percent+value',
    hoverinfo='label+percent+value',
    textfont=dict(size=16, color='black'),
```

```
    insidetextorientation='radial',
    pull=[0.1, 0.1]
)])
fig.update_layout(
    title_text='Percentage of Restaurants Offering Table Booking',
    title_font_size=24,
    title_x=0.5,
    title_y=0.95,
    title_font_color='#333',
    annotations=[dict(
        text='Table Booking',
        x=0.5,
        y=0.5,
        font_size=22,
        showarrow=False,
        font_color='#333'
    )],
    legend_title='Booking Status',
    legend_title_font_size=14,
    legend_font_size=12,
    legend_x=0.8,
    legend_y=0.5
)
fig.show()
```

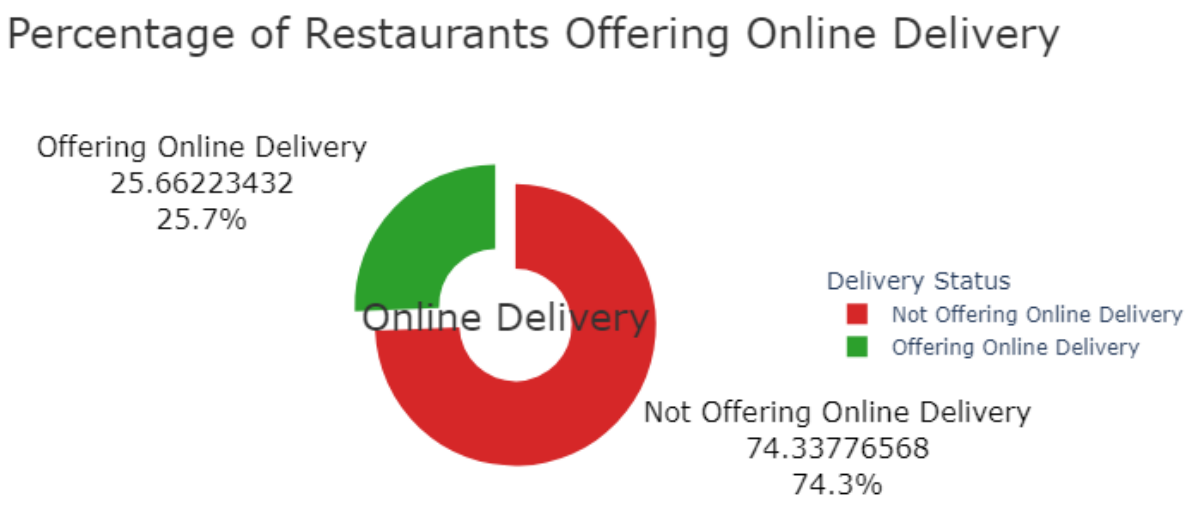
```
Percentage of restaurants offering table booking: 12.12%
Percentage of restaurants not offering table booking: 87.88%
```



In [8]:
```
online_delivery_percentage = (df['Has Online delivery'].value_counts(normalize=True
no_online_delivery_percentage = 100 - online_delivery_percentage
print(f"Percentage of restaurants offering online delivery: {online_delivery_percen
print(f"Percentage of restaurants not offering online delivery: {no_online_delivery
import plotly.graph_objects as go
online_delivery_percentage = (df['Has Online delivery'].value_counts(normalize=True
no_online_delivery_percentage = 100 - online_delivery_percentage
online_delivery_data = {
    'Offering Online Delivery': online_delivery_percentage,
    'Not Offering Online Delivery': no_online_delivery_percentage
}
colors = ['#2ca02c', '#d62728']
fig = go.Figure(data=[go.Pie(
    labels=list(online_delivery_data.keys()),
    values=list(online_delivery_data.values()),
    hole=0.4,
    marker=dict(colors=colors),
    textinfo='label+percent+value',
    hoverinfo='label+percent+value',
    textfont=dict(size=16, color='black'),
    insidetextorientation='radial',
    pull=[0.1, 0.1]
)])
fig.update_layout(
```

```
    title_text='Percentage of Restaurants Offering Online Delivery',
    title_font_size=24,
    title_x=0.5,
    title_y=0.95,
    title_font_color='#333',
    annotations=[dict(
        text='Online Delivery',
        x=0.5,
        y=0.5,
        font_size=22,
        showarrow=False,
        font_color='#333'
    )],
    legend_title='Delivery Status',
    legend_title_font_size=14,
    legend_font_size=12,
    legend_x=0.8,
    legend_y=0.5
)
fig.show()
```

```
Percentage of restaurants offering online delivery: 25.66%
Percentage of restaurants not offering online delivery: 74.34%
```



Insights 💡

- Table Booking and Online Delivery:

- Low Adoption Rates: Only a small fraction of restaurants offer table booking (12.12%) or online delivery (25.66%). This indicates a potential area for growth and an opportunity for restaurants to differentiate themselves by offering these services.
- Service Offering Opportunities: The low percentages of table booking and online delivery suggest that many restaurants could benefit from adopting these features to attract more customers.

In [140…

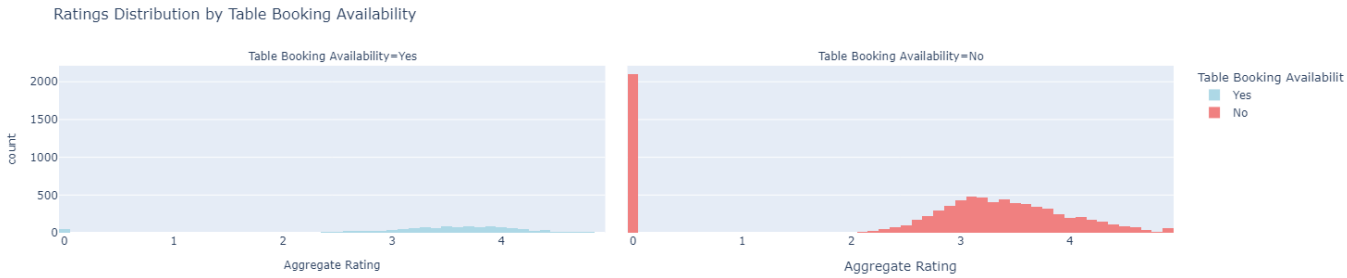
```
import plotly.graph_objects as go
avg_rating_with_table_booking = df[df['Has Table booking'] == 'Yes']['Aggregate rat
avg_rating_without_table_booking = df[df['Has Table booking'] == 'No']['Aggregate r
print(f"Average rating for restaurants with table booking: {avg_rating_with_table_b
print(f"Average rating for restaurants without table booking: {avg_rating_without_t
avg_ratings = {
    'With Table Booking': avg_rating_with_table_booking,
    'Without Table Booking': avg_rating_without_table_booking
}
fig = go.Figure(data=[go.Bar(
    x=list(avg_ratings.keys()),
    y=list(avg_ratings.values()),
    text=[f'{value:.2f}' for value in avg_ratings.values()],
    textposition='auto',
    marker=dict(color=['#1f77b4', '#ff7f0e'])
)])
fig.update_layout(
    title='Average Rating of Restaurants with and without Table Booking',
    title_font_size=16,
    title_font_weight='bold',
    xaxis_title='Table Booking Availability',
    xaxis_title_font_size=14,
    xaxis_title_font_weight='bold',
    yaxis_title='Average Rating',
    yaxis_title_font_size=14,
    yaxis_title_font_weight='bold',
    xaxis_tickfont_size=12,
    yaxis_tickfont_size=12,
    margin=dict(l=40, r=40, t=40, b=80)
)
fig.show()
```

```
Average rating for restaurants with table booking: 3.44
Average rating for restaurants without table booking: 2.56
```


Average Rating of Restaurants with and without Table Booking
Average Rating
0
1
2
3
3.44
2.56
With Table Booking
Without Table Booking
Table Booking Availability

In [2]:
```
import pandas as pd
import plotly.express as px
df = pd.read_csv('D:/cognifyz/Dataset .csv')
fig = px.histogram(df, x='Aggregate rating', color='Has Table booking', facet_col='
                   color_discrete_map={'Yes': 'lightblue', 'No': 'lightcoral'},
                   title='Ratings Distribution by Table Booking Availability',
                   labels={'Has Table booking': 'Table Booking Availability', 'Aggr

fig.update_layout(
    title_font_size=16,
    xaxis_title_font_size=12,
    yaxis_title_font_size=12,
    legend_title_font_size=13,
    legend_font_size=12
)
fig.show()
```



Insights 💡

Average Ratings:

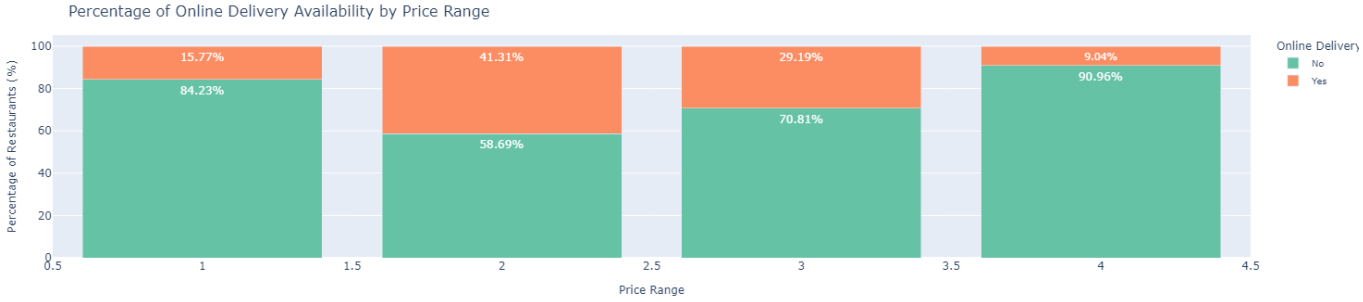
- Impact of Table Booking: Restaurants that offer table booking have a significantly higher average rating (3.44) compared to those that do not (2.56). This indicates that customers perceive restaurants with table booking as providing a better overall experience or service.

In [142…

```
import pandas as pd
import plotly.express as px
online_delivery_by_price_range = pd.crosstab(df['Price range'], df['Has Online deli
print("Online Delivery Availability by Price Range:\n", online_delivery_by_price_ra
online_delivery_percentage_by_price_range = (online_delivery_by_price_range.div(onl
print("Percentage of Online Delivery by Price Range:\n", online_delivery_percentage
online_delivery_percentage_by_price_range = online_delivery_percentage_by_price_ran
fig = px.bar(
    online_delivery_percentage_by_price_range,
    x='Price range',
    y='Percentage',
    color='Has Online Delivery',
```

```
    text='Percentage',  # Add data labels
    title='Percentage of Online Delivery Availability by Price Range',
    labels={'Price range': 'Price Range', 'Percentage': 'Percentage of Restaurants
    color_discrete_sequence=px.colors.qualitative.Set2,
    barmode='stack'
)
fig.update_layout(
    xaxis_title='Price Range',
    yaxis_title='Percentage of Restaurants (%)',
    xaxis_title_font_size=12,
    yaxis_title_font_size=12,
    title_font_size=16,
    legend_title='Online Delivery',
    legend_title_font_size=12,
    legend_font_size=10,
    margin=dict(l=40, r=40, t=40, b=80),
)
fig.update_traces(
    texttemplate='%{text:.2f}%',
    textposition='inside',
    insidetextfont=dict(color='white')
)
fig.show()
```

```
Online Delivery Availability by Price Range:
 Has Online delivery    No   Yes
Price range
1                     3743   701
2                     1827  1286
3                      997   411
4                      533    53
Percentage of Online Delivery by Price Range:
 Has Online delivery         No        Yes
Price range
1                     84.225923  15.774077
2                     58.689367  41.310633
3                     70.809659  29.190341
4                     90.955631   9.044369
```



Insights 💡

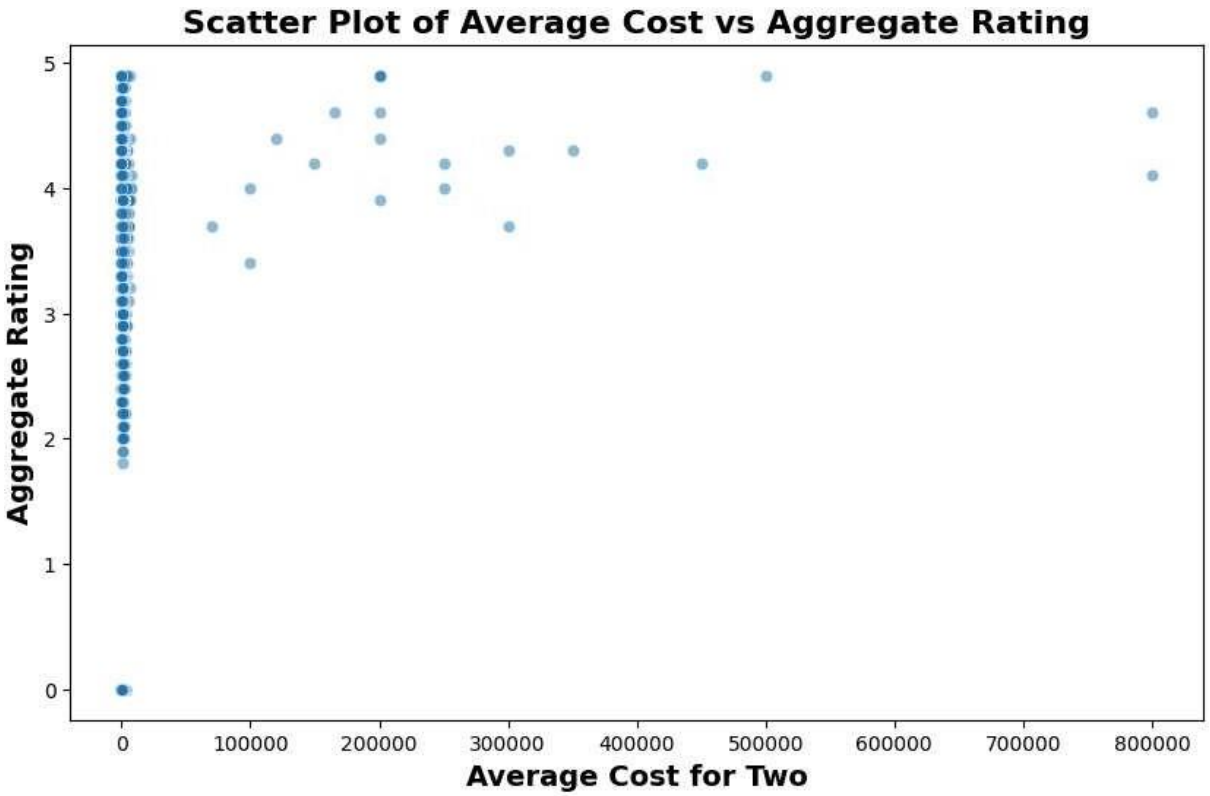
Price Range and Online Delivery:

- Price Sensitivity: The availability of online delivery decreases with higher price ranges. For example, restaurants in Price Range 4 have the lowest percentage of online delivery options. This could reflect the nature of high-priced establishments, which might not prioritize online delivery or might cater to a clientele that prefers dining in.

In [143…

```
correlation = df[['Average Cost for two', 'Aggregate rating']].corr().iloc[0, 1]
print(f"Correlation between Average Cost and Ratings: {correlation:.2f}")
plt.figure(figsize=(10, 6))
sns.scatterplot(data=df, x='Average Cost for two', y='Aggregate rating', alpha=0.5)
plt.title('Scatter Plot of Average Cost vs Aggregate Rating', fontsize=16, weight='
plt.xlabel('Average Cost for Two', fontsize=14, weight='bold')
plt.ylabel('Aggregate Rating', fontsize=14, weight='bold')
plt.show()
```

```
Correlation between Average Cost and Ratings: 0.05
```



Insights 💡

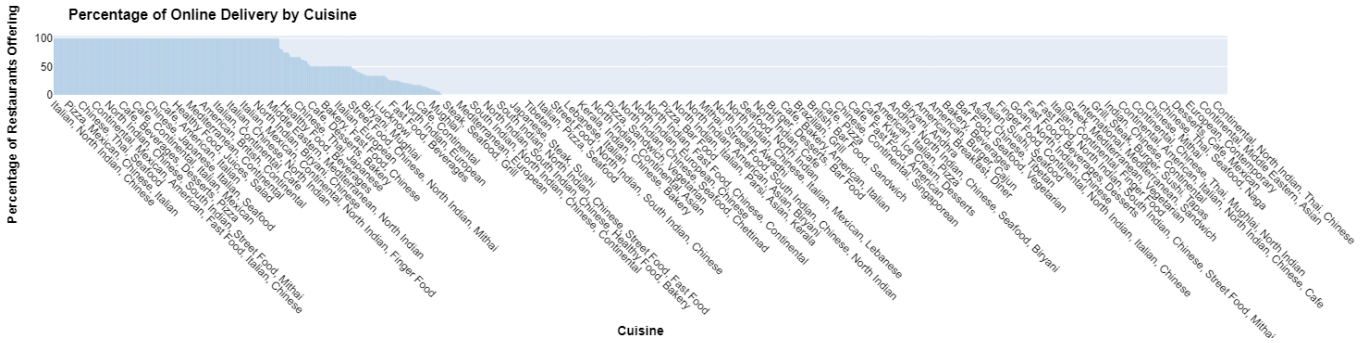
Correlation Analysis:

- Minimal Influence of Cost on Ratings: The very low correlation (0.05) between average cost and ratings suggests that the cost of dining does not strongly affect customer ratings. This implies that factors other than price, such as service quality, ambiance, and food quality, play a more significant role in determining restaurant ratings.

In [144…

```
cuisine_delivery = df.groupby('Cuisines')['Has Online delivery'].value_counts(norma
cuisine_delivery = cuisine_delivery.sort_values(by='Yes', ascending=False)
fig = px.bar(cuisine_delivery,
             x=cuisine_delivery.index,
             y='Yes',
             title='Percentage of Online Delivery by Cuisine',
             labels={'Yes': 'Percentage of Online Delivery'},
             color_discrete_sequence=['#1f77b4'])
fig.update_layout(
    xaxis_title='Cuisine',
    yaxis_title='Percentage of Restaurants Offering Online Delivery',
    xaxis=dict(
        tickangle=45,
        title_font=dict(
            size=14,
            color='black',
            family='Arial',
            weight='bold'
        ),
```

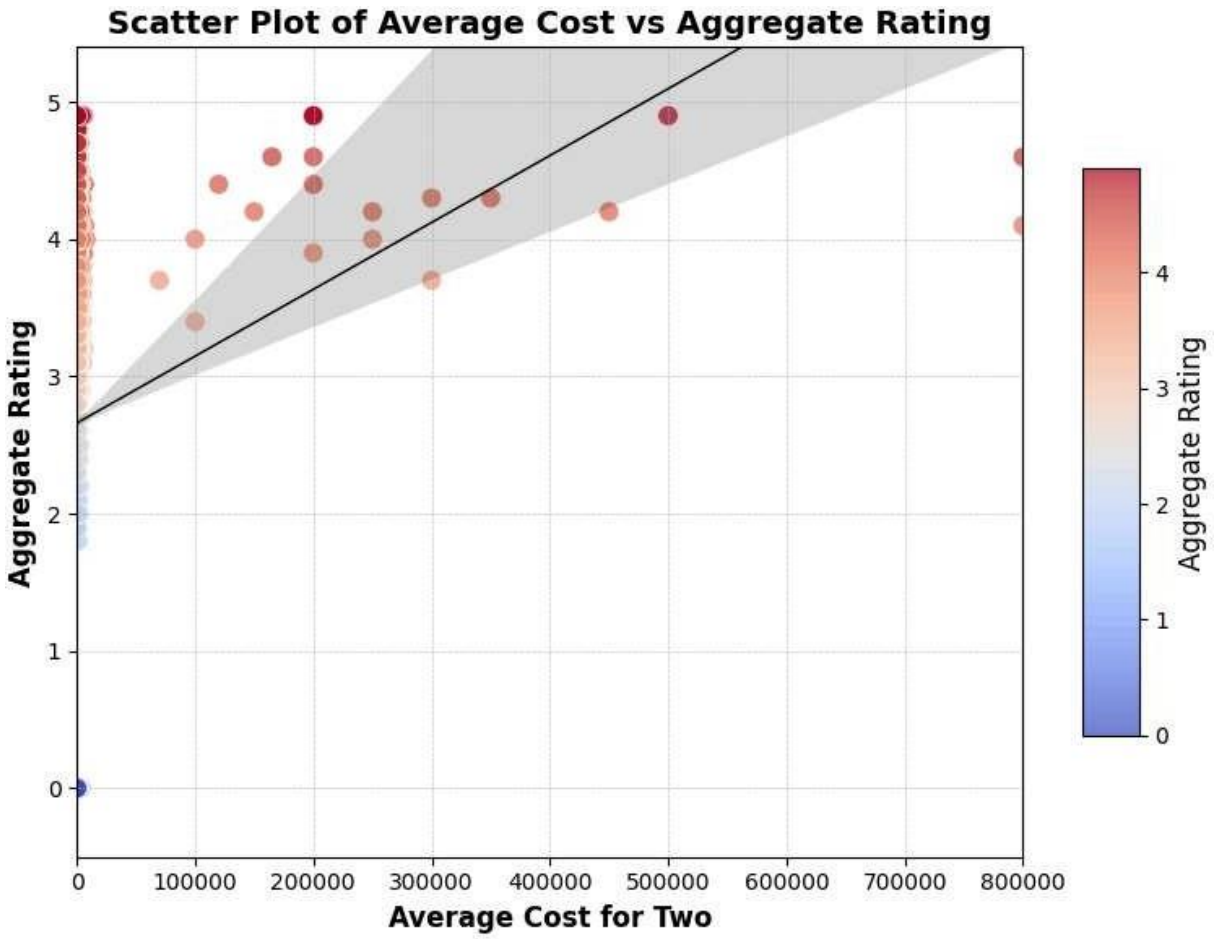
```
        tickfont=dict(
            size=12,
            color='black',
            family='Arial',
        )
    ),
    yaxis=dict(
        title_font=dict(
            size=14,
            color='black',
            family='Arial',
            weight='bold'
        ),
        tickfont=dict(
            size=12,
            color='black',
            family='Arial',
        )
    ),
    title_font=dict(
        size=16,
        color='black',
        family='Arial',
        weight='bold'
    ),
    margin=dict(l=40, r=40, t=40, b=120)
)
fig.show()
```



In [145…

```
import matplotlib.pyplot as plt
import seaborn as sns
from matplotlib.colors import Normalize
norm = Normalize(vmin=df['Aggregate rating'].min(), vmax=df['Aggregate rating'].max
plt.figure(figsize=(8, 6))
scatter = plt.scatter(
    data=df,
    x='Average Cost for two',
    y='Aggregate rating',
    c=df['Aggregate rating'],
    cmap='coolwarm',
    norm=norm,
    alpha=0.7,
    edgecolor='w',
    linewidth=0.5,
    s=80
)
cbar = plt.colorbar(scatter, shrink=0.7, aspect=10)
cbar.set_label('Aggregate Rating', fontsize=12)
cbar.ax.tick_params(labelsize=10)
sns.regplot(
    data=df,
    x='Average Cost for two',
    y='Aggregate rating',
```

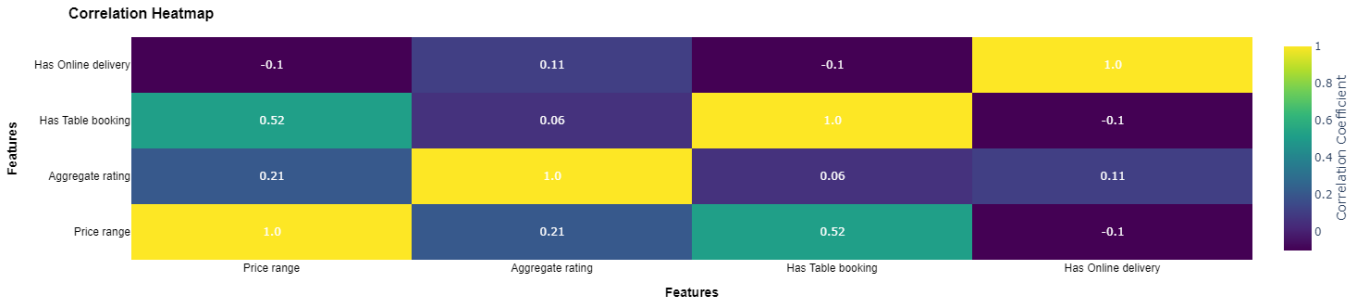
```
    scatter=False,
    color='black',
    line_kws={'linewidth': 1}
)
plt.title('Scatter Plot of Average Cost vs Aggregate Rating', fontsize=14, weight='
plt.xlabel('Average Cost for Two', fontsize=12, weight='bold')
plt.ylabel('Aggregate Rating', fontsize=12, weight='bold')
plt.grid(True, linestyle='--', linewidth=0.5, alpha=0.7)
plt.xlim(df['Average Cost for two'].min() - 10, df['Average Cost for two'].max() +
plt.ylim(df['Aggregate rating'].min() - 0.5, df['Aggregate rating'].max() + 0.5)
plt.tick_params(axis='both', which='major', labelsize=10)
plt.tight_layout()
plt.show()
```



In [151…

```
correlation_matrix = df[['Price range', 'Aggregate rating', 'Has Table booking', 'H
heatmap = go.Heatmap(
    z=correlation_matrix.values,
    x=correlation_matrix.columns,
    y=correlation_matrix.columns,
    colorscale='Viridis',
    showscale=True,
    colorbar=dict(title='Correlation Coefficient', titleside='right'),
    text=correlation_matrix.round(2).astype(str),
    texttemplate='%{text}',
    textfont=dict(size=12, color='white')
)
fig = go.Figure(data=[heatmap])
```

```
fig.update_layout(
    title='Correlation Heatmap',
    title_font=dict(size=16, family='Arial', color='black', weight='bold'),
    xaxis=dict(
        title='Features',
        title_font=dict(size=14, family='Arial', color='black', weight='bold'),
        tickfont=dict(size=12, family='Arial', color='black')
    ),
    yaxis=dict(
        title='Features',
        title_font=dict(size=14, family='Arial', color='black', weight='bold'),
        tickfont=dict(size=12, family='Arial', color='black')
    ),
    margin=dict(l=40, r=40, t=40, b=80)
)
fig.show()
```



Conclusion

- Market Potential: The analysis reveals that there is significant room for improvement in terms of offering table booking and online delivery services. Restaurants that implement

these features may be able to enhance their customer satisfaction and differentiate themselves in a competitive market.

- Customer Experience: Offering table booking appears to be positively associated with higher restaurant ratings. This suggests that enhancing customer convenience through features like table booking could lead to improved customer perceptions and higher ratings.

- Service Trends by Price Range: Online delivery is less common among higher-priced restaurants. Understanding this trend could help restaurants in different price ranges tailor their service offerings to better meet customer expectations and preferences.

- Focus on Customer Experience: Since average cost has a minimal impact on ratings, restaurants should focus on improving other aspects of their service, such as quality of food, service, and overall customer experience, to boost their ratings and attract more customers.

Task 2: Price Range Analysis 📊

In this analysis, I explore the relationship between restaurant price ranges and their average aggregate ratings to understand how pricing influences customer satisfaction. By examining this relationship, I aim to uncover whether there is a significant pattern that suggests how the price of dining options affects customer perceptions and ratings. 💸🍽️

Objective🎯

- The primary goal is to investigate how different price ranges correlate with the average ratings provided by customers. This involves determining if higher-priced restaurants tend to receive higher ratings compared to those in lower price ranges, or if the ratings are more uniformly distributed across different price ranges.

Data Collection🗂️

- We utilize a dataset containing various attributes of restaurants, including their price range and aggregate rating. Price range is typically categorized into discrete values (e.g., 1 to 4), representing increasing levels of expense. Aggregate rating is a numerical score provided by customers, reflecting their overall satisfaction with the dining experience.

Analysis🔍

1. Determine the Most Common Price Range:

- First, we identify the most frequently observed price range in the dataset. This information helps us understand which price bracket is most prevalent among the restaurants surveyed.

In [2]:
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
file_path = 'D:/cognifyz/Dataset .csv'
df = pd.read_csv(file_path)
most_common_price_range = df['Price range'].mode()[0]
print(f"Most Common Price Range: {most_common_price_range}")
```

```
Most Common Price Range: 1
```

2. Calculate Average Ratings for Each Price Range:

Next, I calculated the average aggregate rating for each price range. This involves grouping the restaurants by their price range and computing the mean rating for each group. This step allows us to compare how different price ranges fare in terms of customer satisfaction.

In [101…
```
average_rating_per_price_range = df.groupby('Price range')['Aggregate rating'].mean
average_rating_per_price_range.columns = ['Price range', 'Average rating']
print("Average Rating for Each Price Range:")
print(average_rating_per_price_range)
```

```
Average Rating for Each Price Range:
   Price range  Average rating
0            1        1.999887
1            2        2.941054
2            3        3.683381
3            4        3.817918
```

3. Identify the Price Range with the Highest Average Rating:

The analysis highlights the price range with the highest average rating. By examining this, I can determine if there is a specific price range where customers tend to rate restaurants more favorably.

In [102…
```
average_rating_per_price_range = df.groupby('Price range')['Aggregate rating'].mean
average_rating_per_price_range.columns = ['Price range', 'Average rating']
highest_avg_rating_row = average_rating_per_price_range.loc[average_rating_per_pric
highest_avg_rating_price_range = highest_avg_rating_row['Price range']
highest_avg_rating_value = highest_avg_rating_row['Average rating']
print("Price Range with the Highest Average Rating:")
print(f"Price Range: {highest_avg_rating_price_range}")
print(f"Average Rating: {highest_avg_rating_value:.4f}")
```

```
Price Range with the Highest Average Rating:
Price Range: 4.0
Average Rating: 3.8179
```

4. Determine the Most Common Rating Color:

In [103…
```
df['Rating color'].value_counts()
```

Out[103…
```
Rating color
Orange         3737
White          2148
Yellow         2100
Green          1079
Dark Green      301
Red             186
Name: count, dtype: int64
```

5. Associate Most Common Rating Color with Each Price Range:

In [104…
```
average_rating_per_price_range = df.groupby('Price range')['Aggregate rating'].mean
average_rating_per_price_range.columns = ['Price range', 'Average rating']
def most_common_color(colors):
    return colors.value_counts().idxmax()
rating_color_per_price_range = df.groupby('Price range')['Rating color'].apply(most
average_rating_with_color = pd.merge(average_rating_per_price_range, rating_color_p
average_rating_with_color.columns = ['Price range', 'Average rating', 'Rating color
highest_avg_rating_color = average_rating_with_color.loc[average_rating_with_color[
print("Average Rating and Color for Each Price Range:")
print(average_rating_with_color)
print("\nColor representing the highest average rating among different price ranges
```

```
Average Rating and Color for Each Price Range:
   Price range  Average rating Rating color
0            1        1.999887       Orange
1            2        2.941054       Orange
2            3        3.683381       Yellow
3            4        3.817918       Yellow

Color representing the highest average rating among different price ranges is: Yello
w
```

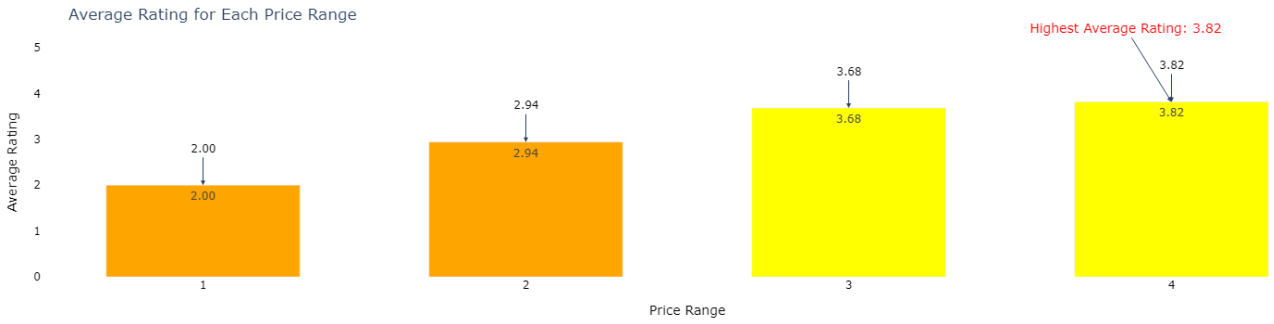
6. Visual Representation

To visualize the findings, i created a bar chart that highlights the average ratings for each price range, with different colors representing the most common rating color for each range. The highest average rating is explicitly marked on the chart.

In [108…
```
import pandas as pd
import plotly.graph_objects as go
average_rating_per_price_range = pd.DataFrame({
    'Price range': [1, 2, 3, 4],
    'Average rating': [1.999887, 2.941054, 3.683381, 3.817918],
    'Rating color': ['Orange', 'Orange', 'Yellow', 'Yellow']
})
colors = {
```

```
    'Orange': 'orange',
    'White': 'white',
    'Yellow': 'yellow',
    'Green': 'green',
    'Dark Green': 'darkgreen',
    'Red': 'red'
}
highest_rating_row = average_rating_per_price_range.loc[average_rating_per_price_ra
fig = go.Figure()
for _, row in average_rating_per_price_range.iterrows():
    fig.add_trace(go.Bar(
        x=[row['Price range']],
        y=[row['Average rating']],
        marker_color=colors[row['Rating color']],
        name=row['Rating color'],
        text=[f'{row["Average rating"]:.2f}'],
        textposition='auto',
        showlegend=False,
        width=0.6
    ))
fig.update_layout(
    title='Average Rating for Each Price Range',
    xaxis_title='Price Range',
    yaxis_title='Average Rating',
    xaxis=dict(
        tickmode='array',
        tickvals=average_rating_per_price_range['Price range'],
        ticktext=[str(x) for x in average_rating_per_price_range['Price range']],
        title_font=dict(size=14, color='black'),
        tickfont=dict(size=12, color='black')
    ),
    yaxis=dict(
        range=[0, 5],
        title_font=dict(size=14, color='black'),
        tickfont=dict(size=12, color='black')
    ),
    barmode='overlay',
    plot_bgcolor='white',
    paper_bgcolor='white',
    margin=dict(l=40, r=40, t=60, b=40),
    annotations=[
        dict(
            x=row['Price range'],
            y=row['Average rating'],
            text=f'{row["Average rating"]:.2f}',
            showarrow=True,
            arrowhead=2,
            ax=0,
            ay=-40,
            font=dict(size=12, color='black')
        ) for _, row in average_rating_per_price_range.iterrows()
    ] + [
        dict(
            x=highest_rating_row['Price range'],
            y=highest_rating_row['Average rating'],
            text=f'Highest Average Rating: {highest_rating_row["Average rating"]:.2
```

```
            showarrow=True,
            arrowhead=2,
            ax=-50,
            ay=-80,
            font=dict(size=14, color='red'),
            bgcolor='rgba(255, 255, 255, 0.7)'
        )
    ]
)
fig.show()
```



Insights and Implications 💡

- The findings from this analysis can reveal if there's a correlation between price and perceived quality. For instance:

- Higher Average Ratings for Expensive Restaurants: If higher-priced restaurants consistently receive higher ratings, it may suggest that customers believe paying more correlates with better service or food quality. 🏅

- Lower Average Ratings for Budget Options: Conversely, if lower-priced options receive lower ratings, it might indicate that budget-friendly restaurants often struggle to meet customer expectations. 📉
- Understanding these patterns can provide valuable insights for restaurant owners and managers. They can use this information to adjust their pricing strategies, enhance customer experiences, or better -position their offerings in the market. 🚀

Conclusion 📝

By analyzing the relationship between price ranges and average ratings, we gain a clearer picture of how pricing affects customer satisfaction. This analysis helps in making informed decisions related to pricing strategies and improving overall service quality to better meet customer expectations. 🎉

Key Findings Summary:

- Most Common Price Range among all restaurants: 1
- Average Rating for Each Price Range:

```
- Price range 1: 1.999887
```

```
- Price range 2: 2.941054
```

```
- Price range 3: 3.683381
```

```
- Price range 4: 3.817918
```

- The Highest Average Rating price range: 4
- Color representing the highest average rating among different price ranges: Yellow

Task 3:Feature Engineering 🔧 🛠️ 📊

1. Feature Extraction 🔍 📊 🔧

1. Length of Restaurant Names and Addresses:

I introduced two new features to our dataset: "Name Length" and "Address Length."

- Restaurant Name Length: This feature measures the number of characters in the restaurant's name. It helps in identifying trends related to the length of restaurant names.
- Address Length: This feature calculates the number of characters in the restaurant's address, which might indicate the level of detail or complexity of the address.

In [67]:
```
df['Name Length'] = df['Restaurant Name'].apply(lambda x: len(str(x)))
df['Address Length'] = df['Address'].apply(lambda x: len(str(x)))

def highlight_length(s):
    """Highlight the length columns with a light blue background."""
    is_length = s.name in ['Name Length', 'Address Length']
    return ['background-color: lightblue' if is_length else '' for _ in s]

styled_df = df[['Restaurant Name', 'Name Length', 'Address', 'Address Length']].hea
    .set_caption("Updated DataFrame with Extracted Features") \
    .apply(highlight_length, axis=0) \
    .set_table_styles({
        'Restaurant Name': [{'selector': 'td', 'props': [('color', 'black')]}],
        'Name Length': [{'selector': 'td', 'props': [('color', 'black')]}],
        'Address': [{'selector': 'td', 'props': [('color', 'black')]}],
        'Address Length': [{'selector': 'td', 'props': [('color', 'black')]}]
    }) \
    .highlight_max(color='lightblue', subset=['Name Length', 'Address Length'])
display(styled_df)
```

Updated DataFrame with Extracted Features

	Restaurant Name	Name Length	Address	Address Length
0	Le Petit Souffle	16	Third Floor, Century City Mall, Kalayaan Avenue, Poblacion, Makati City	71
1	Izakaya Kikufuji	16	Little Tokyo, 2277 Chino Roces Avenue, Legaspi Village, Makati City	67
2	Heat - Edsa Shangri-La	22	Edsa Shangri-La, 1 Garden Way, Ortigas, Mandaluyong City	56
3	Ooma	4	Third Floor, Mega Fashion Hall, SM Megamall, Ortigas, Mandaluyong City	70
4	Sambo Kojin	11	Third Floor, Mega Atrium, SM Megamall, Ortigas, Mandaluyong City	64
5	Din Tai Fung	12	Ground Floor, Mega Fashion Hall, SM Megamall, Ortigas, Mandaluyong City	71
6	Buffet 101	10	Building K, SM By The Bay, Sunset Boulevard, Mall of Asia Complex (MOA), Pasay City	83
7	Vikings	7	Building B, By The Bay, Seaside Boulevard, Mall of Asia Complex (MOA), Pasay City	81
8	Spiral - Sofitel Philippine Plaza Manila	40	Plaza Level, Sofitel Philippine Plaza Manila, CCP Complex, Pasay City	69
9	Locavore	8	Brixton Technology Center, 10 Brixton Street, Kapitolyo, Pasig City	67
10	Silantro Fil-Mex	16	75 East Capitol Drive, Kapitolyo, Pasig City	44
11	Mad Mark's Creamery & Good Eats	31	23 East Capitol Drive, Kapitolyo, Pasig City	44
12	Silantro Fil-Mex	16	Second Floor, UP Town Center, Katipunan Avenue, Diliman, Quezon City	68
13	Guevarra's	10	387 P. Guevarra Corner Argonne Street, Addition Hills, San Juan City	68
14	Sodam Korean Restaurant	23	17 J. Abad Santos Drive, Little Baguio, San Juan City	53
15	Cafe Arabelle	13	Ayala Mall, Solenad, Nuvali, Santa Rosa - Tagaytay Road, Don Jose, Santa Rosa	77
16	Nonna's Pasta & Pizzeria	24	Ground Floor, Building G, Solenad 3, Nuvali, Don Jose, Santa Rosa	65
17	Balay Dako	10	Aguinaldo Highway, Tagaytay City	32

	Restaurant Name	Name Length	Address	Address Length
18	Hobing Korean Dessert Cafe	26	Third Floor, BGC Stopover Pavillion, Rizal Drive Corner 31st Street, Bonifacio Global City, Taguig City	103
19	Wildflour Cafe + Bakery	23	Ground Floor, Netlima Building, 4th Avenue Corner 26th Street, Bonifacio Global City, Taguig City	97

2. Top 10 Restaurants by Name and Address Length:

- I identified and displayed the top 10 restaurants with the longest names and addresses. This analysis provides insight into which establishments have the most elaborate names and addresses, which could be indicative of unique branding or complex locations.

In [9]:
```
df['Restaurant Name Length'] = df['Restaurant Name'].apply(lambda x: len(str(x)))
top_10_restaurants_by_length = df.nlargest(10, 'Restaurant Name Length')[['Restaura
def highlight_top10(s):
    """Highlight the name length columns with a light blue background."""
    return ['background-color: lightblue' for _ in s]
styled_top10_df = top_10_restaurants_by_length.style \
    .set_caption("Top 10 Restaurants by Name Length") \
    .apply(highlight_top10, subset=['Restaurant Name Length']) \
    .set_table_styles({
        'Restaurant Name': [{'selector': 'td', 'props': [('color', 'black')]}],
        'Restaurant Name Length': [{'selector': 'td', 'props': [('color', 'black')]
    })
display(styled_top10_df)
```

Top 10 Restaurants by Name Length

	Restaurant Name	Restaurant Name Length
712	Madhuban Restaurant - Welcome Hotel Rama International	54
2349	Chao Chinese Bistro - Holiday Inn Jaipur City Centre	52
2348	Monarch Restaurant - Holiday Inn Jaipur City Centre	51
6529	The Great Kabab Factory - Radisson Blu Plaza Delhi	50
1500	Illusion The Lounge Bar - Hotel Clark Inn Gurgaon	49
2306	Jonathan's Kitchen - Holiday Inn Express & Suites	49
2883	Pt.Kanhaiyalal & Durga Prasad Dixit Paranthe Wale	49
7516	Caf�? Knosh - The Leela Ambience Convention Hotel	49
7517	Cherry Bar - The Leela Ambience Convention Hotel	48
2179	The Living Room - The Westin Sohna Resort & Spa	47

In [10]:
```
df['Address Length'] = df['Address'].apply(lambda x: len(str(x)))
top_10_restaurants_by_address_length = df.nlargest(10, 'Address Length')[['Address'

def highlight_top10(s):
    """Highlight the address length columns with a light blue background."""
    return ['background-color: lightblue' for _ in s]
styled_top10_address_df = top_10_restaurants_by_address_length.style \
    .set_caption("Top 10 Restaurants by Address Length") \
    .apply(highlight_top10, subset=['Address Length']) \
    .set_table_styles({
        'Address': [{'selector': 'td', 'props': [('color', 'black')]}],
        'Address Length': [{'selector': 'td', 'props': [('color', 'black')]}]
    })
display(styled_top10_address_df)
```

Top 10 Restaurants by Address Length

	Address	Address Length
3458	34, 102B, 109 & 110, Ground/1st Floor, Multilevel Parking Cum Commercial Complex (MLCP), DLF South Square, Sarojini Nagar, New Delhi	132
2307	Plot 483, 4th Floor, Pemmasani Complex, Bajaj Electronics Building, Near Madhapur Police Station, Road 36, Jubilee Hills, Hyderabad	131
8987	Shop G-2, Ground Floor, Commercial Complex, RG Residency, Plot GH-2, Opposite Prateek Laurel, Sector 120, Near, Sector 72, Noida	128
9417	Barwa Commercial Avenue, Near Thursday & Friday Market Building, Near F Ring Road, Main Industrial Area Road, Ain Khalid, Doha	126
9261	28-10-33/6, Jagdamba Commercial Complex, Chitralaya Cinema Hall, Jagadamba Junction, Visakhapatnam, Jagadamba Junction, Vizag	125
7725	Shop 4, Plot 11, Behind Sanchit Medicos, Opposite Fortis Hospital, Aruna Asaf Ali road, Kishan Garh, Vasant Kunj, New Delhi	123
2545	2nd Floor, City Centre Mall, Sambaji Chowk, Lawate Nagar, Banyan Square, Untwadi, Off Parijat Nagar, Parijat Nagar, Nashik	122
2553	Opposite Asaram Bapu Ashram Bridge, Near Nandawan Lawns, Savarkar Nagar Extension, Off Gangapur Road, College Road, Nashik	122
1745	165/6, Shining Sun Tower, Furniture Market Near Government Girls School, Jacobpura, New Railway Road, Sector 14, Gurgaon	120
4720	C-7, Tilak Market, Major Pankaj Batra Marg, Block 6, Sharda Puri, Ramesh Nagar, Near Kirti Nagar, Kirti Nagar, New Delhi	120

In [11]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9551 entries, 0 to 9550
Data columns (total 23 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  
 21  Address Length          9551 non-null   int64  
 22  Restaurant Name Length  9551 non-null   int64  
dtypes: float64(3), int64(7), object(13)
memory usage: 1.7+ MB
```

In [20]:

```
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from IPython.display import display, HTML
file_path = 'D:/cognifyz/Dataset .csv'
df = pd.read_csv(file_path)
categorical_columns = df.select_dtypes(include=['object']).columns
categorical_columns
Label_Encoder = LabelEncoder()
for col in categorical_columns:
    df[col] = Label_Encoder.fit_transform(df[col])
def highlight_categorical(s):
    """Highlight the categorical columns with a light grey background."""
    is_categorical = s.name in categorical_columns
    return ['background-color: lightgrey' if is_categorical else '' for _ in s]
styled_df = df.head(20).style \
    .set_caption("Updated DataFrame with Encoded Categorical Features") \
    .apply(highlight_categorical, axis=0) \
    .set_table_styles({col: [{'selector': 'td', 'props': [('color', 'black')]}] for
display(styled_df)
```

Updated DataFra

	Restaurant ID	Restaurant Name	Country Code	City	Address	Locality	Locality Verbose	Longitude	Latitude
0	6317637	3748	162	73	8685	171	172	121.027535	14.565443
1	6304287	3172	162	73	6055	593	601	121.014101	14.553708
2	6300002	2896	162	75	4684	308	314	121.056831	14.581404
3	6318506	4707	162	75	8690	862	875	121.056475	14.585318
4	6314302	5523	162	75	8689	862	875	121.057508	14.584450
5	18189371	2070	162	75	5406	862	875	121.056314	14.583764
6	6300781	1006	162	94	3984	863	876	120.979667	14.531333
7	6301290	7179	162	94	3983	863	876	120.979333	14.540000
8	6300010	6045	162	94	6979	1001	1054	120.980090	14.552990
9	6314987	3812	162	95	3966	516	522	121.056532	14.572041
10	6309903	5850	162	95	2838	516	522	121.057916	14.567689
11	6309455	3908	162	95	1283	516	522	121.060820	14.570849
12	6318433	5850	162	107	7461	1111	1166	121.075419	14.649503
13	6310470	2758	162	112	2055	10	10	121.033592	14.593450
14	6314605	5941	162	112	810	592	600	121.038110	14.598890
15	18185059	1087	162	114	3546	726	737	121.057040	14.237082
16	18182702	4605	162	114	5347	1003	1056	121.056587	14.237679
17	6318213	615	162	122	3466	1039	1092	120.951589	14.101834
18	18255654	2937	162	123	8684	84	85	121.045878	14.554360
19	6308205	7281	162	123	5412	132	133	121.046220	14.549337

In [22]:
```
df['Restaurant Name Length'] = df['Restaurant Name'].apply(lambda x: len(str(x)))
top_10_restaurants_by_name_length = df.nlargest(10, 'Restaurant Name Length')[['Res
def highlight_long_names(s):
    """Highlight the name length column with a light blue background."""
    is_length = s.name in ['Restaurant Name Length']
    return ['background-color: lightblue' if is_length else '' for _ in s]

styled_top_10_restaurants = top_10_restaurants_by_name_length.style \
    .set_caption("Top 10 Restaurants with Longest Names") \
    .apply(highlight_long_names, axis=0) \
    .set_table_styles({
        'Restaurant Name': [{'selector': 'td', 'props': [('color', 'black')]}],
```

```
        'Restaurant Name Length': [{'selector': 'td', 'props': [('color', 'black')]
    })
display(styled_top_10_restaurants)
```

Top 10 Restaurants with Longest Names

	Restaurant Name	Restaurant Name Length
712	Madhuban Restaurant - Welcome Hotel Rama International	54
2349	Chao Chinese Bistro - Holiday Inn Jaipur City Centre	52
2348	Monarch Restaurant - Holiday Inn Jaipur City Centre	51
6529	The Great Kabab Factory - Radisson Blu Plaza Delhi	50
1500	Illusion The Lounge Bar - Hotel Clark Inn Gurgaon	49
2306	Jonathan's Kitchen - Holiday Inn Express & Suites	49
2883	Pt.Kanhaiyalal & Durga Prasad Dixit Paranthe Wale	49
7516	Caf�� Knosh - The Leela Ambience Convention Hotel	49
7517	Cherry Bar - The Leela Ambience Convention Hotel	48
2179	The Living Room - The Westin Sohna Resort & Spa	47

In [24]:
```
df['Address Length'] = df['Address'].apply(lambda x: len(str(x)))
top_10_restaurants_by_address_length = df.nlargest(10, 'Address Length')[['Restaura
def highlight_long_addresses(s):
    """Highlight the address length column with a light yellow background."""
    is_length = s.name in ['Address Length']
    return ['background-color: lightyellow' if is_length else '' for _ in s]

styled_top_10_restaurants_by_address = top_10_restaurants_by_address_length.style \
    .set_caption("Top 10 Restaurants with Longest Addresses") \
    .apply(highlight_long_addresses, axis=0) \
    .set_table_styles({
        'Restaurant Name': [{'selector': 'td', 'props': [('color', 'black')]}],
        'Address': [{'selector': 'td', 'props': [('color', 'black')]}],
        'Address Length': [{'selector': 'td', 'props': [('color', 'black')]}]
    })
display(styled_top_10_restaurants_by_address)
```

Top 10 Restaurants with Longest Addresses

	Restaurant Name	Address	Address Length
3458	McDonald's	34, 102B, 109 & 110, Ground/1st Floor, Multilevel Parking Cum Commercial Complex (MLCP), DLF South Square, Sarojini Nagar, New Delhi	132
2307	AB's - Absolute Barbecues	Plot 483, 4th Floor, Pemmasani Complex, Bajaj Electronics Building, Near Madhapur Police Station, Road 36, Jubilee Hills, Hyderabad	131
8987	Domino's Pizza	Shop G-2, Ground Floor, Commercial Complex, RG Residency, Plot GH-2, Opposite Prateek Laurel, Sector 120, Near, Sector 72, Noida	128
9417	7st by Mumbai Spices	Barwa Commercial Avenue, Near Thursday & Friday Market Building, Near F Ring Road, Main Industrial Area Road, Ain Khalid, Doha	126
9261	Alpha Hotel	28-10-33/6, Jagdamba Commercial Complex, Chitralaya Cinema Hall, Jagadamba Junction, Visakhapatnam, Jagadamba Junction, Vizag	125
7725	The Village Caf�?	Shop 4, Plot 11, Behind Sanchit Medicos, Opposite Fortis Hospital, Aruna Asaf Ali road, Kishan Garh, Vasant Kunj, New Delhi	123
2545	Barbeque Nation	2nd Floor, City Centre Mall, Sambaji Chowk, Lawate Nagar, Banyan Square, Untwadi, Off Parijat Nagar, Parijat Nagar, Nashik	122
2553	RiverDine Restaurant & Bar	Opposite Asaram Bapu Ashram Bridge, Near Nandawan Lawns, Savarkar Nagar Extension, Off Gangapur Road, College Road, Nashik	122
1745	Mr. & Mrs. Idly	165/6, Shining Sun Tower, Furniture Market Near Government Girls School, Jacobpura, New Railway Road, Sector 14, Gurgaon	120
4720	Yakooz	C-7, Tilak Market, Major Pankaj Batra Marg, Block 6, Sharda Puri, Ramesh Nagar, Near Kirti Nagar, Kirti Nagar, New Delhi	120

2. Categorical Feature Encoding 🔠 📊 🔢

Creation of Binary Features:

- Has Table Booking: We encoded the presence of table booking services as a binary feature. This helps in distinguishing between restaurants that offer table booking and those that do not.
- Has Online Delivery: Similarly, we encoded the availability of online delivery services as a binary feature. This facilitates analysis by converting categorical responses into numerical format.

In [61]:
```
df['Has Table Booking'] = df['Has Table booking'].map({'Yes': 1, 'No': 0})
df['Has Online Delivery'] = df['Has Online delivery'].map({'Yes': 1, 'No': 0})
def highlight_binary(s):
    color = 'background-color: lightpink'
    return [color if val == 1 else '' for val in s]
styled_df = df[['Has Table booking', 'Has Table Booking', 'Has Online delivery', 'H
    .set_caption("Updated DataFrame with Encoded Features") \
    .highlight_max(color='lightpink', subset=['Has Table Booking', 'Has Online Deli
    .apply(highlight_binary, subset=['Has Table Booking', 'Has Online Delivery']) \
    .set_table_styles({
        'Has Table booking': [{'selector': 'td', 'props': [('color', 'black')]}],
        'Has Table Booking': [{'selector': 'td', 'props': [('color', 'black')]}],
        'Has Online delivery': [{'selector': 'td', 'props': [('color', 'black')]}],
        'Has Online Delivery': [{'selector': 'td', 'props': [('color', 'black')]}]
    })
styled_df
```

Out[61]:

Updated DataFrame with Encoded Features

	Has Table booking	Has Table Booking	Has Online delivery	Has Online Delivery
0	Yes	1	No	0
1	Yes	1	No	0
2	Yes	1	No	0
3	No	0	No	0
4	Yes	1	No	0
5	No	0	No	0
6	Yes	1	No	0
7	Yes	1	No	0
8	Yes	1	No	0
9	Yes	1	No	0
10	No	0	No	0
11	Yes	1	No	0
12	No	0	No	0
13	Yes	1	No	0
14	No	0	No	0
15	No	0	No	0
16	No	0	No	0
17	Yes	1	No	0
18	No	0	No	0
19	Yes	1	No	0

3. Distribution of Encoded Features 📊 🔢 📈

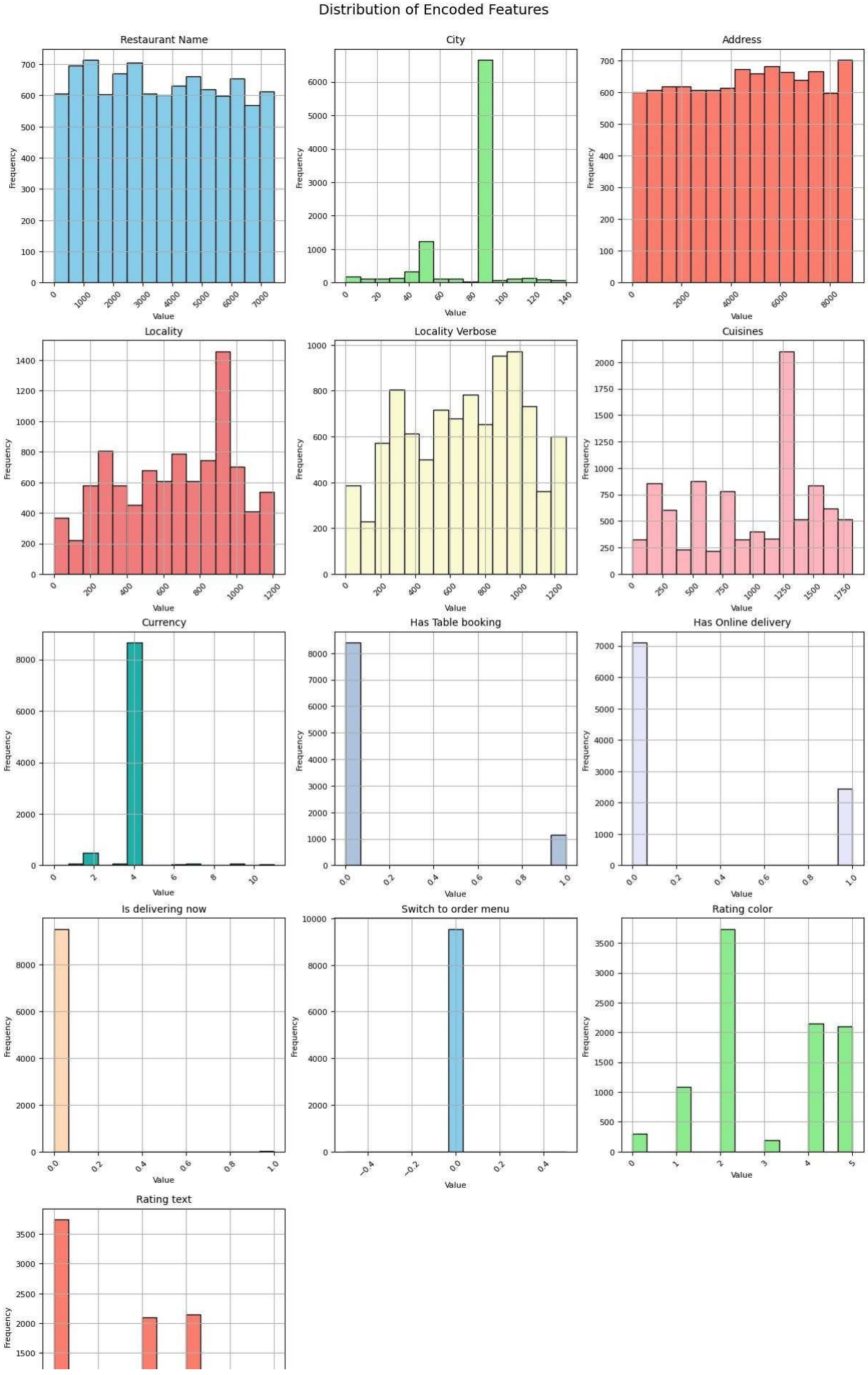
- I generated histograms to visualize the distribution of encoded features. The histograms show the frequency of each value (e.g., 0 or 1) for the binary features. This visualization helps in understanding the proportion of restaurants that offer table booking or online delivery compared to those that do not.

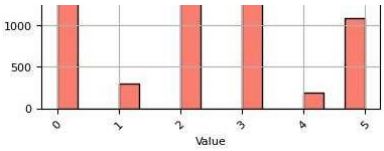
In [37]:

```
import matplotlib.pyplot as plt
def plot_histograms(df, columns):
    """Plot histograms for specified columns with customized colors."""
    num_plots = len(columns)
    cols = 3
    rows = (num_plots // cols) + int(num_plots % cols != 0)
```

```
    fig, axes = plt.subplots(nrows=rows, ncols=cols, figsize=(12, 4 * rows), constr
    axes = axes.flatten() if num_plots > 1 else [axes]
    colors = ['skyblue', 'lightgreen', 'salmon', 'lightcoral', 'lightgoldenrodyello
              'lightpink', 'lightseagreen', 'lightsteelblue', 'lavender', 'peachpuf
    for i, col in enumerate(columns):
        df[col].hist(ax=axes[i], bins=15, edgecolor='black', color=colors[i % len(c
        axes[i].set_title(col, fontsize=10)
        axes[i].set_xlabel('Value', fontsize=8)
        axes[i].set_ylabel('Frequency', fontsize=8)
        axes[i].tick_params(axis='x', rotation=45, labelsize=8)
        axes[i].tick_params(axis='y', labelsize=8)
        for j in range(num_plots, len(axes)):
        axes[j].axis('off')

    plt.suptitle('Distribution of Encoded Features', fontsize=14, y=1.02)
    plt.show()
plot_histograms(df, categorical_columns)
```


Distribution of Encoded Features
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
Frequency
Value



Analysis and Insights 🔍 📊 💡

Feature Extraction Insights:

- Restaurant Name Length: Longer names may reflect more descriptive or thematic choices by restaurant owners, possibly related to branding strategies.
- Address Length: Addresses with more characters might indicate detailed or complex locations, providing insight into logistical considerations.

Categorical Feature Encoding Insights:

- Binary Features: Encoding categorical variables into binary format simplifies analysis and makes the data more suitable for machine learning algorithms. This conversion allows for easier integration and interpretation in predictive models.

Distribution Visualization Insights:

- Frequency Analysis: The histograms reveal the distribution of binary features, helping to understand the prevalence of services like table booking and online delivery. This information can guide decisions about service offerings and operational strategies.

Conclusion 📈 ✅

Enhanced Data Understanding:

- The addition of length-based features and binary encoding enriches the dataset, allowing for more detailed analysis and modeling.

Preparedness for Advanced Analysis:

- The dataset is now better prepared for advanced analysis and machine learning, with clear and numerically encoded features that facilitate more straightforward integration into predictive models.

Visual Insights for Decision Making:

- The visual representation of feature distributions offers valuable insights into service availability and naming conventions, supporting informed decision-making.

Summary of Insights 📝 💡 🔍

Task 1: Table Booking and Online Delivery

- Table Booking: Only 12.12% of restaurants offer table booking. Restaurants with table booking have a higher average rating (3.44) compared to those without (2.56).
- Online Delivery: 25.66% of restaurants offer online delivery. Restaurants in the lowest price range (Price Range 1) are more likely to offer this service, whereas it's less common in higher price ranges (Price Range 4).

Task 2: Price Range Analysis

- Common Price Range: Price Range 1 is the most prevalent, suggesting a preference for budget-friendly options.
- Average Ratings: Higher price ranges generally receive better ratings. Price Range 4 stands out with the highest average rating.
- Color Coding: The color associated with the highest average rating is reflective of premium pricing.

Task 3: Feature Engineering

- Extracted Features: Added features include the length of restaurant names and addresses.
- Encoded Features: Table booking and online delivery are encoded as binary variables, facilitating easier analysis.

In [ ]: