zomato

Exploratory Data Analysis

By Sumit Kumar

About Zomato:

- **Restaurant Aggregator and Food Delivery:** Zomato is a leading multinational restaurant aggregator and food delivery company, providing information about restaurants, prices, and menu items. It also offers food delivery services.
- **Global Presence:** Zomato operates in over 24 countries, helping millions of people decide where to eat every day.
- **5-Point Rating System:** Zomato uses a 5-point classroom-style grading model for restaurant ratings, which helps normalize scores across different cities.
- **Marketing Strategies:** Zomato employs various marketing strategies, including referral programs, loyalty rewards, discounts, and cashbacks, to retain and attract customers.
- **Founders:** The company was founded by Deepinder Goyal and Pankaj Chaddah in 2008, initially as a restaurant guide called "Foodiebay" before evolving into Zomato

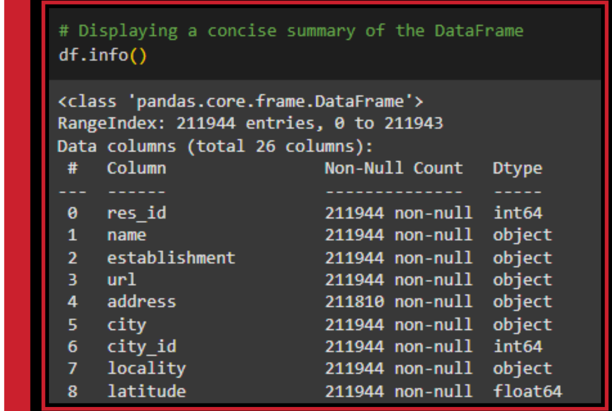
Description of Project:

- **Objective of the Analysis:** The primary goal of this project is to analyze Zomato restaurant data to uncover the key factors that contribute to the success of restaurants, as determined by their ratings.
- **Data Exploration:** By examining various features such as location, cuisine type, pricing, and service offerings, we aim to identify patterns and trends that can influence restaurant ratings.
- **Insight Generation:** The analysis will provide valuable insights that can help restaurant owners understand the crucial elements that drive customer satisfaction and high ratings on Zomato.
- **Decision-Making Support:** The findings from this project will be beneficial for Zomato users, enabling them to make more informed choices when selecting restaurants, and for restaurant owners to strategize their business improvements.
- **Broader Implications:** This project has the potential to contribute to the broader food and beverage industry by highlighting successful practices and offering a data-driven approach to enhance customer experiences and restaurant performance.

Description of Dataset

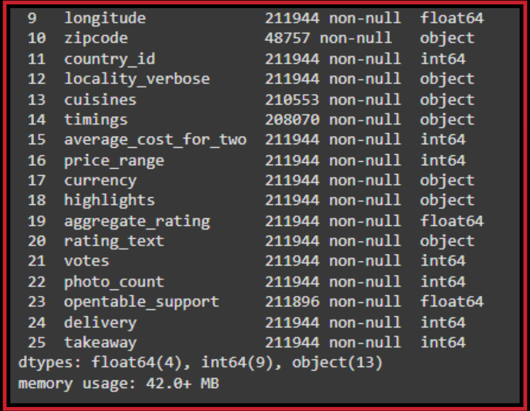
- **Dataset Scope:** Contains detailed information on restaurants across cities, including attributes like names, establishment types, and operational details.
- **Location Details:** Covers city names, addresses, localities, and geographic coordinates for precise mapping and analysis.
- **Customer Feedback:** Includes ratings (scores and descriptive text), votes, and photos, providing insights into customer engagement.Service
- **Highlights:** Features key offerings like delivery, takeaway, dining options, and special highlights such as alcohol service.
- **Data Observations:** While diverse and feature-rich, the dataset includes some missing values in fields like zip codes, cuisines, and operating hours.

Overview of dataset

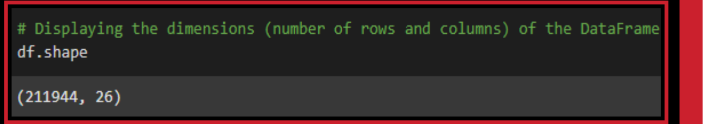


```
# Displaying a concise summary of the DataFrame
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 211944 entries, 0 to 211943
Data columns (total 26 columns):
 #   Column                Non-Null Count   Dtype
---  ------                --------------   -----
 0   res_id                211944 non-null  int64
 1   name                  211944 non-null  object
 2   establishment         211944 non-null  object
 3   url                   211944 non-null  object
 4   address               211810 non-null  object
 5   city                  211944 non-null  object
 6   city_id               211944 non-null  int64
 7   locality              211944 non-null  object
 8   latitude              211944 non-null  float64
```

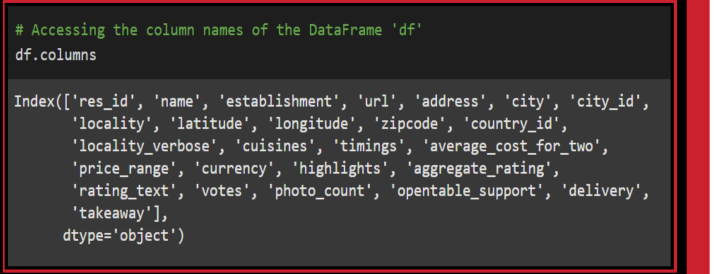


```
 9   longitude             211944 non-null  float64
 10  zipcode               48757 non-null   object
 11  country_id            211944 non-null  int64
 12  locality_verbose      211944 non-null  object
 13  cuisines              210553 non-null  object
 14  timings               208070 non-null  object
 15  average_cost_for_two  211944 non-null  int64
 16  price_range           211944 non-null  int64
 17  currency              211944 non-null  object
 18  highlights            211944 non-null  object
 19  aggregate_rating      211944 non-null  float64
 20  rating_text           211944 non-null  object
 21  votes                 211944 non-null  int64
 22  photo_count           211944 non-null  int64
 23  opentable_support     211896 non-null  float64
 24  delivery              211944 non-null  int64
 25  takeaway              211944 non-null  int64
dtypes: float64(4), int64(9), object(13)
memory usage: 42.0+ MB
```



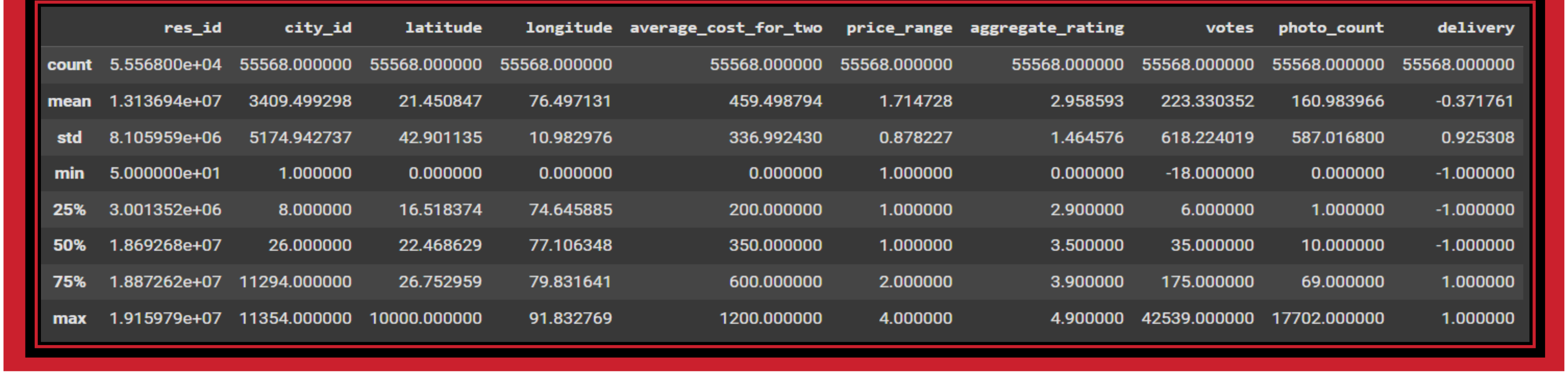
```
# Displaying the dimensions (number of rows and columns) of the DataFrame
df.shape
```

```
(211944, 26)
```



```
# Accessing the column names of the DataFrame 'df'
df.columns
```

```
Index(['res_id', 'name', 'establishment', 'url', 'address', 'city', 'city_id',
       'locality', 'latitude', 'longitude', 'zipcode', 'country_id',
       'locality_verbose', 'cuisines', 'timings', 'average_cost_for_two',
       'price_range', 'currency', 'highlights', 'aggregate_rating',
       'rating_text', 'votes', 'photo_count', 'opentable_support', 'delivery',
       'takeaway'],
      dtype='object')
```



	res_id	city_id	latitude	longitude	average_cost_for_two	price_range	aggregate_rating	votes	photo_count	delivery
count	5.556800e+04	55568.000000	55568.000000	55568.000000	55568.000000	55568.000000	55568.000000	55568.000000	55568.000000	55568.000000
mean	1.313694e+07	3409.499298	21.450847	76.497131	459.498794	1.714728	2.958593	223.330352	160.983966	-0.371761
std	8.105959e+06	5174.942737	42.901135	10.982976	336.992430	0.878227	1.464576	618.224019	587.016800	0.925308
min	5.000000e+01	1.000000	0.000000	0.000000	0.000000	1.000000	0.000000	-18.000000	0.000000	-1.000000
25%	3.001352e+06	8.000000	16.518374	74.645885	200.000000	1.000000	2.900000	6.000000	1.000000	-1.000000
50%	1.869268e+07	26.000000	22.468629	77.106348	350.000000	1.000000	3.500000	35.000000	10.000000	-1.000000
75%	1.887262e+07	11294.000000	26.752959	79.831641	600.000000	2.000000	3.900000	175.000000	69.000000	1.000000
max	1.915979e+07	11354.000000	10000.000000	91.832769	1200.000000	4.000000	4.900000	42539.000000	17702.000000	1.000000

Dataset Summary

Data Shape:

- The dataset has 211,944 rows and 26 columns.

Data Types:

- Integers: res_id, city_id, average_cost_for_two, price_range, votes, photo_count, delivery, takeaway
- Floats: latitude, longitude, aggregate_rating
- Objects: name, establishment, url, address, city, locality, zipcode, country_id, locality_verbose, cuisines, timings, currency, highlights, rating_text

Missing Values:

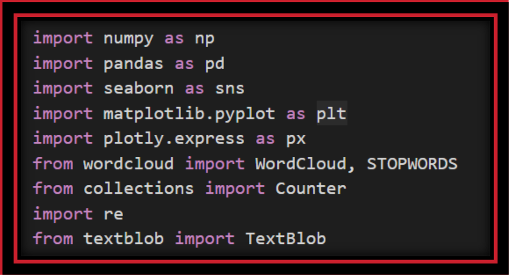
- zipcode: 48,757 null values (approximately 23% of the dataset).
- cuisines: 13,411 null values (approximately 6.3% of the dataset).

Basic Statistics:

- average_cost_for_two has a mean of 459.49, median of 350.00, minimum of 0, and maximum of 1200.
- 25% of restaurants fall into each price_range (1, 2, 3, and 4).
- aggregate_rating has a mean of 2.96, median of 3.50, minimum of 0.00, and maximum of 4.90.
- votes has a mean of 223.33, median of 35.00, minimum of -18.00, and a maximum of 42539.00.
- photo_count has a mean of 160.98, median of 10.00, minimum of 0.00, and a maximum of 17702.00.

Libraries used:

1. NumPy: A library for numerical computing, providing support for large multi-dimensional arrays and matrices, along with a collection of mathematical functions to operate on these arrays.
2. Pandas: A powerful data manipulation library that provides data structures like DataFrame and Series for handling and analyzing structured data efficiently.
3. Seaborn: A data visualization library based on Matplotlib that makes it easy to generate attractive and informative statistical graphics.
4. Matplotlib: A plotting library for creating static, animated, and interactive visualizations in Python.
5. Plotly Express: A library for creating interactive, web-based visualizations with minimal code, often used for dashboards and web applications.
6. WordCloud: A package for generating word clouds from text data, which visually represents the frequency of words in a given corpus.
7. STOPWORDS: A collection of common words (such as 'and', 'the', etc.) used in text analysis to filter out unimportant words during processing.
8. Counter: A class from the collections module used to count the occurrences of elements in an iterable, often applied for text frequency analysis.
9. re: A library for working with regular expressions, providing tools for pattern matching, string searching, and manipulation.
10. TextBlob: A library for natural language processing (NLP) that simplifies tasks like part-of-speech tagging, noun phrase extraction, and sentiment analysis.



```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import plotly.express as px
from wordcloud import WordCloud, STOPWORDS
from collections import Counter
import re
from textblob import TextBlob
```

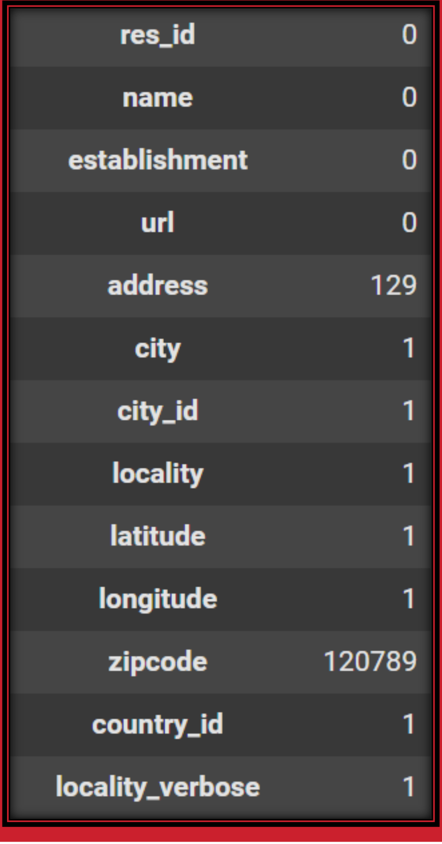
Columns Description:

1. **res_id:** restaurant id
2. **name:** name of the restaurant
3. **establishment:** type of establishment (e.g., quick bites, fine dining)
4. **url:** url of the restaurant's zomato page
5. **address:** full address of the restaurant
6. **city:** city where the restaurant is located
7. **city_id:** id of the city
8. **locality:** locality of the restaurant
9. **latitude:** latitude coordinate
10. **longitude:** longitude coordinate
11. **zipcode:** zip code of the restaurant
12. **country_id:** country id
13. **locality_verbose:** detailed locality description
14. **cuisines:** types of cuisines offered
15. **timings:** operating hours
16. **average_cost_for_two:** average cost for two people
17. **price_range:** price range indicator
18. **currency:** currency of the cost
19. **highlights:** key highlights and features
20. **aggregate_rating:** overall rating
21. **rating_text:** textual representation of the rating
22. **votes:** number of votes
23. **photo_count:** number of photos
24. **opentable_support:** opentable support availability
25. **delivery:** delivery availability
26. **takeaway:** takeaway availability

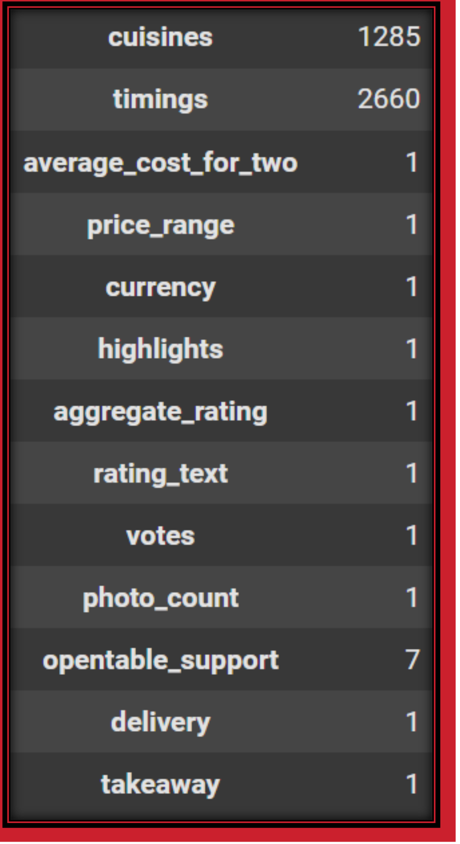
Data Cleaning & Pre-Processing

Checking for the Null Values

1. **Columns with High Missing Values:** The zipcode column has the highest number of missing values (120,789 entries), followed by the timings column with 2,660 missing entries.
2. **Columns with Moderate Missing Values:** The cuisines column has 1,285 missing values, and the address column has 129 missing values.
3. **Columns with Minimal Missing Values:** Several columns have minimal missing values, including city, latitude, longitude, price_range, currency, and others, with missing values ranging from 1 to 7 per column.



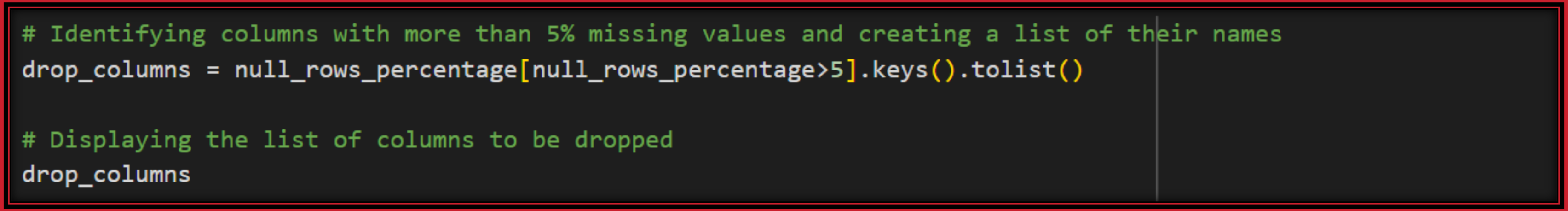
res_id	0
name	0
establishment	0
url	0
address	129
city	1
city_id	1
locality	1
latitude	1
longitude	1
zipcode	120789
country_id	1
locality_verbose	1



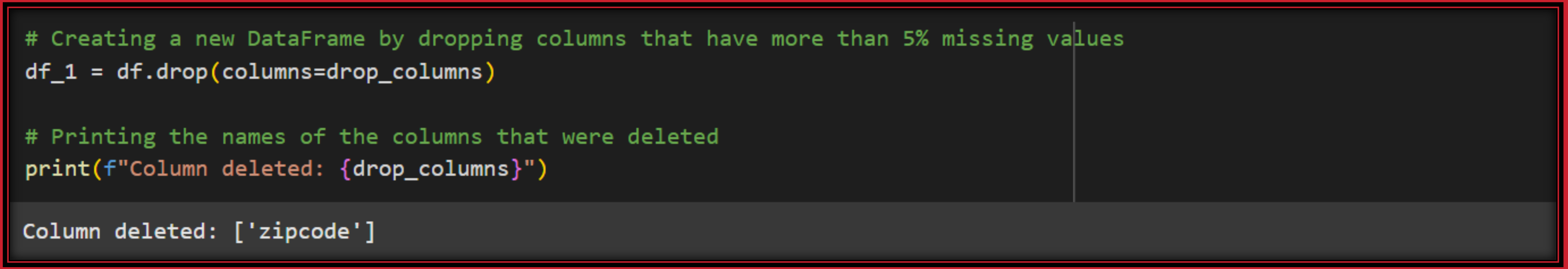
cuisines	1285
timings	2660
average_cost_for_two	1
price_range	1
currency	1
highlights	1
aggregate_rating	1
rating_text	1
votes	1
photo_count	1
opentable_support	7
delivery	1
takeaway	1

Handling Null Values

- Removed columns with more than 5% null values to improve the dataset's quality, and created a new dataframe df_1 after dropping the columns.



```
# Identifying columns with more than 5% missing values and creating a list of their names
drop_columns = null_rows_percentage[null_rows_percentage>5].keys().tolist()

# Displaying the list of columns to be dropped
drop_columns
```



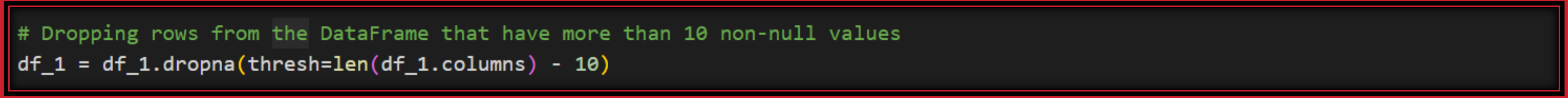
```
# Creating a new DataFrame by dropping columns that have more than 5% missing values
df_1 = df.drop(columns=drop_columns)

# Printing the names of the columns that were deleted
print(f"Column deleted: {drop_columns}")
```

```
Column deleted: ['zipcode']
```

- Dropped rows from the DataFrame that had more than 10 null values to ensure the quality of the dataset.
- Retained rows with sufficient data, resulting in a more reliable df_1 for analysis.

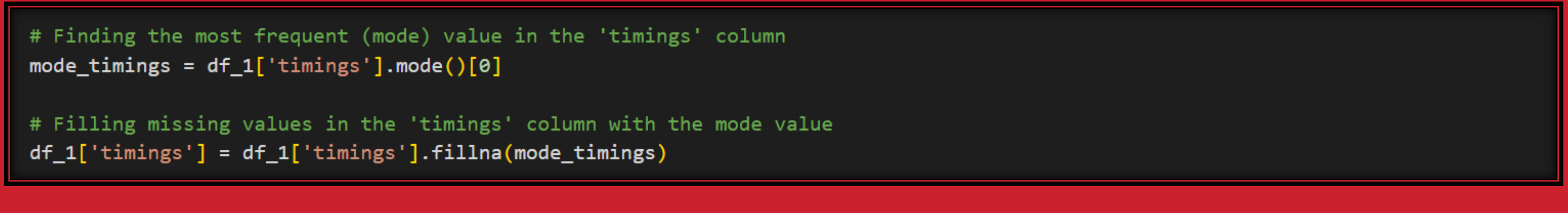
Handling Null Values

- Decided to do this because there were many columns with only 1 null value.
- Checked that all those columns had a common null row and deleted that particular row from the dataset.



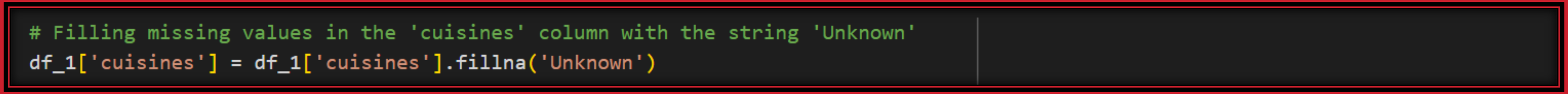
```python
# Dropping rows from the DataFrame that have more than 10 non-null values
df_1 = df_1.dropna(thresh=len(df_1.columns) - 10)
```

- Filled the remaining null values in remaining columns based on their unique values and data type
- Used the mode to fill null values in the timings column.
- Assigned "Unknown" to fill null values in the cuisines column.
- Used the mode to fill null values in the opentable_support column.
- Filled missing values in the address column by concatenating the locality and city columns.

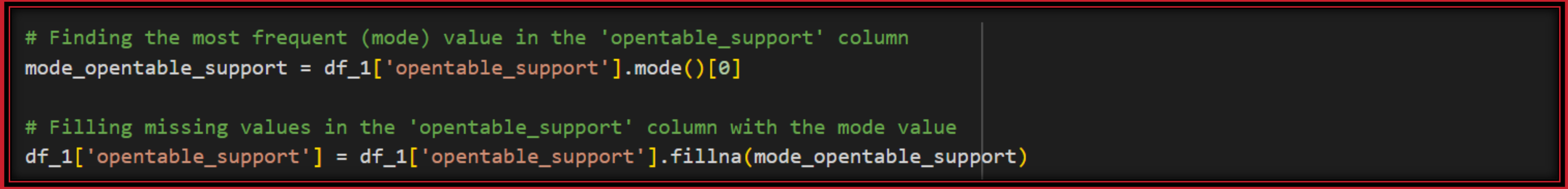


```python
# Finding the most frequent (mode) value in the 'timings' column
mode_timings = df_1['timings'].mode()[0]

# Filling missing values in the 'timings' column with the mode value
df_1['timings'] = df_1['timings'].fillna(mode_timings)
```

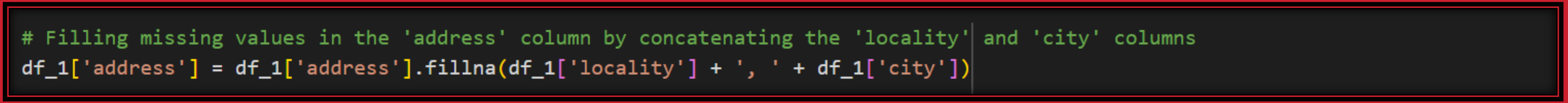
Handling Null Values



```
# Filling missing values in the 'cuisines' column with the string 'Unknown'
df_1['cuisines'] = df_1['cuisines'].fillna('Unknown')
```



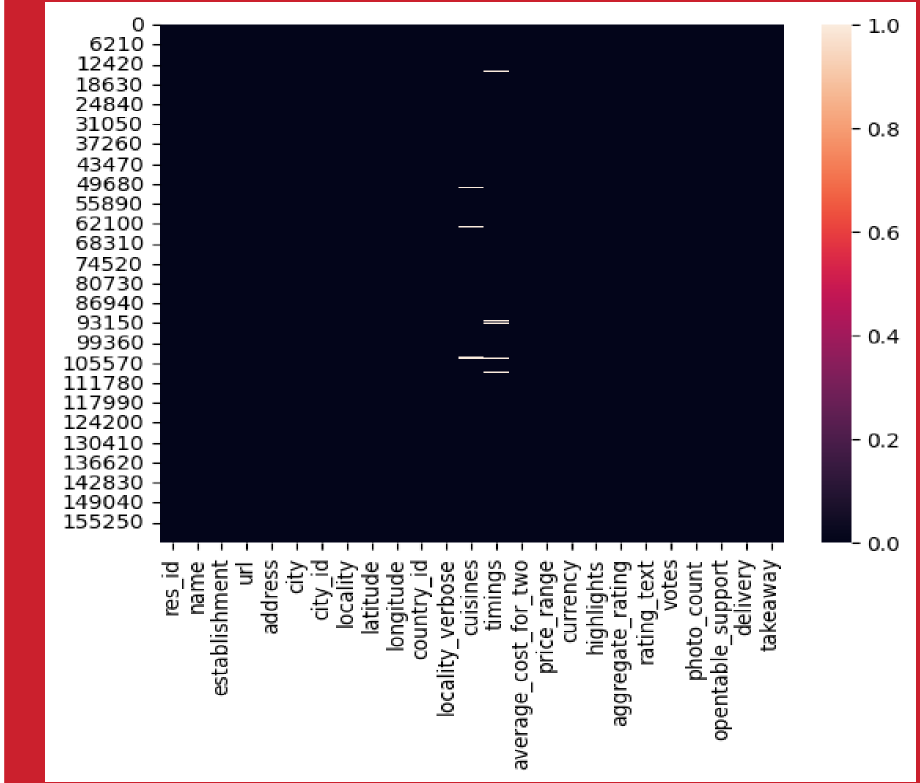
```
# Finding the most frequent (mode) value in the 'opentable_support' column
mode_opentable_support = df_1['opentable_support'].mode()[0]

# Filling missing values in the 'opentable_support' column with the mode value
df_1['opentable_support'] = df_1['opentable_support'].fillna(mode_opentable_support)
```



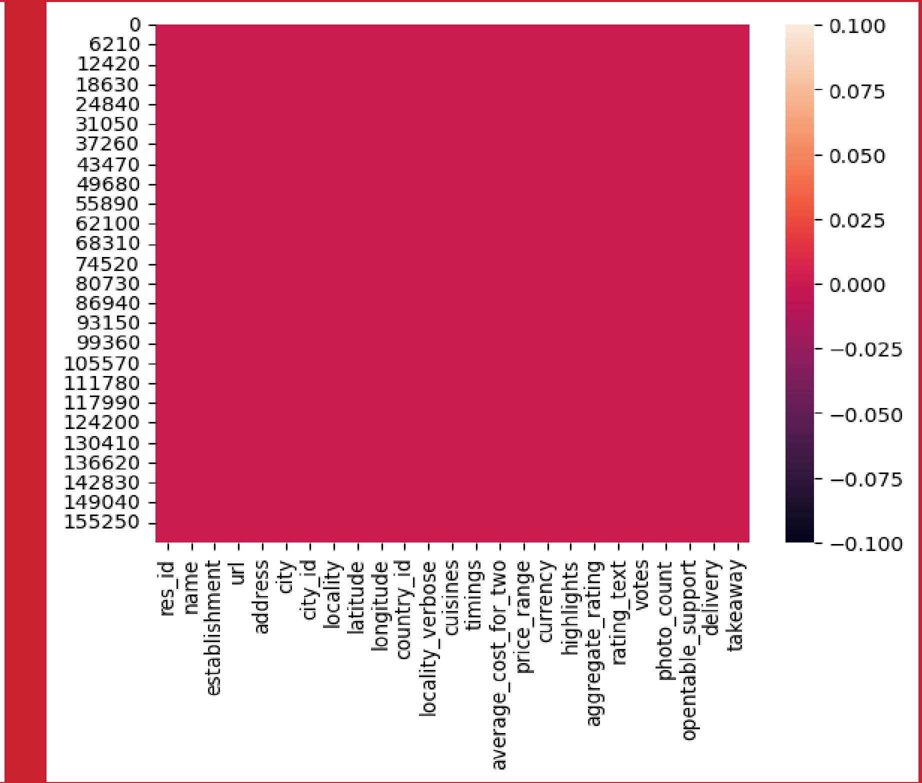
```
# Filling missing values in the 'address' column by concatenating the 'locality' and 'city' columns
df_1['address'] = df_1['address'].fillna(df_1['locality'] + ', ' + df_1['city'])
```

- Null Values Heatmap: Visualizing the presence of null values in the dataset.
- The left-hand heatmap shows the distribution of null values before handling.
- The right-hand heatmap displays the absence of null values after handling.
- This visual comparison underscores the effectiveness of the data cleaning process.

Handling Null Values

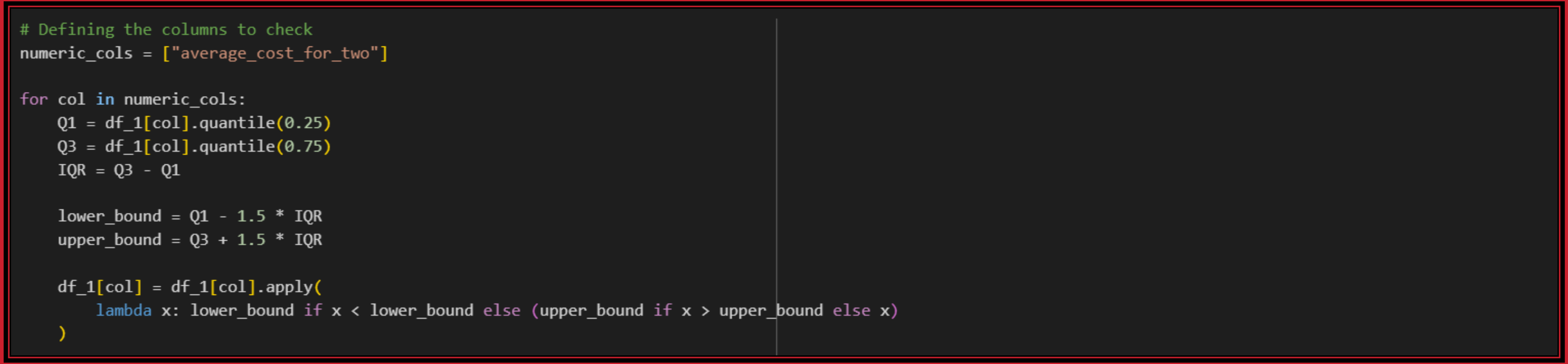


Before



After

Handling Outliers

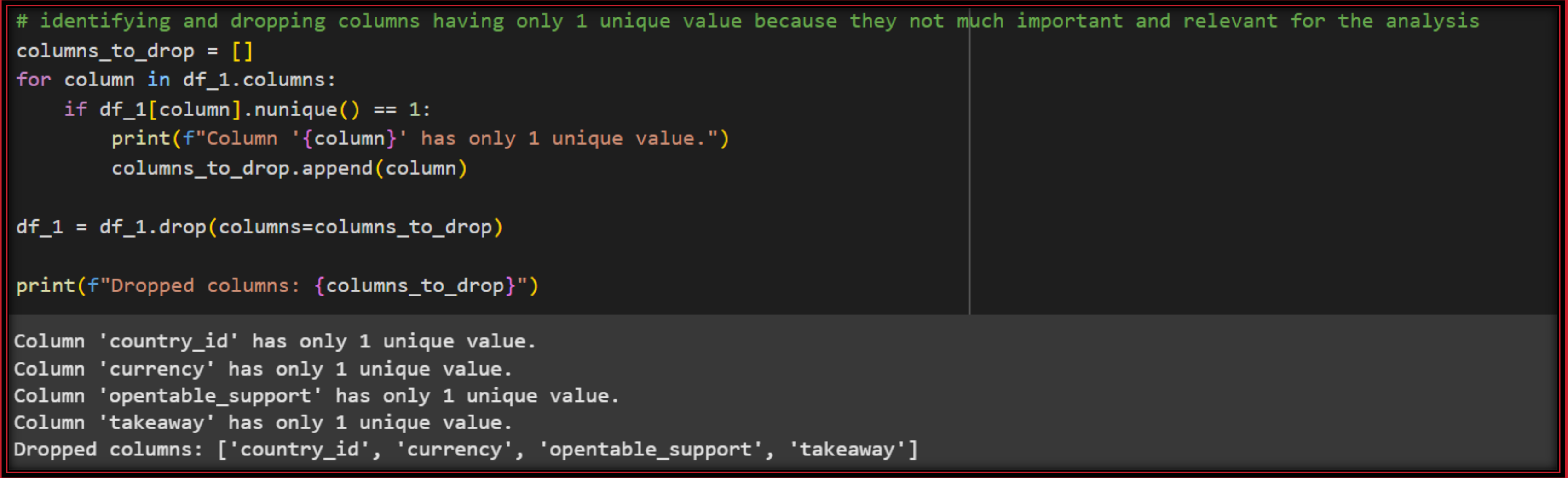


```python
# Defining the columns to check
numeric_cols = ["average_cost_for_two"]

for col in numeric_cols:
    Q1 = df_1[col].quantile(0.25)
    Q3 = df_1[col].quantile(0.75)
    IQR = Q3 - Q1

    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR

    df_1[col] = df_1[col].apply(
        lambda x: lower_bound if x < lower_bound else (upper_bound if x > upper_bound else x)
    )
```

- Outliers in average_cost_for_two were capped using the IQR method to mitigate their impact while preserving the data's distribution.
- average_cost_for_two, unlike features like aggregate_rating and price_range, has a wider range and is more prone to outliers.
- Outliers in average_cost_for_two can heavily skew analyses involving monetary values.
- Other numerical features have inherent limits, reducing outlier influence.
- Focusing on average_cost_for_two balances outlier treatment with data integrity.

Handling Dataset

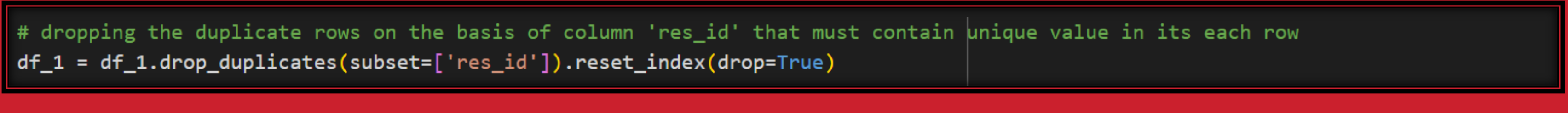
- Removed columns that had only 1 unique value as they do not provide significant information for data analysis.



```python
# identifying and dropping columns having only 1 unique value because they not much important and relevant for the analysis
columns_to_drop = []
for column in df_1.columns:
    if df_1[column].nunique() == 1:
        print(f"Column '{column}' has only 1 unique value.")
        columns_to_drop.append(column)

df_1 = df_1.drop(columns=columns_to_drop)

print(f"Dropped columns: {columns_to_drop}")
```

```
Column 'country_id' has only 1 unique value.
Column 'currency' has only 1 unique value.
Column 'opentable_support' has only 1 unique value.
Column 'takeaway' has only 1 unique value.
Dropped columns: ['country_id', 'currency', 'opentable_support', 'takeaway']
```

- Dropped duplicate rows based on the res_id column, ensuring that each row contains a unique restaurant ID, which is essential for accurate data analysis.



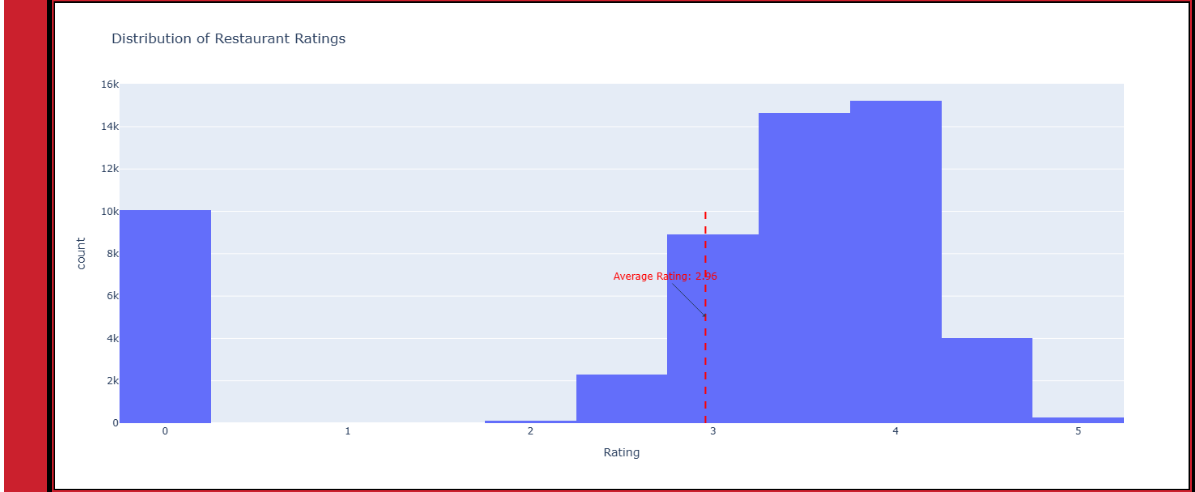
```python
# dropping the duplicate rows on the basis of column 'res_id' that must contain unique value in its each row
df_1 = df_1.drop_duplicates(subset=['res_id']).reset_index(drop=True)
```

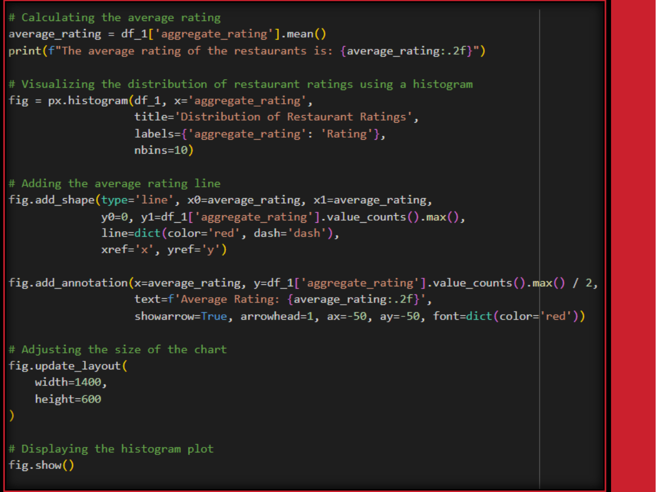
Handling Dataset

- **Dataset Summary:** The initial dataset df had a shape of (161,455, 26). After cleaning, the new dataset df_1 has a shape of (43,253, 21). This reflects the thorough cleaning, ensuring the integrity and reliability of the dataset for analysis.
- **Data Cleaning:** Removed columns with more than 5% null values and created a new dataframe df_1. Dropped rows from the DataFrame with more than 10 null values to retain rows with sufficient data.
- **Null Values Heatmap:** Visualized null values in the dataset. The left-hand heatmap shows the distribution before handling, and the right-hand heatmap shows the absence after handling. This confirms the effectiveness of the cleaning process.
- **Handling Missing Values:** Filled missing values in three specific columns based on their unique values. Used the mode for timings and opentable_support columns, and assigned "Unknown" for cuisines. Filled missing values in address by concatenating locality and city.
- **Dropped Columns and Duplicate Rows:** Removed columns with only 1 unique value as they do not provide significant information. Dropped duplicate rows based on res_id to ensure each row has a unique restaurant ID, reducing noise and improving dataset quality.

Data Analysis and Visualization

- ❑ Calculating and Visualizing Average Restaurant Ratings
- ❑ Analysing the Distribution of Ratings to Understand the Overall Landscape





```python
# Calculating the average rating
average_rating = df_1['aggregate_rating'].mean()
print(f"The average rating of the restaurants is: {average_rating:.2f}")

# Visualizing the distribution of restaurant ratings using a histogram
fig = px.histogram(df_1, x='aggregate_rating',
                   title='Distribution of Restaurant Ratings',
                   labels={'aggregate_rating': 'Rating'},
                   nbins=10)

# Adding the average rating line
fig.add_shape(type='line', x0=average_rating, x1=average_rating,
              y0=0, y1=df_1['aggregate_rating'].value_counts().max(),
              line=dict(color='red', dash='dash'),
              xref='x', yref='y')

fig.add_annotation(x=average_rating, y=df_1['aggregate_rating'].value_counts().max() / 2,
                   text=f'Average Rating: {average_rating:.2f}',
                   showarrow=True, arrowhead=1, ax=-50, ay=-50, font=dict(color='red'))

# Adjusting the size of the chart
fig.update_layout(
    width=1400,
    height=600
)

# Displaying the histogram plot
fig.show()
```

Insights and Recommendations

Key Insights:

- Significant Count at Rating 0: There's a notable number of restaurants with a rating of 0, approximately 7,000. This could indicate either very poor performance or unrated entries.
- Highest Count Between 3 and 4: The most common ratings are between 3 and 4, with the count peaking around 12,000. This suggests that many restaurants are perceived as average to good.
- Average Rating: The average rating is approximately 3.01, indicating that the overall perception of the restaurants is slightly above average.

Recommendations for Zomato:

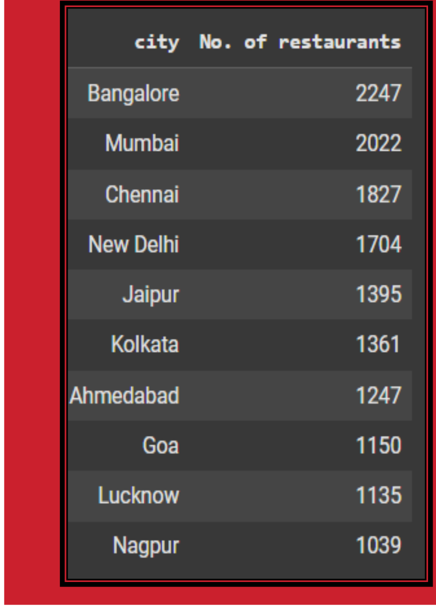
- Address Unrated Entries: Investigate restaurants with a rating of 0 to determine if they are genuinely unrated or have issues that need to be addressed. Ensuring all entries are rated will provide a clearer picture of the quality landscape.
- Highlight Top Performers: Promote restaurants with ratings of 4 or higher as examples of excellence. This can motivate other restaurants to improve their services.

Recommendations for Customers:

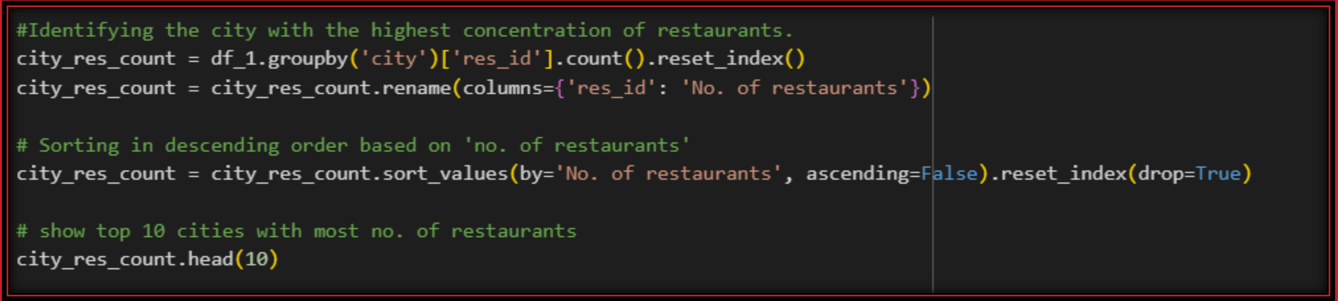
- Focus on Higher-Rated Restaurants: Consider visiting restaurants with ratings of 4 or higher for a better dining experience.
- Provide Honest Feedback: Customers should continue to provide honest and constructive feedback to help restaurants improve their services.

Data Analysis and Visualization

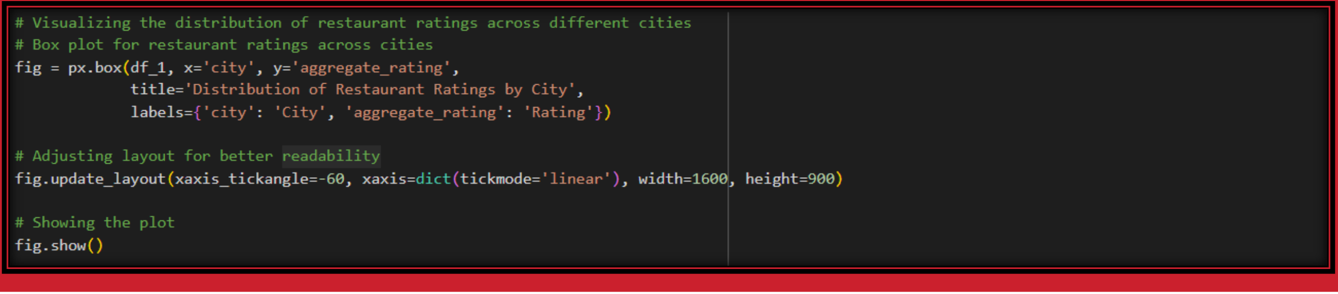
- ❑ City with the Highest Concentration of Restaurants
- ❑ Analysis of Restaurant Rating Distribution Across Cities



city	No. of restaurants
Bangalore	2247
Mumbai	2022
Chennai	1827
New Delhi	1704
Jaipur	1395
Kolkata	1361
Ahmedabad	1247
Goa	1150
Lucknow	1135
Nagpur	1039

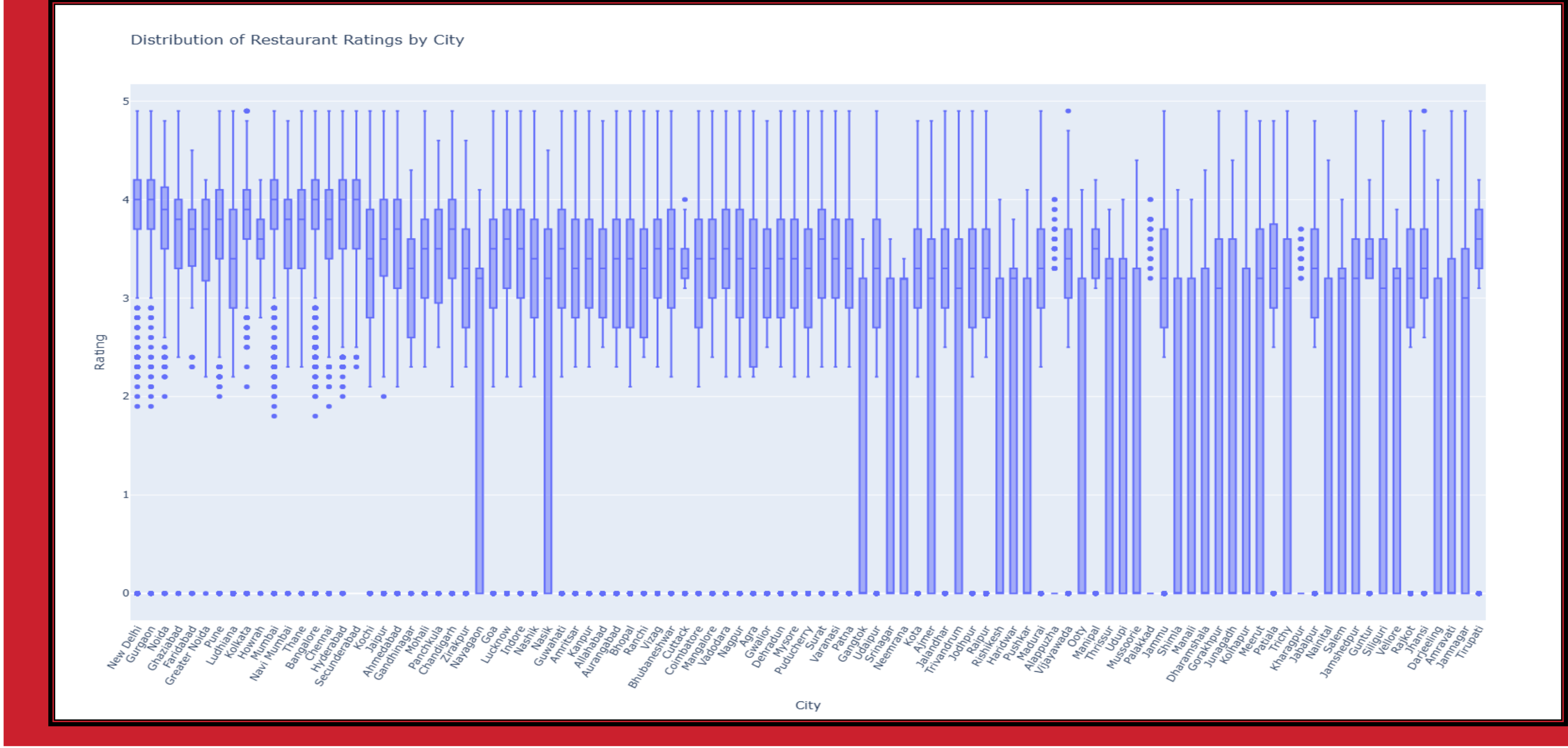


```
#Identifying the city with the highest concentration of restaurants.
city_res_count = df_1.groupby('city')['res_id'].count().reset_index()
city_res_count = city_res_count.rename(columns={'res_id': 'No. of restaurants'})

# Sorting in descending order based on 'no. of restaurants'
city_res_count = city_res_count.sort_values(by='No. of restaurants', ascending=False).reset_index(drop=True)

# show top 10 cities with most no. of restaurants
city_res_count.head(10)
```



```
# Visualizing the distribution of restaurant ratings across different cities
# Box plot for restaurant ratings across cities
fig = px.box(df_1, x='city', y='aggregate_rating',
             title='Distribution of Restaurant Ratings by City',
             labels={'city': 'City', 'aggregate_rating': 'Rating'})

# Adjusting layout for better readability
fig.update_layout(xaxis_tickangle=-60, xaxis=dict(tickmode='linear'), width=1600, height=900)

# Showing the plot
fig.show()
```

Data Analysis and Visualization



Insights and Recommendations

Key Insights:

- **City-wise Restaurant Ratings:** Bangalore, Mumbai, and Chennai show a higher median rating compared to other cities.Cities like Nagpur and Lucknow have a wider spread of ratings, indicating more variability in the quality of restaurants.
- **Consistency in Major Cities:** Major metropolitan areas such as Bangalore, Mumbai, and New Delhi show more consistency in restaurant ratings, with fewer outliers.
- **Variability in Smaller Cities:** Smaller cities like Nagpur and Lucknow have a larger number of outliers, suggesting a diverse range of dining experiences.

Recommendations for Zomato:

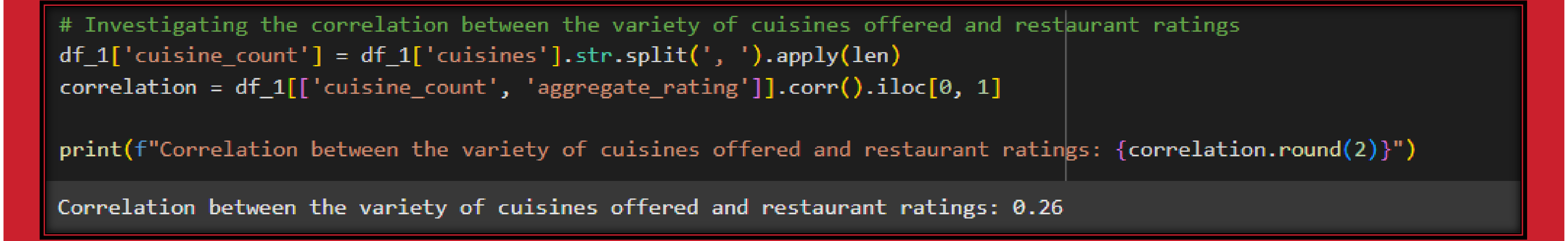
- **Promote High-Rated Restaurants:** Focus on highlighting and promoting restaurants with high ratings in major cities to attract more customers.
- **Support Restaurant Improvement:** Offer support and resources to restaurants in smaller cities to help them improve their services and ratings.

Recommendations for Customers:

- **Explore Consistent High-Rated Options:** Customers in major cities can rely on the consistent high ratings to choose quality dining options.
- **Discover Diverse Experiences:** Customers in smaller cities should explore different restaurants to find unique and diverse dining experiences.

Data Analysis and Visualization

- ❑ Identifying the Most Popular Cuisines
- ❑ Exploring the Relationship Between Cuisine Variety and Restaurant Ratings

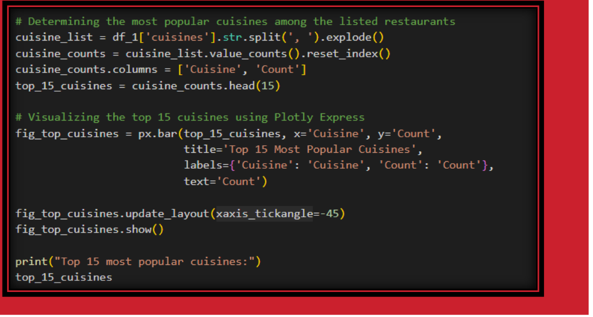


```
# Investigating the correlation between the variety of cuisines offered and restaurant ratings
df_1['cuisine_count'] = df_1['cuisines'].str.split(', ').apply(len)
correlation = df_1[['cuisine_count', 'aggregate_rating']].corr().iloc[0, 1]

print(f"Correlation between the variety of cuisines offered and restaurant ratings: {correlation.round(2)}")
```

```
Correlation between the variety of cuisines offered and restaurant ratings: 0.26
```

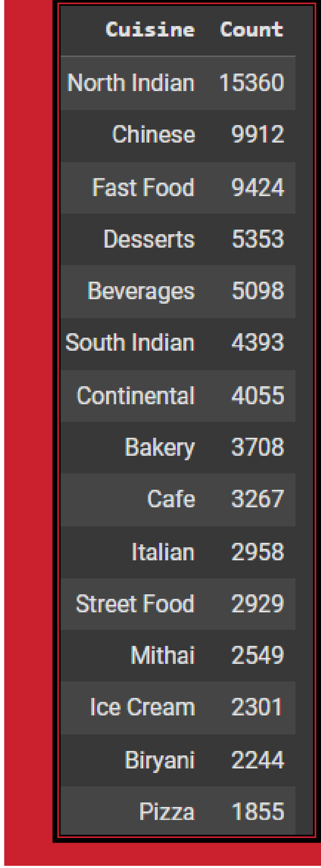
Key Insights about correlation:

- ➢ Positive Correlation: A correlation of 0.26 shows that offering more cuisines is linked to higher restaurant ratings.
- ➢ Diverse Menus Benefit: Restaurants with varied menus tend to have slightly better ratings, suggesting customer preference for diversity.

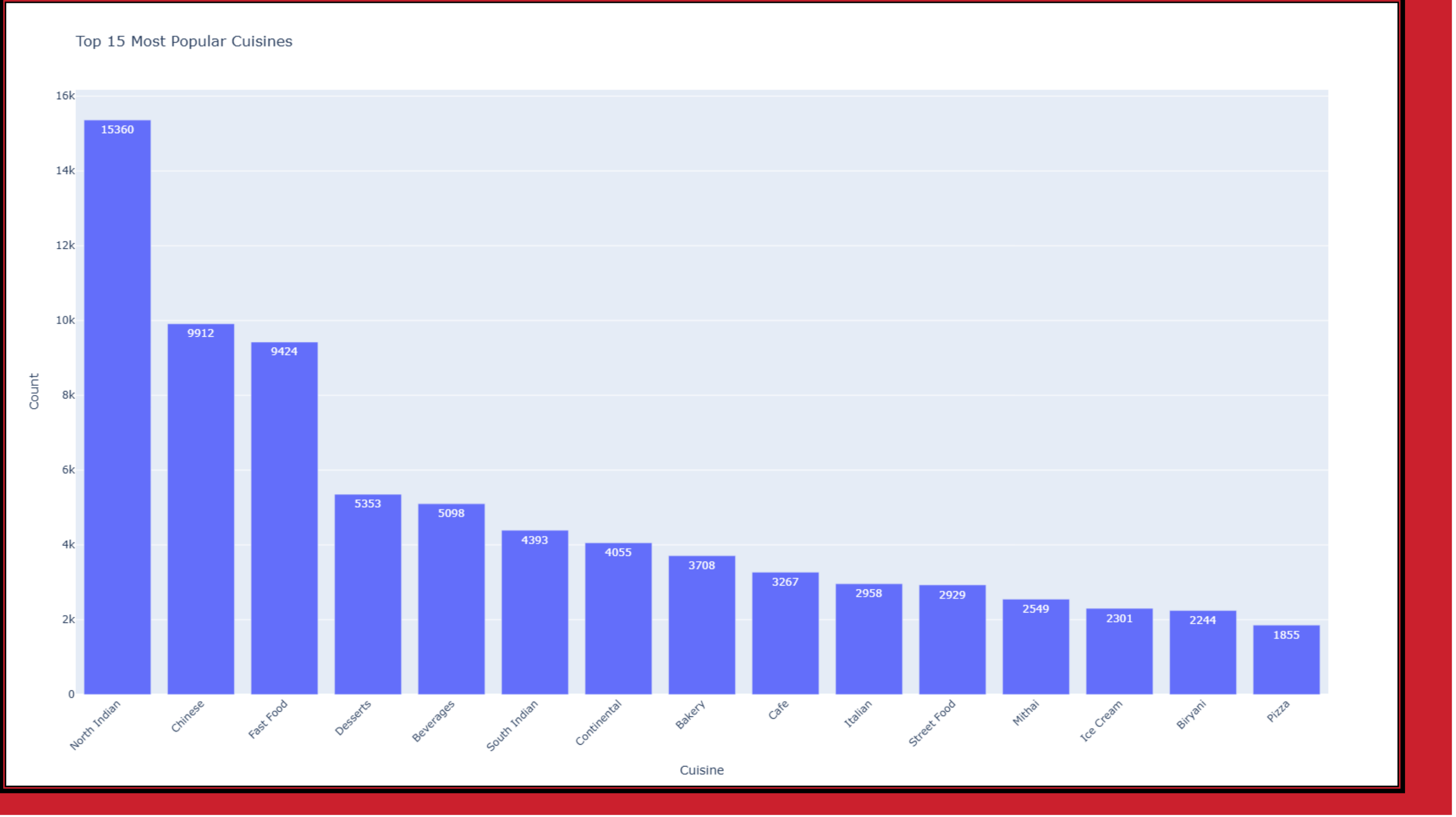


```
# Determining the most popular cuisines among the listed restaurants
cuisine_list = df_1['cuisines'].str.split(', ').explode()
cuisine_counts = cuisine_list.value_counts().reset_index()
cuisine_counts.columns = ['Cuisine', 'Count']
top_15_cuisines = cuisine_counts.head(15)

# Visualizing the top 15 cuisines using Plotly Express
fig_top_cuisines = px.bar(top_15_cuisines, x='Cuisine', y='Count',
                          title='Top 15 Most Popular Cuisines',
                          labels={'Cuisine': 'Cuisine', 'Count': 'Count'},
                          text='Count')

fig_top_cuisines.update_layout(xaxis_tickangle=-45)
fig_top_cuisines.show()

print("Top 15 most popular cuisines:")
top_15_cuisines
```

Data Analysis and Visualization



Cuisine	Count
North Indian	15360
Chinese	9912
Fast Food	9424
Desserts	5353
Beverages	5098
South Indian	4393
Continental	4055
Bakery	3708
Cafe	3267
Italian	2958
Street Food	2929
Mithai	2549
Ice Cream	2301
Biryani	2244
Pizza	1855



Top 15 Most Popular Cuisines

Insights and Recommendations

Key Insights:

- **Popular Cuisines:** North Indian cuisine is the most popular, with 15,360 listings, followed by Chinese (9,912) and Fast Food (9,424). These three cuisines dominate the restaurant scene.
- **Diverse Food Options:** Other popular cuisines include Desserts, Beverages, South Indian, Continental, Bakery, Cafe, and Italian, each with substantial listings, indicating a diverse range of food options available.

Recommendations for Zomato:

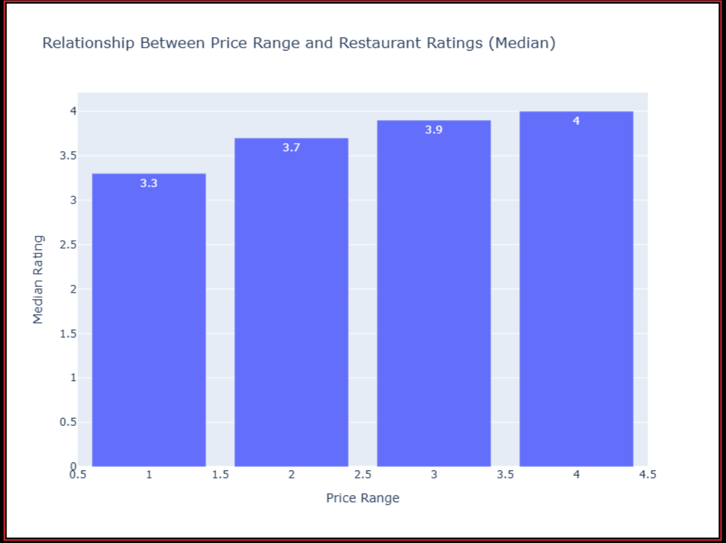
- **Promote Top Cuisines:** Leverage the popularity of North Indian, Chinese, and Fast Food cuisines by promoting these categories to attract more customers.
- **Highlight Diversity:** Emphasize the wide variety of available cuisines to appeal to a broader audience and encourage customers to explore different food options.

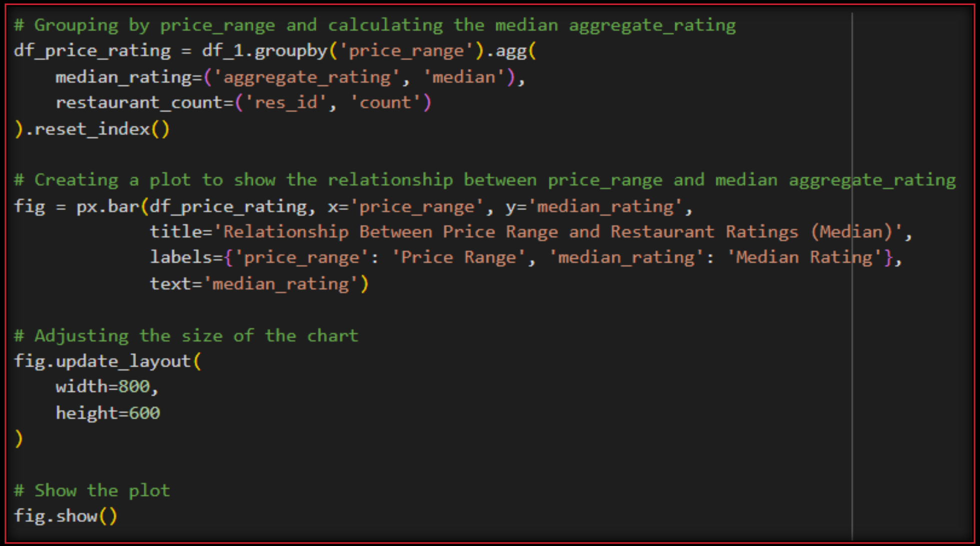
Recommendations for Customers:

- **Explore Popular Choices:** Customers can start with the highly-rated North Indian, Chinese, and Fast Food restaurants for a satisfying dining experience.
- **Try New Cuisines:** Customers can try different cuisines like Desserts, Beverages, and South Indian to discover new favorites and enjoy a diverse culinary journey.

Data Analysis and Visualization

- ❑ Relationship Between Price Range and Restaurant Ratings
- ❑ Visualization of Average Cost for Two Across Price Categories





```python
# Grouping by price_range and calculating the median aggregate_rating
df_price_rating = df_1.groupby('price_range').agg(
    median_rating=('aggregate_rating', 'median'),
    restaurant_count=('res_id', 'count')
).reset_index()

# Creating a plot to show the relationship between price_range and median aggregate_rating
fig = px.bar(df_price_rating, x='price_range', y='median_rating',
             title='Relationship Between Price Range and Restaurant Ratings (Median)',
             labels={'price_range': 'Price Range', 'median_rating': 'Median Rating'},
             text='median_rating')

# Adjusting the size of the chart
fig.update_layout(
    width=800,
    height=600
)

# Show the plot
fig.show()
```

Insights and Recommendations

Key Insights:

- Positive Trend in Ratings: The chart shows that as the price range increases, the median rating of restaurants also tends to increase. Restaurants in the highest price range (Price Range 4) have the highest median rating of 4.0.
- Significant Difference in Ratings: There is a noticeable difference in median ratings across price ranges. Restaurants in the lowest price range (Price Range 1) have a median rating of 3.3, while those in higher price ranges have significantly better ratings.

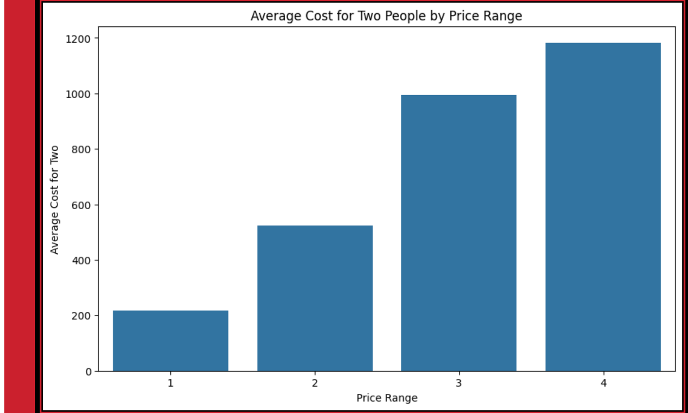
Recommendations for Zomato:

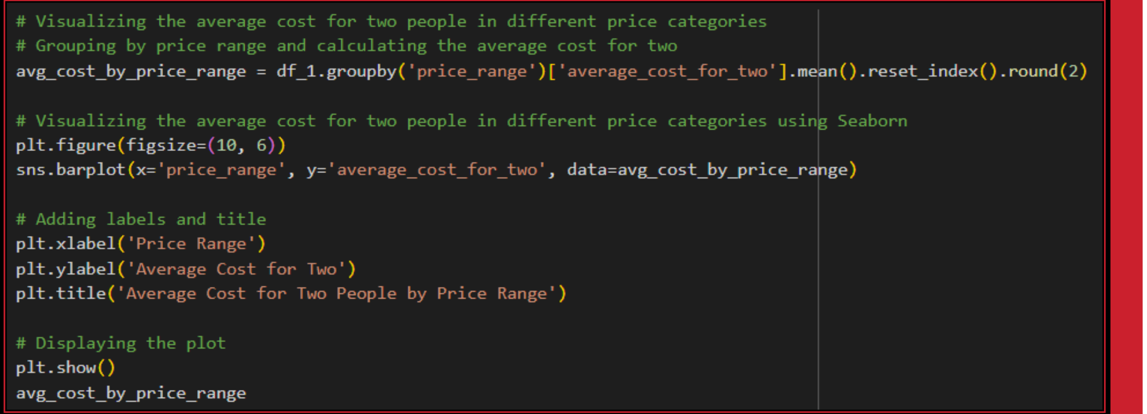
- Promote High-Quality, Higher-Priced Restaurants: Highlight and promote restaurants in the higher price ranges as they tend to have better ratings, attracting customers seeking premium dining experiences.
- Support and Improve Lower-Priced Restaurants: Provide resources and support to help lower-priced restaurants improve their services and ratings, ensuring a balanced quality across all price ranges.

Recommendations for Customers:

- Consider Price-Range Quality: Customers should consider the positive correlation between price range and ratings when choosing restaurants, especially for special occasions.
- Explore Diverse Options: While higher-priced restaurants generally have better ratings, customers should also explore well-rated lower-priced restaurants for good value dining experiences.

Data Analysis and Visualization





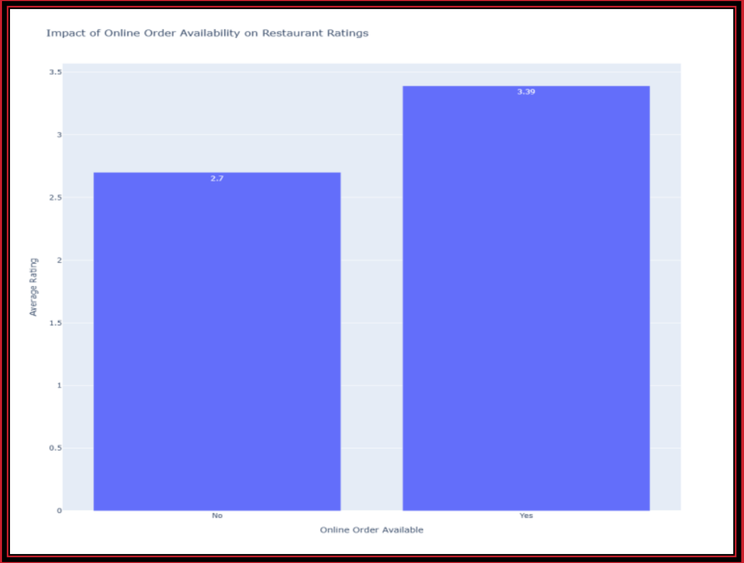
```
# Visualizing the average cost for two people in different price categories
# Grouping by price range and calculating the average cost for two
avg_cost_by_price_range = df_1.groupby('price_range')['average_cost_for_two'].mean().reset_index().round(2)

# Visualizing the average cost for two people in different price categories using Seaborn
plt.figure(figsize=(10, 6))
sns.barplot(x='price_range', y='average_cost_for_two', data=avg_cost_by_price_range)

# Adding labels and title
plt.xlabel('Price Range')
plt.ylabel('Average Cost for Two')
plt.title('Average Cost for Two People by Price Range')

# Displaying the plot
plt.show()
avg_cost_by_price_range
```

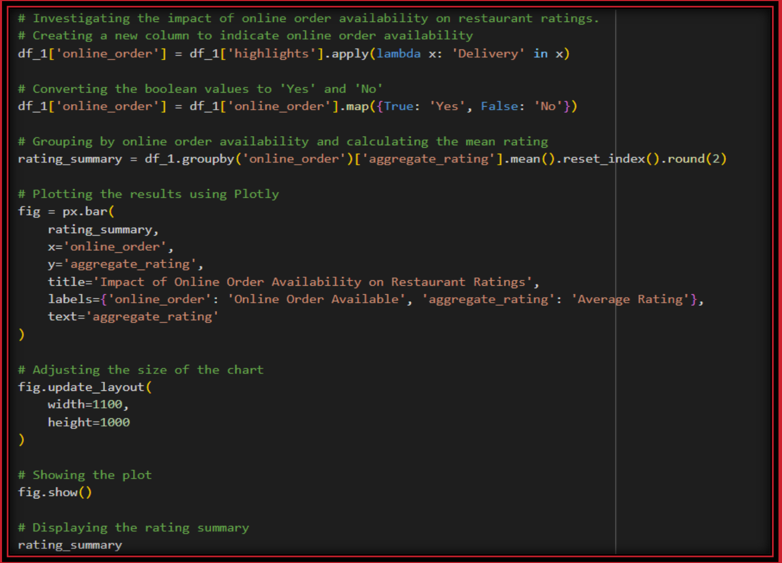
Key Insights:

- **Cost Increase with Price Range:** The average cost for two people increases significantly with the price range. There is a clear upward trend, with the highest cost observed in Price Range 4 and the lowest in Price Range 1.
- **Steep Increase in Higher Price Ranges:** The increase in average cost is most pronounced between Price Range 3 and Price Range 4. This suggests that the jump in cost for higher-end dining options is substantial.
- **Moderate Increase in Middle Ranges:** The increase in average cost between Price Range 1 and 2 and between Price Range 2 and 3 is more gradual compared to the jump between Price Range 3 and 4. This indicates a moderate price difference between these middle-range options.

Data Analysis and Visualization

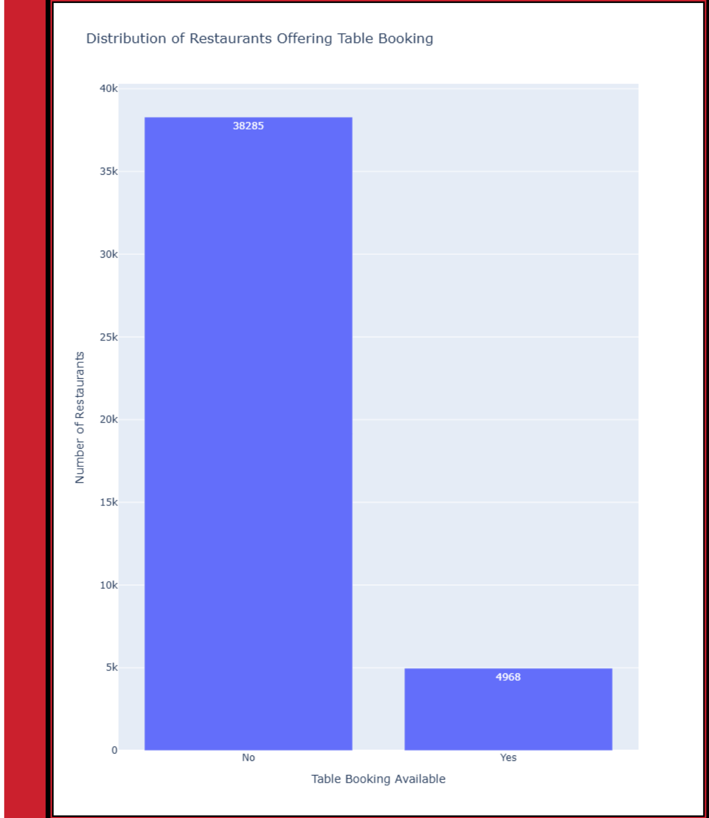
- ❑ Impact of Online Order Availability on Restaurant Ratings
- ❑ Distribution Analysis of Restaurants Offering Table Booking

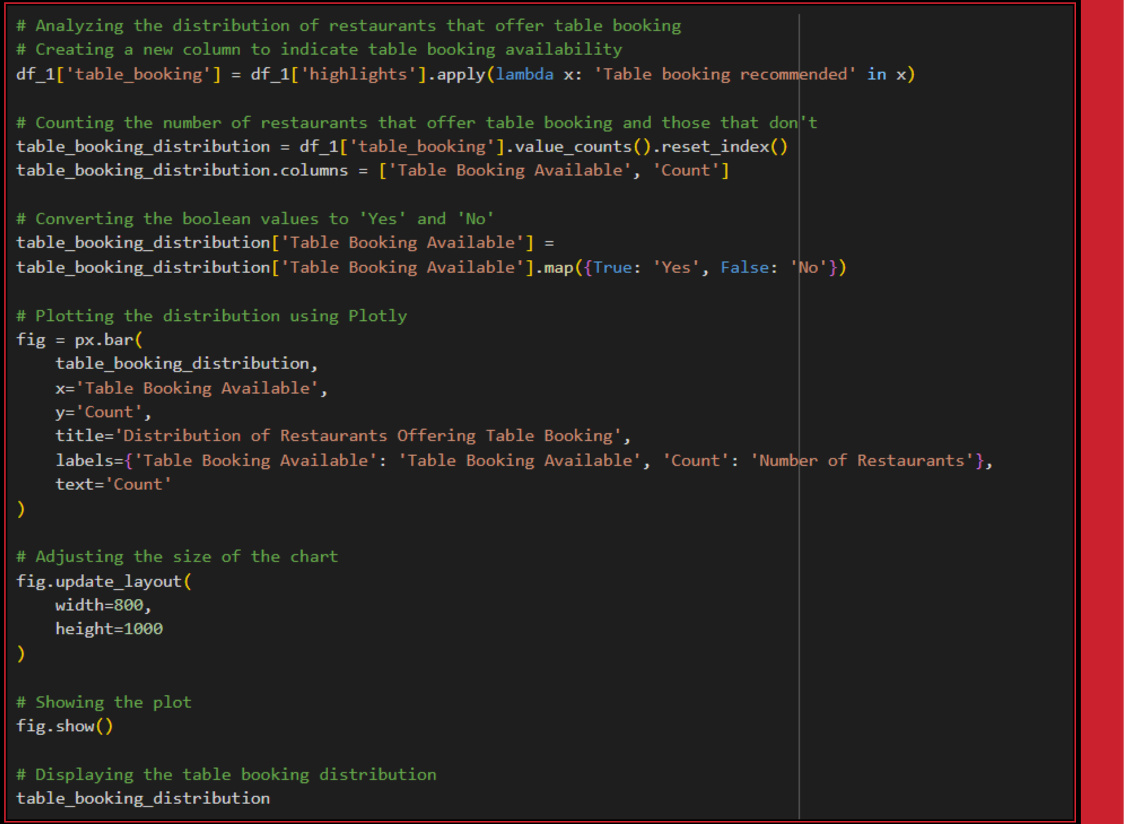




```python
# Investigating the impact of online order availability on restaurant ratings.
# Creating a new column to indicate online order availability
df_1['online_order'] = df_1['highlights'].apply(lambda x: 'Delivery' in x)

# Converting the boolean values to 'Yes' and 'No'
df_1['online_order'] = df_1['online_order'].map({True: 'Yes', False: 'No'})

# Grouping by online order availability and calculating the mean rating
rating_summary = df_1.groupby('online_order')['aggregate_rating'].mean().reset_index().round(2)

# Plotting the results using Plotly
fig = px.bar(
    rating_summary,
    x='online_order',
    y='aggregate_rating',
    title='Impact of Online Order Availability on Restaurant Ratings',
    labels={'online_order': 'Online Order Available', 'aggregate_rating': 'Average Rating'},
    text='aggregate_rating'
)

# Adjusting the size of the chart
fig.update_layout(
    width=1100,
    height=1000
)

# Showing the plot
fig.show()

# Displaying the rating summary
rating_summary
```

Data Analysis and Visualization





```
# Analyzing the distribution of restaurants that offer table booking
# Creating a new column to indicate table booking availability
df_1['table_booking'] = df_1['highlights'].apply(lambda x: 'Table booking recommended' in x)

# Counting the number of restaurants that offer table booking and those that don't
table_booking_distribution = df_1['table_booking'].value_counts().reset_index()
table_booking_distribution.columns = ['Table Booking Available', 'Count']

# Converting the boolean values to 'Yes' and 'No'
table_booking_distribution['Table Booking Available'] =
table_booking_distribution['Table Booking Available'].map({True: 'Yes', False: 'No'})

# Plotting the distribution using Plotly
fig = px.bar(
    table_booking_distribution,
    x='Table Booking Available',
    y='Count',
    title='Distribution of Restaurants Offering Table Booking',
    labels={'Table Booking Available': 'Table Booking Available', 'Count': 'Number of Restaurants'},
    text='Count'
)

# Adjusting the size of the chart
fig.update_layout(
    width=800,
    height=1000
)

# Showing the plot
fig.show()

# Displaying the table booking distribution
table_booking_distribution
```

Insights and Recommendations

Key Insights:

- **Higher Ratings with Delivery Feature:** Restaurants offering the delivery feature have a higher average rating (3.39) compared to those without it (2.7). This indicates that customers tend to rate restaurants with delivery options more favorably.
- **Higher Ratings with Dine-In Feature:** Restaurants offering the dine-in feature also have a higher average rating (3.5) compared to those without it (2.9). This suggests that customers appreciate the option to dine in, leading to better ratings.

Recommendations for Zomato:

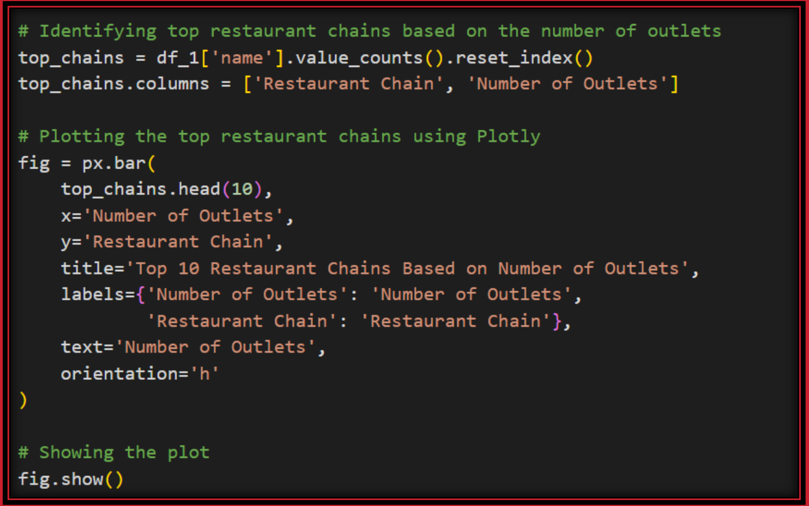
- **Encourage Feature Adoption:** Promote the adoption of both delivery and dine-in features among restaurants by highlighting their positive impact on customer ratings. This can help enhance the overall customer experience and satisfaction.
- **Targeted Promotions for Features:** Create targeted promotions for restaurants offering delivery and dine-in features to attract more customers and boost their visibility on the platform.

Recommendations for Customers:

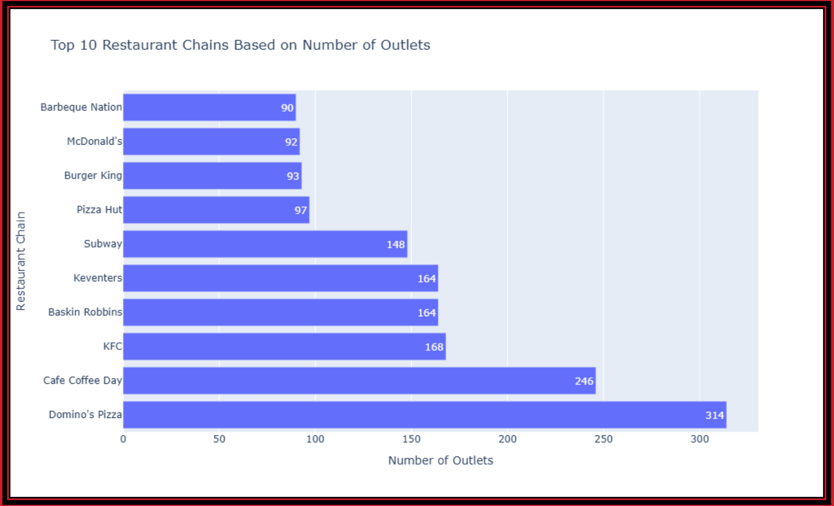
- **Check for Delivery and Dine-In Features:** Customers should look for both delivery and dine-in features when selecting a restaurant to ensure a better dining experience, whether they prefer the convenience of food delivery or the ambiance of dining out.

Data Analysis and Visualization

- ❑ Top Restaurant Chains by Number of Outlets: Identification and Visualization
- ❑ Rating Analysis of Leading Restaurant Chains



```python
# Identifying top restaurant chains based on the number of outlets
top_chains = df_1['name'].value_counts().reset_index()
top_chains.columns = ['Restaurant Chain', 'Number of Outlets']

# Plotting the top restaurant chains using Plotly
fig = px.bar(
    top_chains.head(10),
    x='Number of Outlets',
    y='Restaurant Chain',
    title='Top 10 Restaurant Chains Based on Number of Outlets',
    labels={'Number of Outlets': 'Number of Outlets',
            'Restaurant Chain': 'Restaurant Chain'},
    text='Number of Outlets',
    orientation='h'
)

# Showing the plot
fig.show()
```

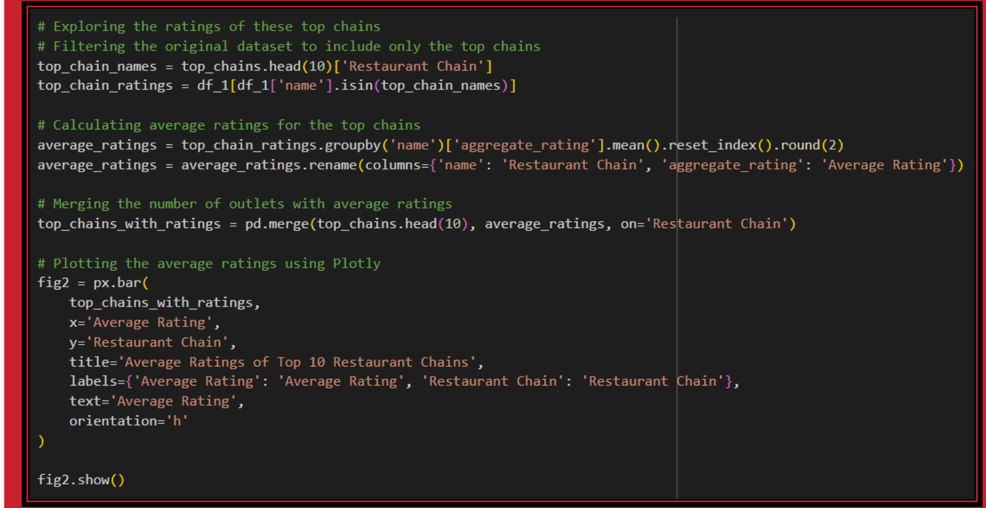


Insights and Recommendations

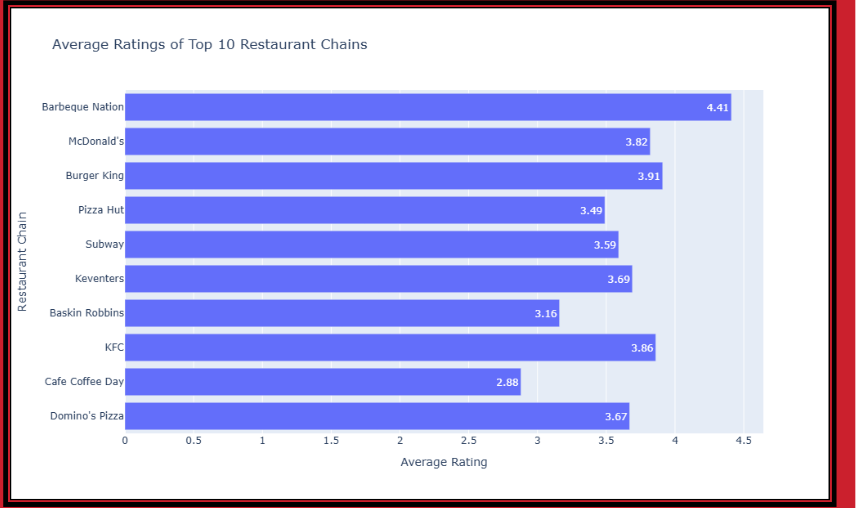
Key Insights:

- **Domino's Pizza Dominance:** Domino's Pizza leads with 314 outlets, showcasing its strong market presence and widespread popularity.
- **Cafe Coffee Day and KFC:** Cafe Coffee Day (246 outlets) and KFC (168 outlets) also have a significant number of outlets, indicating their high customer appeal and extensive reach.
- **Substantial Presence:** Other major chains like Subway (148 outlets) and Pizza Hut (97 outlets) maintain a strong presence, contributing to the competitive landscape.
- **Balanced Competition:** There is a balanced distribution of outlets among the top 10 chains, suggesting a healthy competition and variety for customers.
- **Market Dynamics:** The presence of diverse chains highlights the dynamic and competitive nature of the restaurant industry, offering customers a wide range of dining options.

Data Analysis and Visualization



```
# Exploring the ratings of these top chains
# Filtering the original dataset to include only the top chains
top_chain_names = top_chains.head(10)['Restaurant Chain']
top_chain_ratings = df_1[df_1['name'].isin(top_chain_names)]

# Calculating average ratings for the top chains
average_ratings = top_chain_ratings.groupby('name')['aggregate_rating'].mean().reset_index().round(2)
average_ratings = average_ratings.rename(columns={'name': 'Restaurant Chain', 'aggregate_rating': 'Average Rating'})

# Merging the number of outlets with average ratings
top_chains_with_ratings = pd.merge(top_chains.head(10), average_ratings, on='Restaurant Chain')

# Plotting the average ratings using Plotly
fig2 = px.bar(
    top_chains_with_ratings,
    x='Average Rating',
    y='Restaurant Chain',
    title='Average Ratings of Top 10 Restaurant Chains',
    labels={'Average Rating': 'Average Rating', 'Restaurant Chain': 'Restaurant Chain'},
    text='Average Rating',
    orientation='h'
)

fig2.show()
```



Key Insights:

- **Top Rated Chain:** Barbeque Nation has the highest average rating among the top 10 chains, with an impressive rating of 4.41.
- **Popular Fast Food Chains:** Chains like McDonald's (3.82) and KFC (3.86) have strong ratings, indicating their widespread customer satisfaction.
- **Lowest Rating:** Cafe Coffee Day has the lowest average rating among the top 10 chains, with a rating of 2.88.

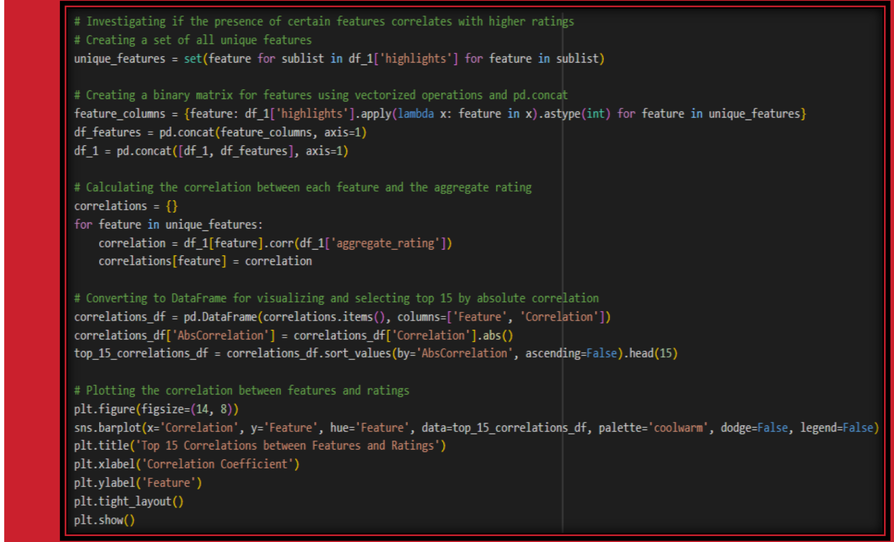
Insights and Recommendations

Key Insights:

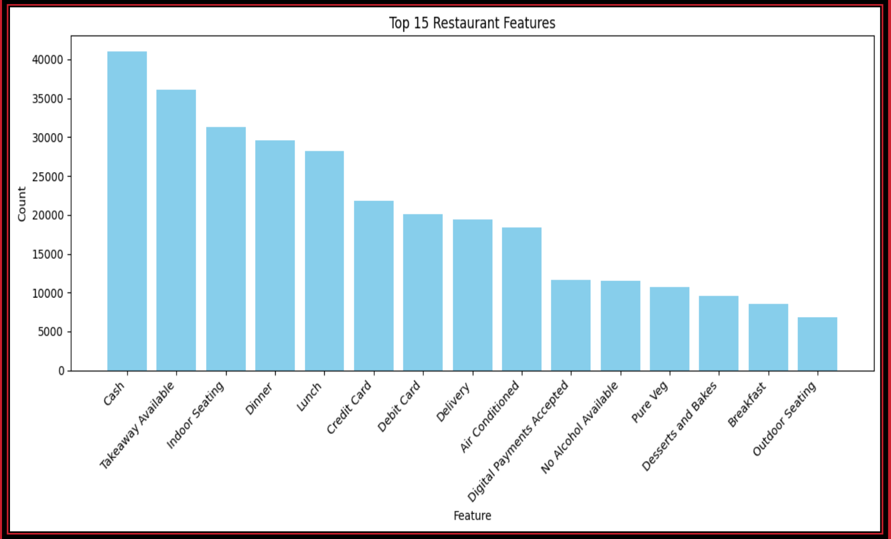
- **Domino's Pizza Dominance:** Domino's Pizza leads with 314 outlets, showcasing its strong market presence and widespread popularity.
- **Cafe Coffee Day and KFC:** Cafe Coffee Day (246 outlets) and KFC (168 outlets) also have a significant number of outlets, indicating their high customer appeal and extensive reach.
- **Substantial Presence:** Other major chains like Subway (148 outlets) and Pizza Hut (97 outlets) maintain a strong presence, contributing to the competitive landscape.
- **Balanced Competition:** There is a balanced distribution of outlets among the top 10 chains, suggesting a healthy competition and variety for customers.
- **Market Dynamics:** The presence of diverse chains highlights the dynamic and competitive nature of the restaurant industry, offering customers a wide range of dining options.

Data Analysis and Visualization

- ❑ Distribution of Restaurants Based on Features like Wi-Fi and Alcohol Availability
- ❑ Correlation Analysis Between Restaurant Features and Ratings



```
# Investigating if the presence of certain features correlates with higher ratings
# Creating a set of all unique features
unique_features = set(feature for sublist in df_1['highlights'] for feature in sublist)

# Creating a binary matrix for features using vectorized operations and pd.concat
feature_columns = {feature: df_1['highlights'].apply(lambda x: feature in x).astype(int) for feature in unique_features}
df_features = pd.concat(feature_columns, axis=1)
df_1 = pd.concat([df_1, df_features], axis=1)

# Calculating the correlation between each feature and the aggregate rating
correlations = {}
for feature in unique_features:
    correlation = df_1[feature].corr(df_1['aggregate_rating'])
    correlations[feature] = correlation

# Converting to DataFrame for visualizing and selecting top 15 by absolute correlation
correlations_df = pd.DataFrame(correlations.items(), columns=['Feature', 'Correlation'])
correlations_df['AbsCorrelation'] = correlations_df['Correlation'].abs()
top_15_correlations_df = correlations_df.sort_values(by='AbsCorrelation', ascending=False).head(15)

# Plotting the correlation between features and ratings
plt.figure(figsize=(14, 8))
sns.barplot(x='Correlation', y='Feature', hue='Feature', data=top_15_correlations_df, palette='coolwarm', dodge=False, legend=False)
plt.title('Top 15 Correlations between Features and Ratings')
plt.xlabel('Correlation Coefficient')
plt.ylabel('Feature')
plt.tight_layout()
plt.show()
```



Insights and Recommendations

Key Insights:

- **Cash Still Reigns:** Cash payments are the most common, highlighting its continued significance in the restaurant industry.
- **Takeaway and Indoor Seating are Popular:** The high frequency of "Takeaway Available" and "Indoor Seating" indicates a strong demand for convenient and comfortable dining options.
- **Digital Payments are on the Rise:** The presence of "Digital Payments Accepted" suggests a growing trend towards cashless transactions.

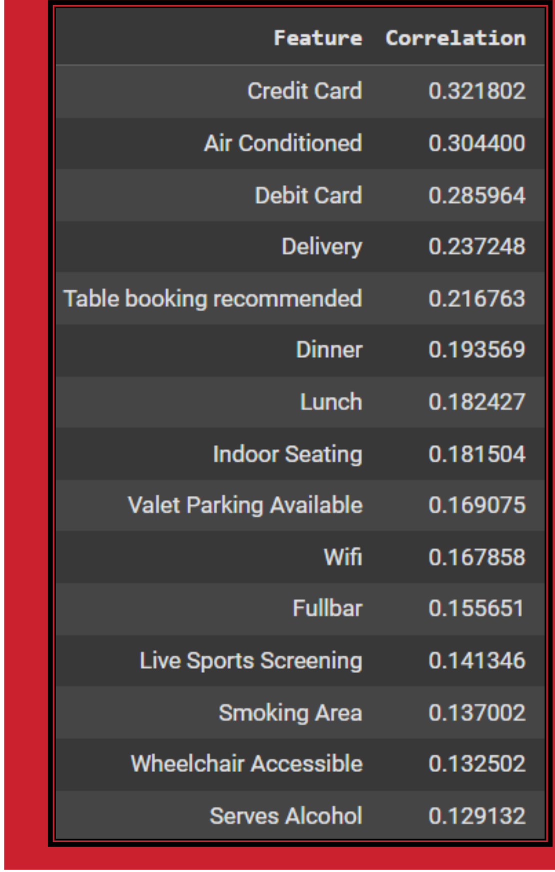
Recommendations for Zomato:

- **Promote Digital Payments:** Incentivize digital payments to encourage their adoption.
- **Highlight Takeaway and Indoor Seating:** Emphasize restaurants with these features to cater to customer preferences.
- **Leverage Data:** Use the data to understand customer behavior and improve platform features.

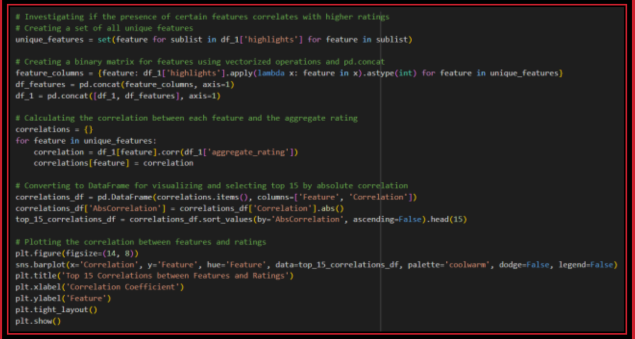
Recommendations for Customers:

- **Explore Digital Payment Options:** Take advantage of discounts and rewards associated with digital payments.
- **Consider Takeaway for Convenience:** Utilize takeaway options for a more convenient dining experience.
- **Prioritize Comfort:** Choose restaurants with indoor seating or other amenities that enhance comfort.

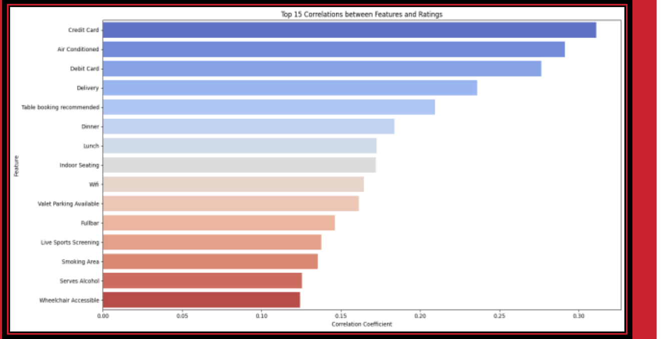
Data Analysis and Visualization



Feature	Correlation
Credit Card	0.321802
Air Conditioned	0.304400
Debit Card	0.285964
Delivery	0.237248
Table booking recommended	0.216763
Dinner	0.193569
Lunch	0.182427
Indoor Seating	0.181504
Valet Parking Available	0.169075
Wifi	0.167858
Fullbar	0.155651
Live Sports Screening	0.141346
Smoking Area	0.137002
Wheelchair Accessible	0.132502
Serves Alcohol	0.129132



```python
# Investigating if the presence of certain features correlates with higher ratings
# Creating a set of all unique features
unique_features = set(feature for sublist in df_1['highlights'] for feature in sublist)

# Creating a binary matrix for features using vectorized operations and pd.concat
feature_columns = {feature: df_1['highlights'].apply(lambda x: feature in x).astype(int) for feature in unique_features}
df_features = pd.concat(feature_columns, axis=1)
df_1 = pd.concat([df_1, df_features], axis=1)

# Calculating the correlation between each feature and the aggregate rating
correlations = {}
for feature in unique_features:
    correlation = df_1[feature].corr(df_1['aggregate_rating'])
    correlations[feature] = correlation

# Converting to DataFrame for visualizing and selecting top 15 by absolute correlation
correlations_df = pd.DataFrame(correlations.items(), columns=['Feature', 'Correlation'])
correlations_df['AbsCorrelation'] = correlations_df['Correlation'].abs()
top_15_correlations_df = correlations_df.sort_values(by='AbsCorrelation', ascending=False).head(15)

# Plotting the correlation between features and ratings
plt.figure(figsize=(14, 8))
sns.barplot(x='Correlation', y='Feature', hue='Feature', data=top_15_correlations_df, palette='coolwarm', dodge=False, legend=False)
plt.title('Top 15 Correlations between Features and Ratings')
plt.xlabel('Correlation Coefficient')
plt.ylabel('Feature')
plt.tight_layout()
plt.show()
```

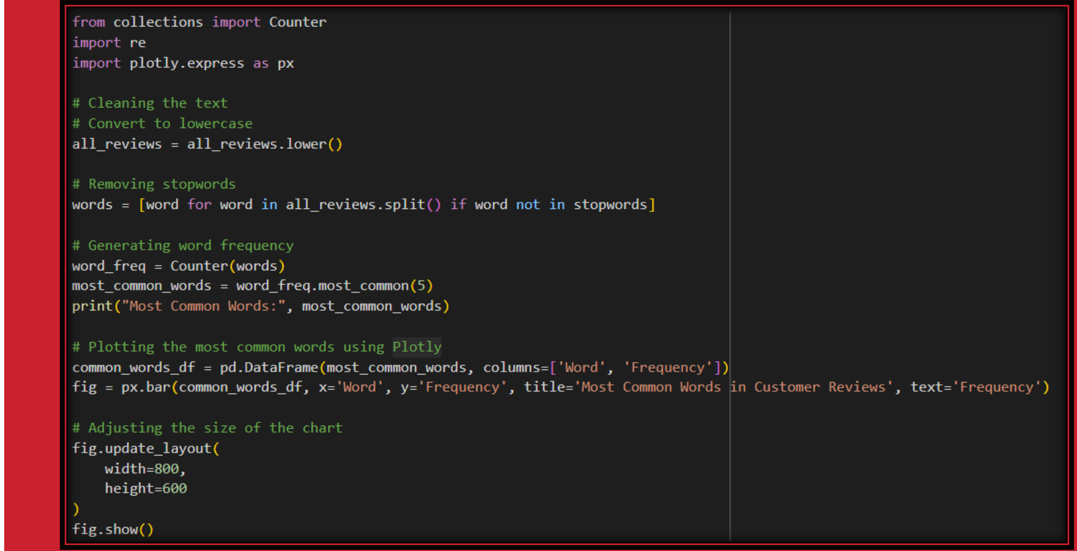


Key Insights:

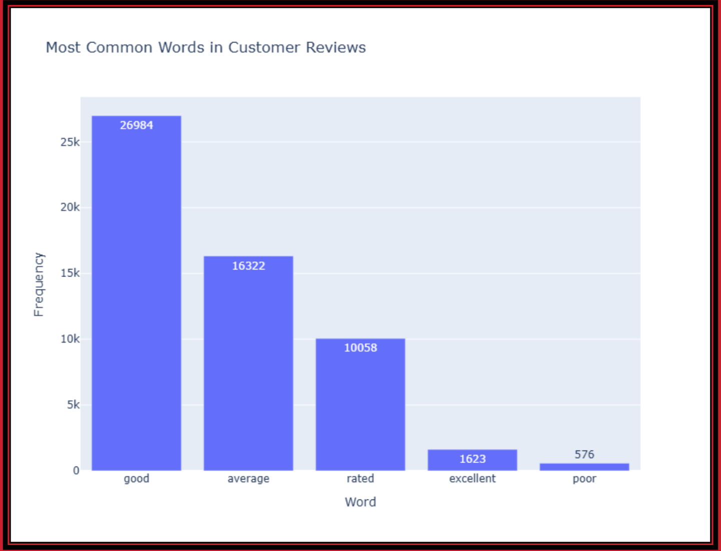
- **Convenience is Key:** Features like credit card/debit card acceptance, delivery, and air conditioning strongly correlate with higher ratings, suggesting that customers value convenience and comfort.
- **Dining Atmosphere Matters:** Indoor seating and the absence of smoking areas are positively correlated with higher ratings, indicating that customers appreciate a pleasant and comfortable dining environment.
- **Live Sports Can Be a Double-Edged Sword:** While not a strong negative correlation, live sports screening might slightly lower ratings. This suggests that while some customers enjoy watching sports, it can potentially disrupt the dining experience for others.

Data Analysis and Visualization

- ❑ Word Cloud Visualization of Customer Reviews: Identifying Positive and Negative Sentiments
- ❑ Analysis of Frequently Mentioned Words and Customer Sentiments



```python
from collections import Counter
import re
import plotly.express as px

# Cleaning the text
# Convert to lowercase
all_reviews = all_reviews.lower()

# Removing stopwords
words = [word for word in all_reviews.split() if word not in stopwords]

# Generating word frequency
word_freq = Counter(words)
most_common_words = word_freq.most_common(5)
print("Most Common Words:", most_common_words)

# Plotting the most common words using Plotly
common_words_df = pd.DataFrame(most_common_words, columns=['Word', 'Frequency'])
fig = px.bar(common_words_df, x='Word', y='Frequency', title='Most Common Words in Customer Reviews', text='Frequency')

# Adjusting the size of the chart
fig.update_layout(
    width=800,
    height=600
)
fig.show()
```



Insights and Recommendations

Key Insights:

- **Positive Sentiment Dominates:** The word "good" appearing most frequently indicates a generally positive sentiment among customers.
- **Neutral Reviews are Significant:** The high frequency of "average" suggests a considerable number of customers are providing neutral or mixed reviews.
- **Lack of Extreme Opinions:** The relatively low frequency of "excellent" and "poor" might suggest that customers are providing more general feedback rather than focusing on specific aspects of the product or service.

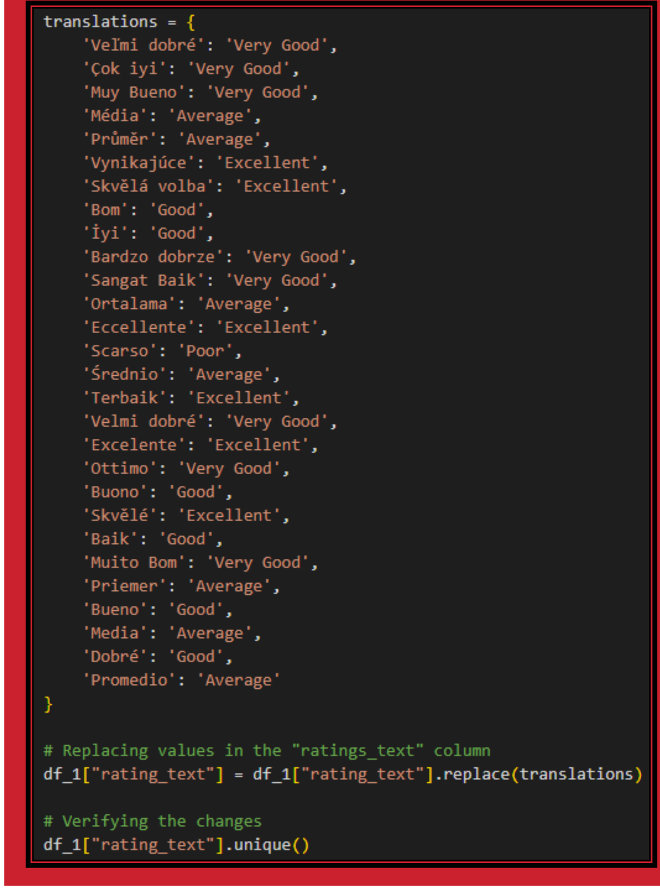
Recommendations for Zomato:

- **Encourage Detailed Feedback:** Implement strategies to encourage customers to provide more specific and detailed feedback.
- **Address Neutral Reviews Proactively:** Analyze neutral reviews to identify common themes and areas for improvement.
- **Promote Positive Feedback:** Actively promote positive reviews and success stories to attract more customers.

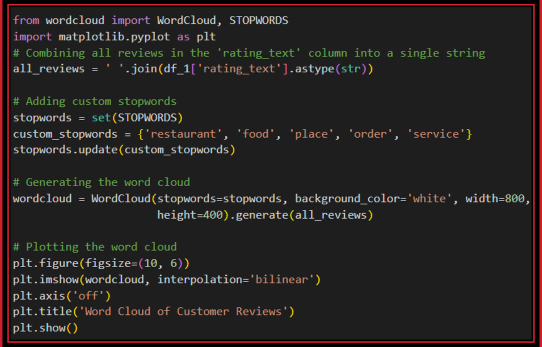
Recommendations for Customers:

- **Provide Specific Feedback:** Contribute more effectively by providing specific and detailed feedback in their reviews.
- **Go Beyond "Good" or "Average":** Provide more nuanced feedback by elaborating on their experience.
- **Utilize Available Features:** Take advantage of features like photo uploads and the option to provide specific feedback on different aspects of the dining experience.

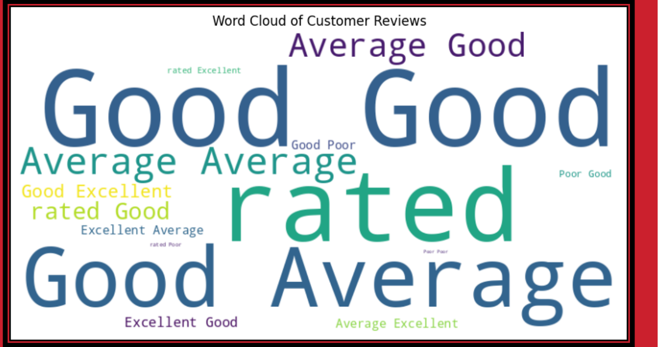
Data Analysis and Visualization



```python
translations = {
    'Veľmi dobré': 'Very Good',
    'Çok iyi': 'Very Good',
    'Muy Bueno': 'Very Good',
    'Média': 'Average',
    'Průměr': 'Average',
    'Vynikajúce': 'Excellent',
    'Skvělá volba': 'Excellent',
    'Bom': 'Good',
    'İyi': 'Good',
    'Bardzo dobrze': 'Very Good',
    'Sangat Baik': 'Very Good',
    'Ortalama': 'Average',
    'Eccellente': 'Excellent',
    'Scarso': 'Poor',
    'Średnio': 'Average',
    'Terbaik': 'Excellent',
    'Velmi dobré': 'Very Good',
    'Excelente': 'Excellent',
    'Ottimo': 'Very Good',
    'Buono': 'Good',
    'Skvělé': 'Excellent',
    'Baik': 'Good',
    'Muito Bom': 'Very Good',
    'Priemer': 'Average',
    'Bueno': 'Good',
    'Media': 'Average',
    'Dobré': 'Good',
    'Promedio': 'Average'
}

# Replacing values in the "ratings_text" column
df_1["rating_text"] = df_1["rating_text"].replace(translations)

# Verifying the changes
df_1["rating_text"].unique()
```



```python
from wordcloud import WordCloud, STOPWORDS
import matplotlib.pyplot as plt
# Combining all reviews in the 'rating_text' column into a single string
all_reviews = ' '.join(df_1['rating_text'].astype(str))

# Adding custom stopwords
stopwords = set(STOPWORDS)
custom_stopwords = {'restaurant', 'food', 'place', 'order', 'service'}
stopwords.update(custom_stopwords)

# Generating the word cloud
wordcloud = WordCloud(stopwords=stopwords, background_color='white', width=800,
                      height=400).generate(all_reviews)

# Plotting the word cloud
plt.figure(figsize=(10, 6))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')
plt.title('Word Cloud of Customer Reviews')
plt.show()
```

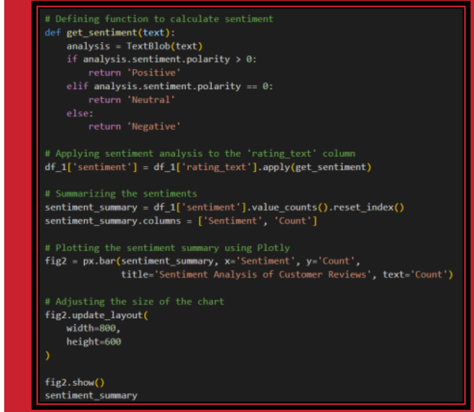


Word Cloud of Customer Reviews

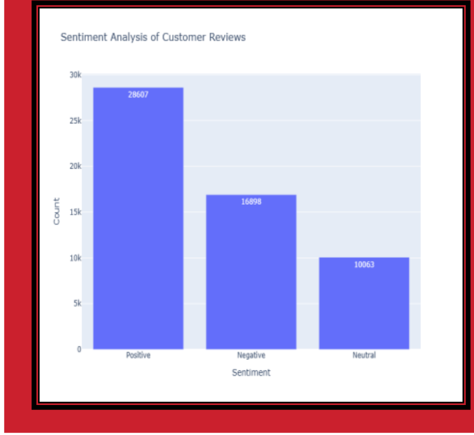
Key Insights:

- **Frequent Positive Reviews:** Words like "Good," "Excellent," and "Rated" are large, indicating many reviews are positive.
- **Neutral Sentiments:** The word "Average" is prominent, showing many reviews describe middling experiences.
- **Varied Opinions:** Words like "Poor" and "Excellent" indicate a range of customer experiences.
- **Common Themes:** Words such as "Service," "Quality," and "Experience" reflect key aspects customers comment on.
- **Sentiment Snapshot:** The size and frequency of words quickly show the general sentiment distribution, mostly positive and average.

Data Analysis and Visualization



```
# Defining function to calculate sentiment
def get_sentiment(text):
    analysis = TextBlob(text)
    if analysis.sentiment.polarity > 0:
        return 'Positive'
    elif analysis.sentiment.polarity == 0:
        return 'Neutral'
    else:
        return 'Negative'

# Applying sentiment analysis to the 'rating_text' column
df_1['sentiment'] = df_1['rating_text'].apply(get_sentiment)

# Summarizing the sentiments
sentiment_summary = df_1['sentiment'].value_counts().reset_index()
sentiment_summary.columns = ['Sentiment', 'Count']

# Plotting the sentiment summary using Plotly
fig2 = px.bar(sentiment_summary, x='Sentiment', y='Count',
              title='Sentiment Analysis of Customer Reviews', text='Count')

# Adjusting the size of the chart
fig2.update_layout(
    width=800,
    height=600
)

fig2.show()
sentiment_summary
```



Key Insights:

- **Positive Sentiment Dominates:** The "Positive" bar is significantly taller than the others, indicating a large portion of the customer reviews express positive sentiments. This is a good sign for the business as it suggests overall customer satisfaction.
- **Negative Sentiment is Significant:** The "Negative" bar is substantial, indicating that a considerable number of customers have expressed negative sentiments in their reviews. This highlights areas for improvement to address customer concerns and enhance the overall experience.
- **Neutral Sentiment is Present:** The "Neutral" bar is also present, indicating that a portion of the customer base has expressed neutral or mixed feelings about the product or service. This presents an opportunity to convert these neutral reviews into positive ones.
- **Data-Driven Insights:** The specific numbers associated with each sentiment category (28607, 16898, and 10063) provide valuable data points for further analysis and decision-making. This data can be used to track trends, compare performance over time, and measure the impact of improvement initiatives.
- **Actionable Insights:** The analysis provides actionable insights. By identifying the reasons behind negative and neutral sentiments, businesses can implement targeted strategies to address customer concerns, improve product or service offerings, and ultimately increase overall customer satisfaction.

Key Insights summary

Restaurant Landscape:

- **Average Restaurant Rating:** The average rating for restaurants is 3.72, indicating generally positive customer experiences.
- **Restaurant Concentration**: New Delhi has the highest concentration of restaurants, suggesting a competitive market in that city.
- **Popular Cuisines:** North Indian, Chinese, and Fast Food are the most popular cuisines among customers.
- **Cuisine Variety:** There's a weak positive correlation between cuisine variety and ratings, implying a slight preference for restaurants with diverse menus.
- **Price and Rating:** Median ratings tend to increase with price range, suggesting higher-priced restaurants generally offer better experiences.

Key Insights summary

Customer Behaviour:

- **Online Ordering:** Restaurants offering online ordering have slightly higher average ratings, indicating customer convenience is valued.
- **Table Booking:** Table booking availability appears to have a negligible impact on ratings.
- **Popular Chains:** Cafe Coffee Day is the most prevalent restaurant chain, suggesting its wide reach and popularity.
- **Positive Sentiment:** Customer reviews frequently use words like "good," "great," and "best," highlighting a positive overall sentiment.
- **Sentiment Analysis:** Sentiment analysis confirms a predominantly positive sentiment among customer reviews.

Key Insights summary

Restaurant Features:

- **Common Features:** "Delivery," "Dinner," and "Lunch" are among the most common restaurant features, indicating a focus on convenience and basic services.
- **Rating-Boosting Features:** Features like "Table booking recommended" and "Air Conditioned" show a weak positive correlation with ratings, suggesting these amenities contribute to a better dining experience.
- **Food Quality:** Features like "Great Food" and "Excellent Food" strongly correlate with positive ratings, emphasizing the importance of food quality.

Recommendations for Zomato

- **Promote Online Ordering:** Encourage more restaurants to offer online ordering and provide incentives for those that do. Highlight this feature prominently on the platform to attract customers seeking convenience.
- **Focus on Food Quality:** Emphasize the importance of food quality to restaurant partners. Consider implementing programs to recognize and reward restaurants known for consistently excellent food.
- **Enhance Convenience:** Promote features like delivery and lunch options, making it easier for customers to find restaurants that cater to their needs.
- **Highlight Amenities:** Encourage restaurants to highlight amenities like air conditioning and table booking, as they contribute to a better dining experience.
- **Improve Restaurant Discoverability:** Enhance search and filtering options to help customers easily find restaurants based on cuisine, price range, location, and features.
- **Leverage Customer Reviews:** Analyze customer reviews to identify areas for improvement and provide targeted feedback to restaurant partners.
- **Personalization:** Offer personalized recommendations to customers based on their past orders, preferences, and location.

Recommendations for Customers

- **Utilize Online Ordering**: Take advantage of online ordering options for a convenient and hassle-free dining experience.
- **Prioritize Food Quality:** Consider restaurant ratings and reviews to find restaurants known for excellent food. Pay attention to features like "Great Food" and "Excellent Food."
- **Explore Amenities:** Look for restaurants that offer amenities like air conditioning and table booking to enhance your dining experience.
- **Leverage Search Filters:** Utilize search filters on Zomato to refine your restaurant search based on cuisine, price range, location, and features.
- **Share Your Feedback:** Provide honest and detailed reviews to help other customers make informed decisions and motivate restaurants to improve their offerings.
- **Try New Restaurants:** Explore a variety of restaurants beyond popular chains to discover hidden gems and expand your culinary horizons.

Conclusion

This comprehensive exploratory data analysis of the Zomato restaurant dataset has unveiled a wealth of valuable insights into the diverse and dynamic Indian restaurant landscape. By meticulously examining various dimensions of the data, including restaurant ratings, customer cuisine preferences, city-specific trends, relationships between pricing and ratings, and the impact of key features, this study provides a nuanced understanding of customer satisfaction and market trends. The analysis also delves into online ordering behaviors, sentiment analysis of customer reviews, and the performance of major restaurant chains, offering actionable recommendations for Zomato to enhance its platform and user experiences. By addressing these findings, such as promoting online ordering, emphasizing food quality, and improving restaurant discoverability, Zomato can further personalize customer interactions, optimize restaurant discovery, and ultimately elevate customer satisfaction, solidifying its position as a leader in the competitive food delivery and dining industry.

Impact and Future Directions

This analysis of Zomato's data provides valuable insights to enhance platform functionality, personalize user experiences, and promote restaurant discovery and customer satisfaction. Restaurants can leverage these findings to improve offerings, address customer feedback, and optimize operations for greater success on the platform. This foundation for data-driven decision-making empowers Zomato to continuously evolve and remain a leading platform in the dynamic Indian restaurant industry.

By implementing these recommendations, Zomato can solidify its position as a leader in online food delivery and restaurant discovery, driving growth and innovation within the industry. This data-driven approach will not only benefit Zomato and its restaurant partners but also empower customers to make more informed dining choices, fostering a vibrant and evolving culinary landscape in India.

Link to access the Dataset and Coding file

https://drive.google.com/drive/folders/1KYiQ7tanYkUG2WDlDedcY8bp4-4tSIqW?usp=sharing

THANK YOU!