zomato

ZOMATO DATA ANALYSIS USING PYTHON

16 August 2025

tasbihaanwaradil@gmail.com

DATASET

	name	online_order	book_table	rate	votes	approx_cost(for two people)	listed_in(type)
0	Jalsa	Yes	Yes	4.1/5	775	800	Buffet
1	Spice Elephant	Yes	No	4.1/5	787	800	Buffet
2	San Churro Cafe	Yes	No	3.8/5	918	800	Buffet
3	Addhuri Udupi Bhojana	No	No	3.7/5	88	300	Buffet
4	Grand Village	No	No	3.8/5	166	600	Buffet
5	Timepass Dinner	Yes	No	3.8/5	286	600	Buffet
6	Rosewood International Hotel - Bar & Restaurant	No	No	3.6/5	8	800	Buffet
7	Onesta	Yes	Yes	4.6/5	2556	600	Cafes
8	Penthouse Cafe	Yes	No	4.0/5	324	700	other
9	Smacznego	Yes	No	4.2/5	504	550	Cafes
10	Village Café	Yes	No	4.1/5	402	500	Cafes
11	Cafe Shuffle	Yes	Yes	4.2/5	150	600	Cafes
12	The Coffee Shack	Yes	Yes	4.2/5	164	500	Cafes
13	Caf-Eleven	No	No	4.0/5	424	450	Cafes
14	San Churro Cafe	Yes	No	3.8/5	918	800	Cafes
15	Cafe Vivacity	Yes	No	3.8/5	90	650	Cafes
16	Catch-up-ino	Yes	No	3.9/5	133	800	Cafes
17	Kirthi's Biryani	Yes	No	3.8/5	144	700	Cafes
18	T3H Cafe	No	No	3.9/5	93	300	Cafes
19	360 Atoms Restaurant And Cafe	Yes	No	3.1/5	13	400	Cafes

📌 Step 1: Importing Python Libraries

Library & Purpose	Import Statement
Pandas – Data manipulation & analysis	import pandas as pd
NumPy – Numerical computing	import numpy as np
Matplotlib – Data visualization	import matplotlib.pyplot as plt
Seaborn – Statistical visualization	import seaborn as sns

💡 These libraries form the foundation for data cleaning, analysis, and visualization in this project.

📌 Step 2: Creating the DataFrame

Action	Code
Read CSV file into DataFrame	data_frame = pd.read_csv('Zomato-data-.csv')
Preview first 20 rows	print(data_frame.head(20))

💡 Pandas read_csv() loads data from a CSV file into a structured DataFrame.

📌 Step 3: Data Cleaning & Preparation

Action	Code

Define function to extract numeric rating	def handle_rate(value):
	  value = str(value).split('/')
	  value = value[0]
	  return float(value)

Apply function to rate column	data_frame['rate'] = data_frame['rate'].apply(handle_rate) print(data_frame.head())

💡 Converted ratings like ‘4.2/5’ into numeric values (e.g., 4.2) for analysis.

📌 Step 3: Data Cleaning & Preparation

Output:

	name	online_order	book_table	rate	votes	approx_cost(for two people)	listed_in(type)
0	Jalsa	Yes	Yes	4.1/5	775	800	Buffet
1	Spice Elephant	Yes	No	4.1/5	787	800	Buffet
2	San Churro Cafe	Yes	No	3.8/5	918	800	Buffet
3	Addhuri Udupi Bhojana	No	No	3.7/5	88	300	Buffet
4	Grand Village	No	No	3.8/5	166	600	Buffet

before

	name	online_order	book_table	rate	votes	approx_cost(for two people)	listed_in(type)
0	Jalsa	Yes	Yes	4.1	775	800	Buffet
1	Spice Elephant	Yes	No	4.1	787	800	Buffet
2	San Churro Cafe	Yes	No	3.8	918	800	Buffet
3	Addhuri Udupi Bhojana	No	No	3.7	88	300	Buffet
4	Grand Village	No	No	3.8	166	600	Buffet

after

💡 Converted ratings like ‘4.2/5’ into numeric values (e.g., 4.2) for analysis.

📌 Step 4: Getting Summary of DataFrame

Action	Code
Display summary info (columns, data types, non-null counts, memory usage)	data_frame.info()

💡 Display summary of DataFrame including columns, data types, and non-null counts..

📌 Step 4: Getting Summary of DataFrame

Output:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 148 entries, 0 to 147
Data columns (total 7 columns):
 #   Column                        Non-Null Count  Dtype  
---  ------                        --------------  -----  
 0   name                          148 non-null    object 
 1   online_order                  148 non-null    object 
 2   book_table                    148 non-null    object 
 3   rate                          148 non-null    float64
 4   votes                         148 non-null    int64  
 5   approx_cost(for two people)   148 non-null    int64  
 6   listed_in(type)               148 non-null    object 
dtypes: float64(1), int64(2), object(4)
memory usage: 8.2+ KB
```

📌 Step 5: Checking for Missing Values

Action	Code
Check each column for missing/null values	print(data_frame.isnull().sum())

💡 Check each column for missing values to identify data gaps.

📌 Step 5: Checking for Missing Values

Output:

```
name                                0
online_order                        0
book_table                          0
rate                                0
votes                               0
approx_cost(for two people)         0
listed_in(type)                     0
dtype: int64
```

💡 There is no NULL value in dataframe.

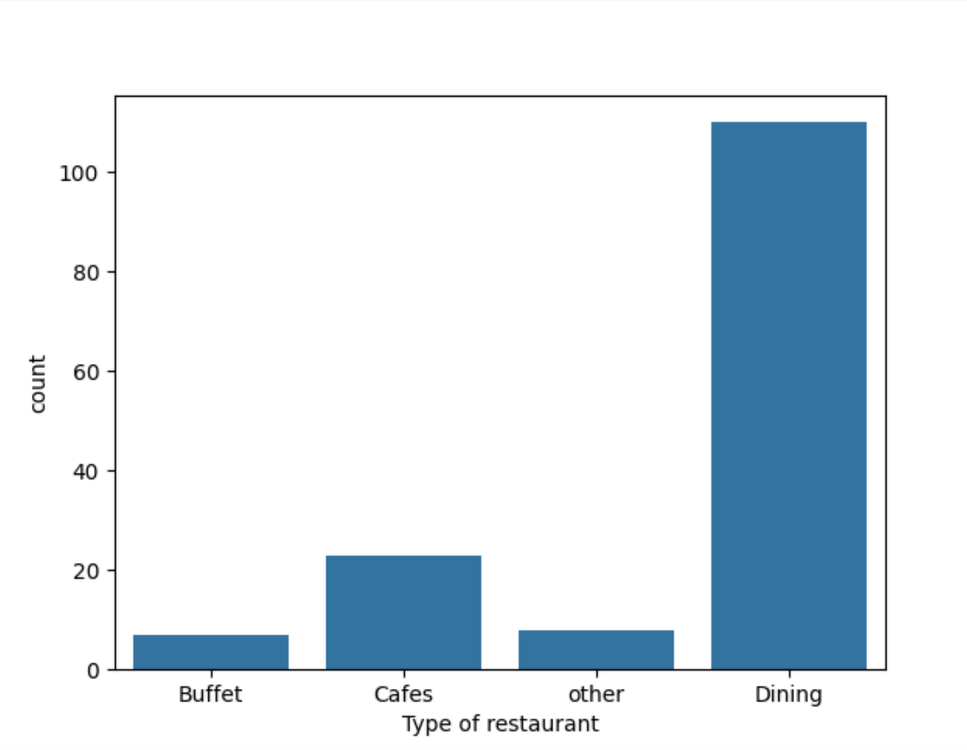
📊 Step 6: Visualizing Restaurant Category Distribution

Action	Code
Plot a bar chart showing the count of restaurants by type (Delivery, Dine-out, Cafes, etc.)	sns.countplot(x=data_frame['listed_in(type)']) plt.xlabel("Type of restaurant") plt.show()

💡 Visualize the distribution of restaurants across different categories.

📊 Step 6: Restaurant Category Distribution

Output:



💡 The majority of restaurants in the dataset fall under the Dining category.

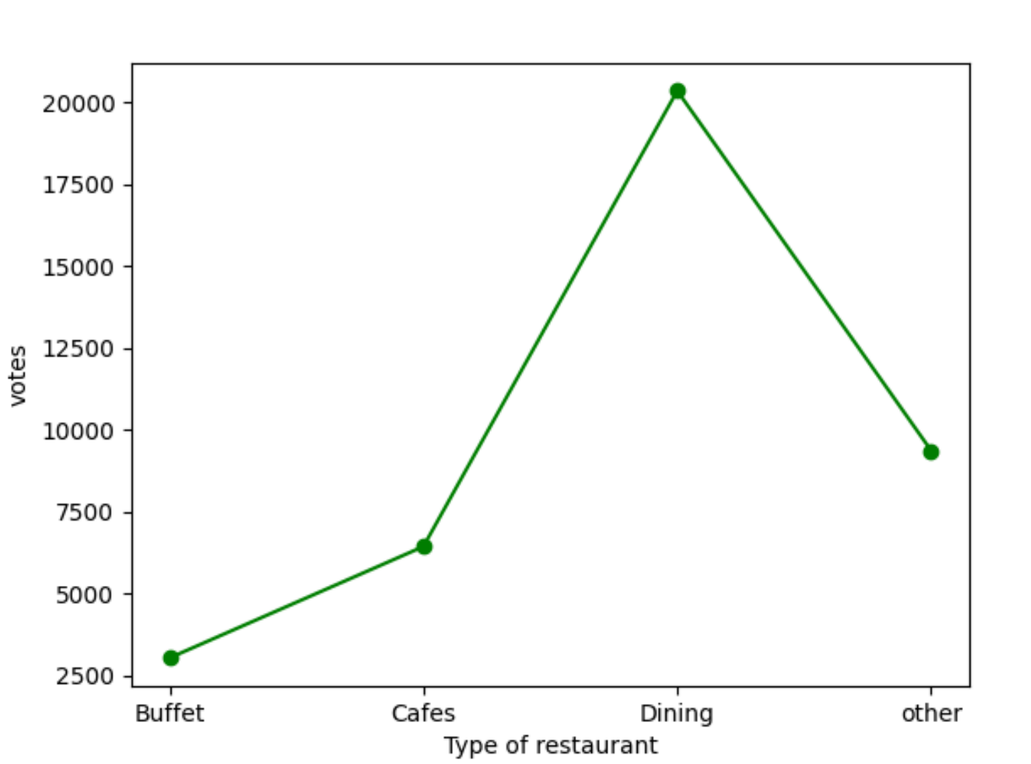
📊 Step 7: Votes by Restaurant Type

Action	Code
Group votes by restaurant type	grouped_data = data_frame.groupby('listed_in(type)') ['votes'].sum()
Convert result to DataFrame	result = pd.DataFrame({'votes': grouped_data})
Plot votes vs. restaurant type	plt.plot(result, c='green', marker='o') plt.xlabel("Type of Restaurant") plt.ylabel("Votes") plt.show()

💡 Identify which restaurant type has the highest ratings.

📊 Step 7: Votes by Restaurant Type

Output:



👉 Casual Dining restaurants received the highest votes among all types.

📊 Step 8: Identify the Most Voted Restaurant

Action	Code
Find the maximum number of votes	max_votes = data_frame['votes'].max()
Identify restaurant(s) with max votes	restaurant_with_max_votes = data_frame.loc[data_frame['votes'] == max_votes, 'name'].tolist()
Display the result	print('Restaurant(s) with the maximum votes:') print(restaurant_with_max_votes)

Output:

```
Restaurant(s) with the maximum votes:
['Empire Restaurant']
```

💡 The “Empire Restaurant” is the restaurant with the highest votes.

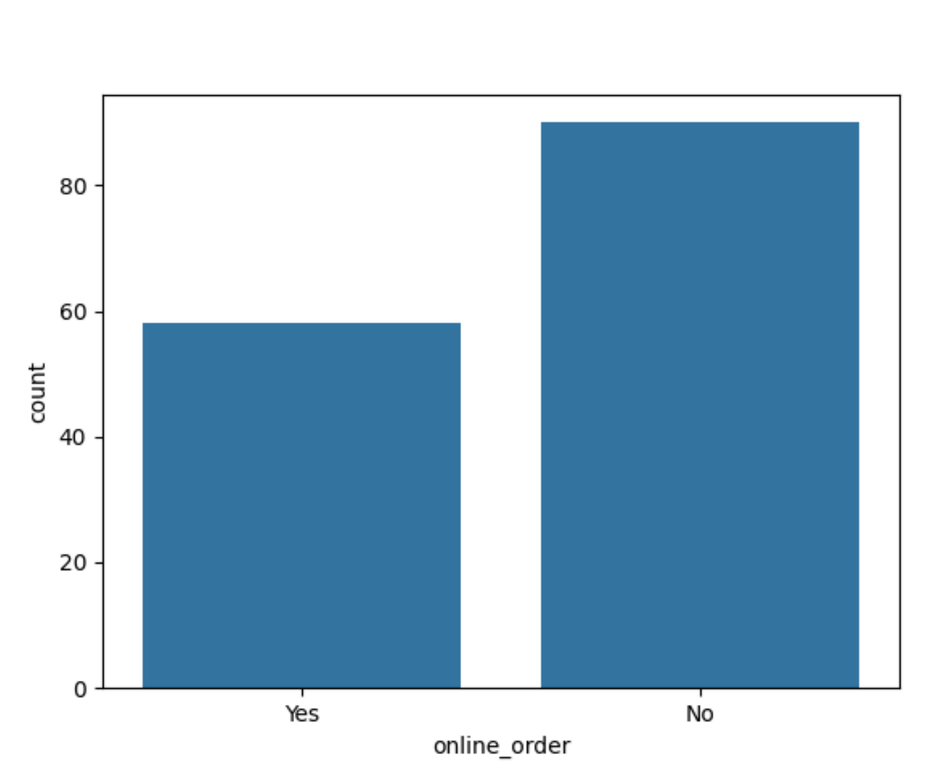
📊 Step 9: Online Order Availability

Action	Code
Plot the count of restaurants offering online orders	sns.countplot(x=data_frame['online_order'])
Display the plot	plt.show()

💡 This plot shows how many restaurants accept online orders versus those that don’t.

📊 Step 9: Online Order Availability

Output:



💡 A majority of the restaurants do not accept online orders.

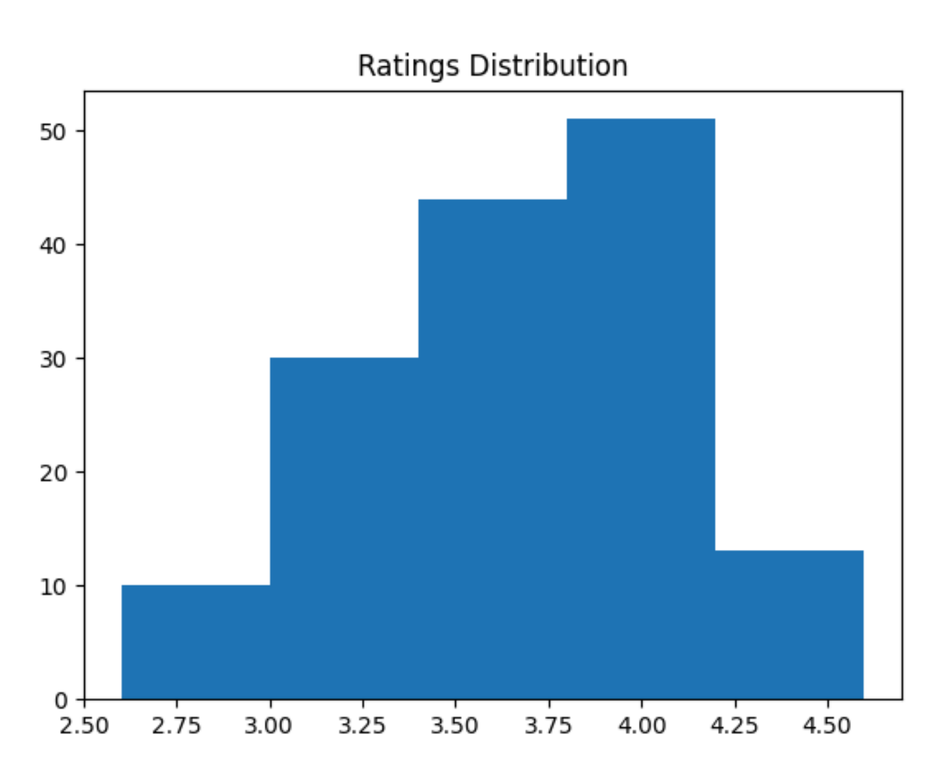
📊 Step 10: Analyze Ratings

Action	Code
Visualize how ratings are distributed across restaurants	plt.hist(data_frame['rate'], bins=5) plt.title("Ratings Distribution") plt.show()

💡 Visualize how restaurant ratings are distributed across different ranges.

📊 Step 10: Analyze Ratings

Output:



💡 The majority of restaurants received ratings ranging from 3.5 to 4.

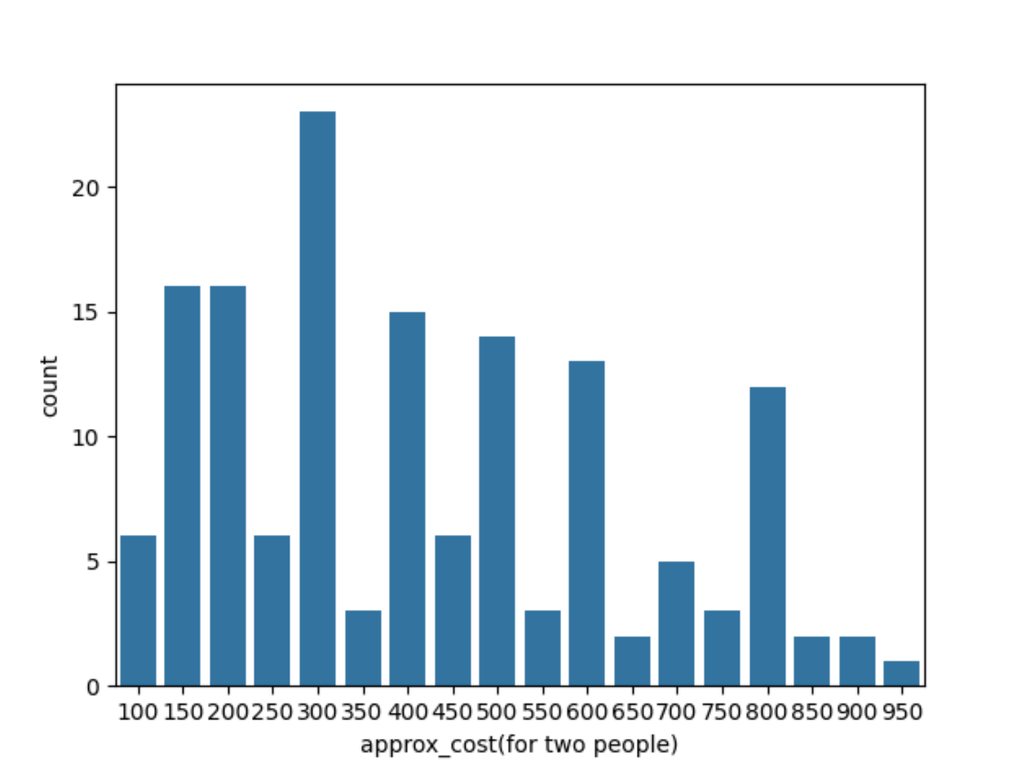
📊 Step 11: Approximate Cost for Two People

Action	Code
Analyze the approx_cost(for two people) column to find the preferred price range	plt.hist(data_frame['rate'], bins=5) plt.title("Ratings Distribution") plt.show()

💡 Visualize the preferred cost range for two people.

📊 Step 11: Approximate Cost for Two People

Output:



💡 The majority of couples prefer restaurants with an approximate cost of ₹300.

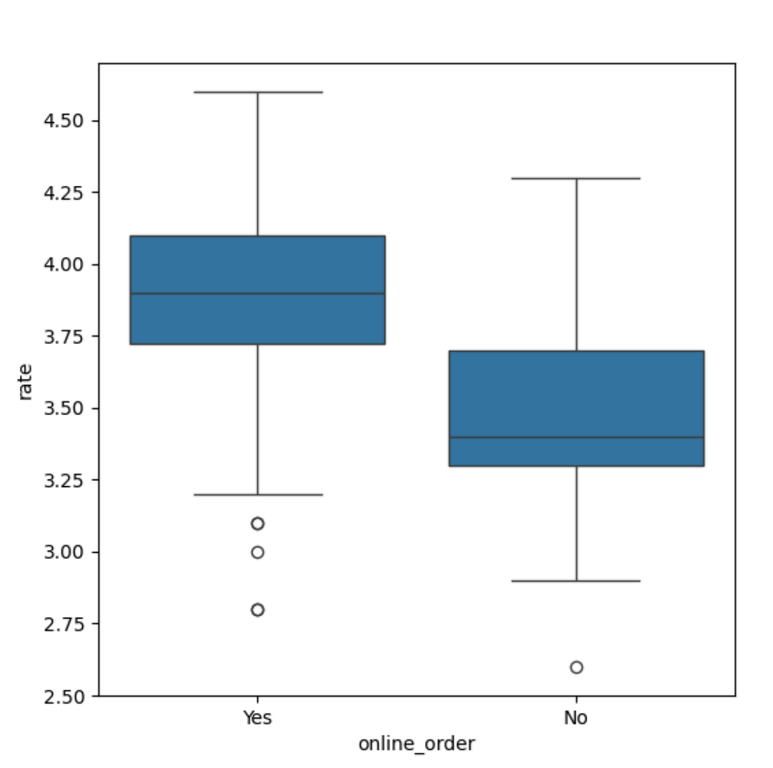
📊 Step 12: Ratings Comparison – Online vs Offline Orders

Action	Code
Compare restaurant ratings between those that accept online orders and those that don’t.	plt.figure(figsize=(6,6)) sns.boxplot(x='online_order', y='rate', data=data_frame) plt.show()

💡 Compare restaurant ratings: Online vs Offline orders.

📊 Step 12: Ratings Comparison – Online vs Offline Orders

Output:



💡 Offline orders received lower ratings compared to online orders, which obtained excellent ratings.

📊 Step 13: Order Mode Preferences by Restaurant Type

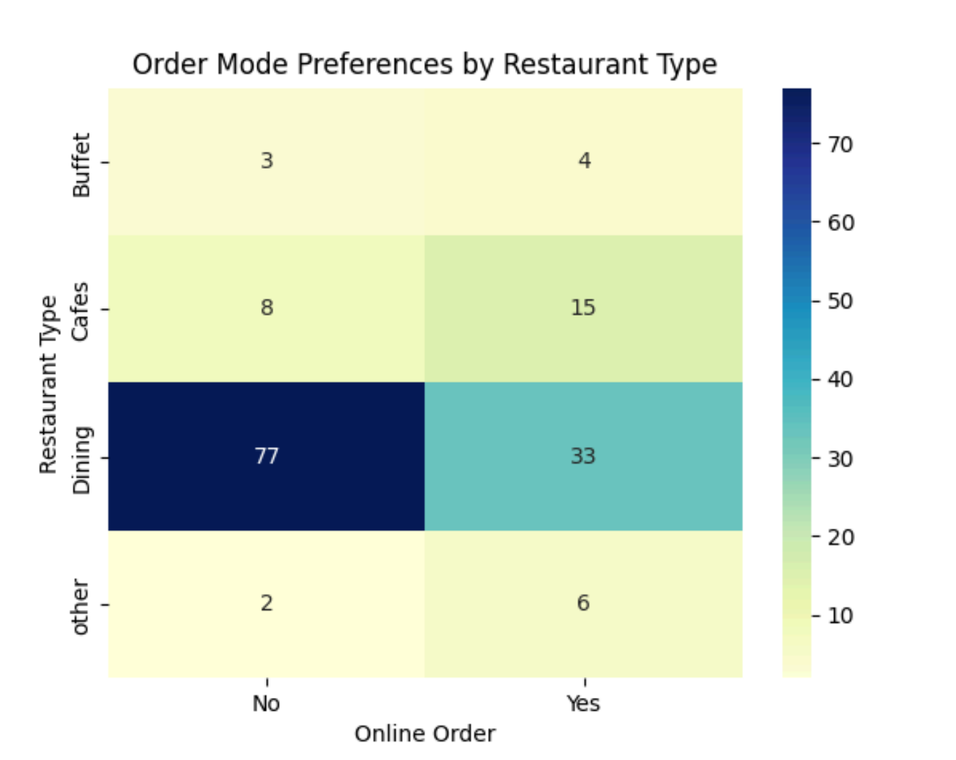
Action	Code

Analyze restaurant type preferences for online vs offline orders using a heatmap.	pivot_table = data_frame.pivot_table(
	  index='listed_in(type)',
	  columns='online_order',
	  aggfunc='size',
	  fill_value=0
	)
	
	sns.heatmap(pivot_table, annot=True, cmap='YlGnBu', fmt='d')
	plt.title('Order Mode Preferences by Restaurant Type')
	plt.xlabel('Online Order')
	plt.ylabel('Restaurant Type')
	plt.show()

💡 Visualize the preferred order mode across different restaurant types.

📊 Step 13: Order Mode Preferences by Restaurant Type

Output:



💡 Dining restaurants primarily accept offline orders, whereas cafes primarily receive online orders.

Found this Insightful?

Connect for the latest data-driven insights, trends, and valuable analysis.

www.linkedin.com/in/tasbiha-anwar-adil