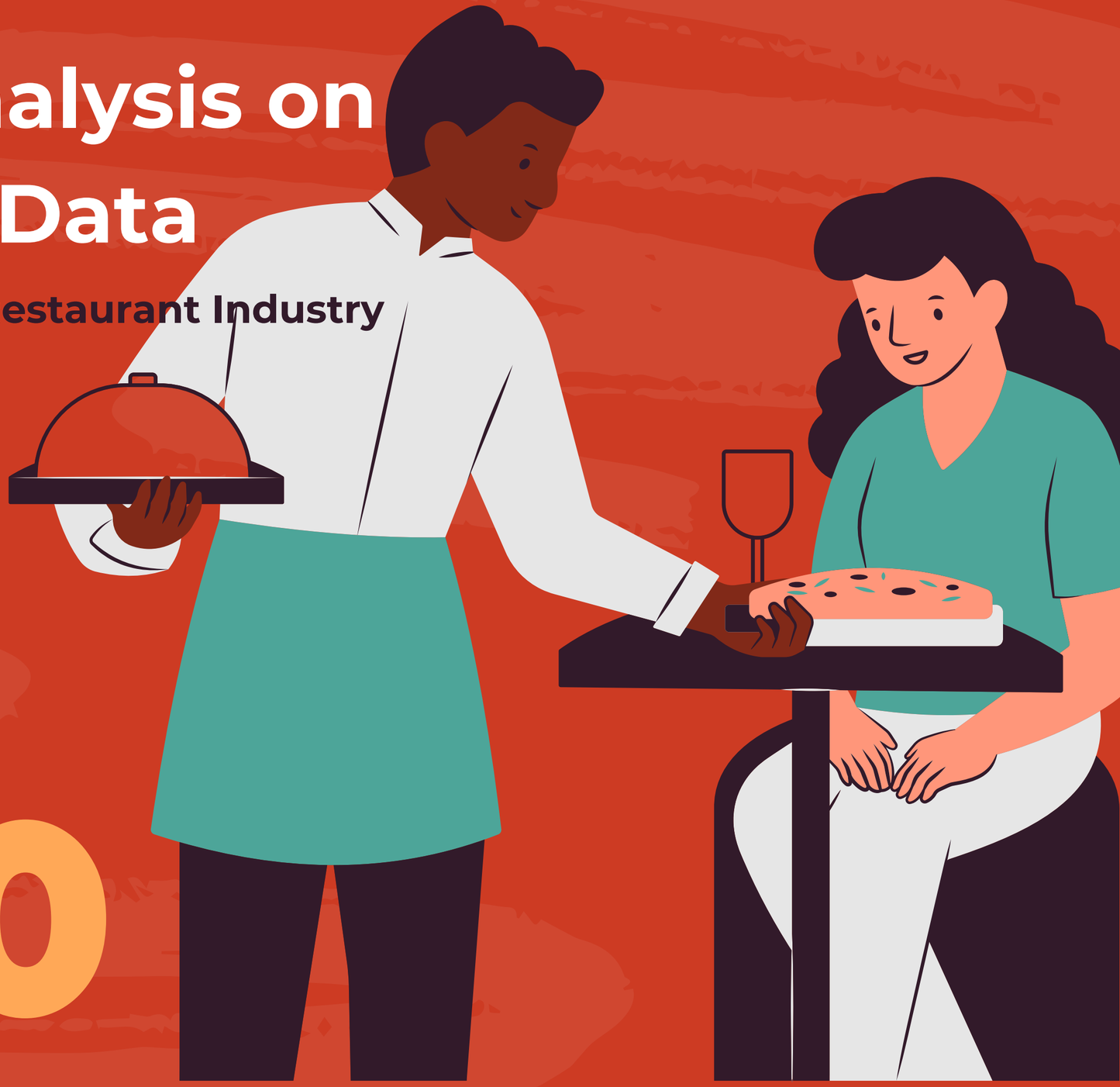
Exploratory Data Analysis on Zomato Restaurant Data

Uncovering Insights and Trends in the Restaurant Industry



ZOMATO

CONTENT

- **Introduction to dataset**
- **Data Preprocessing**
- **Data Transformation**
- **Visualizing Online Order**
- **Visualizing Online Order vs Rate**
- **Relationship Between Restaurant Ratings and Number of Votes**
- **Correlation Analysis of Key Restaurant Metrics**
- **Conclusion**

Introduction to dataset

- This dataset is a collection of restaurants that are registered on Zomato in Bengaluru City.
- In this dataset, we have more than 50000 rows and 17 columns, a fairly large dataset.
- You will be able to get hands-on experience while performing the following tasks and will be able to understand how real-world problem statement analysis is done.

Infomation about the dataset

- It contains 51717 rows and 17 columns
- Columns present in the dataset are given in image

```
In [6]: df.shape   #to know about columns  & rows

Out[6]: (51717, 17)

In [7]: df.columns    #list of columns

Out[7]: Index(['url', 'address', 'name', 'online_order', 'book_table', 'rate', 'votes',
               'phone', 'location', 'rest_type', 'dish_liked', 'cuisines',
               'approx_cost(for two people)', 'reviews_list', 'menu_item',
               'listed_in(type)', 'listed_in(city)'],
              dtype='object')
```

Data Preprocessing

- Data Cleaning: Handling missing values, duplicate entries, and unnecessary data.

```
#  droping unnecessary  data
df = df.drop(['url', 'address', 'phone', 'menu_item', 'dish_liked', 'reviews_list'], axis = 1)
df.head()
```

	name	online_order	book_table	rate	votes	location	rest_type	cuisines	approx_cost(for two people)	listed_in(type)	listed_in(city)
0	Jalsa	Yes	Yes	4.1	775	Banashankari	Casual Dining	North Indian, Mughlai, Chinese	800	Buffet	Banashankari
1	Spice Elephant	Yes	No	4.1	787	Banashankari	Casual Dining	Chinese, North Indian, Thai	800	Buffet	Banashankari
2	San Churro Cafe	Yes	No	3.8	918	Banashankari	Cafe, Casual Dining	Cafe, Mexican, Italian	800	Buffet	Banashankari
3	Addhuri Udupi Bhojana	No	No	3.7	88	Banashankari	Quick Bites	South Indian, North Indian	300	Buffet	Banashankari
4	Grand Village	No	No	3.8	166	Basavanagudi	Casual Dining	North Indian, Rajasthani	600	Buffet	Banashankari

Order Now!

Data Transformation

```
df['Cost2plates'].unique()
```

```
array(['800', '300', '600', '700', '550', '500', '450', '650', '400',
       '900', '200', '750', '150', '850', '100', '1,200', '350', '250',
       '950', '1,000', '1,500', '1,300', '199', '80', '1,100', '160',
       '1,600', '230', '130', '50', '190', '1,700', nan, '1,400', '180',
       '1,350', '2,200', '2,000', '1,800', '1,900', '330', '2,500',
       '2,100', '3,000', '2,800', '3,400', '40', '1,250', '3,500',
       '4,000', '2,400', '2,600', '120', '1,450', '469', '70', '3,200',
       '60', '560', '240', '360', '6,000', '1,050', '2,300', '4,100',
       '00', '3,700', '1,650', '2,700', '4,500', '140'], dtype=object)
```

```
#removig comma
def handlecomma(value):
    value = str(value)
    if ',' in value:
        value = value.replace(',', '')
        return float(value)
    else:
        return float(value)

df['Cost2plates'] = df['Cost2plates'].apply(handlecomma)
df['Cost2plates'].unique()
```

```
array([ 800.,  300.,  600.,  700.,  550.,  500.,  450.,  650.,  400.,
        900.,  200.,  750.,  150.,  850.,  100., 1200.,  350.,  250.,
        950., 1000., 1500., 1300.,  199.,   80., 1100.,  160., 1600.,
        230.,  130.,   50.,  190., 1700.,   nan, 1400.,  180., 1350.,
       2200., 2000., 1800., 1900.,  330., 2500., 2100., 3000., 2800.,
       3400.,   40., 1250., 3500., 4000., 2400., 2600.,  120., 1450.,
        469.,   70., 3200.,   60.,  560.,  240.,  360., 6000., 1050.,
       2300., 4100., 5000., 3700., 1650., 2700., 4500.,  140.])
```

Converting data into a suitable format for analysis, such as normalizing text data, encoding categorical variables, or creating new features (e.g., deriving average ratings).

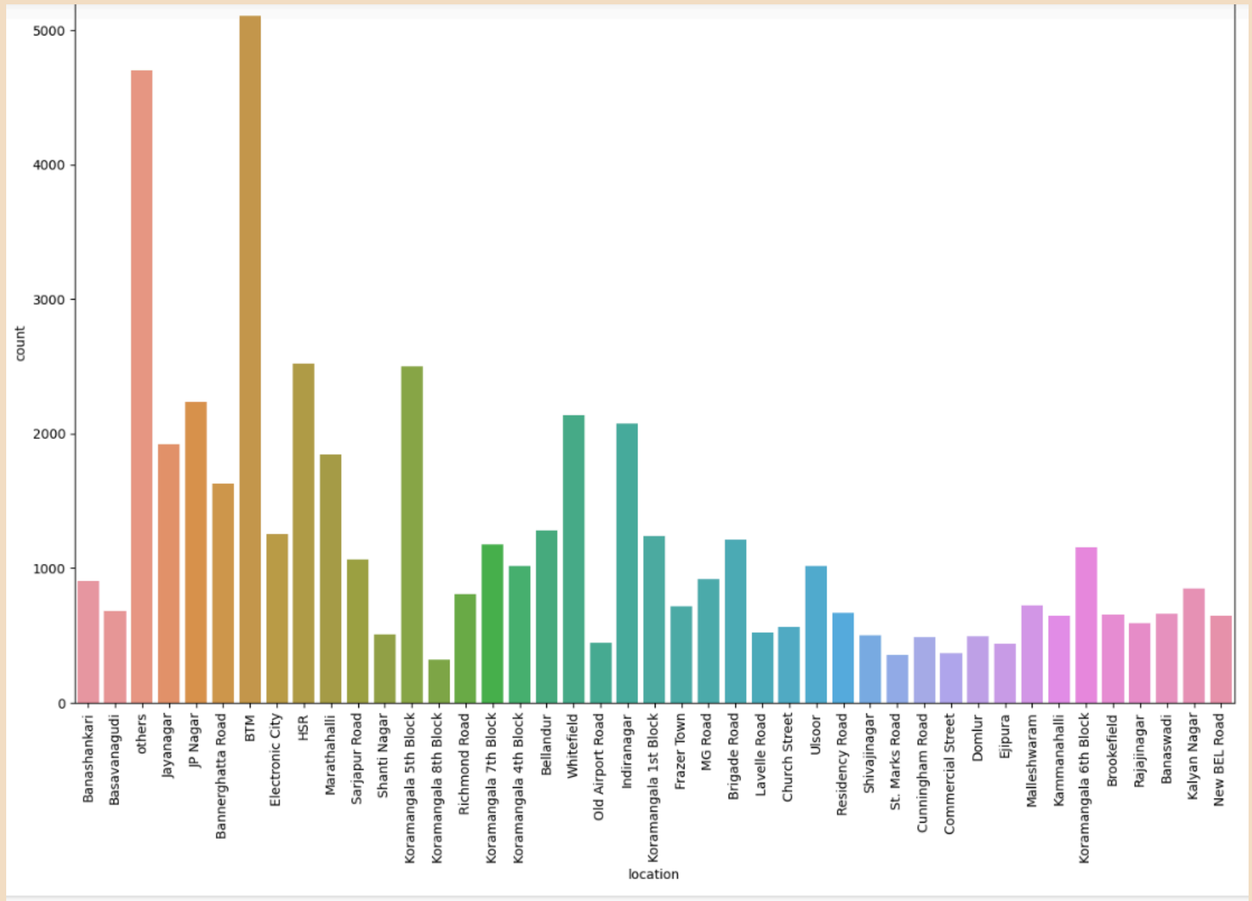
Bar Chart of Restaurant Counts by Location

Objective:

- The visualization provides a clear view of how restaurants are distributed across different locations in the dataset.

Insight:

- It helps identify areas with high restaurant density and areas with fewer restaurants, which can be useful for market analysis or business strategy.

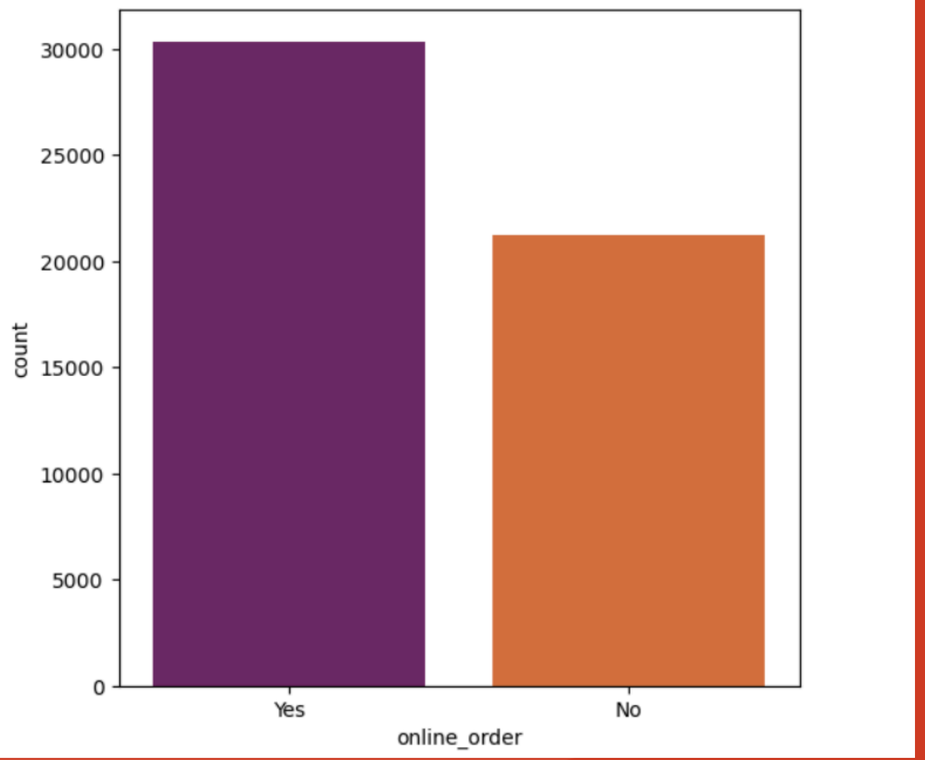


Visualizing Online Order

- A majority of restaurants in the dataset offer online orders."
- "The distribution shows that X axis shows the count of the restaurant."
- "The proportion of restaurants offering online orders or not is shown on Y axis."

```
plt.figure(figsize = (6,6))
sns.countplot(x =df['online_order'], palette = 'inferno')
```

Out[47]: <Axes: xlabel='online_order', ylabel='count'>



Visualizing Online Order vs Rate

Comparison of Ratings:

- "Restaurants that offer online orders generally have higher median ratings compared to those that do not."

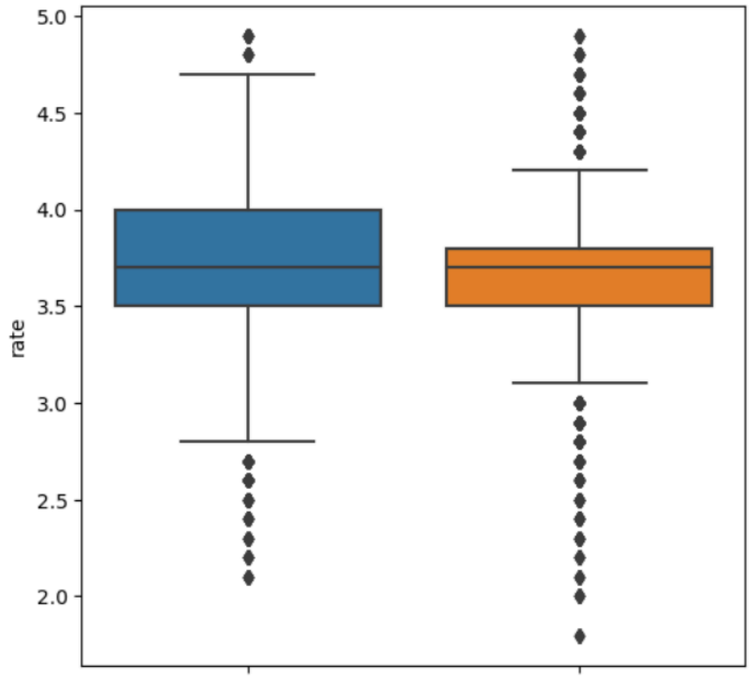
Spread of Ratings:

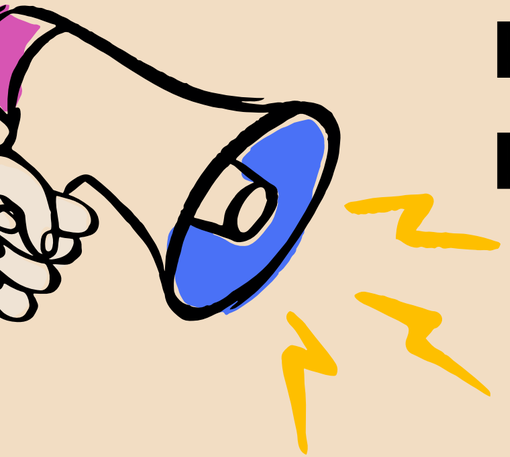
- "The interquartile range (IQR) is wider for restaurants offering online orders, indicating more variability in their ratings."

Outliers:

- "A few restaurants, regardless of online ordering availability, have lower ratings, as shown by the outliers."

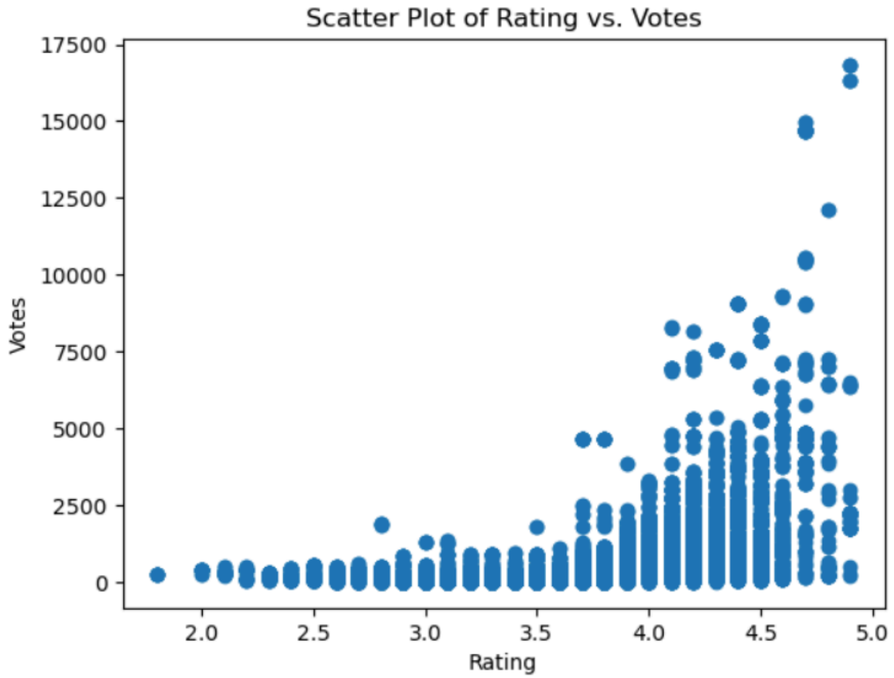
```
In [50]: #Visualizing Online Order vs Rate
         plt.figure(figsize = (6,6))
         sns.boxplot(x = 'online_order', y = 'rate', data = df)

Out[50]: <Axes: xlabel='online_order', ylabel='rate'>
```





Relationship Between Restaurant Ratings and Number of Votes

```
plt.scatter(x=df['rate'], y=df['votes'])
plt.xlabel('Rating')
plt.ylabel('Votes')
plt.title('Scatter Plot of Rating vs. Votes')
plt.show()
```



Positive Correlation:

- "There is a positive correlation between the number of votes and the restaurant ratings."

Distribution of Votes:

- "Restaurants with higher ratings (above 4.0) tend to receive significantly more votes."

Lower Ratings:

- "Restaurants with ratings below 3.0 generally receive fewer votes, indicating less customer engagement."

Clustering:

- "A dense cluster of restaurants is observed between ratings of 3.5 to 4.5, with varying vote counts."

Correlation Analysis of Key Restaurant Metrics

Correlation Between Ratings and Votes:

- "The correlation between rate and votes is moderate (0.43), indicating that higher-rated restaurants tend to receive more votes."

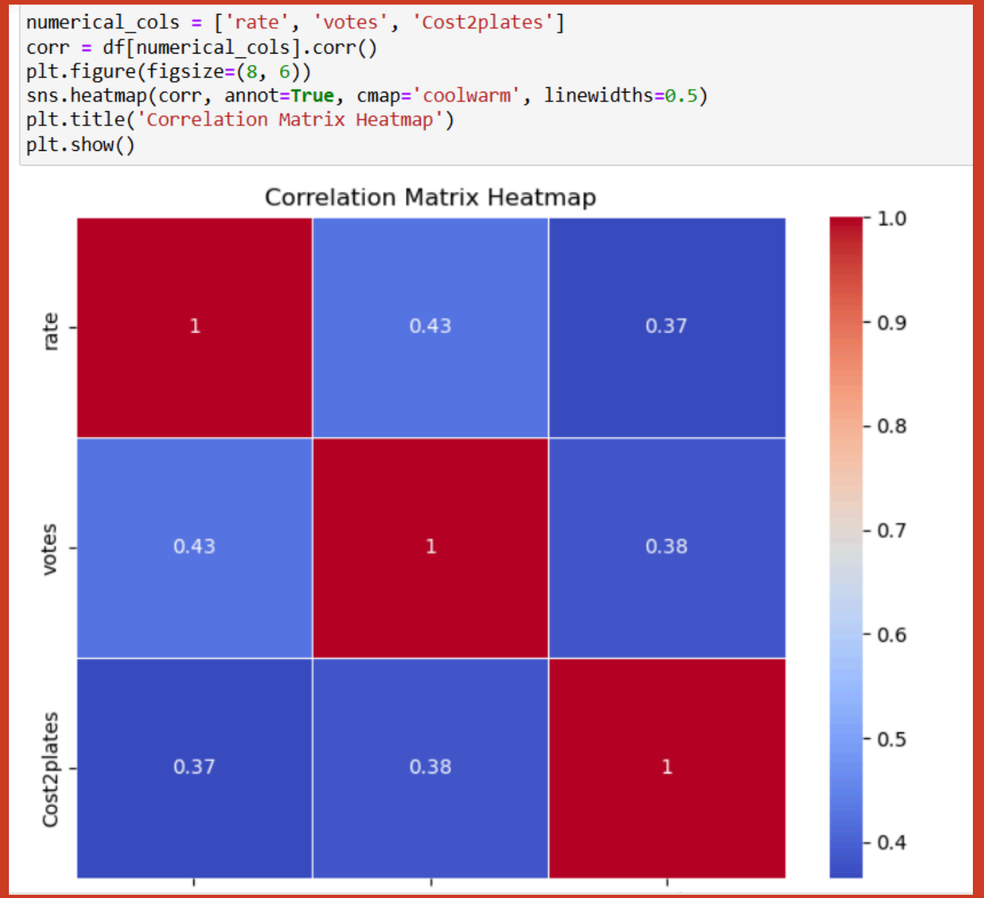
Correlation Between Ratings and Cost:

- "The correlation between rate and Cost2plates is lower (0.37), suggesting that the cost of dining has a weaker association with restaurant ratings."

Correlation Between Votes and Cost:

- "There is a moderate correlation (0.38) between votes and Cost2plates, meaning more expensive restaurants might receive slightly more votes."

```
numerical_cols = ['rate', 'votes', 'Cost2plates']
corr = df[numerical_cols].corr()
plt.figure(figsize=(8, 6))
sns.heatmap(corr, annot=True, cmap='coolwarm', linewidths=0.5)
plt.title('Correlation Matrix Heatmap')
plt.show()
```



Conclusion

- Overall, this project highlighted the significance of exploring the Zomato dataset, providing valuable insights into transaction patterns, customer preferences, and cuisine offerings. ·
- The findings contribute to a deeper understanding of the restaurant business, enabling data-driven decision-making for restaurant owners, managers, and stakeholders.
- Further analysis and exploration can be conducted to delve deeper into specific aspects of the dataset.

Thankyou