Import Libraries

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
```

Load Dataset

```
df = pd.read_csv("/content/zomato.csv")
print(df.head())
print(df.columns)
print(df['online_order'].unique())
```

```
2  https://www.zomato.com/SanchurroBangalore?cont...
3  https://www.zomato.com/bangalore/addhuri-udupi...
4  https://www.zomato.com/bangalore/grand-village...

                                             address                   name  \
0  942, 21st Main Road, 2nd Stage, Banashankari, ...                  Jalsa
1  2nd Floor, 80 Feet Road, Near Big Bazaar, 6th ...         Spice Elephant
2  1112, Next to KIMS Medical College, 17th Cross...        San Churro Cafe
3  1st Floor, Annakuteera, 3rd Stage, Banashankar...  Addhuri Udupi Bhojana
4  10, 3rd Floor, Lakshmi Associates, Gandhi Baza...          Grand Village

  online_order book_table   rate  votes                             phone  \
0          Yes        Yes  4.1/5    775    080 42297555\r\n+91 9743772233
1          Yes         No  4.1/5    787                      080 41714161
2          Yes         No  3.8/5    918                    +91 9663487993
3           No         No  3.7/5     88                    +91 9620009302
4           No         No  3.8/5    166  +91 8026612447\r\n+91 9901210005

      location            rest_type  \
0  Banashankari        Casual Dining
1  Banashankari        Casual Dining
2  Banashankari  Cafe, Casual Dining
3  Banashankari          Quick Bites
4  Basavanagudi        Casual Dining

                                          dish_liked  \
0  Pasta, Lunch Buffet, Masala Papad, Paneer Laja...
1  Momos, Lunch Buffet, Chocolate Nirvana, Thai G...
2  Churros, Cannelloni, Minestrone Soup, Hot Choc...
3                                        Masala Dosa
4                                Panipuri, Gol Gappe

                         cuisines approx_cost(for two people)  \
0  North Indian, Mughlai, Chinese                         800
1     Chinese, North Indian, Thai                         800
2          Cafe, Mexican, Italian                         800
3      South Indian, North Indian                         300
4        North Indian, Rajasthani                         600

                                        reviews_list menu_item  \
0  [('Rated 4.0', 'RATED\n  A beautiful place to ...        []
1  [('Rated 4.0', 'RATED\n  Had been here for din...        []
2  [('Rated 3.0', "RATED\n  Ambience is not that ...        []
3  [('Rated 4.0', "RATED\n  Great food and proper...        []
4  [('Rated 4.0', 'RATED\n  Very good restaurant ...        []

  listed_in(type) listed_in(city)
0          Buffet    Banashankari
1          Buffet    Banashankari
2          Buffet    Banashankari
3          Buffet    Banashankari
4          Buffet    Banashankari
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')
['Yes' 'No']
```

Clean the Data

```
df.drop_duplicates(inplace=True)
df['rate'] = df['rate'].astype(str)
df = df[df['rate'] != 'NEW']
df['rate'] = df['rate'].apply(lambda x: x.replace('/5', '').replace('-', '').strip())
df['rate'] = df['rate'].replace('', '0')
df['rate'] = df['rate'].astype(float)
df['rate'] = df['rate'].fillna(df['rate'].mode()[0])
df.dropna(inplace=True)

df['online_order'] = df['online_order'].map({'Yes': 1, 'No': 0})
df['book_table'] = df['book_table'].map({'Yes': 1, 'No': 0})

print("First 5 rows:\n", df.head())
print("\nData types:\n", df.dtypes)
print("\nNull values in each column:\n", df.isnull().sum())
print("\nSummary statistics:\n", df.describe())
```

```
address                         object
name                            object
online_order                   float64
book_table                     float64
rate                           float64
votes                            int64
phone                           object
location                        object
rest_type                       object
dish_liked                      object
cuisines                        object
approx_cost(for two people)    float64
reviews_list                    object
menu_item                       object
listed_in(type)                 object
listed_in(city)                 object
dtype: object

Null values in each column:
 url                               0
address                          0
name                             0
online_order                 15497
book_table                   15497
rate                             0
votes                            0
phone                            0
location                         0
rest_type                        0
dish_liked                       0
cuisines                         0
approx_cost(for two people)      0
reviews_list                     0
menu_item                        0
listed_in(type)                  0
listed_in(city)                  0
dtype: int64

Summary statistics:
        online_order  book_table          rate         votes  \
count           0.0         0.0  15497.000000  15497.000000
mean            NaN         NaN      3.927386    418.028715
std             NaN         NaN      0.270377    839.525658
min             NaN         NaN      3.200000      0.000000
25%             NaN         NaN      3.800000     93.000000
50%             NaN         NaN      3.900000    189.000000
75%             NaN         NaN      4.100000    426.000000
max             NaN         NaN      4.800000  14726.000000

       approx_cost(for two people)
count                 15497.000000
mean                    510.993741
std                     201.162390
min                      40.000000
25%                     350.000000
50%                     500.000000
75%                     700.000000
max                     950.000000
```

Handling Outliers in 'rate'

```
Q1 = df['rate'].quantile(0.25)
Q3 = df['rate'].quantile(0.75)
IQR = Q3 - Q1
before = df.shape[0]
df = df[(df['rate'] >= Q1 - 1.5 * IQR) & (df['rate'] <= Q3 + 1.5 * IQR)]
after = df.shape[0]
print(f"Outliers removed: {before - after}")
```

```
Outliers removed: 1827
```

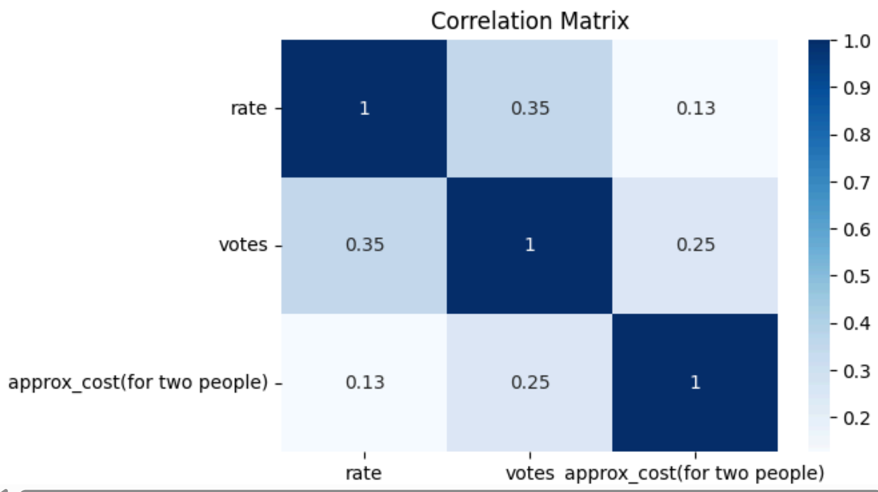
Summary Statistics

```
df.describe()
```

	online_order	book_table	rate	votes
count	21248.000000	21248.000000	21248.000000	21248.000000
mean	0.711455	0.280779	3.992620	631.922534
std	0.453096	0.449390	0.287828	1117.463832
min	0.000000	0.000000	3.200000	0.000000
25%	0.000000	0.000000	3.800000	109.000000
50%	1.000000	0.000000	4.000000	238.000000
75%	1.000000	1.000000	4.200000	654.000000
max	1.000000	1.000000	4.800000	14956.000000

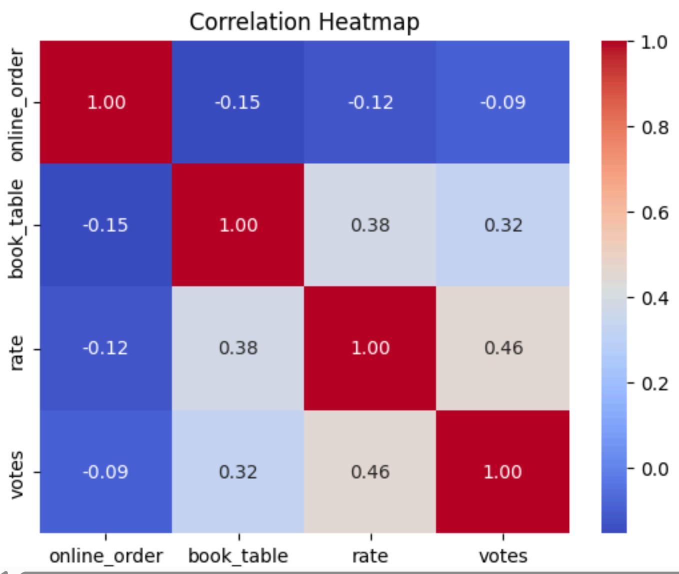
Correlation matrix

```
cols = ['rate', 'votes', correct_cost_col]
if all(col in df.columns for col in cols):

    corr = df[cols].corr()
    plt.figure(figsize=(6, 4))
    sns.heatmap(corr, annot=True, cmap='Blues')
    plt.title('Correlation Matrix')
    plt.show()
else:
    print(f"Error: Not all required columns {cols} found in DataFrame.")
```



Heatmap

```
import seaborn as sns
import matplotlib.pyplot as plt

corr = df.corr(numeric_only=True)
sns.heatmap(corr, annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Correlation Heatmap')
plt.show()
```

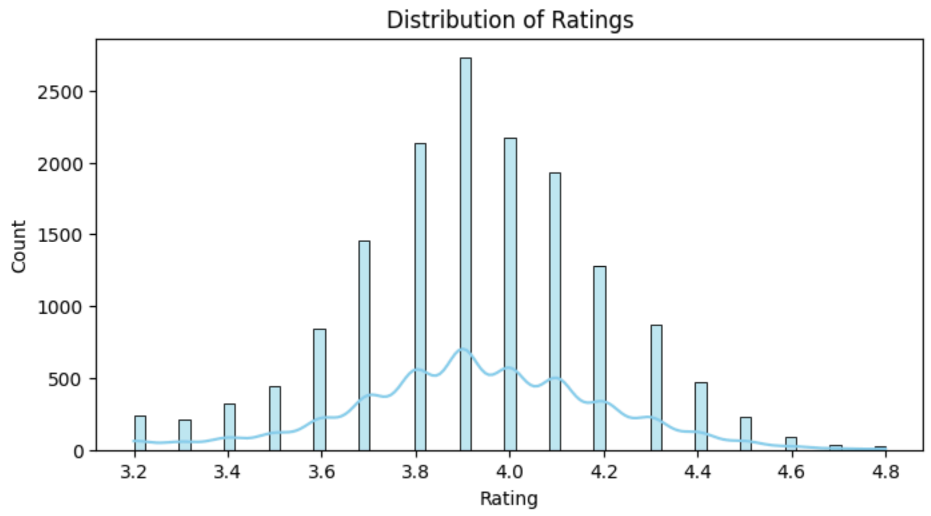


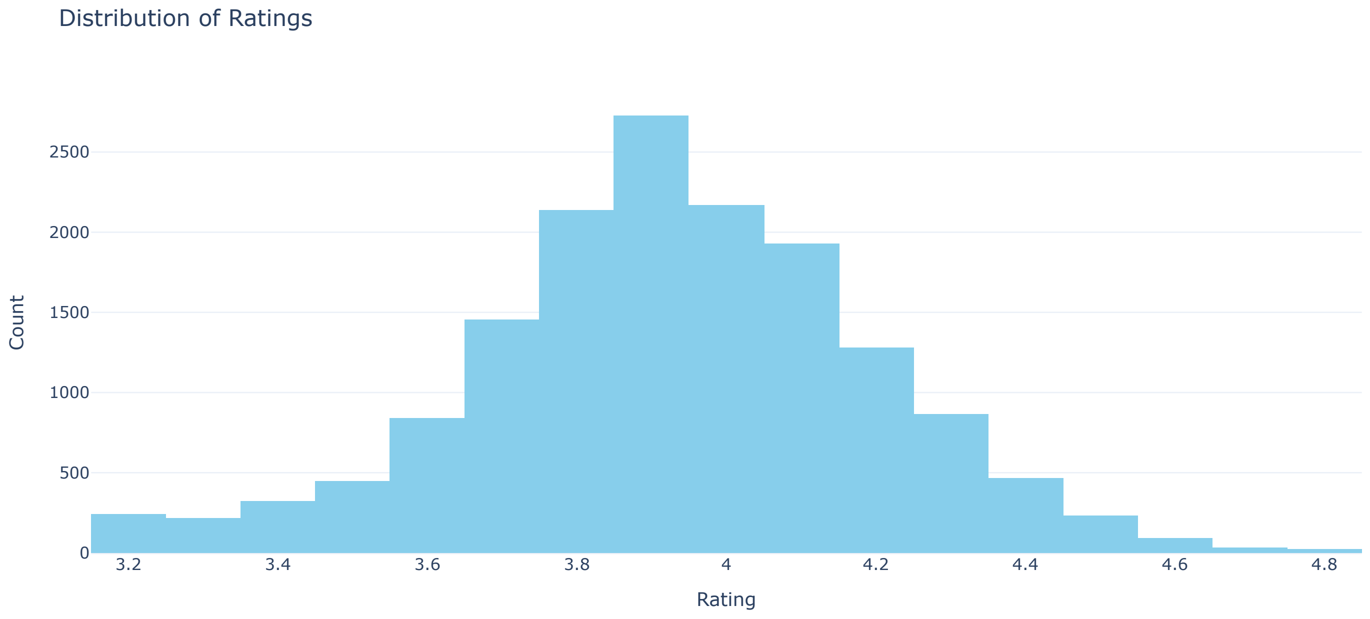
Data Visualizations

a. Rating Distribution

```
df['rate'] = df['rate'].astype(str).str.extract(r'(\d+\.?\d*)')[0]
df['rate'] = pd.to_numeric(df['rate'], errors='coerce')
df = df.dropna(subset=['rate'])
print("Number of valid ratings:", len(df))
print("Unique ratings sample:", df['rate'].unique()[:10])

# Plot with seaborn
plt.figure(figsize=(8, 4))
sns.histplot(df['rate'], kde=True, color='skyblue')
plt.title('Distribution of Ratings')
plt.xlabel('Rating')
plt.ylabel('Count')
plt.show()

# Plot with Plotly Express
import plotly.express as px
fig = px.histogram(df, x='rate', nbins=30, title='Distribution of Ratings',
                   labels={'rate': 'Rating'},
                   color_discrete_sequence=['skyblue'])
fig.update_layout(xaxis_title='Rating', yaxis_title='Count', template='plotly_white')
fig.show()
```

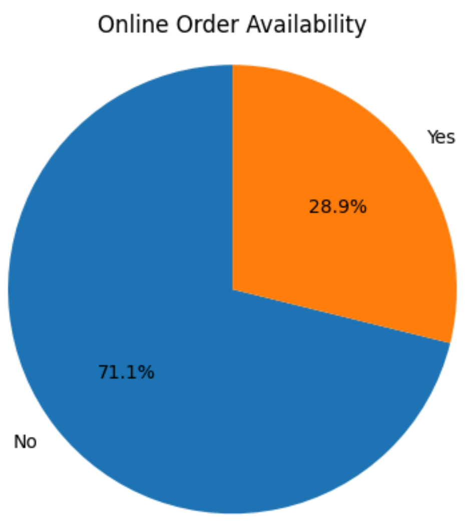
```
Number of valid ratings: 15497
Unique ratings sample: [4.1 3.8 3.7 4.6 4.  4.2 3.9 3.6 4.4 4.3]
```





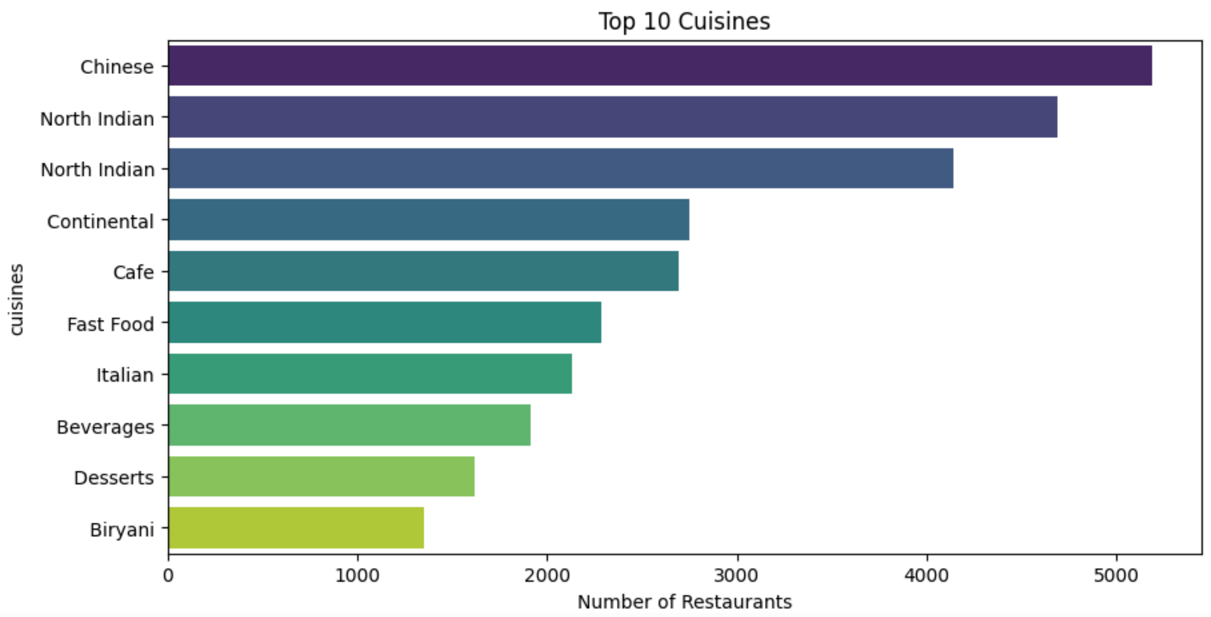
b. Online Order Pie Chart

```
labels = ['No', 'Yes']
values = df['online_order'].value_counts()
plt.pie(values, labels=labels, autopct='%1.1f%%', startangle=90)
plt.title('Online Order Availability')
plt.axis('equal')
plt.show()
```



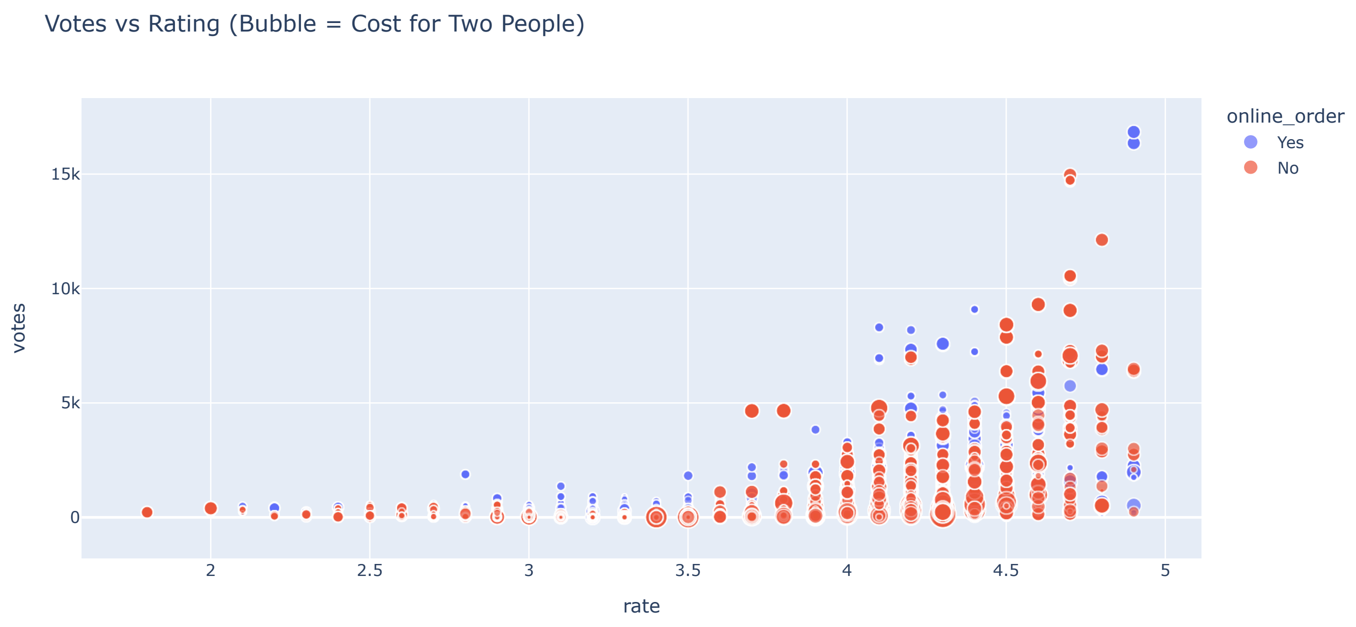
c. Top 10 Cuisines

```
df['cuisines'] = df['cuisines'].astype(str)
cuisines_series = df['cuisines'].str.split(',').explode()
top_cuisines = cuisines_series.value_counts().head(10)
plt.figure(figsize=(10,5))
sns.barplot(x=top_cuisines.values, y=top_cuisines.index, palette='viridis', hue=top_cuisines.index, legend=False)
plt.title('Top 10 Cuisines')
plt.xlabel('Number of Restaurants')
plt.show()
```



d. Interactive: Votes vs Rating

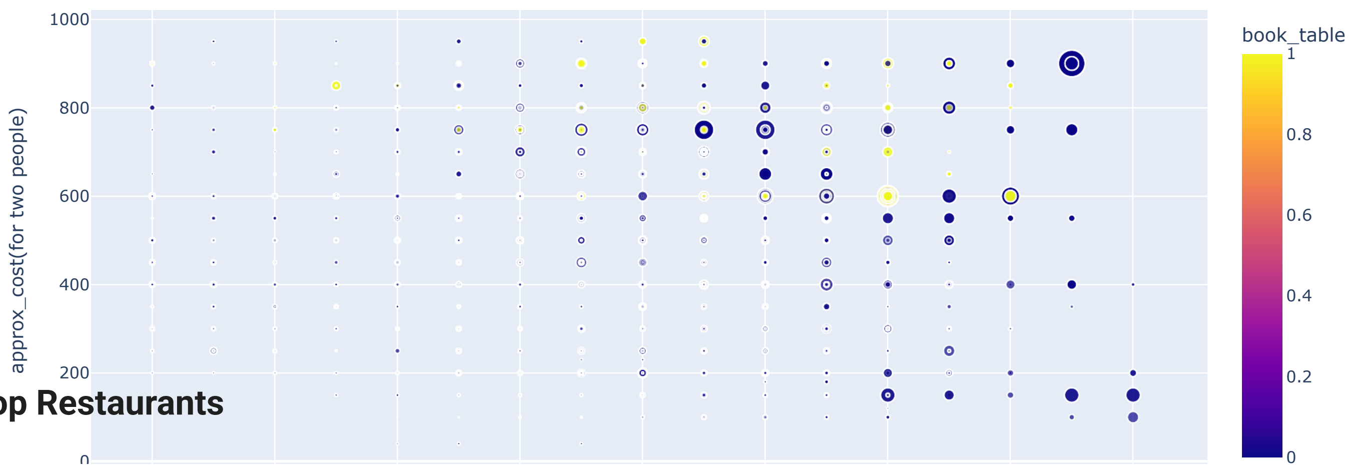
```
import pandas as pd
import plotly.express as px
import plotly.io as pio

pio.renderers.default = 'colab'

df['rate'] = df['rate'].astype(str).str.extract(r'(\d+\.?\d*)')

df['votes'] = pd.to_numeric(df['votes'], errors='coerce')
df['approx_cost(for two people)'] = df['approx_cost(for two people)'].astype(str).str.replace(',', '')
df['approx_cost(for two people)'] = pd.to_numeric(df['approx_cost(for two people)'], errors='coerce')
df_clean = df.dropna(subset=['rate', 'votes', 'approx_cost(for two people)', 'online_order']).copy()
df_clean.loc[:, 'online_order'] = df_clean['online_order'].astype(str)

if not df_clean.empty:
    fig = px.scatter(
        df_clean,
        x='rate',
        y='votes',
        size='approx_cost(for two people)',
        color='online_order',
        hover_data=['name', 'location'],
        title='Votes vs Rating (Bubble = Cost for Two People)'
    )
    fig.show()
else:
    print("⚠️ No valid data to plot after cleaning.")
```



e. Interactive: Cost vs Rating

```
fig = px.scatter(df, x='rate', y='approx_cost(for two people)',
                 size='votes', color='book_table',
                 hover_data=['name', 'location', 'cuisines'],
                 title='Cost vs Rating (Bubble Plot)')
fig.show()
```

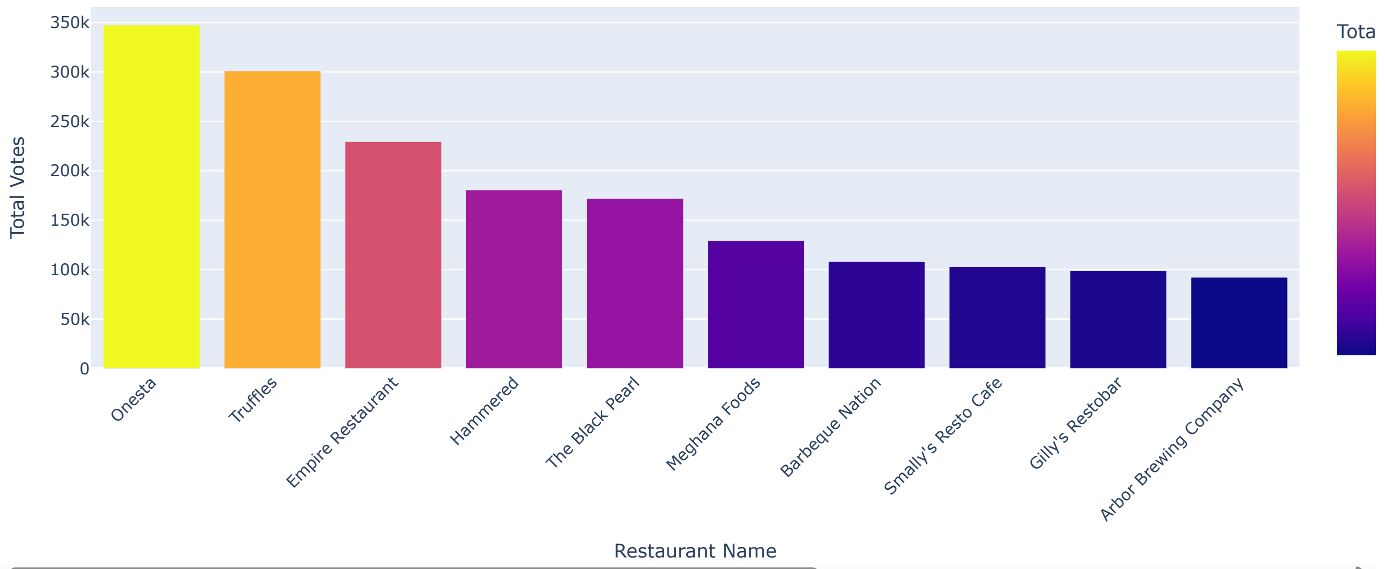
Cost vs Rating (Bubble Plot)



Top Restaurants

```
import plotly.express as px

top_restaurants = df.groupby('name')['votes'].sum().sort_values(ascending=False).head(10).reset_index()
fig = px.bar(
    top_restaurants,
    x='name',
    y='votes',
    title='Top 10 Restaurants by Votes',
    labels={'name': 'Restaurant Name', 'votes': 'Total Votes'},
    color='votes'
)
fig.update_layout(xaxis_tickangle=-45)
fig.show()
```

Top 10 Restaurants by Votes



Data Insights Summary

Rating Distribution: Most restaurants are rated between 3.0 and 4.5, showing generally positive customer reviews.