In [1]:
```python
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
```

In [4]:
```python
df = pd.read_csv("Datasets/zomato.csv")
```

In [6]:
```python
df.head()
```

Out[6]:

	Unnamed: 0	restaurant name	restaurant type	rate (out of 5)	num of ratings	avg cost (two people)	online_order	table booking	cuisines type	U
0	0	#FeelTheROLL	Quick Bites	3.4	7	200	No	No	Fast Food	
1	1	#L-81 Cafe	Quick Bites	3.9	48	400	Yes	No	Fast Food, Beverages	
2	2	#refuel	Cafe	3.7	37	400	Yes	No	Cafe, Beverages	
3	3	'@ Biryani Central	Casual Dining	2.7	135	550	Yes	No	Biryani, Mughlai, Chinese	
4	4	'@ The Bbq	Casual Dining	2.8	40	700	Yes	No	BBQ, Continental, North Indian, Chinese, Bever...	

In [10]:
```python
df.drop(columns=['Unnamed: 0','Unnamed: 9'],inplace=True)
```

In [11]:
```python
df.head()
```

Out[11]:

	restaurant name	restaurant type	rate (out of 5)	num of ratings	avg cost (two people)	online_order	table booking	cuisines type	
0	#FeelTheROLL	Quick Bites	3.4	7	200	No	No	Fast Food	
1	#L-81 Cafe	Quick Bites	3.9	48	400	Yes	No	Fast Food, Beverages	Byresandra,Tav
2	#refuel	Cafe	3.7	37	400	Yes	No	Cafe, Beverages	Ba
3	'@ Biryani Central	Casual Dining	2.7	135	550	Yes	No	Biryani, Mughlai, Chinese	
4	'@ The Bbq	Casual Dining	2.8	40	700	Yes	No	BBQ, Continental, North Indian, Chinese, Bever...	

In [13]:

```
df.columns
```

Out[13]:

```
Index(['restaurant name', 'restaurant type', 'rate (out of 5)',
       'num of ratings', 'avg cost (two people)', 'online_order',
       'table booking', 'cuisines type', 'area', 'local address'],
      dtype='object')
```

In [14]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7105 entries, 0 to 7104
Data columns (total 10 columns):
 #   Column                 Non-Null Count  Dtype  
---  ------                 --------------  -----  
 0   restaurant name        7105 non-null   object 
 1   restaurant type        7105 non-null   object 
 2   rate (out of 5)        7105 non-null   float64
 3   num of ratings         7105 non-null   int64  
 4   avg cost (two people)  7105 non-null   int64  
 5   online_order           7105 non-null   object 
 6   table booking          7105 non-null   object 
 7   cuisines type          7105 non-null   object 
 8   area                   7105 non-null   object 
 9   local address          7105 non-null   object 
dtypes: float64(1), int64(2), object(7)
memory usage: 555.2+ KB
```

In [16]:

```
df.isnull().sum()
```

Out[16]:
```
restaurant name          0
restaurant type          0
rate (out of 5)          0
num of ratings           0
avg cost (two people)    0
online_order             0
table booking            0
cuisines type            0
area                     0
local address            0
dtype: int64
```

In [17]:
```
df.describe()
```

Out[17]:

	rate (out of 5)	num of ratings	avg cost (two people)
count	7105.000000	7105.000000	7105.000000
mean	3.480619	188.921042	535.952006
std	0.574133	592.171049	463.554352
min	0.000000	1.000000	0.000000
25%	3.200000	16.000000	300.000000
50%	3.500000	40.000000	400.000000
75%	3.800000	128.000000	600.000000
max	4.900000	16345.000000	6000.000000

In [18]:
```
df['online_order'] = df['online_order'].map({'Yes': 1, 'No': 0})
df['table booking'] = df['table booking'].map({'Yes': 1, 'No': 0})
```

In [19]:
```
df.head()
```

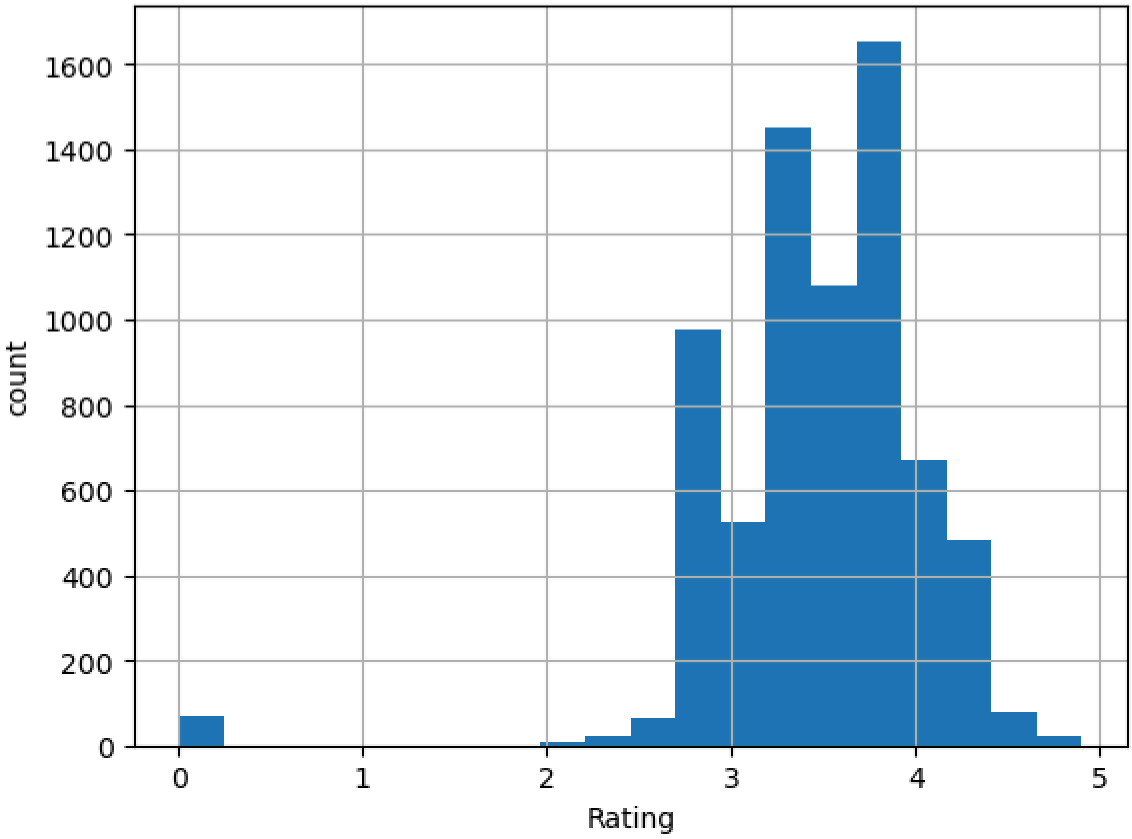
Out[19]:

	restaurant name	restaurant type	rate (out of 5)	num of ratings	avg cost (two people)	online_order	table booking	cuisines type	
0	#FeelTheROLL	Quick Bites	3.4	7	200	0	0	Fast Food	
1	#L-81 Cafe	Quick Bites	3.9	48	400	1	0	Fast Food, Beverages	Byresandra,Tav
2	#refuel	Cafe	3.7	37	400	1	0	Cafe, Beverages	Ba
3	'@ Biryani Central	Casual Dining	2.7	135	550	1	0	Biryani, Mughlai, Chinese	
4	'@ The Bbq	Casual Dining	2.8	40	700	1	0	BBQ, Continental, North Indian, Chinese, Bever...	

In [23]:
```
df['rate (out of 5)'].hist(bins=20)
plt.xlabel('Rating')
```

```
plt.ylabel('count')
```

Out[23]: Text(0, 0.5, 'count')



In [1]:
```
###“Most Zomato restaurants are rated between 3 and 4, indicating generally satisfactory dinin
###while extreme low or high ratings are relatively rare.”

###small bar near 0 may represent:
##unrated restaurants
###placeholder or incorrect values (can be removed during cleaning if needed)
```

In [25]:
```
df.shape
```

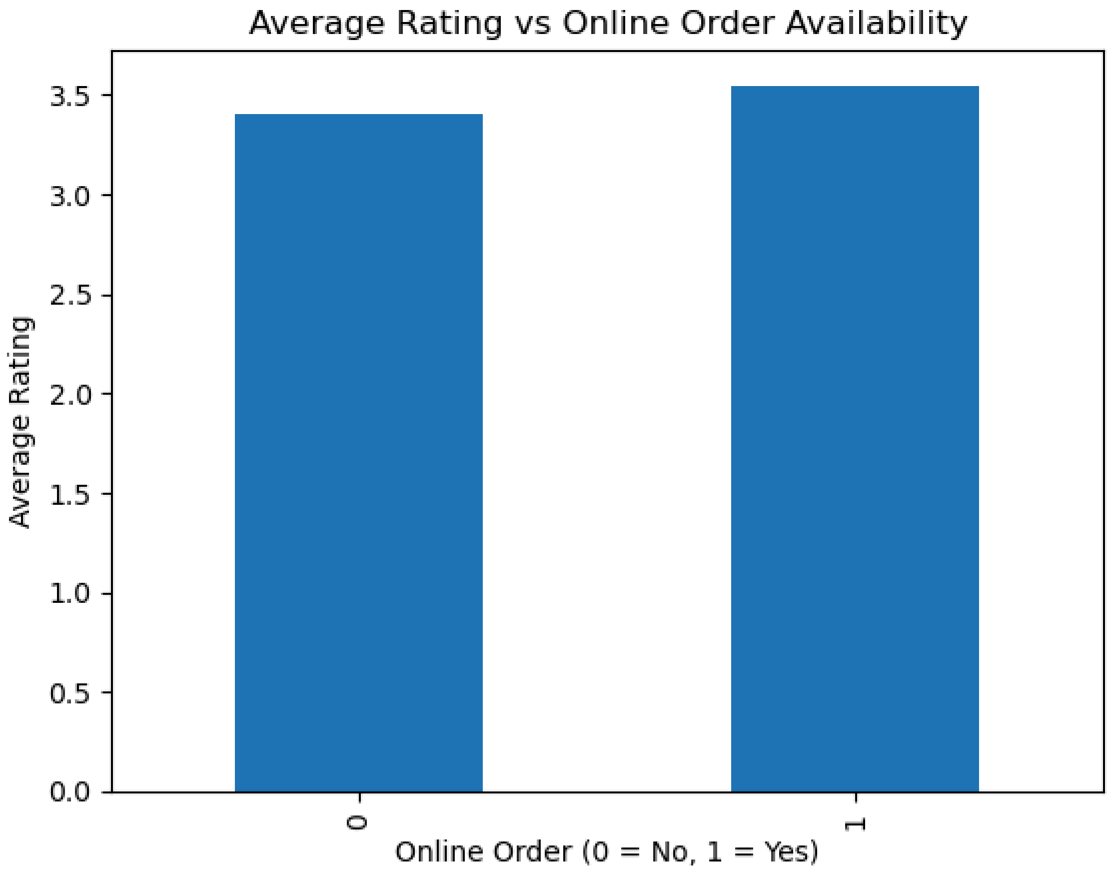
Out[25]: (7105, 10)

In [26]:
```
df.groupby('online_order')['rate (out of 5)'].mean()
```

Out[26]:
```
online_order
0    3.408200
1    3.546257
Name: rate (out of 5), dtype: float64
```

In [ ]:
```
#Online Order = 0 (Not Available):  ⭐ 3.41

#Online Order = 1 (Available):  ⭐ 3.55
```

In [29]:
```
df.groupby('online_order')['rate (out of 5)'].mean().plot(kind='bar')
plt.xlabel('Online Order (0 = No, 1 = Yes)')
plt.ylabel('Average Rating')
plt.title('Average Rating vs Online Order Availability')
plt.show()
```



In [ ]:
```
df['main_cuisine'] = df['cuisines type'].str.split(',').str[0]
```

In [39]:
```
df['main_cuisine'].value_counts().head(10)
```

Out[39]:
```
main_cuisine
North Indian    1929
South Indian     826
Chinese          451
Cafe             449
Biryani          406
Fast Food        386
Continental      251
Bakery           229
Desserts         207
Andhra           191
Name: count, dtype: int64
```

In [3]:
```
## 🍽️ Cuisine Popularity Analysis

###This section analyzes the **most common cuisines** listed in the Zomato restaurant dataset

### 📊 Top 10 Most Popular Cuisines
```

In [30]:
```
df.groupby('table booking')['rate (out of 5)'].mean()
```

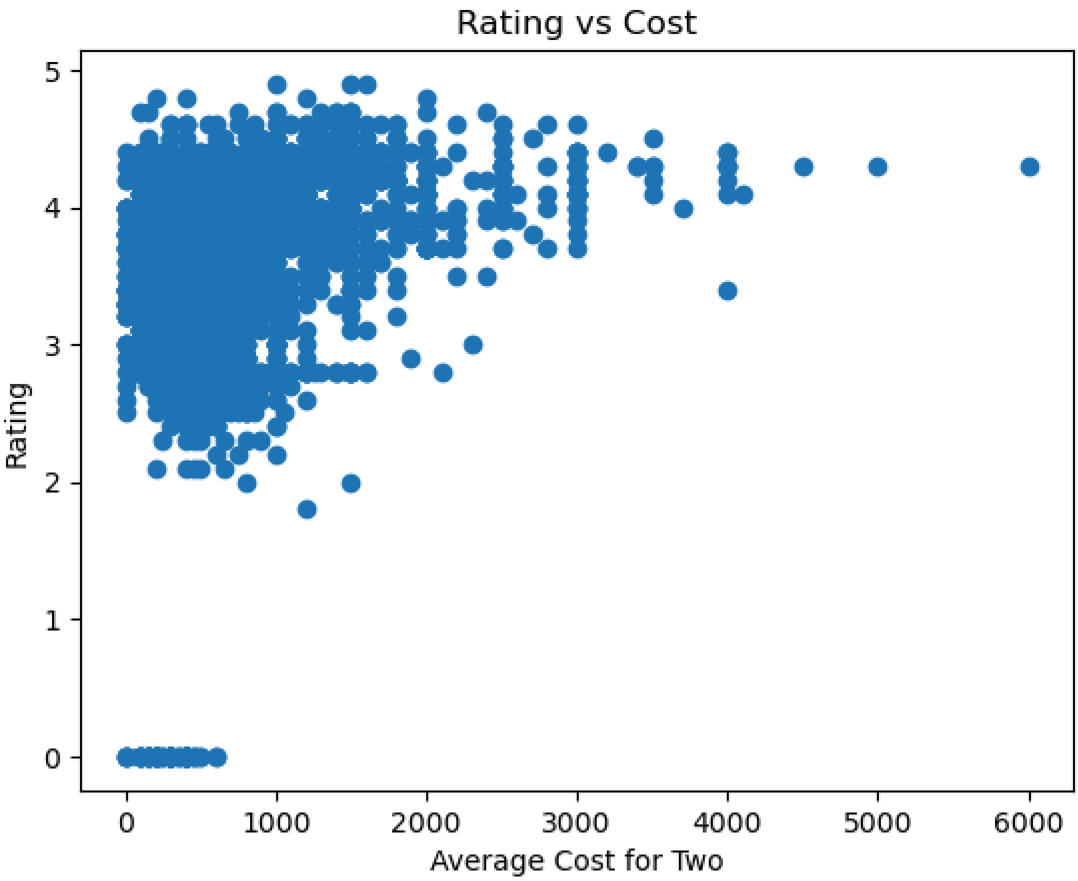
Out[30]:
```
table booking
0    3.415658
1    4.036022
Name: rate (out of 5), dtype: float64
```

In [32]:
```
df['table booking'].value_counts()
```

Out[32]:
```
table booking
0    6361
1     744
Name: count, dtype: int64
```

In [33]:

```
plt.scatter(df['avg cost (two people)'], df['rate (out of 5)'])
plt.xlabel('Average Cost for Two')
plt.ylabel('Rating')
plt.title('Rating vs Cost')
plt.show()
```

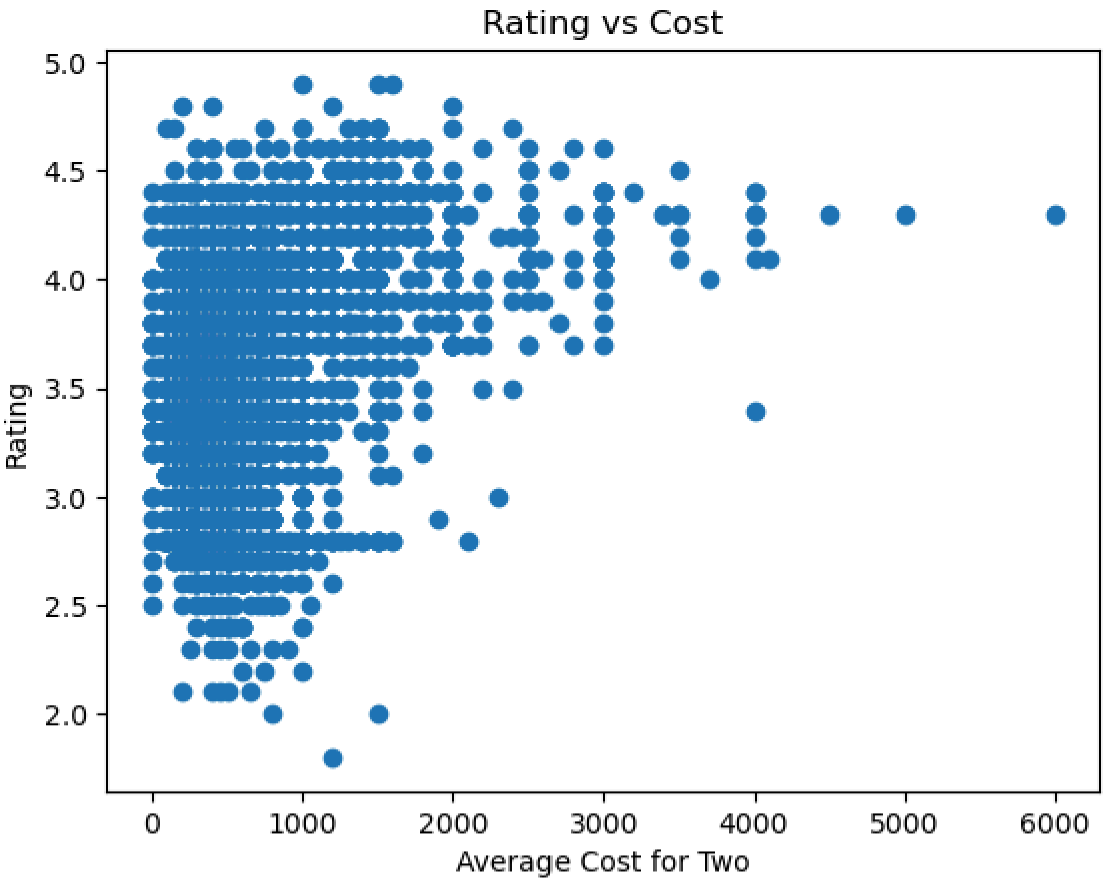


In [34]:

```
df = df[df['rate (out of 5)'] > 0]
```

In [36]:

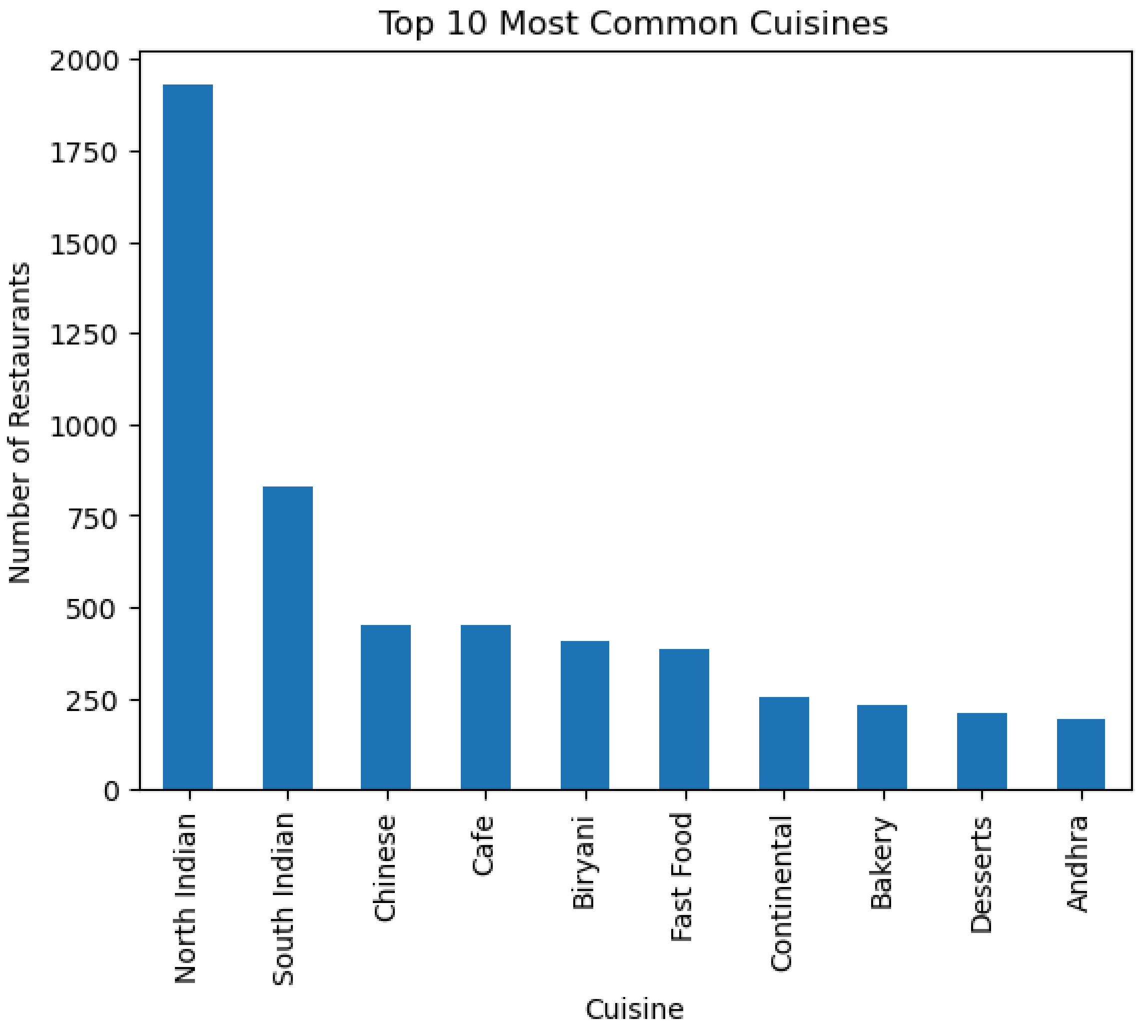
```
plt.scatter(df['avg cost (two people)'], df['rate (out of 5)'])
plt.xlabel('Average Cost for Two')
plt.ylabel('Rating')
plt.title('Rating vs Cost')
plt.show()
```



In [41]:
```
df.groupby('main_cuisine')['rate (out of 5)'].mean().sort_values(ascending=False).head(10)
```

Out[41]:
```
main_cuisine
African           4.500000
Malaysian         4.450000
Sushi             4.400000
Parsi             4.400000
Middle Eastern    4.400000
Singaporean       4.300000
German            4.300000
French            4.266667
Vietnamese        4.260000
Modern Indian     4.246154
Name: rate (out of 5), dtype: float64
```

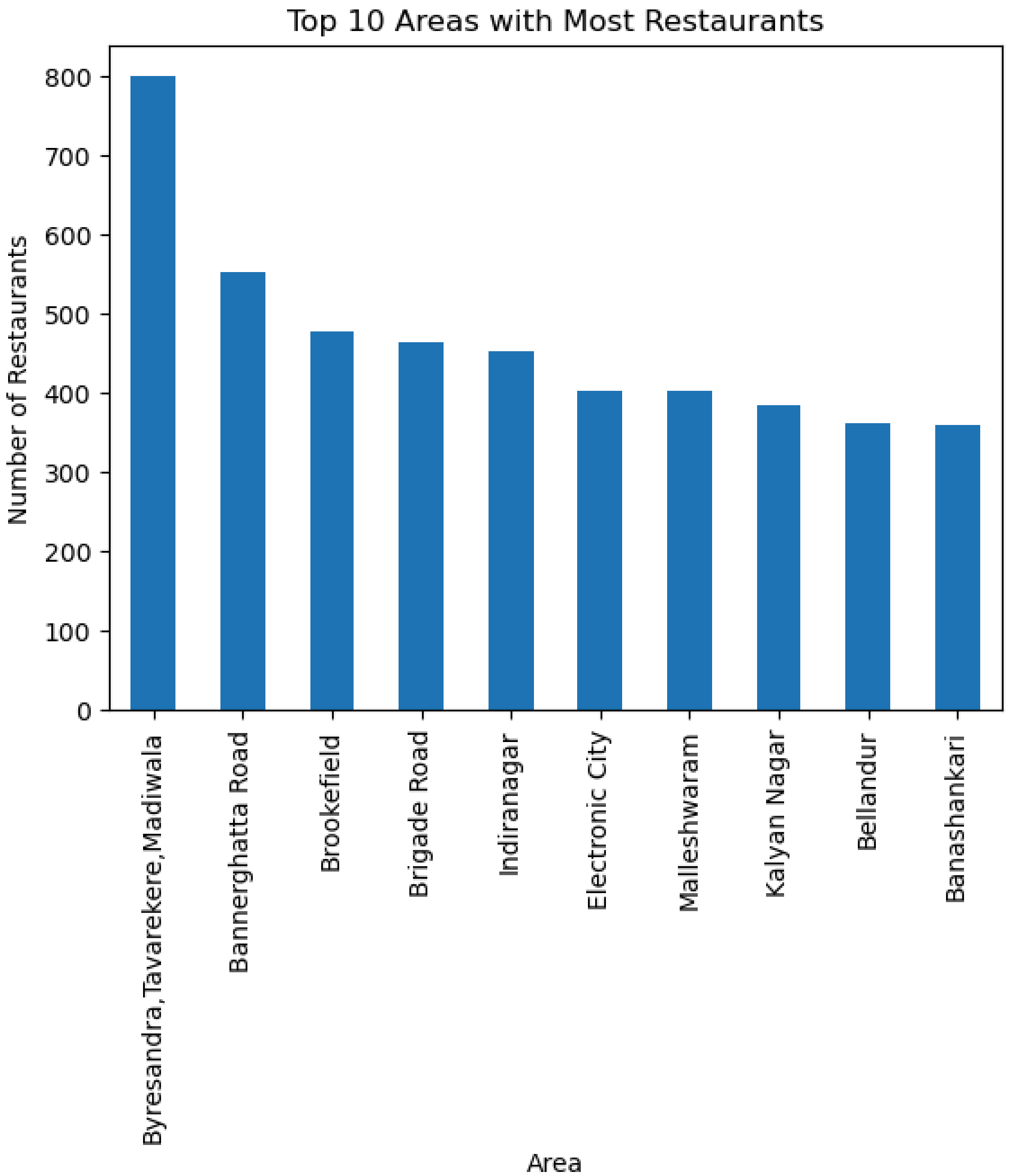
In [42]:
```
df['main_cuisine'].value_counts().head(10).plot(kind='bar')
plt.xlabel('Cuisine')
plt.ylabel('Number of Restaurants')
plt.title('Top 10 Most Common Cuisines')
plt.show()
```



In [43]:
```
df['area'].value_counts().head(10)
```

Out[43]:
```
area
Byresandra,Tavarekere,Madiwala    798
Bannerghatta Road                 552
Brookefield                       477
Brigade Road                      464
Indiranagar                       452
Electronic City                   403
Malleshwaram                      402
Kalyan Nagar                      384
Bellandur                         361
Banashankari                      359
Name: count, dtype: int64
```

In [44]:
```
df['area'].value_counts().head(10).plot(kind='bar')
plt.xlabel('Area')
plt.ylabel('Number of Restaurants')
plt.title('Top 10 Areas with Most Restaurants')
plt.show()
```



In [45]:
```
df.groupby('area')['rate (out of 5)'].mean().sort_values(ascending=False).head(10)
```

Out[45]:
```
area
Brigade Road                      3.684267
Lavelle Road                      3.630496
Malleshwaram                      3.629602
Church Street                     3.606494
Indiranagar                       3.592920
Byresandra,Tavarekere,Madiwala    3.583960
Banashankari                      3.568802
Koramangala 6th Block             3.566667
MG Road                           3.540625
Koramangala 7th Block             3.530556
Name: rate (out of 5), dtype: float64
```

In [46]:
```
df.groupby('area')['avg cost (two people)'].mean().sort_values(ascending=False).head(10)
```

Out[46]:
```
area
Lavelle Road            857.801418
Brigade Road            835.107759
Church Street           740.259740
Whitefield              694.444444
Indiranagar             668.893805
Old Airport Road        587.640449
Malleshwaram            576.666667
Koramangala 4th Block   561.666667
MG Road                 543.750000
Bellandur               542.742382
Name: avg cost (two people), dtype: float64
```

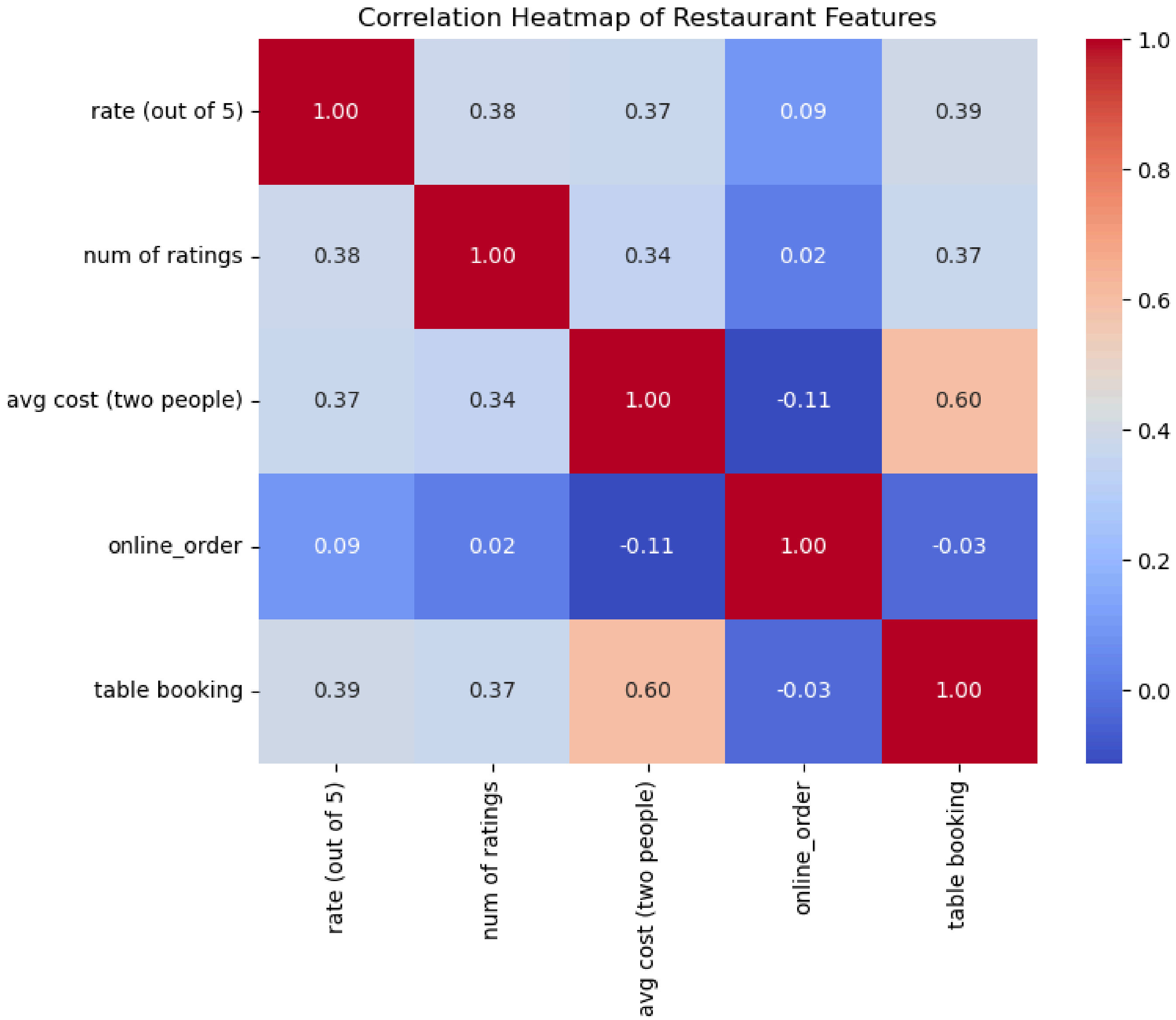
In [47]:
```
num_df = df[['rate (out of 5)', 'num of ratings', 'avg cost (two people)',
             'online_order', 'table booking']]
```

In [48]:
```
corr = num_df.corr()
corr
```

Out[48]:

	rate (out of 5)	num of ratings	avg cost (two people)	online_order	table booking
rate (out of 5)	1.000000	0.380258	0.372906	0.086265	0.387304
num of ratings	0.380258	1.000000	0.338897	0.024637	0.369643
avg cost (two people)	0.372906	0.338897	1.000000	-0.114089	0.603905
online_order	0.086265	0.024637	-0.114089	1.000000	-0.029080
table booking	0.387304	0.369643	0.603905	-0.029080	1.000000

In [49]:
```
import seaborn as sns
import matplotlib.pyplot as plt

plt.figure(figsize=(8,6))
sns.heatmap(corr, annot=True, cmap='coolwarm', fmt='.2f')
plt.title('Correlation Heatmap of Restaurant Features')
plt.show()
```



In [9]:
```
## 🔥 Correlation Heatmap Analysis

#The correlation heatmap represents the linear relationships between numerical restaurant fea

### 📊 Key Insights
 #**Table booking and ratings** show a moderate positive correlation, indicating better dinin
  #**Number of ratings** has a positive relationship with ratings, suggesting trust and visib
#**Average cost** has only a weak influence on ratings.
 #**Online ordering** shows minimal direct impact on ratings.
 #Average cost and table booking** are strongly correlated, representing premium restaurants.

### 🧠 Conclusion
#Service quality indicators such as table booking influence restaurant ratings more than pric
```

In [50]:
```
X = df[['avg cost (two people)', 'num of ratings', 'online_order', 'table booking']]
y = df['rate (out of 5)']
```

In [51]:
```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.3, random_state=42
)
```

In [52]:
```
from sklearn.linear_model import LinearRegression

lr = LinearRegression()
lr.fit(X_train, y_train)
```

Out[52]:

▾ LinearRegression

▶ Parameters

In [53]:

```
y_pred = lr.predict(X_test)
```

In [55]:

```
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score

mae = mean_absolute_error(y_test, y_pred)
rmse = np.sqrt(mean_squared_error(y_test, y_pred))
r2 = r2_score(y_test, y_pred)

print("MAE:", mae)
print("RMSE:", rmse)
print("R2 Score:", r2)
```

```
MAE: 0.3216606182087909
RMSE: 0.4126356805238793
R2 Score: 0.1768664178500432
```

In [56]:

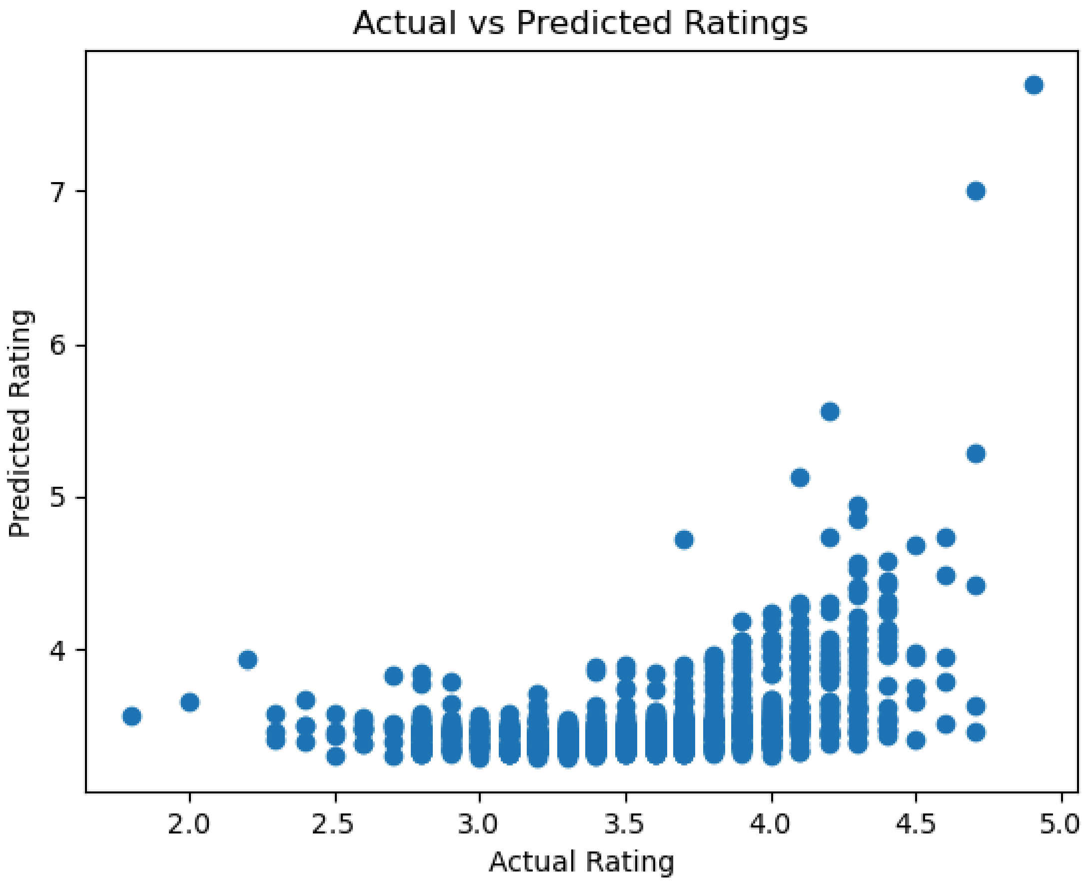
```
feature_importance = pd.DataFrame({
    'Feature': X.columns,
    'Coefficient': lr.coef_
})

feature_importance
```

Out[56]:

	Feature	Coefficient
0	avg cost (two people)	0.000183
1	num of ratings	0.000230
2	online_order	0.094668
3	table booking	0.267586

In [57]:

```
plt.scatter(y_test, y_pred)
plt.xlabel('Actual Rating')
plt.ylabel('Predicted Rating')
plt.title('Actual vs Predicted Ratings')
plt.show()
```



In [11]:
```
## 📈 Linear Regression Coefficient Analysis

#The table below shows the coefficients of the Linear Regression model, indicating
#the impact of each feature on restaurant ratings.

### 🔢 Key Interpretations
# **Table booking** has the strongest positive impact on ratings.
 #**Online ordering** moderately improves ratings.
# **Number of ratings** has a small positive influence.
# **Average cost** has negligible direct effect on ratings.

### 🧠 Conclusion
#Service-related features contribute more to restaurant ratings than pricing,
#which explains the limited performance of a linear model.
```

In [ ]: