In [2]:
```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
```

In [3]:
```
df=pd.read_csv('Amazon _Customer_Survey.csv')
```

In [4]:
```
df
```

Out[4]:

	Customer_ID	age	Gender	Purchase_Frequency	Purchase_Categories	Customer_Reviews_Importance	Review_Left	Review_Reliability	Review_Helpfulness	Rating_Accuracy	Shopping_Satisfaction	Service_Appreciation
0	1	23	Female	Few times a month	Beauty and Personal Care	1	Yes	Occasionally	Yes	1	1	Competitive prices
1	2	23	Female	Once a month	Clothing and Fashion	1	No	Heavily	Yes	3	2	Wide product selection
2	3	24	Prefer not to say	Few times a month	Groceries and Gourmet Food,Clothing and Fashion	2	No	Occasionally	No	3	3	Competitive prices
3	4	24	Female	Once a month	Beauty and Personal Care,Clothing and Fashion;...	5	Yes	Heavily	Yes	3	4	Competitive prices
4	5	22	Female	Less than once a month	Beauty and Personal Care,Clothing and Fashion	1	No	Heavily	Yes	2	2	Competitive prices
...	...	...	...	...	...	...	...	...	...	...	...	...
597	598	23	Female	Once a week	Beauty and Personal Care	4	Yes	Moderately	Sometimes	3	4	Competitive prices
598	599	23	Female	Once a week	Clothing and Fashion	3	Yes	Heavily	Sometimes	3	3	Product recommendations
599	600	23	Female	Once a month	Beauty and Personal Care	3	Yes	Occasionally	Sometimes	2	3	Wide product selection
600	601	23	Female	Few times a month	Beauty and Personal Care,Clothing and Fashion;...	1	No	Heavily	Yes	2	2	Wide product selection
601	602	23	Female	Once a week	Clothing and Fashion	3	Yes	Moderately	Sometimes	3	3	Product recommendations

602 rows × 12 columns

In [5]:
```
# define bin from 18 to 25, 26 to 35, 36 to 45 and 45+
# The last bin includes ages 46 and above
bins = [18, 25, 35, 45, float('inf')]
age_group = ['young','adults','aged adults','seniors']
df['age_group']=pd.cut(df["age"],bins,labels = age_group)
```

In [6]:
```
#displays only age and age group column
df[['age','age_group']]
```

Out[6]:

	age	age_group
0	23	young
1	23	young
2	24	young
3	24	young
4	22	young
...	...	...
597	23	young
598	23	young
599	23	young
600	23	young
601	23	young

602 rows × 2 columns

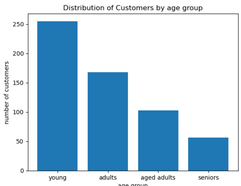
In [7]:
```
#displays only senior customer
#boolean indexing
df.loc[df['age_group'] == 'seniors'].head()
```

Out[7]:

	Customer_ID	age	Gender	Purchase_Frequency	Purchase_Categories	Customer_Reviews_Importance	Review_Left	Review_Reliability	Review_Helpfulness	Rating_Accuracy	Shopping_Satisfaction	Service_Appreciation	age_g
20	21	64	Male	Once a week	Groceries and Gourmet Food	3	No	Occasionally	Yes	1	2	Competitive prices	se
78	79	47	Prefer not to say	Less than once a month	others	2	No	Never	Sometimes	2	2	User-friendly website/app interface	se
79	80	54	Others	Multiple times a week	Groceries and Gourmet Food,Beauty and Personal...	4	Yes	Heavily	Yes	5	5	Competitive prices	se
80	81	58	Male	Once a month	Clothing and Fashion;others	5	No	Never	No	3	3	Competitive prices	se
81	82	53	Female	Less than once a month	Home and Kitchen	3	No	Moderately	Yes	2	1	Competitive prices	se

In [8]:
```
age_group_counts= df['age_group'].value_counts()
```

In [9]:
```
plt.bar(age_group_counts.index, age_group_counts.values)
plt.xlabel('age group')
plt.ylabel('number of customers')
plt.title('Distribution of Customers by age group')
plt.show()
```



In [10]:
```
#converting categories into numerical values
from sklearn.preprocessing import LabelEncoder
label_encoder = LabelEncoder()
df['gender_encoded'] = label_encoder.fit_transform(df['Gender'])
df['Review_Left_encoded'] = label_encoder.fit_transform(df['Review_Left'])
df['Purchase_Frequency_encoded'] = label_encoder.fit_transform(df['Purchase_Frequency'])
df
```

Out[10]:

	Customer_ID	age	Gender	Purchase_Frequency	Purchase_Categories	Customer_Reviews_Importance	Review_Left	Review_Reliability	Review_Helpfulness	Rating_Accuracy	Shopping_Satisfaction	Service_Appreciation	age_
0	1	23	Female	Few times a month	Beauty and Personal Care	1	Yes	Occasionally	Yes	1	1	Competitive prices	
1	2	23	Female	Once a month	Clothing and Fashion	1	No	Heavily	Yes	3	2	Wide product selection	
2	3	24	Prefer not to say	Few times a month	Groceries and Gourmet Food,Clothing and Fashion	2	No	Occasionally	No	3	3	Competitive prices	
3	4	24	Female	Once a month	Beauty and Personal Care,Clothing and Fashion;...	5	Yes	Heavily	Yes	3	4	Competitive prices	
4	5	22	Female	Less than once a month	Beauty and Personal Care,Clothing and Fashion	1	No	Heavily	Yes	2	2	Competitive prices	
...	...	...	...	...	...	...	...	...	...	...	...	...	
597	598	23	Female	Once a week	Beauty and Personal Care	4	Yes	Moderately	Sometimes	3	4	Competitive prices	
598	599	23	Female	Once a week	Clothing and Fashion	3	Yes	Heavily	Sometimes	3	3	Product recommendations	
599	600	23	Female	Once a month	Beauty and Personal Care	3	Yes	Occasionally	Sometimes	2	3	Wide product selection	
600	601	23	Female	Few times a month	Beauty and Personal Care,Clothing and Fashion;...	1	No	Heavily	Yes	2	2	Wide product selection	
601	602	23	Female	Once a week	Clothing and Fashion	3	Yes	Moderately	Sometimes	3	3	Product recommendations	

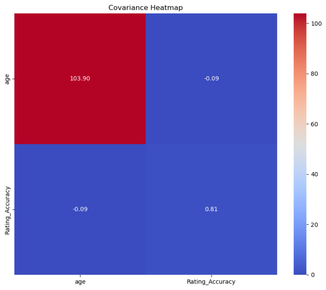
602 rows × 16 columns

In [11]:
```
select_column = ['age','Rating_Accuracy']
covariance_matrix = df[select_column].cov()
covariance_matrix
```

Out[11]:

	age	Rating_Accuracy
age	103.903875	-0.093565
Rating_Accuracy	-0.093565	0.809539

In [12]:
```
import seaborn as sns
plt.figure(figsize=(10, 8))
sns.heatmap(covariance_matrix, annot=True, cmap='coolwarm', fmt='.2f')
plt.title('Covariance Heatmap')
plt.show()
```



In [13]:
```
#check independent or not using chi-square test
from scipy.stats import chi2_contingency
contingency_table = pd.crosstab(df['Gender'], df['Purchase_Frequency'])
contingency_table
```

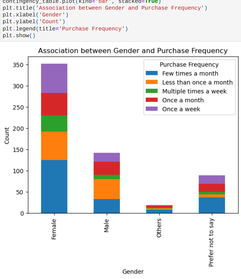
Out[13]:

Purchase_Frequency	Few times a month	Less than once a month	Multiple times a week	Once a month	Once a week
Gender					
Female	125	67	38	53	69
Male	33	47	10	31	21
Others	8	3	2	4	2
Prefer not to say	37	7	6	19	20

In [14]:
```
chi2, p, dof, expected = chi2_contingency(contingency_table)
print("Chi-Square Value:", chi2)
print("P-value:", p)
print("Degrees of Freedom:", dof)
print("Expected Frequencies Table:\n", expected)
```

```
Chi-Square Value: 33.888000387515973
P-value: 0.0007025219073103722
Degrees of Freedom: 12
Expected Frequencies Table:
 [[118.69787442  72.56498339  32.74418605  62.56478405  65.48837209]
 [ 47.88372093  29.24916944  13.20930233  25.23020266  26.41860465]
 [  6.40697674   3.91362126   1.76744186   3.37707641   3.53488372]
 [ 30.01162791  18.32222591   8.27906977  15.81893688  16.55813953]]
```

In [15]:
```
contingency_table.plot(kind='bar', stacked=True)
plt.title('Association between Gender and Purchase Frequency')
plt.xlabel('Gender')
plt.ylabel('Count')
plt.legend(title='Purchase Frequency')
plt.show()
```



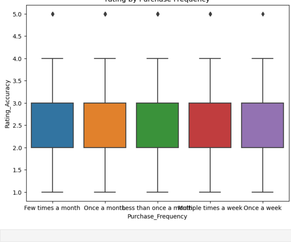
In [16]:
```
#independent t-test
from scipy.stats import ttest_ind
product_rating_yes = df[df['Purchase_Frequency'] == 'Yes']['Rating_Accuracy']
product_rating_no = df[df['Purchase_Frequency'] == 'No']['Rating_Accuracy']
t_statistic, p_value = ttest_ind(product_rating_yes, product_rating_no)

print("T-Statistic:", t_statistic)
print("P-Value:", p_value)

if p_value < 0.05:
    print("The difference is statistically significant.")
else:
    print("There is no statistically significant difference.")
```

```
T-Statistic: nan
P-Value: nan
There is no statistically significant difference.
```

In [17]:
```
#boxplot
plt.figure(figsize=(8,6))
sns.boxplot(x=df['Purchase_Frequency'],y=df['Rating_Accuracy'])
plt.title('rating by Purchase Frequency')
plt.xlabel('Purchase_Frequency')
plt.ylabel('Rating_Accuracy')
plt.show()
```



In [ ]:
```
```