In [7]:
```python
import pandas as pd
import matplotlib.pyplot as plt
customers=pd.read_csv(r"F:\Downloads\Customers.csv")
products=pd.read_csv(r"F:\Downloads\Products.csv")
transactions=pd.read_csv(r"F:\Downloads\Transactions.csv")
```

In [9]:
```python
print("Customers Dataset Overview:")
print(customers.head(), customers.info(), customers.describe())

print("Products Dataset Overview:")
print(products.head(), products.info(), products.describe())

print("Transactions Dataset Overview:")
print(transactions.head(), transactions.info(), transactions.describe())
```

```
Customers Dataset Overview:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 4 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   CustomerID    200 non-null    object
 1   CustomerName  200 non-null    object
 2   Region        200 non-null    object
 3   SignupDate    200 non-null    object
dtypes: object(4)
memory usage: 6.4+ KB
  CustomerID        CustomerName         Region  SignupDate
0      C0001    Lawrence Carroll  South America  2022-07-10
1      C0002      Elizabeth Lutz           Asia  2022-02-13
2      C0003      Michael Rivera  South America  2024-03-07
3      C0004  Kathleen Rodriguez  South America  2022-10-09
4      C0005         Laura Weber           Asia  2022-08-15 None        CustomerID      CustomerName
Region  SignupDate
count         200               200            200         200
unique        200               200              4         179
top         C0001  Lawrence Carroll  South America  2024-11-11
freq            1                 1             59           3
Products Dataset Overview:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100 entries, 0 to 99
Data columns (total 4 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   ProductID    100 non-null    object 
 1   ProductName  100 non-null    object 
 2   Category     100 non-null    object 
 3   Price        100 non-null    float64
dtypes: float64(1), object(3)
memory usage: 3.3+ KB
  ProductID              ProductName     Category   Price
0      P001     ActiveWear Biography        Books  169.30
1      P002    ActiveWear Smartwatch  Electronics  346.30
2      P003  ComfortLiving Biography        Books   44.12
3      P004            BookWorld Rug   Home Decor   95.69
4      P005          TechPro T-Shirt     Clothing  429.31 None             Price
count  100.000000
mean   267.551700
std    143.219383
min     16.080000
25%    147.767500
50%    292.875000
75%    397.090000
max    497.760000
Transactions Dataset Overview:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 7 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   TransactionID    1000 non-null   object 
 1   CustomerID       1000 non-null   object 
 2   ProductID        1000 non-null   object 
 3   TransactionDate  1000 non-null   object 
 4   Quantity         1000 non-null   int64  
 5   TotalValue       1000 non-null   float64
 6   Price            1000 non-null   float64
dtypes: float64(2), int64(1), object(4)
memory usage: 54.8+ KB
  TransactionID CustomerID ProductID      TransactionDate  Quantity  \
0        T00001      C0199      P067  2024-08-25 12:38:23         1   
1        T00112      C0146      P067  2024-05-27 22:23:54         1   
2        T00166      C0127      P067  2024-04-25 07:38:55         1   
3        T00272      C0087      P067  2024-03-26 22:55:37         2   
4        T00363      C0070      P067  2024-03-21 15:10:10         3   

   TotalValue   Price  
0      300.68  300.68  
1      300.68  300.68  
2      300.68  300.68  
3      601.36  300.68  
4      902.04  300.68   None           Quantity   TotalValue       Price
count  1000.000000  1000.000000  1000.00000
mean      2.537000   689.995560   272.55407
std       1.117981   493.144478   140.73639
min       1.000000    16.080000    16.08000
25%       2.000000   295.295000   147.95000
50%       3.000000   588.880000   299.93000
75%       4.000000  1011.660000   404.40000
max       4.000000  1991.040000   497.76000
```

In [15]:
```python
data=transactions.merge(customers,on="CustomerID").merge(products,on="ProductID")
```

In [17]:
```python
# Task 2: Lookalike Model
import pandas as pd
from sklearn.metrics.pairwise import cosine_similarity
from sklearn.preprocessing import StandardScaler
customer_profiles=data.groupby('CustomerID').agg({
    'Quantity': 'sum',          # Total products purchased
    'TotalValue': 'sum',        # Total money spent
    'ProductID': 'count',       # Total transactions
    'Region': 'first',          # Region of the customer
}).reset_index()
```

In [21]:
```python
customer_profiles = pd.get_dummies(customer_profiles, columns=['Region'], drop_first=True)
```

In [23]:
```python
# Standardize features for similarity computation
scaler = StandardScaler()
scaled_features = scaler.fit_transform(customer_profiles.iloc[:, 1:])
```

In [25]:
```python
# Compute pairwise cosine similarity
similarity_matrix = cosine_similarity(scaled_features)
```

In [27]:
```python
# Map customers to their similar customers with scores
lookalike_results = {}
for idx, customer_id in enumerate(customer_profiles['CustomerID']):
    # Get similarity scores for this customer and sort them
    similar_indices = similarity_matrix[idx].argsort()[::-1]  # Sort in descending order
    similar_customers = [
        (customer_profiles['CustomerID'].iloc[i], similarity_matrix[idx][i])
        for i in similar_indices if i != idx  # Exclude the customer itself
    ]
    # Take the top 3 similar customers
    lookalike_results[customer_id] = similar_customers[:3]
```

In [29]:
```python
# Create a DataFrame for the first 20 customers (CustomerID: C0001 to C0020)
lookalike_data = {
    "CustomerID": [],
    "SimilarCustomers": []
}
```

In [31]:
```python
for customer_id, similar_customers in list(lookalike_results.items())[:20]:  # First 20 customers
    lookalike_data["CustomerID"].append(customer_id)
    lookalike_data["SimilarCustomers"].append(similar_customers)

lookalike_df = pd.DataFrame(lookalike_data)
```

In [33]:
```python
# Save results to Lookalike.csv
lookalike_df.to_csv("Lookalike.csv", index=False)

# Display the first 5 lookalike results
print(lookalike_df.head())
```

```
  CustomerID                                   SimilarCustomers
0      C0001  [(C0107, 0.9964160629333633), (C0137, 0.995700...
1      C0002  [(C0142, 0.9887986276382208), (C0177, 0.966505...
2      C0003  [(C0190, 0.9663449212719497), (C0133, 0.963972...
3      C0004  [(C0113, 0.9950141093849689), (C0102, 0.983592...
4      C0005  [(C0186, 0.9975070362104175), (C0159, 0.996987...
```

In [ ]:
```python
#Explanation of the Code:
#1.Data Preprocessing:
#*Merged Customers.csv, Products.csv, and Transactions.csv to create a single dataset.
#*Aggregated key features (e.g., total products purchased, total value spent) for each customer to cons

#Feature Engineering:
#*Encoded categorical features (Region) into numerical values using one-hot encoding.
#*Standardized features using StandardScaler for uniform scaling.

#Similarity Calculation:
```

```
#*Calculated pairwise cosine similarity between customer profiles using cosine_similarity from sklearn.

#Top 3 Recommendations:
#For each customer, identified the 3 most similar customers (excluding the customer itself) based on s
```