In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.metrics.pairwise import cosine_similarity
```

In [2]:
```
# Load the datasets
customers_df = pd.read_csv('Customers.csv')
products_df = pd.read_csv('Products.csv')
transactions_df = pd.read_csv('Transactions.csv')
```

In [5]:
```
# Convert date columns to datetime format
customers_df['SignupDate'] = pd.to_datetime(customers_df['SignupDate'])
transactions_df['TransactionDate'] = pd.to_datetime(transactions_df['TransactionDat
```

In [7]:
```
# Merge datasets
merged_df = transactions_df.merge(customers_df, on='CustomerID', how='inner')
merged_df = merged_df.merge(products_df, on='ProductID', how='inner')
merged_df
```

Out[7]:

	TransactionID	CustomerID	ProductID	TransactionDate	Quantity	TotalValue	Price_x	Custo
0	T00001	C0199	P067	2024-08-25 12:38:23	1	300.68	300.68	Andr
1	T00112	C0146	P067	2024-05-27 22:23:54	1	300.68	300.68	Britta
2	T00166	C0127	P067	2024-04-25 07:38:55	1	300.68	300.68	Kathry
3	T00272	C0087	P067	2024-03-26 22:55:37	2	601.36	300.68	Travis
4	T00363	C0070	P067	2024-03-21 15:10:10	3	902.04	300.68	Timo
...	...	...	...	...	...	...	...	
995	T00630	C0031	P093	2024-10-08 23:58:14	2	609.88	304.94	
996	T00672	C0165	P044	2024-07-28 00:09:49	4	75.28	18.82	Juan
997	T00711	C0165	P044	2024-06-11 15:51:14	4	75.28	18.82	Juan
998	T00878	C0165	P044	2024-09-24 21:15:21	3	56.46	18.82	Juan
999	T00157	C0169	P044	2024-11-09 09:07:36	2	37.64	18.82	Jeni

1000 rows × 13 columns

In [10]:
```
# ==================
# Feature Engineering
# ==================

# Calculate total spending per customer
customer_spending = merged_df.groupby('CustomerID')['TotalValue'].sum().reset_index
customer_spending.columns = ['CustomerID', 'TotalSpending']
```

In [11]:
```
# Calculate transaction count per customer
customer_transactions = merged_df.groupby('CustomerID')['TransactionID'].count().re
customer_transactions.columns = ['CustomerID', 'TransactionCount']
print(customer_transactions)
```

```
    CustomerID  TransactionCount
0        C0001                 5
1        C0002                 4
2        C0003                 4
3        C0004                 8
4        C0005                 3
..         ...               ...
194      C0196                 4
195      C0197                 3
196      C0198                 2
197      C0199                 4
198      C0200                 5

[199 rows x 2 columns]
```

In [12]:
```
# Calculate product category preferences per customer
category_preferences = merged_df.pivot_table(index='CustomerID', columns='Category'
print(category_preferences)
```

```
Category CustomerID  Books  Clothing  Electronics  Home Decor
0              C0001      1         0            3           1
1              C0002      0         2            0           2
2              C0003      0         1            1           2
3              C0004      3         0            2           3
4              C0005      0         0            2           1
..               ...    ...       ...          ...         ...
194            C0196      1         1            0           2
195            C0197      0         0            2           1
196            C0198      0         1            1           0
197            C0199      0         0            2           2
198            C0200      1         2            1           1

[199 rows x 5 columns]
```

In [13]:
```
# Merge all features into a single dataset
customer_features = customers_df.merge(customer_spending, on='CustomerID', how='le
customer_features = customer_features.merge(customer_transactions, on='CustomerID',
customer_features = customer_features.merge(category_preferences, on='CustomerID',
customer_features
```

Out[13]:

	CustomerID	CustomerName	Region	SignupDate	TotalSpending	TransactionCount	Books
0	C0001	Lawrence Carroll	South America	2022-07-10	3354.52	5.0	1.0
1	C0002	Elizabeth Lutz	Asia	2022-02-13	1862.74	4.0	0.0
2	C0003	Michael Rivera	South America	2024-03-07	2725.38	4.0	0.0
3	C0004	Kathleen Rodriguez	South America	2022-10-09	5354.88	8.0	3.0
4	C0005	Laura Weber	Asia	2022-08-15	2034.24	3.0	0.0
...	...	...	...	...	...	...	...
195	C0196	Laura Watts	Europe	2022-06-07	4982.88	4.0	1.0
196	C0197	Christina Harvey	Europe	2023-03-21	1928.65	3.0	0.0
197	C0198	Rebecca Ray	Europe	2022-02-27	931.83	2.0	0.0
198	C0199	Andrea Jenkins	Europe	2022-12-03	1979.28	4.0	0.0
199	C0200	Kelly Cross	Asia	2023-06-11	4758.60	5.0	1.0

200 rows × 10 columns

In [14]:
```
# Fill missing values with 0
customer_features.fillna(0, inplace=True)
```

In [15]:
```
# Normalize numerical features
from sklearn.preprocessing import StandardScaler
from sklearn.metrics.pairwise import cosine_similarity


scaler = StandardScaler()
numerical_features = ['TotalSpending', 'TransactionCount'] + list(category_preferer
customer_features[numerical_features] = scaler.fit_transform(customer_features[nume
```

In [16]:
```
# Similarity Calculation
# ----------------------
# Calculate cosine similarity between all customers
similarity_matrix = cosine_similarity(customer_features[numerical_features])
similarity_df = pd.DataFrame(similarity_matrix, index=customer_features['CustomerID
```

In [17]:
```
# Find top 3 lookalikes for each customer
lookalike_map = {}
for customer_id in customer_features['CustomerID']:
    similar_customers = similarity_df[customer_id].sort_values(ascending=False).ilo
    lookalike_map[customer_id] = [(sim_id, round(score, 4)) for sim_id, score in si
```

In [18]:
```
# Prepare Lookalike.csv
lookalike_output = []
for cust_id, lookalikes in lookalike_map.items():
    lookalike_output.append({'CustomerID': cust_id, 'Lookalikes': lookalikes})
```

In [19]:
```
lookalike_df = pd.DataFrame(lookalike_output)
lookalike_df.to_csv('Lookalike.csv', index=False)
```

In [20]:
```
# Output for Customers C0001 - C0020
subset_lookalikes = lookalike_df[lookalike_df['CustomerID'].isin([f'C{str(i).zfill(
print(subset_lookalikes)
```

```
   CustomerID                                          Lookalikes
0       C0001  [(C0069, 0.9473), (C0127, 0.8748), (C0190, 0.8...
1       C0002  [(C0133, 0.9681), (C0062, 0.8986), (C0134, 0.8...
2       C0003  [(C0166, 0.9944), (C0031, 0.9744), (C0158, 0.9...
3       C0004  [(C0090, 0.918), (C0122, 0.9119), (C0017, 0.90...
4       C0005  [(C0197, 0.9997), (C0007, 0.9906), (C0140, 0.8...
5       C0006  [(C0135, 0.9132), (C0187, 0.7755), (C0185, 0.7...
6       C0007  [(C0005, 0.9906), (C0197, 0.9868), (C0120, 0.8...
7       C0008  [(C0162, 0.9358), (C0154, 0.8942), (C0113, 0.8...
8       C0009  [(C0198, 0.9187), (C0029, 0.9179), (C0033, 0.8...
9       C0010  [(C0061, 0.9181), (C0176, 0.9141), (C0042, 0.9...
10      C0011  [(C0126, 0.9552), (C0171, 0.9067), (C0193, 0.8...
11      C0012  [(C0065, 0.9739), (C0136, 0.9319), (C0104, 0.9...
12      C0013  [(C0067, 0.9497), (C0105, 0.941), (C0102, 0.84...
13      C0014  [(C0151, 0.9999), (C0097, 0.9997), (C0060, 0.9...
14      C0015  [(C0123, 0.9989), (C0014, 0.9564), (C0151, 0.9...
15      C0016     [(C0183, 1.0), (C0107, 0.9962), (C0105, 0.89)]
16      C0017  [(C0075, 0.9551), (C0090, 0.9417), (C0194, 0.9...
17      C0018  [(C0023, 0.8811), (C0168, 0.8687), (C0068, 0.7...
18      C0019  [(C0191, 0.9555), (C0174, 0.8714), (C0070, 0.8...
19      C0020  [(C0130, 0.9987), (C0180, 0.9653), (C0095, 0.9...
```

In [ ]:
```

```