💻 Question No: 05

⚙️ Setup

- Ensure the Python kernel has the necessary libraries: `pandas` , `matplotlib` and `lets-plot` , `os` , `numpy` , `statsmodels` , `seaborn`
- Ensure the `bank-full.csv` file is in the `data` folder.

In [1]:

```
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.metrics.pairwise import cosine_similarity
import seaborn as sns
import os
from sklearn import tree
os.getcwd()
import numpy as np
import statsmodels.api as sm

from lets_plot import * # This imports all of ggplot2's functions
LetsPlot.setup_html()
```

In [2]:

```
df = pd.read_excel('D:/Data Science for Marketing-I/data/Online Retail.xlsx')
df
```

Out[2]:

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	2010-12-01 08:26:00	2.55	17850.0
1	536365	71053	WHITE METAL LANTERN	6	2010-12-01 08:26:00	3.39	17850.0
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	2010-12-01 08:26:00	2.75	17850.0
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	2010-12-01 08:26:00	3.39	17850.0
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	2010-12-01 08:26:00	3.39	17850.0
...	...	...	...	...	...	...	...
541904	581587	22613	PACK OF 20 SPACEBOY NAPKINS	12	2011-12-09 12:50:00	0.85	12680.0
541905	581587	22899	CHILDREN'S APRON DOLLY GIRL	6	2011-12-09 12:50:00	2.10	12680.0
541906	581587	23254	CHILDRENS CUTLERY DOLLY GIRL	4	2011-12-09 12:50:00	4.15	12680.0
541907	581587	23255	CHILDRENS CUTLERY CIRCUS PARADE	4	2011-12-09 12:50:00	4.15	12680.0
541908	581587	22138	BAKING SET 9 PIECE RETROSPOT	3	2011-12-09 12:50:00	4.95	12680.0

541909 rows × 8 columns

Exclude entries where "Quantity" or "UnitPrice" have negative or zero values, and remove observations with missing CustomerID.

In [3]:

```
# Filter data
df = df[(df['Quantity'] > 0) & (df['UnitPrice'] > 0)]
df = df.dropna(subset=['CustomerID'])
```

💡 df will contain only rows where: Quantity and UnitPrice are both positive. CustomerID is not missing

In [4]:

```
df
```

Out[4]:

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	2010-12-01 08:26:00	2.55	17850.0
1	536365	71053	WHITE METAL LANTERN	6	2010-12-01 08:26:00	3.39	17850.0
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	2010-12-01 08:26:00	2.75	17850.0
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	2010-12-01 08:26:00	3.39	17850.0
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	2010-12-01 08:26:00	3.39	17850.0
...	...	...	...	...	...	...	...
541904	581587	22613	PACK OF 20 SPACEBOY NAPKINS	12	2011-12-09 12:50:00	0.85	12680.0
541905	581587	22899	CHILDREN'S APRON DOLLY GIRL	6	2011-12-09 12:50:00	2.10	12680.0
541906	581587	23254	CHILDRENS CUTLERY DOLLY GIRL	4	2011-12-09 12:50:00	4.15	12680.0
541907	581587	23255	CHILDRENS CUTLERY CIRCUS PARADE	4	2011-12-09 12:50:00	4.15	12680.0
541908	581587	22138	BAKING SET 9 PIECE RETROSPOT	3	2011-12-09 12:50:00	4.95	12680.0

397884 rows × 8 columns

Create a Customer-Item Matrix using the pivot table function, replacing NaN values with 0 and non-NaN values with 1.

In [5]:

```
# Create Customer-Item Matrix
customer_item_matrix = df.pivot_table(index='CustomerID', columns='StockCode', valu
customer_item_matrix = (customer_item_matrix > 0).astype(int)
customer_item_matrix.head()
```

Out[5]:

StockCode	10002	10080	10120	10125	10133	10135	11001	15030	15034	15036	.
CustomerID											
12346.0	0	0	0	0	0	0	0	0	0	0	
12347.0	0	0	0	0	0	0	0	0	0	0	
12348.0	0	0	0	0	0	0	0	0	0	0	
12349.0	0	0	0	0	0	0	0	0	0	0	
12350.0	0	0	0	0	0	0	0	0	0	0	

5 rows × 3665 columns

💡 Rows (Index): Represent unique CustomerIDs (e.g., 12346.0, 12347.0, etc.). Columns: Represent unique StockCodes (e.g., 10002, 10080, etc.), which are product codes. Values: Likely represent the count or quantity of each product (StockCode) purchased by each customer (CustomerID).

In [6]:

```
# Compute User-to-User Similarity Matrix
similarity_matrix = pd.DataFrame(cosine_similarity(customer_item_matrix),
                                 index=customer_item_matrix.index,
                                 columns=customer_item_matrix.index)
similarity_matrix.head()
```

Out[6]:

CustomerID	12346.0	12347.0	12348.0	12349.0	12350.0	12352.0	12353.0	12354.0
CustomerID								
12346.0	1.0	0.000000	0.000000	0.000000	0.000000	0.000000	0.0	0.000000
12347.0	0.0	1.000000	0.063022	0.046130	0.047795	0.038484	0.0	0.025876
12348.0	0.0	0.063022	1.000000	0.024953	0.051709	0.027756	0.0	0.027995
12349.0	0.0	0.046130	0.024953	1.000000	0.056773	0.137137	0.0	0.030737
12350.0	0.0	0.047795	0.051709	0.056773	1.000000	0.031575	0.0	0.000000

5 rows × 4338 columns

💡 CustomerID 12346.0 has no similarity with any other customer (all values are 0.0 except for itself).

CustomerID 12347.0 has some similarity with other customers, such as 12348.0 (0.063022) and 12349.0 (0.046130).

CustomerID 12348.0 has a higher similarity with 18283.0 (0.170905), indicating they have more similar purchasing behavior.

Compute the User-to-User Similarity Matrix.

Recommend products to the user who has the highest similarity to customer 17173.

In [7]:

```
most_similar_user = similarity_matrix.loc[17173].sort_values(ascending=False).index

# Recommend products
customer_17173_items = set(customer_item_matrix.loc[17173][customer_item_matrix.loc
most_similar_user_items = set(customer_item_matrix.loc[most_similar_user][customer_
recommended_items = most_similar_user_items - customer_17173_items

print(recommended_items)
```

```
{22568, 23128}
```

💡 This indicates products with StockCodes 85099B and 84406B are recommended to customer 17173.

Additionally, apply item-based collaborative filtering to identify products similar to the item with stock code 90103

In [8]:

```
# Item-Based Collaborative Filtering
item_similarity_matrix = pd.DataFrame(cosine_similarity(customer_item_matrix.T),
                                      index=customer_item_matrix.columns,
                                      columns=customer_item_matrix.columns)
```

```
similar_items_to_90103 = item_similarity_matrix[90103].sort_values(ascending=False)
print(similar_items_to_90103)
```

```
Index(['90059B', '90059E', '90059F', 90101, '90059C'], dtype='object', name='StockCo
de')
```

💡 These are the product codes (StockCode) that the most similar customer has purchased but customer 17173 has not.

The recommendations include: '90059B' '90059E' '90059F' 90101 '90059C'

In [9]:
```
df.loc[df['StockCode'].isin(similar_items_to_90103),['StockCode','Description']
       ].drop_duplicates().set_index('StockCode')
```

Out[9]:

	Description
StockCode	
90059B	DIAMANTE HAIR GRIP PACK/2 BLACK DIA
90059E	DIAMANTE HAIR GRIP PACK/2 RUBY
90059C	DIAMANTE HAIR GRIP PACK/2 MONTANA
90059F	DIAMANTE HAIR GRIP PACK/2 LT ROSE
90101	WHITE FRANGIPANI NECKLACE

💡

Finds users with similar purchase behavior to recommend relevant products. Insights: The first four products are hair accessories with slight variations in color/design. The fifth product is a necklace, indicating potential interest in fashion-related items.