Task 2: Lookalike Model

In [1]:
```
import pandas as pd
from sklearn.preprocessing import OneHotEncoder, StandardScaler
from sklearn.metrics.pairwise import cosine_similarity
import csv
```

In [2]:
```
# Load datasets
customers = pd.read_csv("Customers.csv")
products = pd.read_csv("Products.csv")
transactions = pd.read_csv("Transactions.csv")

# Preprocessing
# Drop unnecessary columns
customers_processed = customers.drop(columns=['CustomerID', 'CustomerName'])
```

In [3]:
```
# Encode 'Region' column
encoder = OneHotEncoder()
region_encoded = encoder.fit_transform(customers_processed[['Region']]).toarray()
region_encoded_df = pd.DataFrame(region_encoded, columns=encoder.get_feature_names_out(['Region']))

# Convert 'SignupDate' to numeric (days since the earliest signup date)
customers_processed['SignupDate'] = pd.to_datetime(customers_processed['SignupDate'])
customers_processed['DaysSinceSignup'] = (customers_processed['SignupDate'] - customers_processed['SignupDate'].min()).dt.days
customers_processed = customers_processed.drop(columns=['Region', 'SignupDate'])
```

In [4]:
```
# Combine numeric data with encoded data
final_features = pd.concat([customers_processed.reset_index(drop=True), region_encoded_df], axis=1)

# Standardize the features
scaler = StandardScaler()
final_features_scaled = scaler.fit_transform(final_features)
```

In [5]:
```
# Compute similarity
similarity_matrix = cosine_similarity(final_features_scaled)

# Create similarity DataFrame
similarity_df = pd.DataFrame(similarity_matrix, index=customers['CustomerID'], columns=customers['CustomerID'])
```

In [6]:
```
# Generate lookalikes for the first 20 customers
lookalikes = {}
for customer_id in customers['CustomerID'][:20]:
    similar_customers = similarity_df[customer_id].sort_values(ascending=False).iloc[1:4]
    lookalikes[customer_id] = list(zip(similar_customers.index, similar_customers.values))

# Save lookalikes to a CSV file
with open("Lookalike.csv", mode="w", newline="") as file:
    writer = csv.writer(file)
    writer.writerow(["CustomerID", "SimilarCustomers"])
    for cust_id, similars in lookalikes.items():
        writer.writerow([cust_id, similars])

print("Lookalike CSV generated successfully.")
```

```
Lookalike CSV generated successfully.
```

In [7]:
```
# Now read the saved CSV and display it
lookalike_df = pd.read_csv("Lookalike.csv")
print(lookalike_df)
```

```
   CustomerID                                     SimilarCustomers
0       C0001  [('C0112', 0.9999934679217143), ('C0025', 0.99...
1       C0002  [('C0134', 0.9999845314129499), ('C0045', 0.99...
2       C0003  [('C0126', 0.9997195890105688), ('C0052', 0.99...
3       C0004  [('C0108', 0.9998871383108665), ('C0102', 0.99...
4       C0005  [('C0159', 0.9999451367443317), ('C0106', 0.99...
5       C0006  [('C0076', 0.9985577703727381), ('C0052', 0.99...
6       C0007  [('C0159', 0.9984824549509707), ('C0005', 0.99...
7       C0008  [('C0189', 0.9999238166556718), ('C0183', 0.99...
8       C0009  [('C0121', 0.9994280145949601), ('C0170', 0.99...
9       C0010  [('C0062', 0.9999990418223423), ('C0199', 0.99...
10      C0011  [('C0168', 0.9999988928106186), ('C0171', 0.99...
11      C0012  [('C0187', 0.9999822030221266), ('C0195', 0.99...
12      C0013  [('C0190', 0.9998734425366048), ('C0032', 0.99...
13      C0014  [('C0044', 0.9997681922150611), ('C0057', 0.99...
14      C0015  [('C0185', 0.9988111004387746), ('C0016', 0.99...
15      C0016  [('C0008', 0.9999046560368468), ('C0189', 0.99...
16      C0017  [('C0069', 0.9999956642434136), ('C0051', 0.99...
17      C0018  [('C0049', 0.9999950115052263), ('C0144', 0.99...
18      C0019  [('C0149', 0.9999178272796675), ('C0132', 0.99...
19      C0020  [('C0036', 0.999567927973277), ('C0065', 0.999...
```

In [8]:

```
pd.set_option('display.max_colwidth', None)  # Allow full content in a column to be displayed
pd.set_option('display.max_rows', None)  # Display all rows (if needed)

# Print the full DataFrame
print(lookalike_df)
```

	CustomerID	SimilarCustomers
0	C0001	[('C0112', 0.9999934679217143), ('C0025', 0.9999739916869222), ('C0071', 0.9999417505631495)]
1	C0002	[('C0134', 0.9999845314129499), ('C0045', 0.999937605464356), ('C0040', 0.999787344339684)]
2	C0003	[('C0126', 0.9997195890105688), ('C0052', 0.9997195890105688), ('C0076', 0.9993086221473566)]
3	C0004	[('C0108', 0.9998871383108665), ('C0102', 0.9995145168893103), ('C0192', 0.9994721032009289)]
4	C0005	[('C0159', 0.9999451367443317), ('C0106', 0.9989686627560788), ('C0173', 0.9980392964424282)]
5	C0006	[('C0076', 0.9985577703727381), ('C0052', 0.9977416568193173), ('C0126', 0.9977416568193173)]
6	C0007	[('C0159', 0.9984824549509707), ('C0005', 0.9978508144622327), ('C0175', 0.9978215970299322)]
7	C0008	[('C0189', 0.9999238166556718), ('C0183', 0.9999060163308237), ('C0016', 0.9999046560368468)]
8	C0009	[('C0121', 0.9994280145949601), ('C0170', 0.9989438392701946), ('C0064', 0.9944764587651896)]
9	C0010	[('C0062', 0.9999990418223423), ('C0199', 0.9998634382707906), ('C0019', 0.9992291947506864)]
10	C0011	[('C0168', 0.9999988928106186), ('C0171', 0.9999822265580522), ('C0096', 0.9993800895541761)]
11	C0012	[('C0187', 0.9999822030221266), ('C0195', 0.9988862962568205), ('C0085', 0.9986091308590731)]
12	C0013	[('C0190', 0.9998734425366048), ('C0032', 0.9990445479847451), ('C0129', 0.9988524858911861)]
13	C0014	[('C0044', 0.9997681922150611), ('C0057', 0.9991426539203047), ('C0124', 0.9971579832161084)]
14	C0015	[('C0185', 0.9988111004387746), ('C0016', 0.9980895995316794), ('C0008', 0.997141296478379)]
15	C0016	[('C0008', 0.9999046560368468), ('C0189', 0.9996580334949515), ('C0183', 0.9996213675439523)]
16	C0017	[('C0069', 0.9999956642434136), ('C0051', 0.999725733434161), ('C0041', 0.9994835901443253)]
17	C0018	[('C0049', 0.9999950115052263), ('C0144', 0.9999558141121888), ('C0035', 0.9999455222593414)]
18	C0019	[('C0149', 0.9999178272796675), ('C0132', 0.999590699525703), ('C0037', 0.9994574594023078)]
19	C0020	[('C0036', 0.999567927973277), ('C0065', 0.9994213569723714), ('C0194', 0.9989275184262444)]

In [ ]: