TASK 2

```
# Import necessary libraries
import pandas as pd
from sklearn.preprocessing import LabelEncoder, MinMaxScaler
from sklearn.metrics.pairwise import cosine_similarity



#  Load Datasets
customers = pd.read_csv("Customers.csv")
products = pd.read_csv("Products.csv")
transactions = pd.read_csv("Transactions.csv")


#  Merge Datasets
# Merge transactions with customers and products
merged_data = transactions.merge(customers, on="CustomerID").merge(products, on="ProductID")

# Rename conflicting columns to avoid ambiguity
merged_data.rename(columns={'Price_y': 'Price', 'Price_x': 'TransactionPrice'}, inplace=True)


# Preprocessing and Feature Engineering
# Encode the 'Region' column into numerical format
le_region = LabelEncoder()
merged_data['Region'] = le_region.fit_transform(merged_data['Region'])



# Aggregate features for each customer
customer_profiles = merged_data.groupby('CustomerID').agg({
    'Region': 'first',               # First region associated with the customer
    'Price': 'mean',                 # Average price of products purchased
    'Quantity': 'sum',               # Total quantity purchased
    'TotalValue': 'sum'              # Total transaction value
}).reset_index()



# Normalize Features
# Scale numerical features to a 0-1 range
scaler = MinMaxScaler()
normalized_features = scaler.fit_transform(customer_profiles.iloc[:, 1:])  # Exclude CustomerID


#  Calculate Cosine Similarity
# Compute pairwise cosine similarity
similarity_matrix = cosine_similarity(normalized_features)

# Create a similarity DataFrame for better handling
similarity_df = pd.DataFrame(
    similarity_matrix,
    index=customer_profiles['CustomerID'],
    columns=customer_profiles['CustomerID']
)


#  Generate Lookalike Recommendations
# Find top 3 similar customers for the first 20 customers (C0001 to C0020)
top_lookalikes = {}
for cust_id in customer_profiles['CustomerID'][:20]:
    # Get top 3 most similar customers (exclude the customer itself)
    similar_customers = similarity_df[cust_id].sort_values(ascending=False)[1:4]
    top_lookalikes[cust_id] = [(sim_cust_id, round(sim_score, 3)) for sim_cust_id, sim_score in similar_customers.item


# Save Recommendations to CSV
output_file = "GOKUL_SHANMUGAM_Lookalike.csv"
with open(output_file, "w") as f:
    f.write("CustomerID,Lookalikes\n")
    for cust_id, lookalikes in top_lookalikes.items():
```

```
        f.write(f"{cust_id},\"{lookalikes}\"\n")

print(f"Lookalike recommendations saved to {output_file}")

# Optional: Print sample recommendations
print("\nSample Recommendations:")
for cust_id, recommendations in list(top_lookalikes.items())[:5]:
    print(f"CustomerID: {cust_id} -> Lookalikes: {recommendations}")
```

```
Lookalike recommendations saved to GOKUL_SHANMUGAM_Lookalike.csv

Sample Recommendations:
CustomerID: C0001 -> Lookalikes: [('C0137', 0.999), ('C0191', 0.999), ('C0011', 0.999)]
CustomerID: C0002 -> Lookalikes: [('C0088', 0.998), ('C0027', 0.998), ('C0106', 0.994)]
CustomerID: C0003 -> Lookalikes: [('C0190', 0.999), ('C0147', 0.998), ('C0174', 0.996)]
CustomerID: C0004 -> Lookalikes: [('C0113', 0.999), ('C0034', 0.997), ('C0169', 0.996)]
CustomerID: C0005 -> Lookalikes: [('C0007', 1.0), ('C0146', 1.0), ('C0115', 0.998)]
```

Loading...

```
#  Calculate Cosine Similarity
# Compute cosine similarity between customer profiles
similarity_matrix = cosine_similarity(normalized_features)

# Convert similarity matrix to a DataFrame for easier handling
similarity_df = pd.DataFrame(
    similarity_matrix,
    index=customer_profiles['CustomerID'],  # Row labels are Customer IDs
    columns=customer_profiles['CustomerID']  # Column labels are Customer IDs
)

# Generate Recommendations for the First 20 Customers
# Focus on the first 20 customers (C0001 to C0020)
top_lookalikes = {}
for cust_id in customer_profiles['CustomerID'][:20]:  # Loop through the first 20 customers
    # Sort similarity scores in descending order and exclude the customer itself
    similar_customers = similarity_df[cust_id].sort_values(ascending=False)[1:4]
    # Store the top 3 similar customers and their scores
    top_lookalikes[cust_id] = [(sim_cust_id, round(sim_score, 3)) for sim_cust_id, sim_score in similar_customers.items(

# Save Recommendations to Lookalike.csv
output_file = "GOKUL_SHANMUGAM_Lookalike.csv"
with open(output_file, "w") as f:
    f.write("CustomerID,Lookalikes\n")
    for cust_id, lookalikes in top_lookalikes.items():
        f.write(f"{cust_id},\"{lookalikes}\"\n")

# Display a sample of the results
print("\nRecommendations for First 20 Customers:")
for cust_id, recommendations in list(top_lookalikes.items())[:5]:  # Display the first 5 for preview
    print(f"CustomerID: {cust_id} -> Lookalikes: {recommendations}")
```

```
Recommendations for First 20 Customers:
CustomerID: C0001 -> Lookalikes: [('C0137', 0.999), ('C0191', 0.999), ('C0011', 0.999)]
CustomerID: C0002 -> Lookalikes: [('C0088', 0.998), ('C0027', 0.998), ('C0106', 0.994)]
CustomerID: C0003 -> Lookalikes: [('C0190', 0.999), ('C0147', 0.998), ('C0174', 0.996)]
CustomerID: C0004 -> Lookalikes: [('C0113', 0.999), ('C0034', 0.997), ('C0169', 0.996)]
CustomerID: C0005 -> Lookalikes: [('C0007', 1.0), ('C0146', 1.0), ('C0115', 0.998)]
```

Start coding or generate with AI.

Start coding or generate with AI.

Start coding or generate with AI.

Start coding or generate with AI.

Start coding or generate with AI.

Loading...