# Task 2: Lookalike Model

### Task Overview

You will create a **Lookalike Model** that recommends three similar customers based on their profile and transaction history. The recommendations should use both customer and product data, and you need to compute a similarity score for each pair of customers.

### Step 1: Import Libraries

Start by importing the required Python libraries:

```
import pandas as pd

import numpy as np

from sklearn.metrics.pairwise import cosine_similarity

from sklearn.preprocessing import MinMaxScaler, LabelEncoder
```

### Step 2: Load and Merge the Datasets

```
customers = pd.read_csv("Customers.csv")

products = pd.read_csv("Products.csv")

transactions = pd.read_csv("Transactions.csv")
```

Merge them into a single dataset:

```
data = transactions.merge(customers,
on='CustomerID').merge(products, on='ProductID'
```

### Step 3: Data Preprocessing

### 1. Encode Categorical Data

Encode categorical features like Region and Category to numerical values:

```
le_region = LabelEncoder()

le_category = LabelEncoder()
```

```
data['Region'] = le_region.fit_transform(data['Region'])
data['Category'] = le_category.fit_transform(data['Category'])
```

**2. Aggregate Customer Profiles**

Aggregate data at the CustomerID level to create profiles:

```
customer_profiles = data.groupby('CustomerID').agg({
    'Region': 'first',                  # Region of the customer
    'TotalValue': 'sum',                # Total value of purchases
    'Quantity': 'sum',                  # Total quantity purchased
    'Category': lambda x: x.mode()[0]   # Most purchased category
}).reset_index()
```

**3. Normalize Numerical Features**

Normalize TotalValue and Quantity for fair comparison:

```
scaler = MinMaxScaler()
customer_profiles[['TotalValue', 'Quantity']] = scaler.fit_transform(customer_profiles[['TotalValue', 'Quantity']])
```

---

**Step 4: Compute Customer Similarity**

**1. Create Feature Matrix**

Use relevant features from customer_profiles for similarity calculations:

```
features = customer_profiles[['Region', 'TotalValue', 'Quantity', 'Category']]
```

**2. Compute Cosine Similarity**

Calculate similarity between customers using cosine similarity:

```
similarity_matrix = cosine_similarity(features)
```

```
similarity_df = pd.DataFrame(similarity_matrix,
index=customer_profiles['CustomerID'],
columns=customer_profiles['CustomerID'])
```

---

**Step 5: Extract Top 3 Similar Customers**

**1. Find Top 3 Similar Customers**

For each of the first 20 customers (CustomerID: C0001 - C0020), extract their top 3 similar customers:

```
lookalike_dict = {}

for customer_id in customer_profiles['CustomerID'][:20]:  # First 20 customers
    similar_customers = similarity_df[customer_id].nlargest(4).iloc[1:] # Exclude self (largest similarity score)
    lookalike_dict[customer_id] = [(similar_cust_id, round(similarity, 4)) for similar_cust_id, similarity in similar_customers.items()]
```

---

**Step 6: Save Lookalike Data**

```
import csv

with open('Lookalike.csv', 'w', newline='') as file:
    writer = csv.writer(file)
    writer.writerow(['CustomerID', 'Lookalikes'])

    for key, value in lookalike_dict.items():
        writer.writerow([key, value])
```

---

**Step 7: Deliverables**

**1. Jupyter Notebook**

1. Data loading and merging steps.
2. Preprocessing code (encoding, aggregation, normalization).
3. Similarity calculation logic.
4. Recommendation extraction.
5. Results visualization (optional).

**2. Lookalike.csv**

This file should have two columns:

- CustomerID: The customer ID for whom recommendations are made.
- Lookalikes: A list of tuples with recommended customer IDs and their similarity scores.

**Example:**

CustomerID,Lookalikes

C0001,"[(C0005, 0.92), (C0010, 0.89), (C0020, 0.87)]"

C0002,"[(C0007, 0.95), (C0015, 0.90), (C0018, 0.88)]"