Lookalike Model Development Code:

```
import pandas as pd
from sklearn.preprocessing import LabelEncoder, MinMaxScaler
from sklearn.metrics.pairwise import cosine_similarity

# Load the data
customers_file = 'Customers.csv'
products_file = 'Products.csv'
transactions_file = 'Transactions.csv'

customers_df = pd.read_csv(customers_file)
products_df = pd.read_csv(products_file)
transactions_df = pd.read_csv(transactions_file)

# Merge datasets
transactions_products_df = pd.merge(transactions_df, products_df, on="ProductID")
full_data = pd.merge(transactions_products_df, customers_df, on="CustomerID")

# Feature engineering
# Total spending per customer
total_spending =
full_data.groupby("CustomerID")["TotalValue"].sum().rename("TotalSpending")

# Total transactions per customer
transaction_count =
full_data.groupby("CustomerID")["TransactionID"].nunique().rename("TransactionCount
")

# Average transaction value
avg_transaction_value = (total_spending /
transaction_count).rename("AvgTransactionValue")

# Preferred categories (most frequent category per customer)
preferred_category = (
full_data.groupby(["CustomerID", "Category"])["TransactionID"]
.count()
.reset_index()
.sort_values(["CustomerID", "TransactionID"], ascending=[True, False])
.drop_duplicates(subset="CustomerID")[["CustomerID", "Category"]]
.rename(columns={"Category": "PreferredCategory"})
)

# Combine features into customer profiles
customer_profiles = pd.DataFrame(total_spending).merge(transaction_count,
on="CustomerID")
customer_profiles = customer_profiles.merge(avg_transaction_value, on="CustomerID")
customer_profiles = customer_profiles.merge(preferred_category, on="CustomerID")

# Encode PreferredCategory
```

```
encoder = LabelEncoder()
customer_profiles["PreferredCategoryEncoded"] =
encoder.fit_transform(customer_profiles["PreferredCategory"])

# Normalize numerical features
scaler = MinMaxScaler()
customer_profiles_scaled = scaler.fit_transform(
customer_profiles[["TotalSpending", "TransactionCount", "AvgTransactionValue",
"PreferredCategoryEncoded"]]
)

# Compute cosine similarity
similarity_matrix = cosine_similarity(customer_profiles_scaled)
customer_ids = customer_profiles["CustomerID"].tolist()

# Generate recommendations
recommendations = {}
for idx, customer_id in enumerate(customer_ids):
sim_scores = list(enumerate(similarity_matrix[idx]))
sim_scores = sorted(sim_scores, key=lambda x: x[1], reverse=True)
sim_scores = [(customer_ids[i], score) for i, score in sim_scores if i != idx]
recommendations[customer_id] = sim_scores[:3]

# Convert recommendations to a DataFrame
lookalike_data = {
"CustomerID": [],
"Recommendations": []
}
for cust_id, recs in recommendations.items():
lookalike_data["CustomerID"].append(cust_id)
lookalike_data["Recommendations"].append(
[{"CustomerID": rec[0], "Score": round(rec[1], 4)} for rec in recs]
)

lookalike_df = pd.DataFrame(lookalike_data)

# Save to CSV
lookalike_csv_path = 'Lookalike.csv'
lookalike_df.to_csv(lookalike_csv_path, index=False)

print(f"Lookalike recommendations saved to {lookalike_csv_path}")
```

Output: Lookalike_New.csv