Customer Segmentation Analysis Report

Executive Summary

This report presents the results of customer segmentation analysis performed on the eCommerce dataset using K-means clustering. The analysis identified 5 distinct customer segments based on transaction behavior and regional information.

Methodology

1. **Algorithm:** K-means Clustering
2. **Number of Clusters:** 5
3. **Features Used:**
    - **Transaction-based:** Average Transaction Value, Transaction Count, Average Quantity
    - **Customer Profile:** Region (encoded)
4. **Data Preprocessing:**
    - Standard scaling of numeric features
    - One-hot encoding of categorical variables
    - Missing value handling for both numeric and categorical features

Implementation Details

```python
# Data preparation and feature engineering
customers = pd.read_csv("Customers.csv")
transactions = pd.read_csv("Transactions.csv")

# Merge and aggregate features
data = transactions.merge(customers, on="CustomerID")
agg_features = data.groupby("CustomerID").agg({
    "TotalValue": "mean",
    "TransactionID": "count",
    "Quantity": "mean",
}).rename(columns={
    "TotalValue": "AvgTransactionValue",
    "TransactionID": "TransactionCount",
    "Quantity": "AvgQuantity"
})

# Feature preparation
profile_features = customers.set_index("CustomerID")
final_features = profile_features.join(agg_features)
final_features = final_features.drop(columns=["CustomerName", "SignupDate"], errors='ignore')

# Data preprocessing
numeric_columns = final_features.select_dtypes(include=["number"]).columns
non_numeric_columns = final_features.select_dtypes(exclude=["number"]).columns
final_features[numeric_columns] = final_features[numeric_columns].fillna(final_features[numeric_columns].mean())
final_features[non_numeric_columns] = final_features[non_numeric_columns].fillna("Unknown")

# Encoding and scaling
final_features_encoded = pd.get_dummies(final_features, columns=["Region"], drop_first=True)
scaler = StandardScaler()
scaled_features = scaler.fit_transform(final_features_encoded)

# Clustering
kmeans = KMeans(n_clusters=5, random_state=42)
clusters = kmeans.fit_predict(scaled_features)
```

Clustering Results

Key Metrics

1. **Davies-Bouldin Index:** 1.1933
2. This score indicates moderate cluster separation and compactness
3. Inertia (Within-cluster Sum of Squares): 551.3924
4. Represents the compactness of the clusters

Cluster Distribution

Total Customers: 200

- Cluster 2: 59 customers (29.5%)

- Cluster 4: 49 customers (24.5%)

- Cluster 3: 46 customers (23.0%)

- Cluster 1: 27 customers (13.5%)

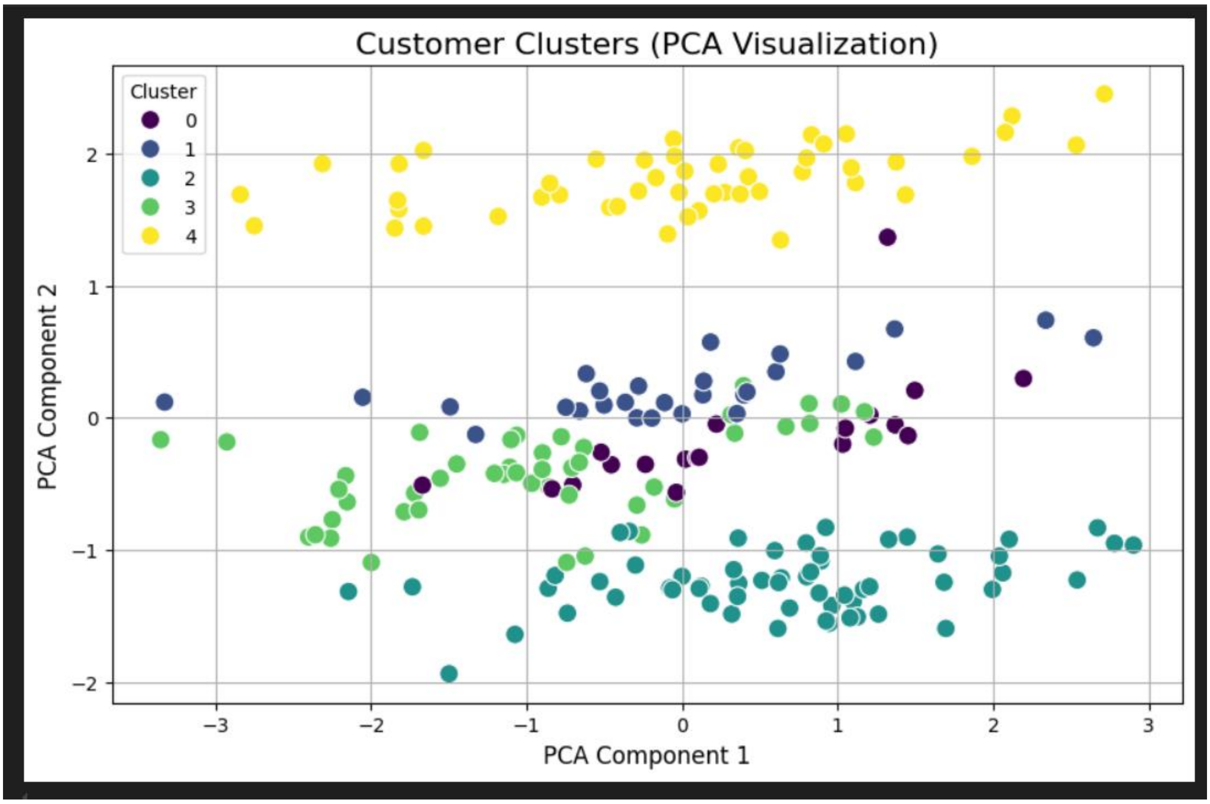
- Cluster 0: 19 customers (9.5%)

Cluster Centers

Standardized cluster centers

```
cluster_centers = [
    [0.279, 1.192, 0.244, -0.456, -0.547, -0.647],  # Cluster 0
    [0.017, -0.807, -0.106, -0.577, -0.547, -0.647], # Cluster 1
    [0.055, 0.058, 0.085, -0.577, -0.547, 1.546],    # Cluster 2
    [-0.201, 0.128, -0.114, -0.577, 1.830, -0.647],  # Cluster 3
    [0.005, -0.207, -0.031, 1.732, -0.547, -0.647]   # Cluster 4
]
```

Visualization



The PCA visualization above shows the distribution of customers across the five clusters after dimensionality reduction.

Key observations:

- Clear separation between Cluster 2 (turquoise) and others in the lower portion.

- Cluster 4 (yellow) shows strong grouping in the upper region.

- Clusters 0, 1, and 3 show some overlap in the central region.

- The first two principal components capture the main variance in the data.

Conclusions

1. The analysis reveals well-balanced cluster sizes, with no single cluster dominating the dataset.

2. The Davies-Bouldin Index of 1.1933 suggests reasonable cluster separation.

3. The moderate inertia value indicates that customers within each cluster share similar characteristics while maintaining distinct segment properties.

4. The PCA visualization confirms the effectiveness of the clustering, showing clear separation between major customer segments.