Clustering

1. Introduction

This report outlines the customer segmentation analysis performed on the eCommerce Transactions dataset using KMeans clustering. The objective is to identify distinct customer segments based on their purchasing behavior.

2. Data Preparation

The following code was used to prepare the data for clustering:

```
# Prepare data for clustering

clustering_data = merged_data.groupby('CustomerID').agg({

    'TotalValue': 'sum',

    'Quantity': 'mean'

}).reset_index()
```

3. Standardization of Features

Standardization was performed to ensure that the features are on the same scale before applying KMeans clustering.

```
from sklearn.preprocessing import StandardScaler



# Standardize features
```

```
scaler = StandardScaler()

scaled_clustering_data = scaler.fit_transform(clustering_data.iloc[:, 1:])  # Exclude 'CustomerID'
```

4. KMeans Clustering

The KMeans algorithm was applied to segment customers into distinct groups. The Davies-Bouldin Index was used to evaluate the clustering performance.

```
from sklearn.cluster import KMeans

from sklearn.metrics import davies_bouldin_score



# KMeans clustering

db_scores = []

for k in range(2, 11):  # Test clusters from 2 to 10

    kmeans = KMeans(n_clusters=k, random_state=42).fit(scaled_clustering_data)

    db_index = davies_bouldin_score(scaled_clustering_data, kmeans.labels_)

    db_scores.append(db_index)



# Plot DB Index

plt.plot(range(2, 11), db_scores, marker='o')

plt.title('Davies-Bouldin Index vs. Number of Clusters')

plt.xlabel('Number of Clusters')

plt.ylabel('DB Index')

plt.show()
```

5. Results

The optimal number of clusters was determined based on the Davies-Bouldin Index, and the final clustering results were visualized.

```
# Final clustering with optimal clusters

optimal_k = db_scores.index(min(db_scores)) + 2

final_kmeans = KMeans(n_clusters=optimal_k, random_state=42).fit(scaled_clustering_data)

# Visualize clusters

plt.scatter(scaled_clustering_data[:, 0], scaled_clustering_data[:, 1], c=final_kmeans.labels_, cmap='viridis')

plt.title('Customer Segments')

plt.xlabel('Feature 1 (TotalValue or Quantity)')

plt.ylabel('Feature 2 (Quantity or Price)')

plt.show()
```

6. Conclusion

The clustering analysis revealed distinct customer segments, which can be targeted for personalized marketing strategies. Understanding these segments allows for more effective resource allocation and marketing efforts.