Customer Segmentation Report

Introduction

Customer segmentation is a crucial aspect of marketing and business strategy, allowing companies to tailor their products and services to different customer groups. This report details the process and findings of customer segmentation using hierarchical clustering and KMeans clustering techniques. The analysis is based on data from `Customers.csv` and `Transactions.csv`.

Methodology

Data Preparation

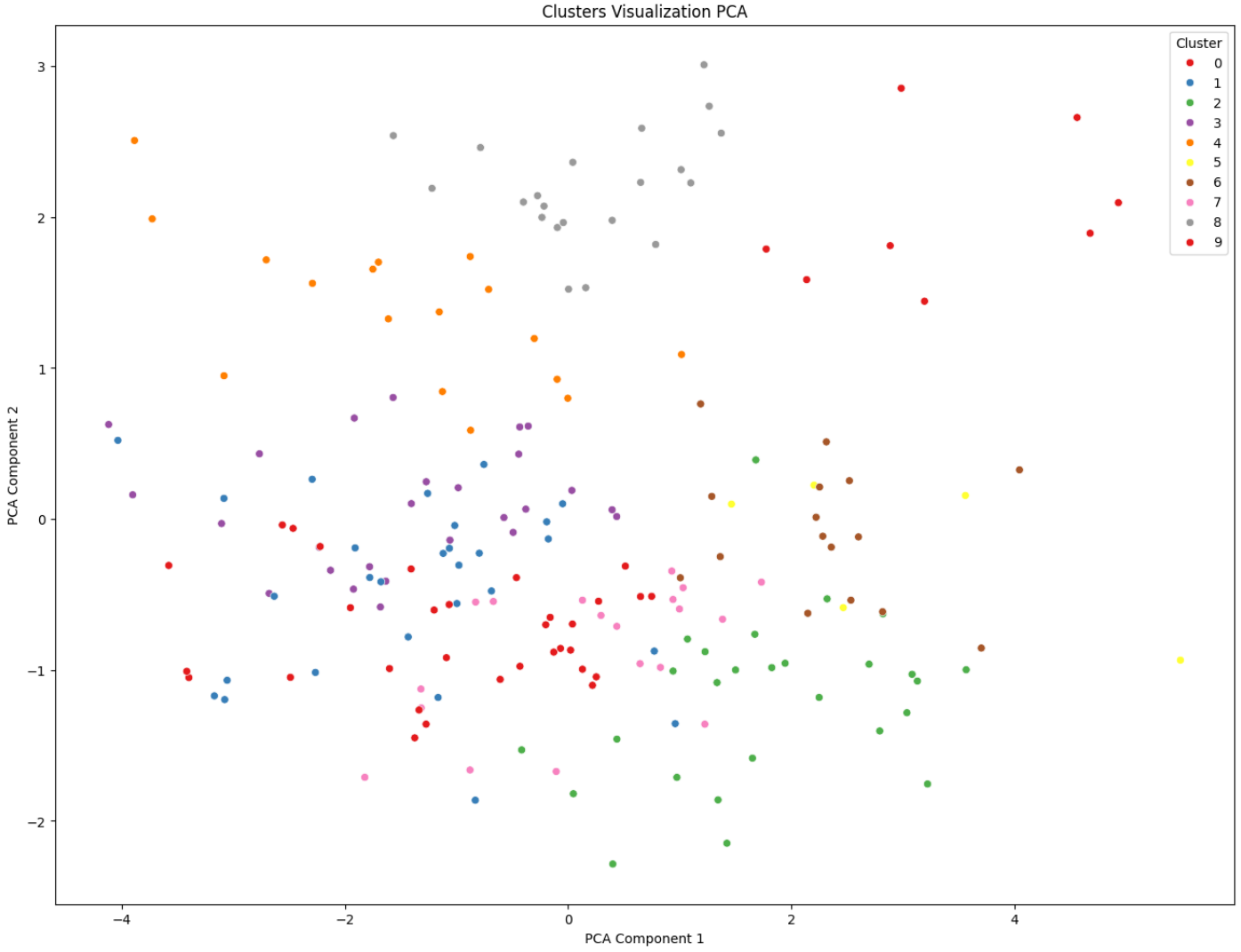
1. **Data Loading** : Loaded Customers.csv and Transactions.csv using pandas.
2. **Data Aggregation** :
    1. Aggregated transaction data to calculate total transactions , total quantity , total revenues, average transaction value , and product variance for each customer.
3. **Data merging** : Merged the aggregated transaction data with customer data.
4. **Categorical Data Encoding** : Encoded the Region column using OneHotEncding.
5. **Data scaling** : Scaled the numerical features using StandardScaler.

Clustering

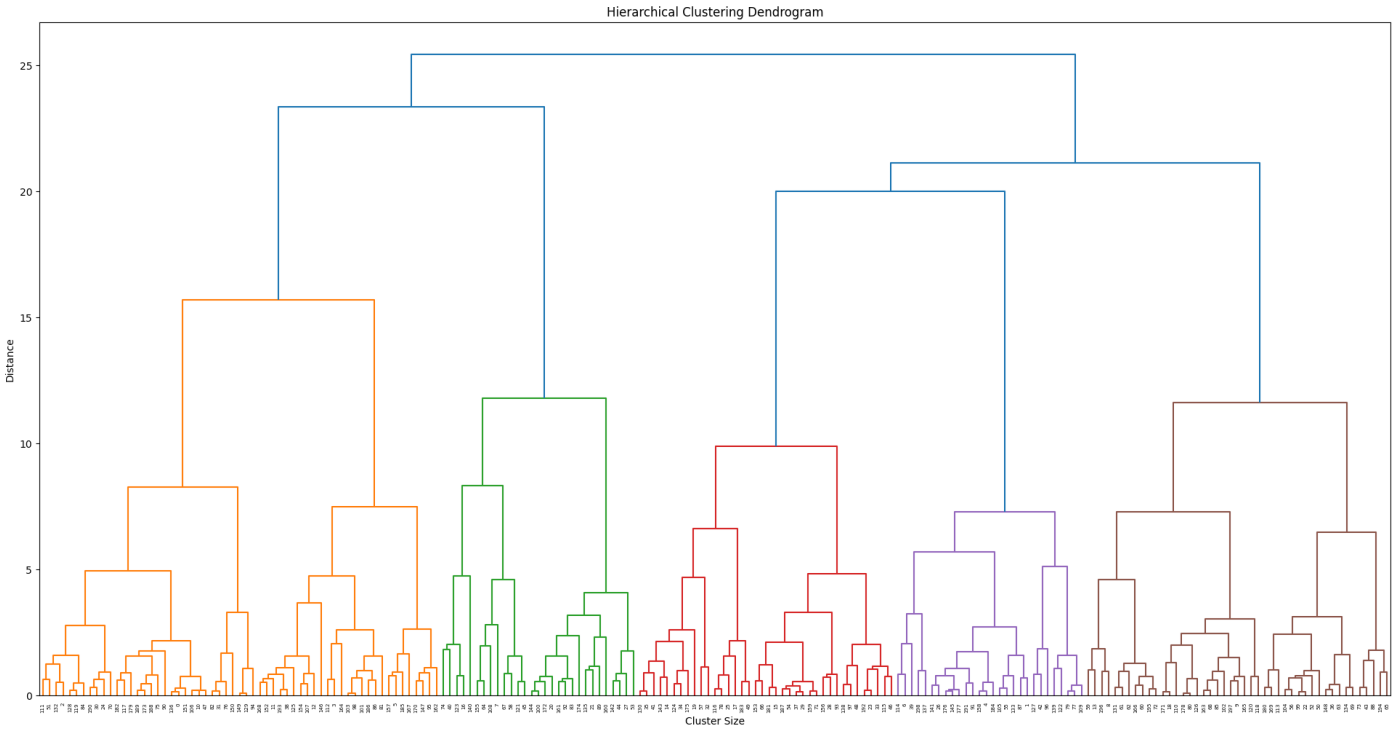
1. **Hierarchical Clustering (Agglomerative Clustering)**:
    - Performed hierarchical clustering for a range of clusters (2 to 10).
    - Calculated clustering metrics: Davies-Bouldin Index, Silhouette Score, and Calinski-Harabasz Index.
    - Determined the optimal number of clusters based on the clustering metrics.
2. **KMeans Clustering**:
    - Performed KMeans clustering for a range of clusters (2 to 10).
    - Calculated clustering metrics: Davies-Bouldin Index, Silhouette Score, and Calinski-Harabasz Index.
    - Determined the optimal number of clusters based on the clustering metrics.

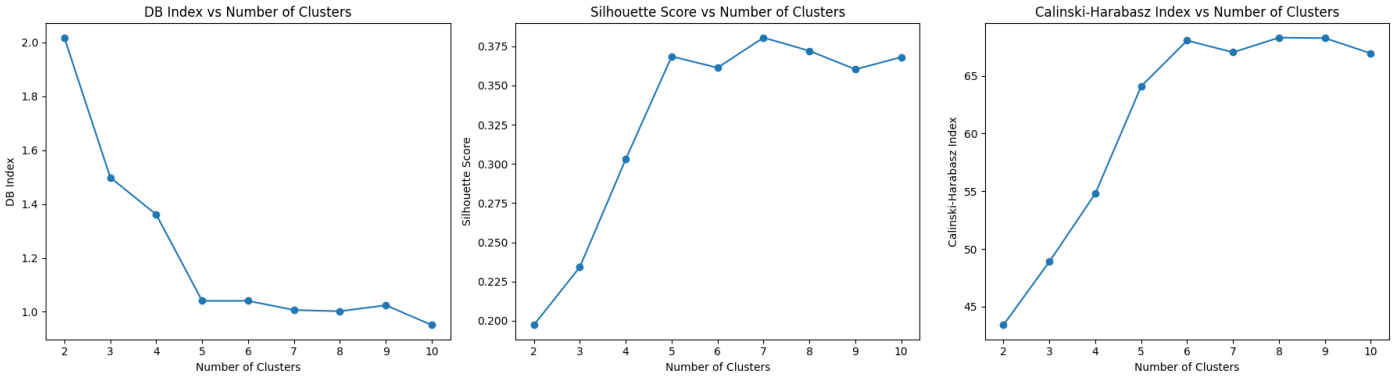
Visualization

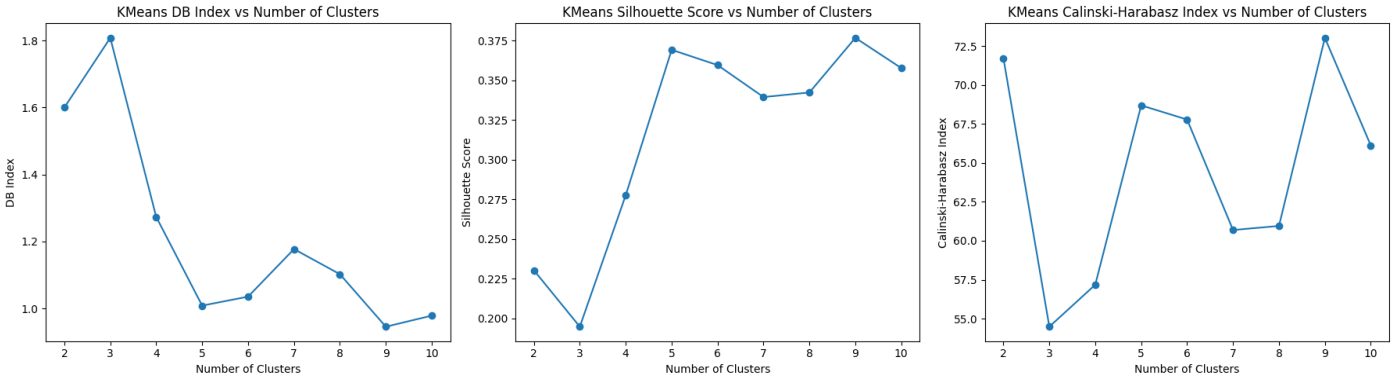
- **Created Scatter plots using PCA components to visulaize the clusters.**



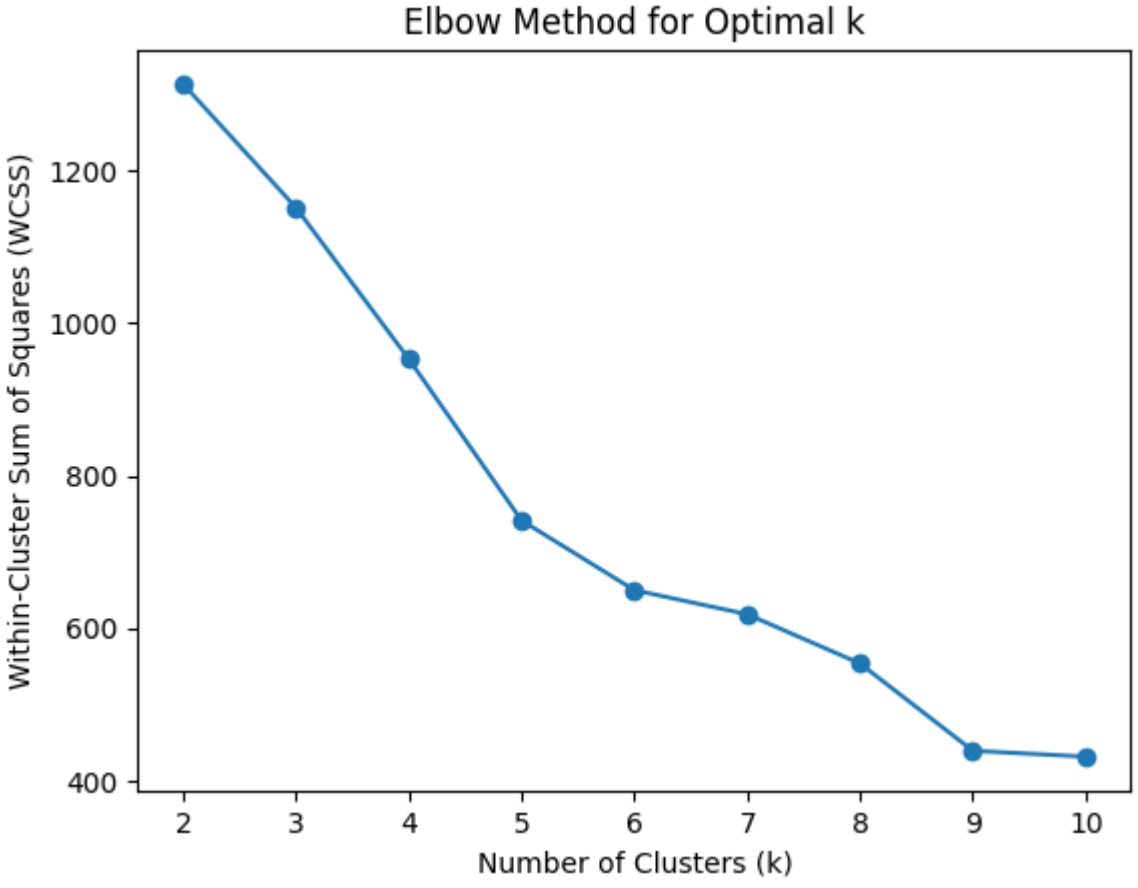
- # Hierarchical Clustering (Agglomerative Clustering)
- In this graph db index is calculated for every number from 2 to 10 and also other two metrics (Silhouette and Calinski Harabasz Index)
- In the linking image we can see 10 clusters are formed while the 10^{th} cluster is very small it also shows that 9 clusters are enough but with Agglomerative Clustering DB indexing we have minimum value at point 10.





- ## K-Means Clustering
- In this graph db index is calculated for every number from 2 to 10 and also other two metrics (Silhouette and Calinski Harabasz Index)
- In this graph we can see the huge drop between 2 and 5 , but minimum value we are getting is at point 9 so according to K-Means Clustering optimal number of clusters are 9.
- 

WCSS (Elbow method)

- Elbow method can provide us additional insights and potentially confirm your findings.
- Here the maximum drop we can see between point 2 to 5 and after them there are very minimal drops. So optimal value for DB index could be 5.
- 

Using 5 Clusters to Calculate DB Index

```
k= 5

kmeans = KMeans(n_clusters=k, random_state=42)
kmeans_cluster_labels = kmeans.fit_predict(scaled_data)

kmeans_db_index = davies_bouldin_score(scaled_data, kmeans_cluster_labels)
agglo_cluster = AgglomerativeClustering(n_clusters=k, linkage='ward')
agglo_cluster_labels = agglo_cluster.fit_predict(scaled_data)
agglo_db_index = davies_bouldin_score(scaled_data, agglo_cluster_labels)
print(f"Kmeans DB index = {kmeans_db_index:.3f}")
print(f"Agglomerative DB Index = {agglo_db_index:.3f}")
```

```
Kmeans DB index = 1.0087180296821647
Agglomerative DB index = 1.0406119422641784
```

I have made these reports using markdown in obsidian app therefore they are minimalistic