Hierarchical Clustering for Customer Segmentation: A Data Mining Perspective

Shulabh Bhattarai

December 22, 2024

Abstract

This tutorial is aimed to explore the applications and use of hierarchical clustering to segment customers in the Dataset (Mall Customer Segmentation Dataset{Mall_Customers.csv}from Kaggle). This dataset includes information on customer demographics and spending behavior. In this tutorial, data cleaning and standardizing are performed to prepare the data for further exploration. Then, I employed Hierarchical clustering with Ward's linkage to identify unique customer segments based on the features in the dataset. The clusters obtained by applying these data science techniques are then analyzed to understand their characteristics and potential implications for other marketing strategies. I then visualized the clusters using scatter plots to analyze the relationships between customer attributes. This tutorial demonstrates the effectiveness of hierarchical clustering in helping to provide insights from customer data. This exploration is aimed to aid business by tailoring their marking efforts and improve customer satisfaction.

Access the code to this tutorial here: https://github.com/shulabhb/Hierarchical-Clustering-for-Customer-Segmentation.git

1 Introduction

Customer Segmentation is a powerful technique in data mining that involves segmenting customers into distinct groups based on similar characteristics. Hierarchical clustering[MC17] is a popular unsupervised machine learning algorithm that can be used to identify these customer segments. In this tutorial, we will explore the application of hierarchical clustering to segment customers using the Mall Customers Segmentation dataset. This dataset contains age, gender, annual income, and spending score for 200 users. By cleaning and standardizing the data, we will apply hierarchical clustering to identify and analyze customer segments and their characteristics.

2 Data Preparation and Exploration

2.1 Data Loading and Cleaning

First, we start by loading the dataset Mall_customers.csv into a python environment and use pandas to handle data manipulation and analysis. I performed basic exploration to understand the dataset's structure, check for inconsistencies and missing values. I explored the data using function like data.head(), data.info(), and data.describe() which are basic Exploratory Data Analysis functions. Since this dataset is meant for educational purposes, there were no invalid fields and the dataset is very well structured.

2.2 Data Standardization

To ensure that the numeric features with different scales contribute equally to the clustering, we need to standardize these numerical features like age, income and spending scores. Data standardization involves scaling the features to have a mean of 0 and a standard deviation of 1 and this standardization is done using sci-learn kit.

Using: ***'from sklearn.preprocessing import StandardScaler'***

After data cleaning and standardizing, this new data is saved as a csv file (cleaned_data.csv). We will use this dataset with standardized numeric features to cluster the customer and conduct further analysis.

2.3 Exploratory Data Analysis

Before I begin with clustering, I use visualizations to understand the characteristic of different customer segments. After running the eda.py file, 7 .png files are saved to results/visualization folder. These visualizations include Histogram and Box Plot to visualize the distribution of age, annual income, and spending score. I also furthered produced scatter plots to explore the relationship between age and spending score, as well as annual income and spending score.

3 Hierarchical Clustering

Hierarchical Clustering creates a hierarchy of clusters, starting with each data point as a single cluster and merging the closest clusters iteratively until all data points belong to a single cluster. To identify distinct customer segments, I employ hierarchical clustering with Ward's linkage[GRS19]. This method iteratively merges the two closest clusters, whose closeness will be determined using Euclidean Distance. The Euclidean distance is calculated withing the linkage function from spicy.cluster.hierarchy module. Ward's linkage will help with minimizing the increase in the sum of squared errors (SSE) when merging clusters. In other words, it helps to create compact and well-separated clusters.

3.1 Coding Implementation

We first load the data and use linkage function to minimize SSE and employ agglomerative clustering to produce a dendrogram. Agglomerative clustering[Mül11] is a type of hierarchical clustering technique where data points are initially treated as individual clusters and then progressively merged together based on their similarity, eventually forming one large cluster - essentially a "bottom-up" approach to grouping data points into clusters; it is also known as "AGNES" (Agglomerative Nesting). Initially we assume our clustering size to be 10 or 20: producing dendrogram first will help us determine the appropriate number of clusters. Then we plot the dendrogram. The code for this part is given as 1.

```python
def cluster_data(X, labels, output_dir, plot_title, file_prefix, n_clusters=10, linkage_method='ward'):
    """Performing hierarchical clustering and save dendrogram."""
    print(f"\nPerforming clustering for {plot_title}...")
    Z = linkage(X, method=linkage_method)

    # Plot Dendrogram
    plt.figure(figsize=(16, 10))
    plt.title(plot_title)
    plt.subplots_adjust(bottom=0.3)
    dendrogram(
        Z, labels=labels, leaf_rotation=90, leaf_font_size=10, truncate_mode=None,
        show_contracted=False, color_threshold=1.5 * max(Z[:, 2])
    )
    plt.ylabel("Distance")
    plt.xlabel("Customer IDs")
    dendrogram_path = os.path.join(output_dir, f'{file_prefix}_detailed_dendrogram.png')
    plt.tight_layout()
    plt.savefig(dendrogram_path)
    plt.close()
    print(f"Detailed Dendrogram saved to: {dendrogram_path}")

    # Performing Agglomerative Clustering
    model = AgglomerativeClustering(n_clusters=n_clusters, linkage=linkage_method)
    cluster_labels = model.fit_predict(X)
    return cluster_labels
```

Figure 1: Hierarchical Clustering Code Implementation

3.2 Dendrogram Interpretation

A Dendrogram2 . is used to visualize the hierarchical structure of the clusters we obtained. The vertical axis represents the distance between the clusters and the horizontal axis represents the customer IDs

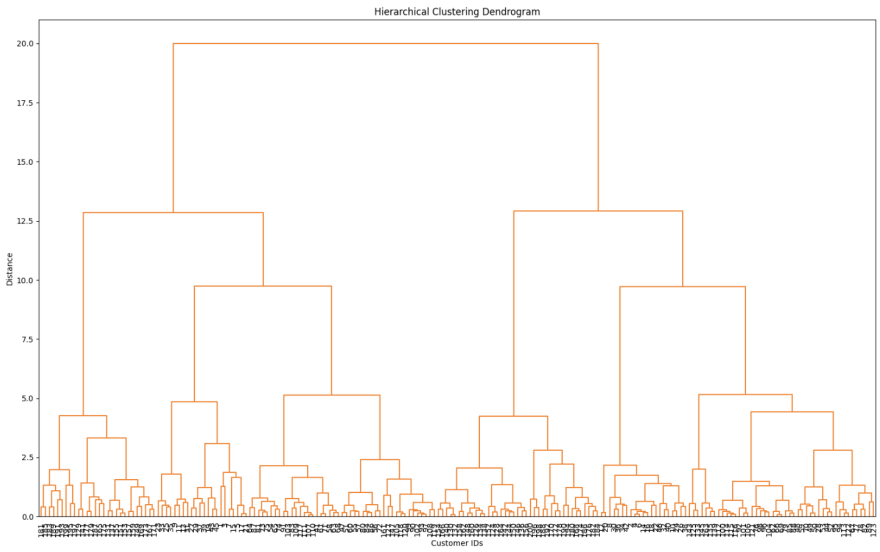

Figure 2: Dendrogram visualizing Hierarchical Clustering

in the cleaned_dataset.csv. The very large distances between the horizontal lines suggests us potential clusters. In this case, there is a significant gap (between 10 and 15 on the y-axis), which shows that we could cu the dendrogram at this level to form clusters.

4 Optimal Number of Clusters

Now to identify the number of clusters, look for the longest vertical line in the dendrogram that is not intersected by any horizontal line. If we decide to cut between 10 and 15 on the y-axis, our optimal cluster size is 4-5, which we will validate with elbow method.

4.1 Elbow Method

I used the Elbow Method[HR20] to identify the optimal number of clusters for optimal classification of the customers. By plotting WCSS(Within-Clusters Sum of Square) values for cluster sizes ranging from 1 to 10, the "elbow point", where the WCSS reduction diminishes drastically is observed at 4 clusters as evident in 3.

4.2 Implementation

Now after I confirm the number of clusters to 4 after validating with the Elbow Method, I now set the number of clusters to 4 and save the clustering details as as a csv file named customer_segmentation_clusters.csv. This file will be used to visualize the clusters and learn about the attributes of these clusters. The code implementation is given in fig 4

5 Cluster Evaluation and Insights

5.1 Visualization

I produced scatter plots to see the relationships between Age vs Spending Score in 5 and Annual Income vs Spending Score in 5.

5.2 Interpretation

From the visualizations of scatter plot of the clusters, we can see how distinctively the clusters are separated in the Income vs Spending Score plot6 and somehow distributed in a pattern in the Age vs

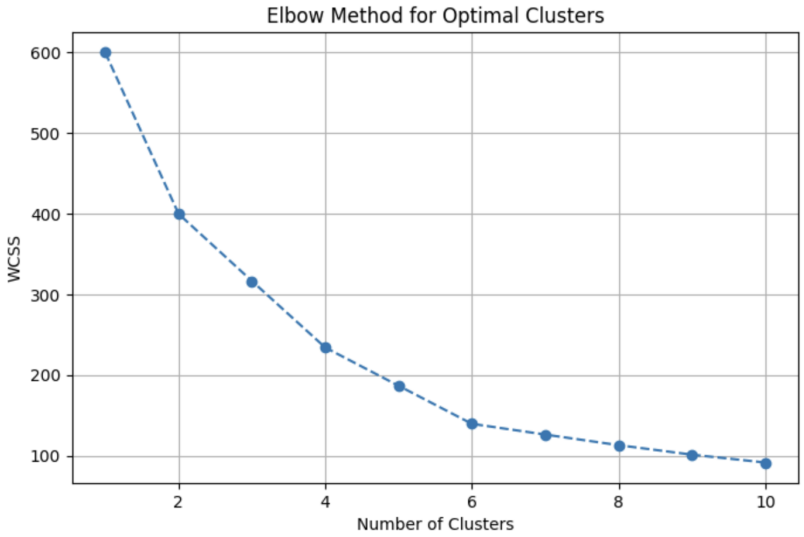

Figure 3: Elbow Method

```python
def save_cluster_results(data, cluster_labels, output_dir, file_prefix):
    """Save clustering results to CSV."""
    data['Cluster'] = cluster_labels
    output_file = os.path.join(output_dir, f'{file_prefix}_clusters.csv')
    data.to_csv(output_file, index=False)
    print(f"Cluster results saved to: {output_file}\n")

if __name__ == "__main__":
    # Loading cleaned data
    data = load_cleaned_data(CLEANED_DATA_PATH)

    # Performing clustering on numeric features
    features = ['Age', 'Annual Income (k$)', 'Spending Score (1-100)']
    X = data[features].values

    # Performing Clustering
    cluster_labels = cluster_data(
        X, labels=data['CustomerID'].astype(str).values, output_dir=CLUSTER_OUTPUT_DIR,
        plot_title="Hierarchical Clustering Dendrogram", file_prefix="customer_segmentation",
        n_clusters=4, linkage_method='ward'
    )

    # Saveing Clustering Results
    save_cluster_results(data, cluster_labels, CLUSTER_OUTPUT_DIR, "customer_segmentation")

    print("Clustering process completed! Proceed to evaluation.py for detailed analysis.")
```

Figure 4: Code implementation to save clustering details as a csv file

Figure 5: Age vs Spending Score

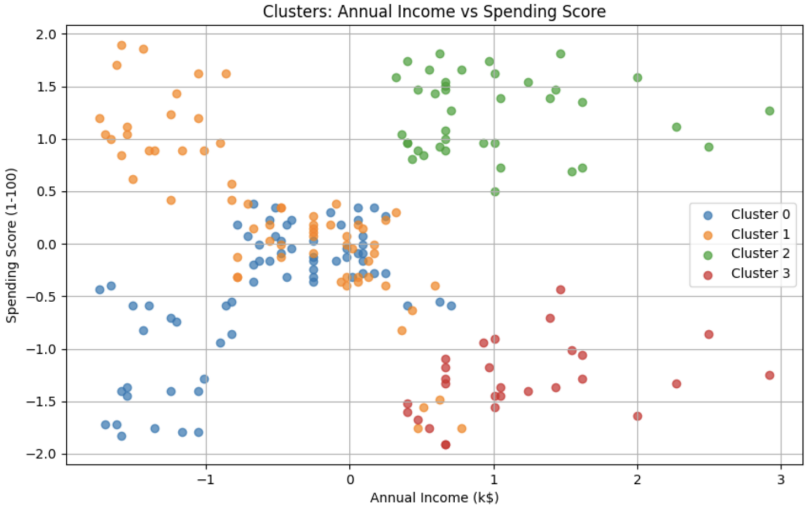

Figure 6: Income vs Spending Score

Spending Score plot 6. From that we can conclude the following for each cluster:

1. Cluster 0 (blue) : This cluster represents low spending customers with low income.
2. Cluster 1 (orange) : This cluster represents high spending customers with low income.
3. Cluster 2 (green) : This cluster represents high spending customers with high income.
4. Cluster 3 (red) : This cluster represents low spending customers with high income.

6 Real Life Use Case Scenario

The clusters we obtained from applying hierarchical clustering helped us segment customers into 4 different groups according to their spending habits. So now, for each of these clustered users, we can tailor our marketing strategies as a business owner. Example:

1. Cluster 0: For these low income, low spending customers, we can promote more deals on clearance items.
2. Cluster 1: For these high spending, low income customers, we could potentially promote payment plans for their big purchases.
3. Cluster 2: For these high spending, high income customers, we can promote big and exclusive deals and start a loyalty program as they are the most important customers for business.

7 Conclusion

This tutorial is aimed to demonstrate the use of hierarchical clustering for customer segmentation. And by leveraging EDA, the Elbow Method, and Ward's linkage, this tutorial identifies meaningful customer segments that can drive marketing efforts.

References

[GRS19] Anna Großwendt, Heiko Röglin, and Melanie Schmidt. Analysis of ward's method. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 2939–2957. SIAM, 2019.

[HR20] Hestry Humaira and R Rasyidah. Determining the appropiate cluster number using elbow method for k-means algorithm. In *Proceedings of the 2nd Workshop on Multidisciplinary and Applications (WMA) 2018, 24-25 January 2018, Padang, Indonesia*, 2020.

[MC17] Fionn Murtagh and Pedro Contreras. Algorithms for hierarchical clustering: an overview, ii. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 7(6):e1219, 2017.

[Mül11] Daniel Müllner. Modern hierarchical, agglomerative clustering algorithms. *arXiv preprint arXiv:1109.2378*, 2011.