STATS/CSE 780 - Homework Assignment 3

Name: Amanpreet Singh (400672477)

2025-11-07

Assignment Questions

1. Dataset Selection and Description

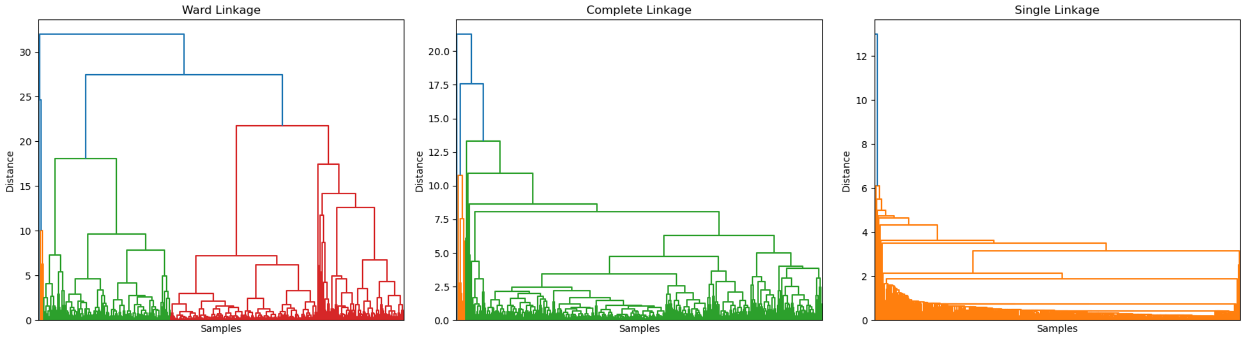
The dataset used in this study is the Wholesale Customers Dataset, obtained from the UCI Machine Learning Repository. It contains information about 440 wholesale clients of a Portuguese distributor, with the goal of segmenting customers based on their annual spending patterns. The dataset includes eight attributes, of which six are continuous variables representing annual expenditures on different product categories, Fresh, Milk, Grocery, Frozen, Detergents_Paper, and Delicatessen and two are categorical variables, Channel (Horeca or Retail) and Region (Lisbon, Oporto, or Other). For the purpose of this analysis, the categorical variables were dropped, and only the continuous features were used. The data is clean, with no missing values. The continuous features were scaled using StandardScaler to ensure that all variables contribute equally to the analysis.

Clustering is suitable for this dataset because the primary objective is to group customers with similar purchasing behaviors without any predefined labels. By identifying natural groupings, businesses can better understand customer segments, tailor marketing strategies, and optimize inventory planning.

2. Cluster Analysis

Hierarchical Clustering

The Agglomerative Hierarchical Clustering analysis on the dataset was conducted using various linkage methods to explore how distance metrics affect cluster formation. The dendrograms for Ward, Complete, and Single linkages reveal distinct hierarchical structures, with Ward linkage producing more balanced, compact clusters, while Single linkage exhibited the typical "chaining" effect, where points are sequentially linked based on minimal pairwise distances rather than overall group cohesion. Although the Single and Complete linkage methods achieved the highest silhouette score of 0.8638 for two clusters, this can be misleading, the high score reflects the presence of very tight subgroups and large inter-cluster gaps, not necessarily meaningful cluster compactness. In practice, such chaining can occur when a few customers have extreme spending behaviors that "bridge" between groups, artificially connecting distant observations into one elongated cluster. In contrast, the Ward linkage with a silhouette score of 0.7925 produced more interpretable clusters by minimizing within-cluster variance. Moreover, increasing the number of clusters to three reduced silhouette scores across all methods, suggesting that the data naturally forms two main customer groups with distinct purchasing patterns



K-Means Clustering

Building on the hierarchical clustering results, which showed that the data fits best into two clear clusters, K-Means was used to check if a centroid-based method could find a similar pattern. The results showed slightly better performance for three clusters (silhouette = 0.4583) than for two (silhouette = 0.3998), but both were much lower than the hierarchical scores. This suggests that K-Means could not capture the same strong separation found earlier, likely because customer spending patterns are irregular and not spherical in shape. In real terms, many customers share similar buying habits but differ in overall spending volume, causing overlap between groups.

K-Means Clustering After PCA

To further improve clustering performance, Principal Component Analysis (PCA) was applied to reduce dimensionality while retaining around 85% of the explained variance within the first two components. After transforming the data, K-Means clustering was performed on the reduced features. The best result was achieved using two clusters with two principal components, giving a silhouette score of 0.6984, which is lower than the best hierarchical score but significantly higher than K-Means on the original data. This shows that PCA helped remove noise and redundant information, leading to better separation of customers in the reduced space. However, increasing the number of components or clusters did not further improve results, indicating that the main structure of the data can already be captured effectively by two components.

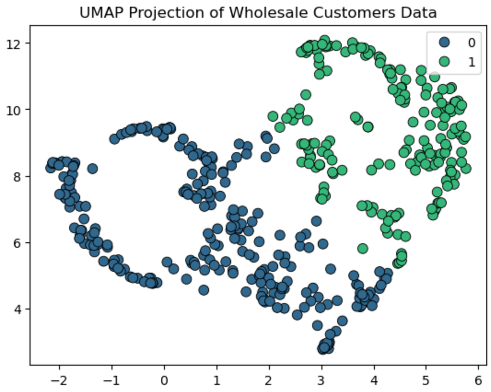
3. Clustering methods comparisions

The clustering analysis reveals notable differences between the methods. First, Agglomerative Clustering produces the most distinct and well-separated clusters, with ward linkage capturing the natural groupings of customers based on spending patterns. In contrast, KMeans shows more overlap between clusters, reflecting its reliance on centroids and the assumption of spherical clusters, which may not match the true distribution of the data. Second, comparing hierarchical clustering with PCA, the latter reduces cluster separation because projecting high-dimensional data into fewer dimensions can obscure subtle differences, whereas Agglomerative Clustering retains the full structure of the data. These comparisons indicate that hierarchical clustering best preserves natural groupings, while KMeans and PCA provide complementary perspectives that highlight global trends and finer variations in customer behavior.

4. UMAP Visualization and Discussion

Compared to Agglomerative Clustering, UMAP captures the overall separation of customers but produces slightly less compact clusters, with some points bridging groups. Its silhouette score of 0.4312 indicates moderate separation, reflecting local overlap caused by the nonlinear embedding. Overall, UMAP provides an intuitive visualization that aligns with hierarchical clustering while highlighting the dataset's inherent structure and variability.

A comparison with the Channel variable shows that Cluster 0 mostly contains Channel 1 customers (248 out of 256) and Cluster 1 mostly contains Channel 2 customers (134 out of 184), suggesting that UMAP clusters roughly correspond to the Horeca vs. Retail segmentation while still capturing some cross-channel variability.



References

Dua, Dheeru, and Casey Graff. 2019. "UCI Machine Learning Repository: Wholesale Customers Data Set." https://archive.ics.uci.edu/ml/datasets/wholesale+customers.

Jain, Anil K. 2010. "Data Clustering: 50 Years Beyond k-Means." *Pattern Recognition Letters* 31 (8): 651–66.

Kaufman, L., and P. J. Rousseeuw. 2009. *Finding Groups in Data: An Introduction to Cluster Analysis*. Wiley.

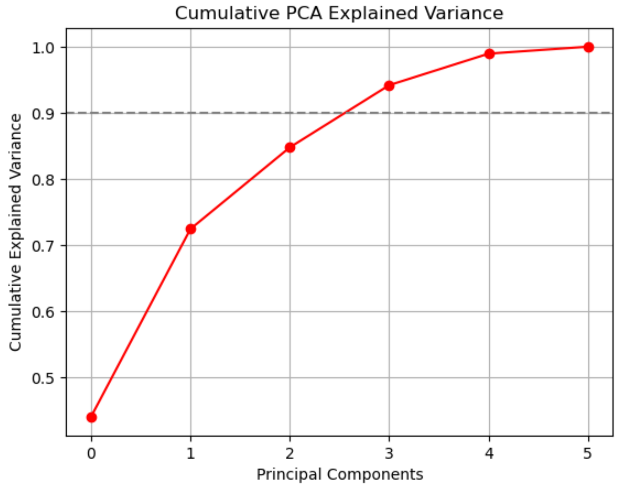
McInnes, Leland, John Healy, and James Melville. 2018. "UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction." https://arxiv.org/abs/1802.03426.

Rousseeuw, Peter J. 1987. "Silhouettes: A Graphical Aid to the Interpretation and Validation of Cluster Analysis." *Journal of Computational and Applied Mathematics* 20: 53–65.

(Dua and Graff 2019) (Jain 2010) (Kaufman and Rousseeuw 2009) (McInnes, Healy, and Melville 2018) (Rousseeuw 1987)

Supplemental Material

- Note: GitHub Copilot was used to assist with code generation and error handling.



```
# Data manipulation
import pandas as pd
import numpy as np

# Visualization
import matplotlib.pyplot as plt
import seaborn as sns

# Preprocessing
from sklearn.preprocessing import StandardScaler

# Clustering
from sklearn.cluster import KMeans, AgglomerativeClustering
from scipy.cluster.hierarchy import linkage, dendrogram

# Evaluation
from sklearn.metrics import silhouette_score

# Dimensionality Reduction
from sklearn.decomposition import PCA
from umap.umap_ import UMAP
```

```
df = pd.read_csv("Wholesale customers data.csv")
```

```
df.shape
df.columns
df.head()
```

	Channel	Region	Fresh	Milk	Grocery	Frozen	Detergents_Paper	Delicassen
0	2	3	12669	9656	7561	214	2674	1338
1	2	3	7057	9810	9568	1762	3293	1776
2	2	3	6353	8808	7684	2405	3516	7844
3	1	3	13265	1196	4221	6404	507	1788
4	2	3	22615	5410	7198	3915	1777	5185

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 440 entries, 0 to 439
Data columns (total 8 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   Channel           440 non-null    int64
 1   Region            440 non-null    int64
 2   Fresh             440 non-null    int64
 3   Milk              440 non-null    int64
 4   Grocery           440 non-null    int64
 5   Frozen            440 non-null    int64
 6   Detergents_Paper  440 non-null    int64
 7   Delicassen        440 non-null    int64
dtypes: int64(8)
memory usage: 27.6 KB
```

```
# dropping categorical variables
df = df.drop(columns=["Channel", "Region"])
```

```
scaler = StandardScaler()
X_scaled = scaler.fit_transform(df)
```

```
# Agglomerative Clustering

linkages = ['single', 'ward', 'complete', 'average']
clusters = [2, 3]
results = []
for link in linkages:
    for n in clusters:
        # 'ward' only supports Euclidean distance, so skip if not allowed
        if link == 'ward' and n < 2:
            continue
        agg = AgglomerativeClustering(n_clusters=n, linkage=link)
        labels = agg.fit_predict(X_scaled)
        score = silhouette_score(X_scaled, labels)
        results.append({'Clusters': n, 'Linkage': link, 'Silhouette Score': score})

# make a neat table
df_results = pd.DataFrame(results).sort_values(by='Silhouette Score', ascending=False)
print(df_results.to_string(index=False))
```

```
 Clusters  Linkage  Silhouette Score
        2   single          0.863801
        2 complete          0.863801
        2  average          0.863801
        3   single          0.796648
        2     ward          0.792457
        3  average          0.767580
        3 complete          0.711531
        3     ward          0.264609
```

```
# Define linkage methods
methods = ['ward', 'complete', 'single']

plt.figure(figsize=(18, 5))

# Generate a subplot for each linkage method
for i, method in enumerate(methods, 1):
    plt.subplot(1, 3, i)
    Z = linkage(X_scaled, method=method)
    dendrogram(Z, no_labels=True)
    plt.title(f'{method.capitalize()} Linkage')
    plt.xlabel('Samples')
    plt.ylabel('Distance')

plt.tight_layout()
plt.show()
```

```
results = []

# test for 2 and 3 clusters
for n in [2, 3]:
    kmeans = KMeans(n_clusters=n, random_state=42)
    labels = kmeans.fit_predict(X_scaled)
    score = silhouette_score(X_scaled, labels)
    results.append({'Clusters': n, 'Model': 'KMeans', 'Silhouette Score': score})

# make a DataFrame and sort descending
df_kmeans = pd.DataFrame(results).sort_values(by='Silhouette Score', ascending=False)
print(df_kmeans.to_string(index=False))
```

```
 Clusters  Model  Silhouette Score
        3 KMeans          0.458263
        2 KMeans          0.399828
```

```python
results = []

# Try PCA with 2 and 3 components
for n_pc in [2, 3]:
    pca = PCA(n_components=n_pc)
    X_pca = pca.fit_transform(X_scaled)

    # Cluster on reduced data
    for n_clusters in [2, 3]:
        kmeans_pca = KMeans(n_clusters=n_clusters, random_state=42)
        labels = kmeans_pca.fit_predict(X_pca)
        score = silhouette_score(X_pca, labels)

        results.append({
            'PCs': n_pc,
            'Clusters': n_clusters,
            'Silhouette Score': score
        })

# Display results
df_pca = pd.DataFrame(results).sort_values(by='Silhouette Score', ascending=False)
print(df_pca.to_string(index=False))
```

```
 PCs  Clusters  Silhouette Score
   2         2          0.698393
   3         2          0.577820
   3         3          0.390530
   2         3          0.381729
```

```python
pca = PCA().fit(X_scaled)
cum_var = np.cumsum(pca.explained_variance_ratio_)

plt.plot(cum_var, 'o-', color='red')
plt.axhline(0.9, color='gray', ls='--')
plt.xlabel('Principal Components')
plt.ylabel('Cumulative Explained Variance')
plt.title('Cumulative PCA Explained Variance')
plt.grid(True)
plt.show()
```

```
# UMAP + Clustering + Visualization
X_umap = UMAP(n_neighbors=15, min_dist=0.1, random_state=42).fit_transform(X_scaled)
labels = AgglomerativeClustering(n_clusters=2, linkage='ward').fit_predict(X_umap)

sns.scatterplot(x=X_umap[:,0], y=X_umap[:,1], hue=labels, palette="viridis", s=60, edgecolor='
plt.title("UMAP Projection of Wholesale Customers Data")
plt.show()
```

```
comparison = pd.crosstab(labels, df['Channel'])
print(comparison)
```

```
Channel    1    2
row_0
0        248    8
1         50  134
```

```
print(f"Silhouette Score: {silhouette_score(X_umap, labels):.4f}")
```

```
Silhouette Score: 0.4312
```