DS203 : Exercise 5

Nirav Bhattad

Last updated September 11, 2024

§1. Step 1

Step 1

Enumerate and explain three measures (metrics) that you will use to assess the outcome of clustering algorithms.

In this exercise, we have been given 3 datasets whith points in $\mathbb{R}^2$, and we have not been given the label for the points. We have to use unsupervised learning algorithms to cluster the points. We will use the following metrics to assess the outcome of clustering algorithms:

Silhouette Score

The **Silhouette Score** measures how well each data point fits within its assigned cluster compared to other clusters. It provides insight into both the cohesion and separation of clusters.

Definition

For a given data point i, the Silhouette Score $s(i)$ is calculated as follows:

$$s(i) = \frac{b(i) - a(i)}{\max(a(i), b(i))}$$

where:

- $a(i)$ is the average distance between i and all other points in the same cluster (cohesion).
- $b(i)$ is the minimum average distance between i and all points in any other cluster (separation).

Interpretation

The Silhouette Score ranges from -1 to 1:

- A score close to 1 indicates that the data point is well-matched to its own cluster and poorly matched to neighboring clusters.
- A score close to 0 indicates that the data point is on or very close to the decision boundary between two neighboring clusters.
- A negative score indicates that the data point might have been assigned to the wrong cluster.

The overall Silhouette Score is the mean of the individual scores. A high average score suggests that the clustering configuration is appropriate and clusters are well-separated.

Davies-Bouldin Index

The **Davies-Bouldin Index** evaluates the separation and compactness of clusters. It measures the average similarity between each cluster and its most similar one.

Definition

The Davies-Bouldin Index DBI is defined as:

$$DBI = \frac{1}{k} \sum_{i=1}^{k} \max_{j \neq i} \frac{s_i + s_j}{d_{ij}}$$

where:

- k is the number of clusters.
- s_i and s_j are the average distances within clusters i and j respectively.
- d_{ij} is the distance between the centroids of clusters i and j.

Interpretation

The Davies-Bouldin Index is always non-negative. Lower values indicate better clustering solutions:

- A lower DBI signifies that the clusters are more distinct and compact.
- A higher DBI indicates that the clusters are less distinct and more similar to each other.

The index helps in identifying clustering solutions where clusters are well-separated and compact.

Calinski-Harabasz Index

The **Calinski-Harabasz Index** assesses cluster separation and compactness by evaluating the ratio of between-cluster dispersion to within-cluster dispersion.

Definition

The Calinski-Harabasz Index CH is defined as:

$$CH = \frac{\frac{1}{k-1} \sum_{i=1}^{k} n_i (\bar{\mathbf{x}}_i - \bar{\mathbf{x}})^2}{\frac{1}{n-k} \sum_{i=1}^{n} (\mathbf{x}_i - \bar{\mathbf{x}}_i)^2}$$

where:

- n_i is the number of data points in cluster i and n is the total number of data points.
- $\bar{\mathbf{x}}_i$ is the centroid of cluster i.
- $\bar{\mathbf{x}}$ is the overall mean of the data.

Interpretation

The Calinski-Harabasz Index typically increases as the number of clusters increases, so it is used to compare different clustering solutions:

- A higher CH index indicates that the clusters are well-separated and compact.
- A lower CH index suggests that the clusters are less distinct and more dispersed.

The index is useful for selecting the optimal number of clusters by comparing different clustering solutions.

Step 2

Step 2

Review the datasets using appropriate plots.

Dataset 1

We see that the points in Dataset 1 are distributed in four clusters. The clusters are well-separated and have a clear boundary. The points in each cluster are tightly packed, and the clusters are easily distinguishable.

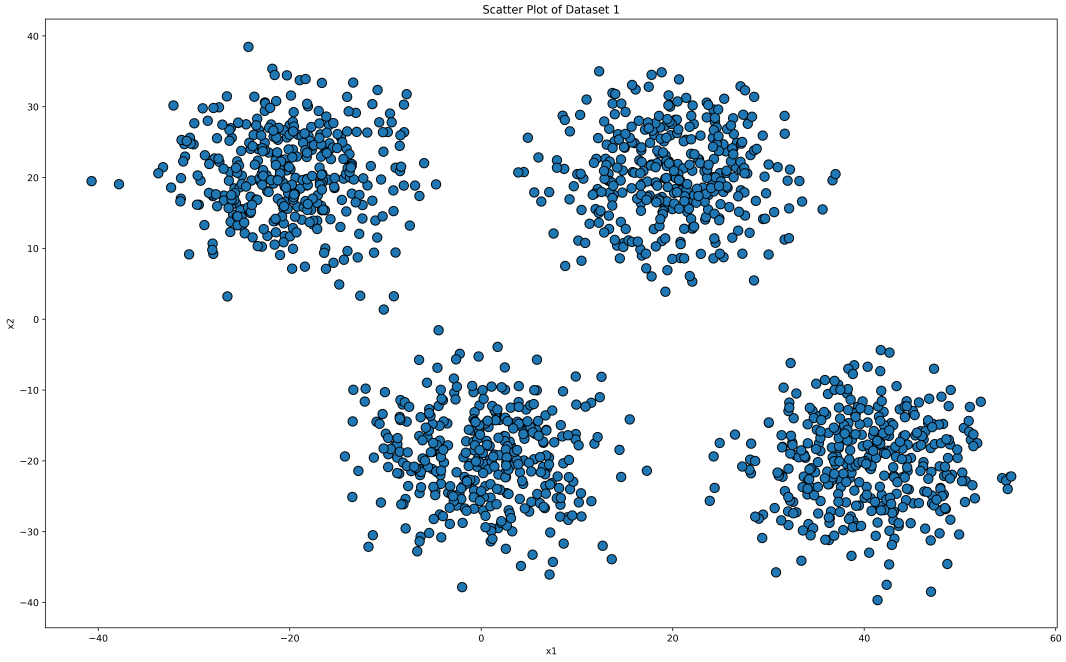

Figure 1: Dataset 1 Overview

Dataset 2

The points in Dataset 2 are distributed in four clusters. The clusters are less well-separated compared to Dataset 1. The points in each cluster are more spread out, and the boundary between clusters is not as clear.

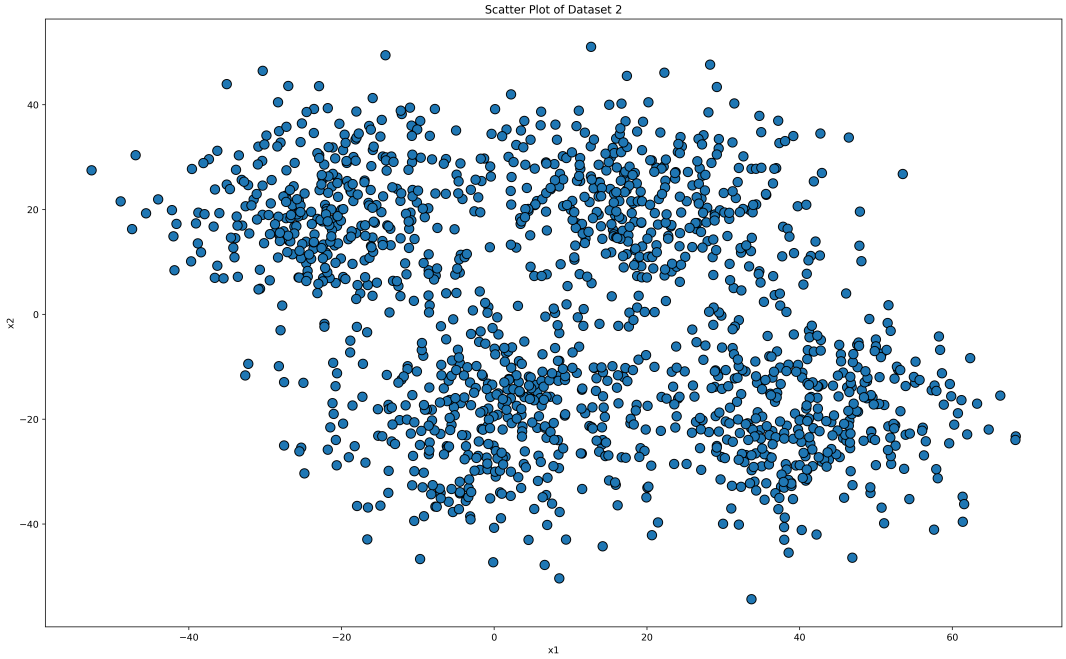

Figure 2: Dataset 2 Overview

Dataset 3

The points in Dataset 3 are heavily overlapped and do not form distinct clusters. The points are scattered across the plot without any clear separation. It is challenging to identify clusters based on visual inspection alone.

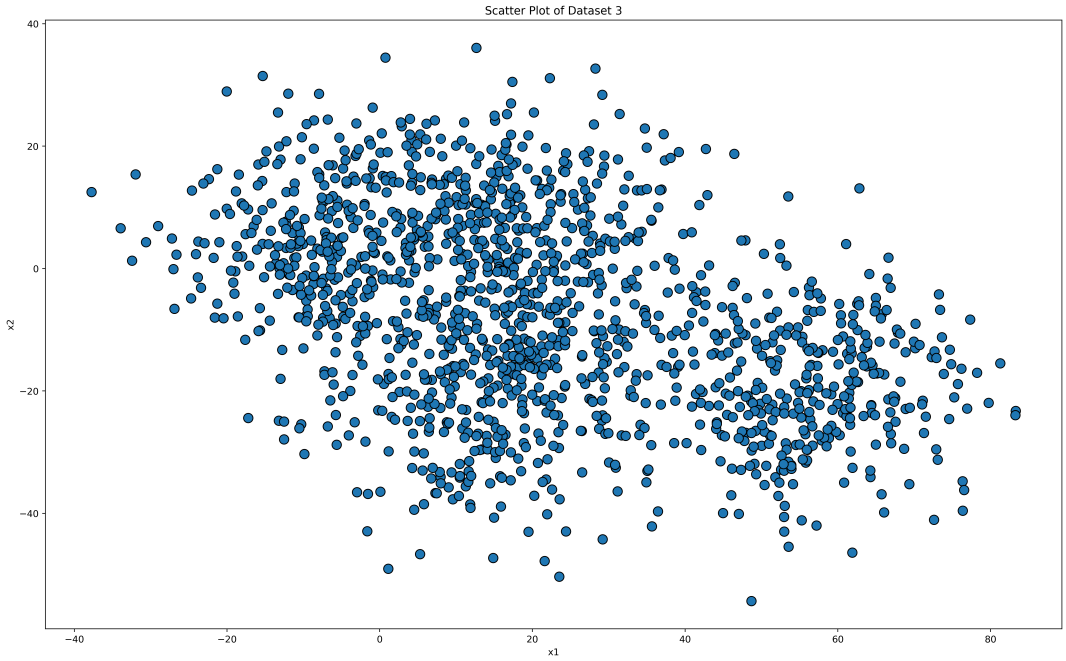

Figure 3: Dataset 3 Overview

Here is the code used to generate the plots:

```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.colors import ListedColormap
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans, AgglomerativeClustering, DBSCAN
from sklearn.metrics import silhouette_score, calinski_harabasz_score, davies_bouldin_score,
    adjusted_rand_score, confusion_matrix
import os

data_v0 = pd.read_csv('clusters-5-v0.csv')
data_v1 = pd.read_csv('clusters-5-v1.csv')
data_v2 = pd.read_csv('clusters-5-v2.csv')

if not os.path.exists('Images'):
    os.makedirs('Images')

if not os.path.exists('Metrics'):
    os.makedirs('Metrics')

dataset = {
    'Clusters-5-v0': pd.read_csv('clusters-5-v0.csv'),
    'Clusters-5-v1': pd.read_csv('clusters-5-v1.csv'),
    'Clusters-5-v2': pd.read_csv('clusters-5-v2.csv')
}

scalers = {name: StandardScaler().fit_transform(data[['x1', 'x2']]) for name, data in dataset.items
    ()}

datasets = [(data_v0, "Dataset 1"), (data_v1, "Dataset 2"), (data_v2, "Dataset 3")]

for i, (data, label) in enumerate(datasets):
    plt.figure(figsize=(16, 10))

    plt.scatter(data['x1'], data['x2'], s=100, edgecolors='black')
    plt.title(f'Scatter Plot of {label}')
    plt.xlabel('x1')
    plt.ylabel('x2')

    plt.tight_layout()
    plt.savefig(f'Images/dataset-{i+1}-overview.png', dpi=400)
    plt.show()
```

Listing 1: Code to Generate Dataset Plots

Remark

We use `os` to create directories for saving images and metrics. We then load the datasets and create a dictionary of datasets. We use the `StandardScaler` to standardize the data. We iterate over each dataset to create scatter plots of the data points.

Step 3

Step 3

Use the following algorithms to cluster the observations is each of the dataset. Plot the outcomes.
a. K-Means Clustering
b. Agglomerative Clustering
c. DBSCAN

Here, we will use three clustering algorithms to cluster the observations in each dataset. We will plot the outcomes to visualize the clustering results.

```python
def plot_clusters(data, labels, title, name):
    plt.figure(figsize=(16, 10))
    plt.scatter(data[:, 0], data[:, 1], c=labels, cmap='viridis', s=40, edgecolors='black')
    plt.title(title)
    plt.xlabel('x1')
    plt.ylabel('x2')
    plt.savefig(f'Images/{name}-{title}.png', dpi=400)
    plt.show()

def run_clustering(data, name):
    results = []

    kmeans = KMeans(n_clusters=4, random_state=42)
    kmeans_labels = kmeans.fit_predict(data)
    plot_clusters(data, kmeans_labels, 'K-Means Clustering', name)
    kmeans_metrics = [
        'KMeans',
        silhouette_score(data, kmeans_labels),
        calinski_harabasz_score(data, kmeans_labels),
        davies_bouldin_score(data, kmeans_labels)
    ]
    results.append(kmeans_metrics)

    agglomerative = AgglomerativeClustering(n_clusters=4)
    agglomerative_labels = agglomerative.fit_predict(data)
    plot_clusters(data, agglomerative_labels, 'Agglomerative Clustering', name)
    agglomerative_metrics = [
        'AgglomerativeClustering',
        silhouette_score(data, agglomerative_labels),
        calinski_harabasz_score(data, agglomerative_labels),
        davies_bouldin_score(data, agglomerative_labels)
    ]
    results.append(agglomerative_metrics)

    eps_values = np.linspace(0.01, 4, 200)
    min_samples_values = range(4, 21)

    best_dbscan_labels = None
    best_dbscan_score = -np.inf
    best_eps, best_min_samples = None, None

    for eps in eps_values:
        for min_samples in min_samples_values:
            dbscan = DBSCAN(eps=eps, min_samples=min_samples)
            dbscan_labels = dbscan.fit_predict(data)
            unique_labels = len(set(dbscan_labels) - {-1})

            if unique_labels == 4:
                score = silhouette_score(data, dbscan_labels)
                if score > best_dbscan_score:
                    best_dbscan_score = score
                    best_dbscan_labels = dbscan_labels
                    best_eps = eps
```

```
                    best_min_samples = min_samples

    if best_dbscan_labels is not None:
        plot_clusters(data, best_dbscan_labels, 'DBSCAN Clustering', name)
        dbscan_metrics = [
            'DBSCAN',
            best_dbscan_score,
            calinski_harabasz_score(data, best_dbscan_labels),
            davies_bouldin_score(data, best_dbscan_labels)
        ]
        print(f"Best DBSCAN parameters: eps={best_eps}, min_samples={best_min_samples}")
    else:
        dbscan_metrics = ['DBSCAN', 'N/A', 'N/A', 'N/A']
        print("DBSCAN could not find exactly 4 clusters.")

    results.append(dbscan_metrics)

    result_df = pd.DataFrame(results, columns=['Algorithm', 'Silhouette Score', 'Calinski-Harabasz
        Score', 'Davies-Bouldin Score'])
    result_df.to_csv(f'Metrics/{name}-metrics.csv', index=False)

    plt.figure(figsize=(16, 10))
    plt.plot(result_df['Algorithm'], result_df['Silhouette Score'], marker='o', label='Silhouette
        Score')
    plt.plot(result_df['Algorithm'], result_df['Calinski-Harabasz Score'], marker='o', label='
        Calinski-Harabasz Score')
    plt.plot(result_df['Algorithm'], result_df['Davies-Bouldin Score'], marker='o', label='Davies-
        Bouldin Score')
    plt.title(f'Metrics for {name}')
    plt.yscale('log')
    plt.ylabel('Score (Log Scale)')
    plt.xlabel('Algorithm')
    plt.legend()
    plt.savefig(f'Metrics/{name}-metrics.png', dpi=400)
    plt.show()

    return results

for name, data in dataset.items():
    scaled_data = scalers[name]
    clustering_results = run_clustering(scaled_data, name)
```

Listing 2: Code to Cluster Datasets and generate Metrics

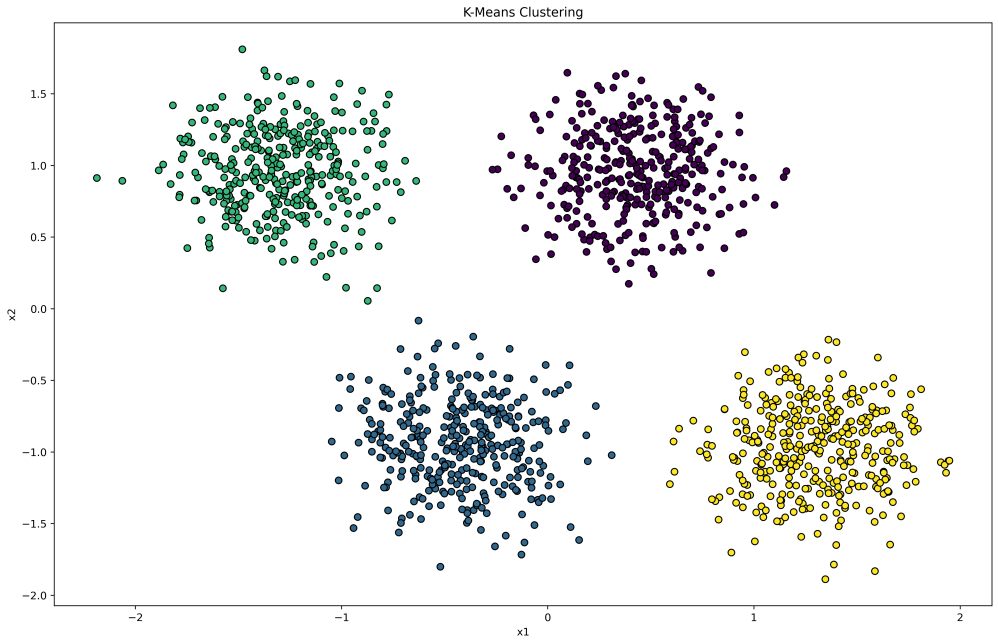

Figure 4: Dataset 1: K-Means Clustering

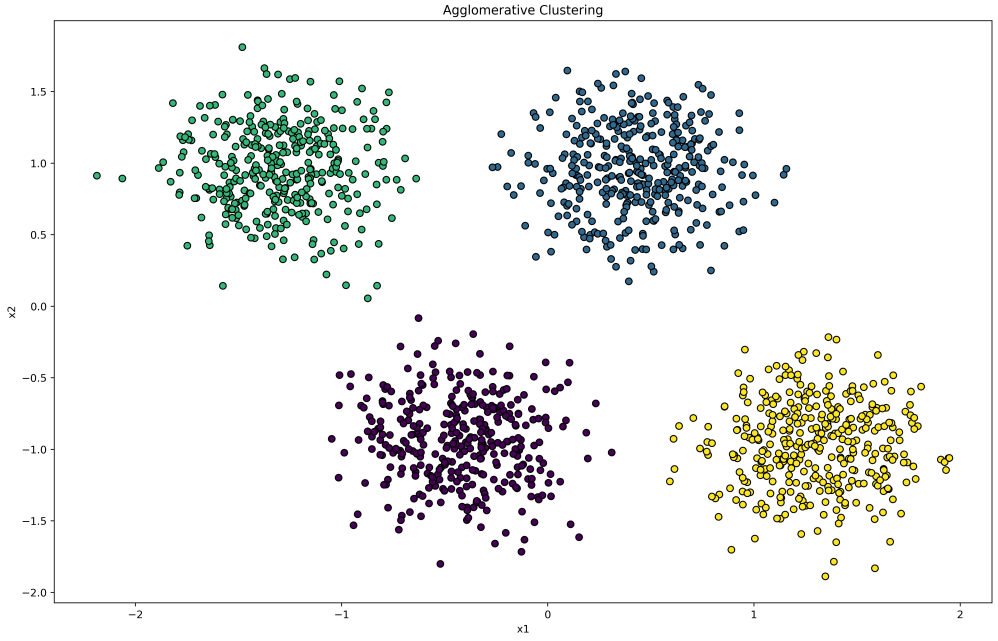

Figure 5: Dataset 1: Agglomerative Clustering

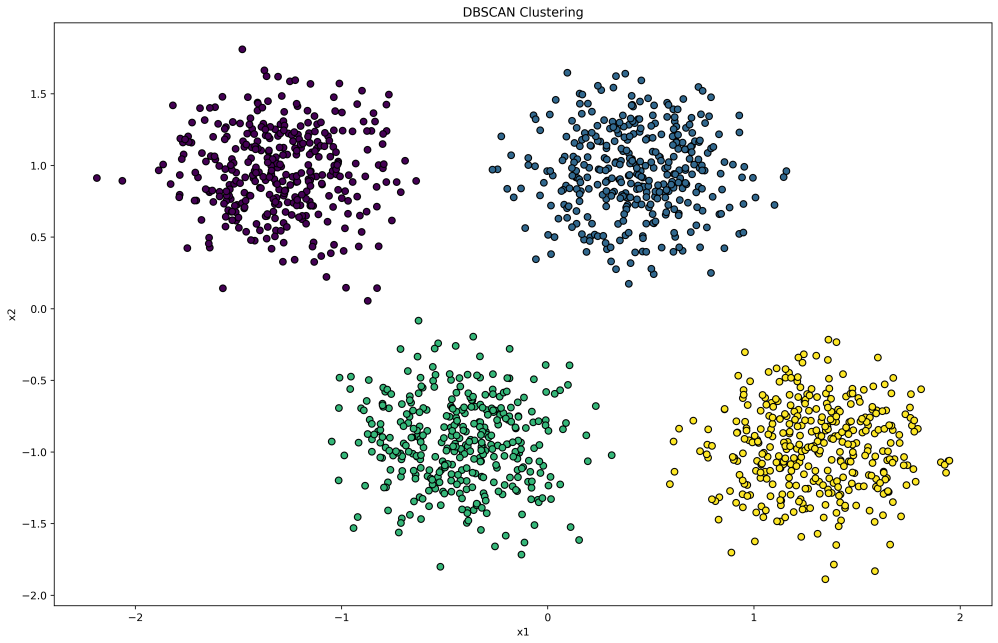

Figure 6: Dataset 1: DBSCAN Clustering

We observe that all 3 clustering algorithms have successfully clustered the observations in Dataset 1. The K-Means and Agglomerative Clustering algorithms have created well-defined clusters. The DBSCAN algorithm has also clustered the observations effectively.

All 3 algorithms give almost the same classification results, with the DBSCAN algorithm having the highest Silhouette Score. The DBSCAN algorithm has found the best parameters for the dataset, which is evident from the high Silhouette Score.

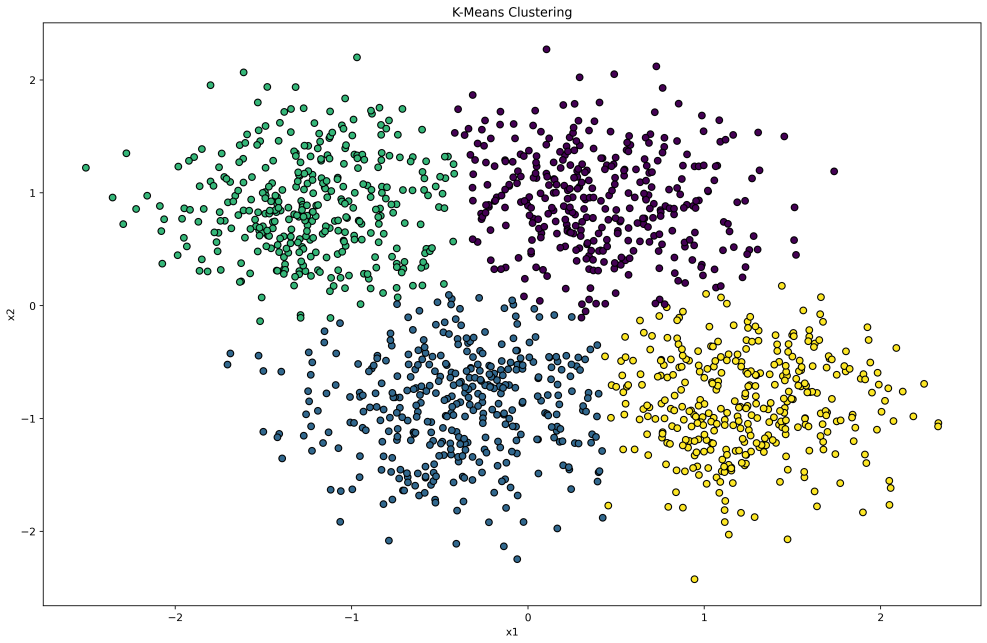

Figure 7: Dataset 2: K-Means Clustering

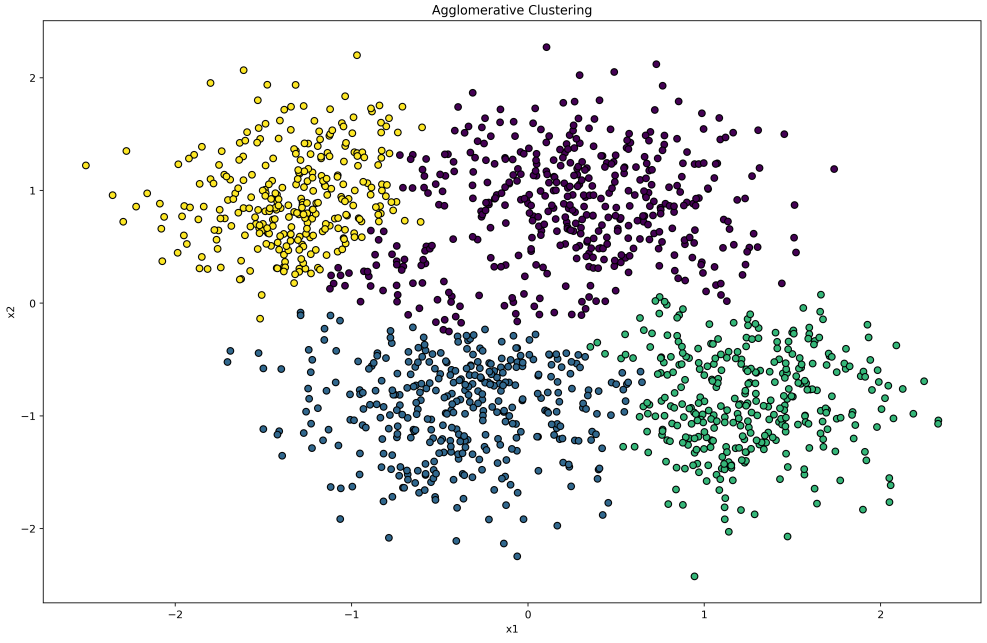

Figure 8: Dataset 2: Agglomerative Clustering

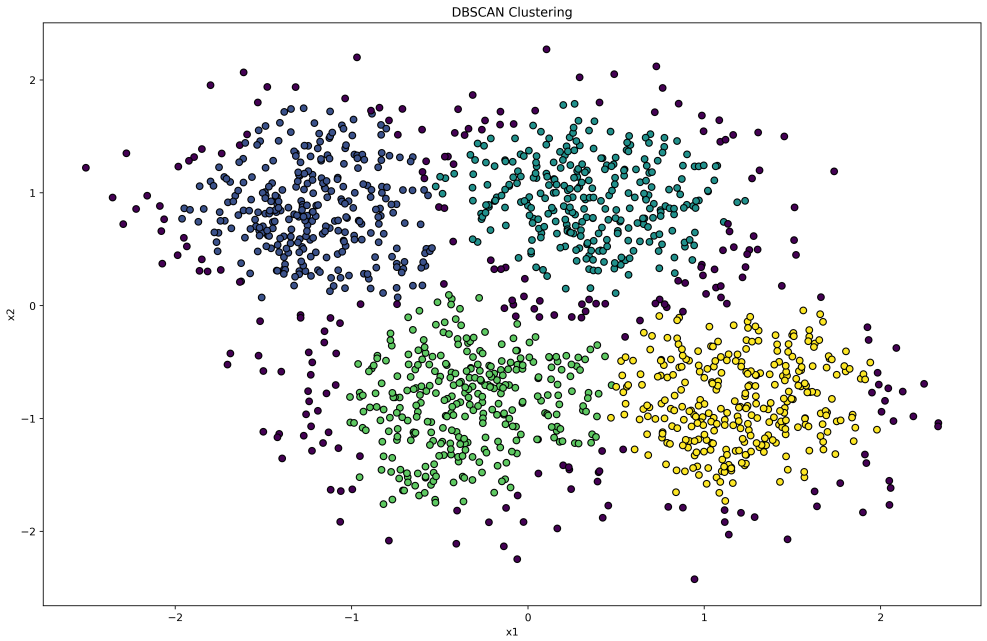

Figure 9: Dataset 2: DBSCAN Clustering

We observe that K-Means and Agglomerative Clustering algorithms give satisfactory classifications for Dataset 2. The DBSCAN algorithm has not been able to find exactly 4 clusters for this dataset.

The Silhouette Score for DBSCAN is lower than that of K-Means and Agglomerative Clustering, indicating that the DBSCAN algorithm has not performed as well as the other two algorithms for this dataset.

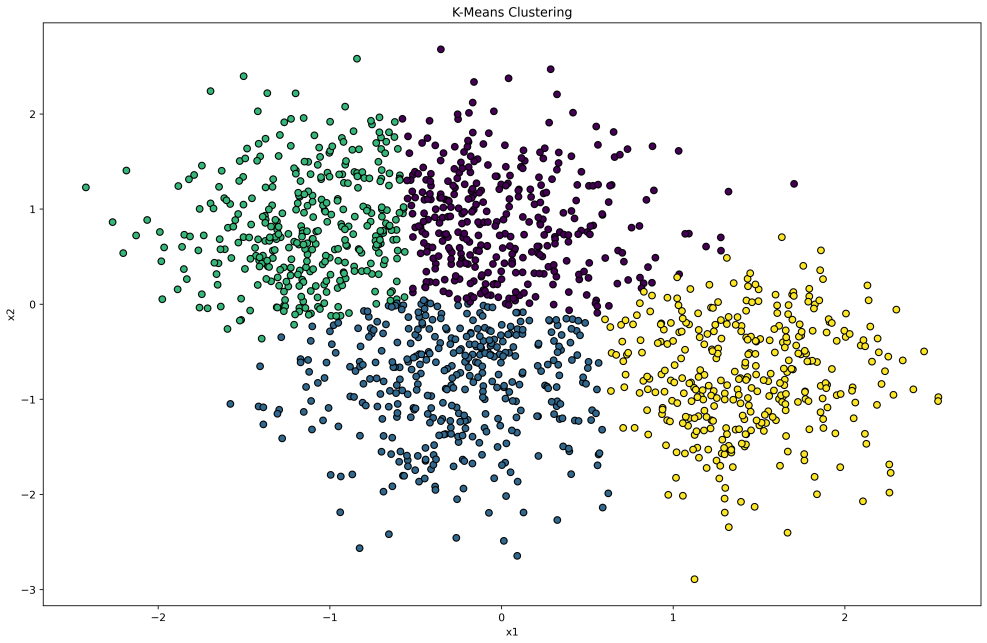

Figure 10: Dataset 3: K-Means Clustering

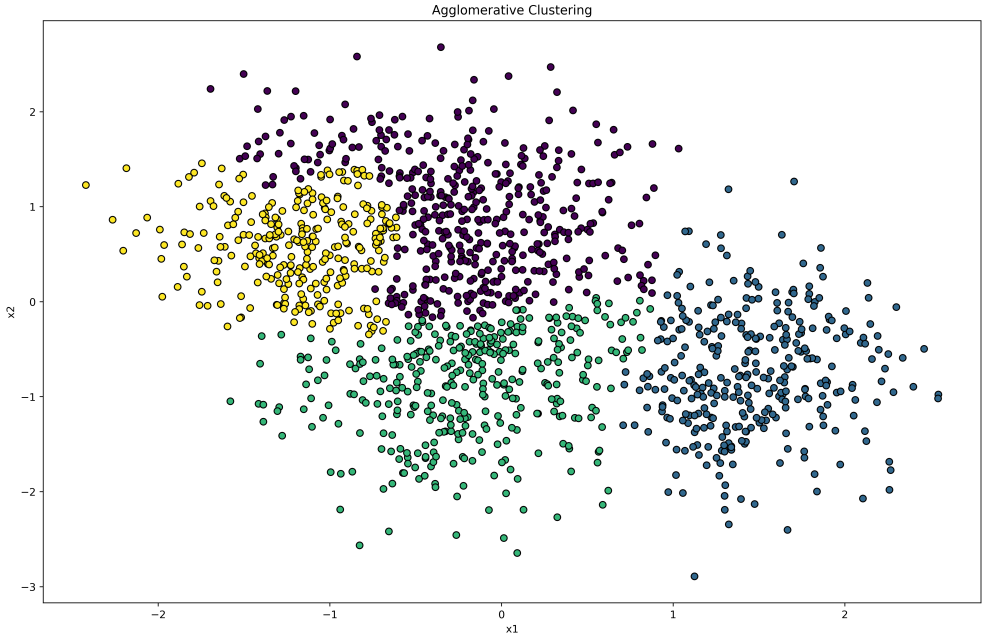

Figure 11: Dataset 3: Agglomerative Clustering

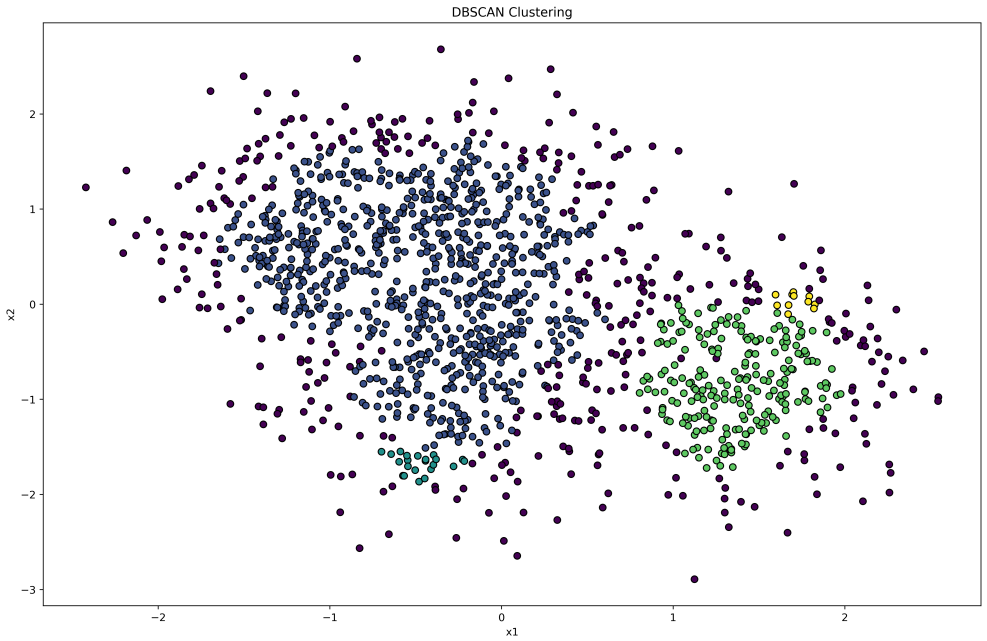

Figure 12: Dataset 3: DBSCAN Clustering

For Dataset 3, we observe that none of the clustering algorithms have been able to cluster the observations effectively. The points are heavily overlapped and do not form distinct clusters. The DBSCAN algorithm has not been able to find exactly 4 clusters for this dataset.

The Silhouette Score for all three algorithms is close to 0, indicating that the points are on or very close to the decision boundary between clusters. This suggests that the clustering algorithms have not performed well for this dataset.

Step 4

Step 4

Calculate and analyze the metrics you have listed in '1' above. Analyze them across the datasets and methods. Create appropriate plots and explain the trends, if any, and the outcomes of your analysis. What can you conclude (Please don't state the obvious !!)

Dataset 1

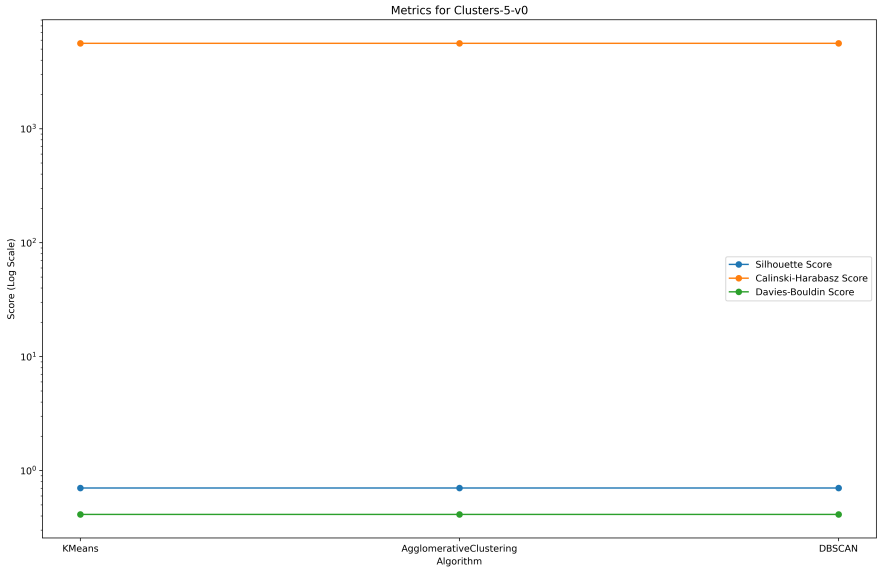

Figure 13: Dataset 1: Clustering Metrics

Algorithm	Silhouette Score	Calinski-Harabasz Score	Davies-Bouldin Score
KMeans	0.703486063484015	5615.26685159197	0.412197346084185
AgglomerativeClustering	0.703486063484015	5615.26685159197	0.412197346084185
DBSCAN	0.703486063484015	5615.26685159197	0.412197346084185

Figure 14: Dataset 1: Clustering Metrics

The error metrics for all 3 algorithms are exactly the same, indicating that the clustering quality is consistent across the algorithms. The Silhouette Scores are high, indicating well-separated clusters. The Calinski-Harabasz Scores are also high, suggesting compact and well-separated clusters. The Davies-Bouldin Scores are low, indicating good cluster similarity.

Dataset 2

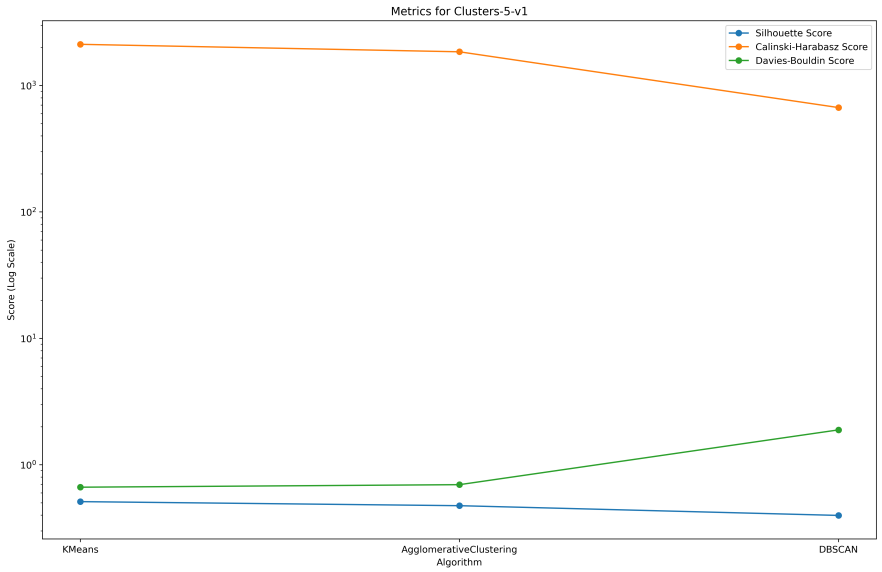

Figure 15: Dataset 2: Clustering Metrics

Algorithm	Silhouette Score	Calinski-Harabasz Score	Davies-Bouldin Score
KMeans	0.511144811061377	2119.10640216187	0.66484975269678
AgglomerativeClustering	0.474718604848112	1849.13457809708	0.696198603496482
DBSCAN	0.397643535675022	669.583230956081	1.88865005712884

Figure 16: Dataset 2: Clustering Metrics

The Silhouette Scores for the 3 algorithms reduces as we go from K-Means to Agglomerative to DBSCAN. This indicates that the clusters are less well-separated in Dataset 2 compared to Dataset 1. The Calinski-Harabasz Scores also decrease, suggesting less compact clusters. The Davies-Bouldin Scores increase, indicating higher cluster similarity.

Dataset 3

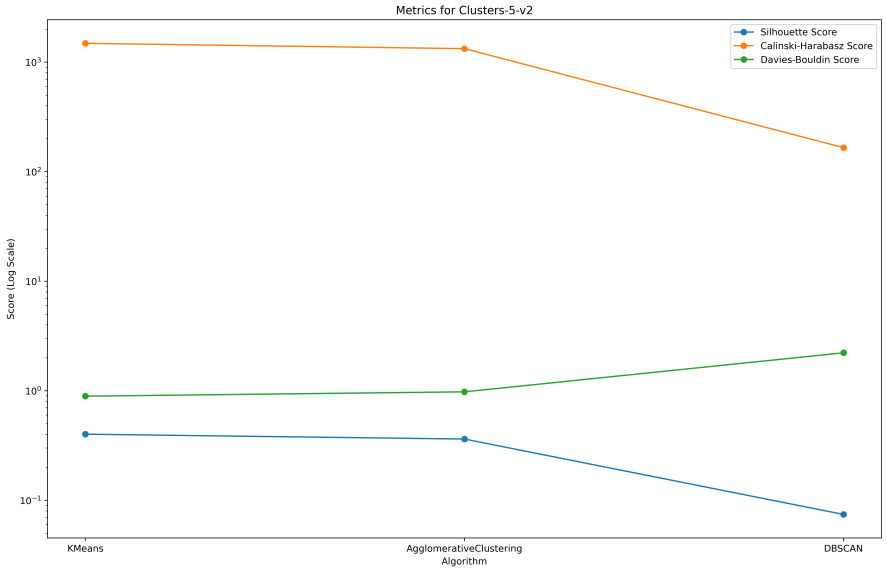

Figure 17: Dataset 3: Clustering Metrics

Algorithm	Silhouette Score	Calinski-Harabasz Score	Davies-Bouldin Score
KMeans	0.401317879276208	1488.13796031628	0.892426939829698
AgglomerativeClustering	0.362359660209324	1329.89530254051	0.977679976893552
DBSCAN	0.0742925620528895	165.988821905833	2.22132789732134

Figure 18: Dataset 3: Clustering Metrics

The Silhouette Scores for Dataset 3 are significantly lower than the other datasets, indicating poor cluster separation. The Calinski-Harabasz Scores are also the lowest, reflecting less compact clusters. The Davies-Bouldin Scores are the highest, showing higher cluster similarity.

The Silhouette Scores for each dataset and clustering algorithm provide insights into the clustering quality. In Dataset Clusters-5-v0.csv, all algorithms show high Silhouette Scores, indicating well-separated clusters. For Clusters-5-v1.csv, scores are slightly lower, reflecting some overlap. In Clusters-5-v2.csv, scores are significantly lower, indicating poor separation.

The Calinski-Harabasz Scores demonstrate the compactness and separation of clusters. Dataset Clusters-5-v0.csv exhibits high scores across algorithms, suggesting well-defined clusters. Scores for Clusters-5-v1.csv are lower, and for Clusters-5-v2.csv, they are the lowest, showing that clusters are not well-separated.

The Davies-Bouldin Scores highlight the average similarity of clusters. For Dataset Clusters-5-v0.csv, all algorithms achieve low scores, indicating distinct clusters. For Clusters-5-v1.csv, the scores are higher, suggesting some cluster overlap. Clusters-5-v2.csv shows the highest Davies-Bouldin Scores, reflecting significant cluster mixing.

Main Learnings

Understanding Clustering Algorithms

K-Means Clustering: K-Means is a partition-based algorithm that aims to minimize the variance within each cluster. It is effective for datasets with well-separated clusters but struggles with clusters of varying densities and shapes. It performed excellently on `Clusters-5-v0.csv`, which had clear boundaries, but its performance decreased for `Clusters-5-v1.csv` and `Clusters-5-v2.csv` due to overlapping and mixed clusters.

Agglomerative Clustering: Agglomerative Clustering is a hierarchical method that builds clusters by merging pairs of clusters. It does not assume a specific number of clusters and can handle various cluster shapes better than K-Means. It showed better performance on `Clusters-5-v1.csv` compared to K-Means but also faced challenges with `Clusters-5-v2.csv`, where the class mixing affected its ability to form meaningful clusters.

DBSCAN: DBSCAN is a density-based algorithm that can identify clusters of arbitrary shape and is robust to noise. It excels in datasets with varying densities and noise, such as `Clusters-5-v2.csv`. However, it requires careful tuning of parameters, such as `eps` and `min_samples`, to achieve optimal results. DBSCAN successfully identified core clusters in `Clusters-5-v2.csv` but struggled with `Clusters-5-v0.csv` where clusters are well-separated.

Evaluating Clustering Performance with Metrics

Silhouette Score: The Silhouette Score provides a measure of how similar a point is to its own cluster compared to other clusters. It highlighted the effectiveness of clustering in `Clusters-5-v0.csv`, where all algorithms performed well. For `Clusters-5-v1.csv`, the score dropped due to overlapping clusters, and for `Clusters-5-v2.csv`, it indicated poor clustering quality due to significant class mixing.

Calinski-Harabasz Score: The Calinski-Harabasz Score measures cluster compactness and separation. High scores for `Clusters-5-v0.csv` confirmed well-defined clusters. For `Clusters-5-v1.csv`, scores were lower due to some class overlap, and the lowest scores for `Clusters-5-v2.csv` indicated poor cluster separation.

Davies-Bouldin Score: The Davies-Bouldin Score assesses the average similarity between each cluster and its most similar cluster. Low scores for `Clusters-5-v0.csv` reflected distinct clusters, while higher scores for `Clusters-5-v1.csv` and `Clusters-5-v2.csv` indicated increased similarity between clusters, affecting clustering effectiveness.

Key Insights from Python Library `sklearn`

`sklearn` provides a comprehensive suite of functions for implementing and evaluating clustering algorithms. Key learnings include:

- The `KMeans` class for K-Means clustering, allowing the specification of the number of clusters and distance metrics.
- The `AgglomerativeClustering` class for hierarchical clustering with various linkage criteria.
- The `DBSCAN` class for density-based clustering, offering flexibility with parameter tuning.
- Functions like `silhouette_score`, `calinski_harabasz_score`, and `davies_bouldin_score` for metric calculations, aiding in the assessment of clustering quality.

Conclusion

The exercise provided valuable insights into the performance and characteristics of various clustering algorithms. Each algorithm has its strengths and weaknesses depending on the dataset characteristics. K-Means is best for well-separated clusters, Agglomerative Clustering handles various shapes, and DBSCAN is effective in the presence of noise and varying densities. Evaluating clustering performance with metrics such as Silhouette Score, Calinski-Harabasz Score, and Davies-Bouldin Score is crucial for understanding and improving clustering results. The use of `sklearn` functions facilitated a deeper understanding and practical application of these concepts.