Department of Computer Engineering

Bu-Ali Sina University

Machine Learning Course

Assignment 2: Comparing Various Clustering Algorithms

By:

Ali Nafisi

Course Professor:

Dr. Muharram Mansoorizadeh

Fall 2023

TABLE OF CONTENTS

1 Introduction **1**

2 Methodology **2**

2.1 K-means 2

2.2 DBSCAN 2

2.3 Agglomerative clustering 3

2.4 Gaussian mixture model (GMM) 4

3 Experimental Results **6**

3.1 Datasets 6

3.1.1 Synthetic Datasets 6

3.1.2 Real-world Dataset (Adult) 6

3.2 Evaluation Metrics 7

3.2.1 Silhouette Score 7

3.2.2 Adjusted Rand Index (ARI) 8

3.2.3 Inertia 8

3.3 Results 9

4 Conclusion **13**

1 INTRODUCTION

In recent years, clustering algorithms have become essential tools in various fields, facilitating the exploration and understanding of underlying patterns within diverse datasets. This project delves into the comparative analysis of four prominent clustering algorithms: K-means, DBSCAN (Density-Based Spatial Clustering of Applications with Noise), Agglomerative clustering, and Gaussian Mixture Model (GMM). The primary goal is to assess the performance of these algorithms across different types of datasets, encompassing both synthetic and real-world data.

Clustering, a form of unsupervised learning, involves grouping similar data points together to uncover intrinsic structures within the data. Each algorithm in focus employs distinct approaches to achieve this objective, ranging from partitioning the data space into clusters (K-means) to identifying clusters based on density (DBSCAN) and hierarchical merging of data points (Agglomerative clustering). GMM, on the other hand, leverages probabilistic models to represent data as a mixture of Gaussian distributions.

In the subsequent sections, we present the methodologies employed for each clustering algorithm, discuss the datasets used for experimentation (including synthetic datasets and a real-world dataset - 'Adult'), and outline the evaluation metrics applied to quantify the performance of the algorithms. The experimental results, presented in Section 3, provide insights into the algorithms' effectiveness in capturing underlying patterns within the diverse datasets.

This comprehensive exploration aims to contribute valuable insights into the strengths and limitations of each clustering algorithm, paving the way for informed decision-making in their application to various real-world scenarios. The project concludes with a summary of findings and potential avenues for future research in the realm of clustering algorithms.

2 METHODOLOGY

2.1 K-means

K-means is a popular clustering algorithm widely used in machine learning and data analysis. The primary objective of K-means is to partition a dataset into distinct, non-overlapping groups or clusters based on similarities among data points. The "K" in K-means refers to the number of clusters that the algorithm aims to identify within the dataset.

The algorithm operates through an iterative process that involves assigning data points to clusters and adjusting the cluster centroids until a convergence criterion is met. Initially, K-means randomly selects K data points as the initial cluster centroids. Subsequently, each data point is assigned to the cluster whose centroid is closest, typically using Euclidean distance. After the assignment step, the centroids of the clusters are recalculated based on the data points currently belonging to each cluster. This process of assignment and centroid update iterates until convergence, where the centroids remain stable or the change falls below a predefined threshold.

One challenge in K-means is determining the optimal value of K, the number of clusters. The elbow method and silhouette analysis are commonly used techniques to assist in selecting an appropriate K value. The elbow method involves plotting the sum of squared distances between data points and their assigned centroids for different values of K. The "elbow" in the resulting graph represents a point where increasing the number of clusters does not significantly decrease the sum of squared distances, suggesting an optimal K value. Silhouette analysis, on the other hand, measures how similar an object is to its own cluster compared to other clusters and provides a graphical representation to aid in selecting K.

While K-means is efficient and relatively straightforward, it has limitations. It assumes clusters with similar sizes, spherical shapes, and similar variances, which may not always align with the characteristics of real-world datasets. Additionally, the algorithm's results can be sensitive to the initial placement of centroids, and it may converge to suboptimal solutions in some cases. Despite these limitations, K-means remains a valuable tool for clustering tasks, especially in situations where the data has clear cluster structures and the assumptions of the algorithm are met.

2.2 DBSCAN

Density-Based Spatial Clustering of Applications with Noise (DBSCAN) is a clustering algorithm designed to identify clusters of arbitrary shapes in a dataset based on the density of data points. Unlike traditional clustering algorithms like K-means, DBSCAN does not assume that clusters have a specific geometric shape and can discover clusters with varying sizes and shapes. This makes it

particularly effective in scenarios where the clusters are not well-defined or have irregular boundaries.

DBSCAN works by defining two key parameters: epsilon (ε), which represents the radius around a data point, and the minimum number of points (MinPts) required within this radius for a data point to be considered a core point. The algorithm categorizes data points into three types: core points, border points, and noise points. Core points are those with at least MinPts neighbors within the specified radius ε, while border points have fewer than MinPts neighbors but fall within the ε radius of a core point. Noise points, on the other hand, do not meet the criteria for core or border points.

The algorithm starts by randomly selecting an unvisited data point and determining whether it is a core point. If so, DBSCAN expands the cluster by recursively adding reachable core points and their border points. This process continues until no more points can be added to the cluster. The algorithm then moves on to another unvisited point and repeats the process until all points are either assigned to a cluster or labeled as noise.

DBSCAN has several advantages, including its ability to discover clusters of arbitrary shapes, handling noise effectively, and not requiring a predefined number of clusters. Additionally, it is less sensitive to the order in which data points are processed. However, choosing appropriate values for ε and MinPts can be challenging, and the algorithm may struggle with datasets of varying densities or with large differences in cluster sizes.

In summary, DBSCAN is a powerful density-based clustering algorithm suitable for datasets with irregular cluster shapes and varying densities. Its flexibility makes it a valuable tool in situations where other traditional clustering algorithms might struggle to produce meaningful results.

2.3 Agglomerative clustering

Agglomerative clustering is a hierarchical clustering algorithm that builds a hierarchy of clusters by successively merging or grouping similar data points or clusters. The algorithm begins by treating each data point as an individual cluster and then iteratively combines the most similar clusters until a single overarching cluster is formed. The resulting hierarchy is often visualized as a dendrogram, which displays the sequential merging of clusters.

The process of agglomerative clustering involves defining a measure of similarity between clusters, often referred to as a linkage criterion. Popular linkage criteria include complete linkage, single linkage, and average linkage. Complete linkage measures the maximum distance between any two points in different clusters, while single linkage calculates the minimum distance between points in different clusters. Average linkage computes the average distance between all pairs of points in different clusters. The choice of linkage criterion influences the characteristics of the resulting clusters and dendrogram.

At each step of the agglomerative clustering process, the algorithm identifies the two most similar clusters based on the chosen linkage criterion and merges them into a new cluster. This process continues until all data points are part of a single cluster. The hierarchy of clusters can then be cut at a specific level to obtain a desired number of clusters.

Agglomerative clustering is versatile and can be applied to various types of data. Its hierarchical nature provides a multi-scale view of the data's structure, allowing users to explore different levels of granularity in the clustering results. However, the algorithm's computational complexity can be a drawback for large datasets, as the time complexity is typically $O(n^3)$, where n is the number of data points.

Despite its computational cost, agglomerative clustering remains a widely used algorithm, particularly when the hierarchical structure of clusters is of interest or when the data exhibits nested or overlapping clusters. The interpretability of the resulting dendrogram can aid in understanding relationships between clusters at different levels of abstraction.

2.4 Gaussian mixture model (GMM)

A Gaussian Mixture Model (GMM) is a probabilistic model widely used for clustering and density estimation. GMM assumes that the data is generated from a mixture of several Gaussian distributions, each characterized by its own mean and covariance matrix. The model represents the underlying structure of the data as a combination of these Gaussian components, each contributing to different aspects or modes of the data distribution.

The GMM clustering process involves estimating the parameters of the Gaussian distributions that best fit the observed data. This typically involves the Expectation-Maximization (EM) algorithm, which iteratively refines the parameters until convergence. The EM algorithm has two main steps: the E-step, where the probabilities of each data point belonging to each cluster are computed, and the M-step, where the parameters of the Gaussian distributions are updated based on these probabilities.

One of the strengths of GMM is its flexibility in modeling complex data distributions. Unlike K-means, GMM does not assume that clusters are spherical or have equal variance. Instead, it can model clusters with different shapes, sizes, and orientations by adjusting the parameters of the Gaussian components. This makes GMM particularly useful for datasets where the underlying cluster structure is not well represented by simple geometric shapes.

GMM also provides probabilistic cluster assignments for each data point, allowing for a more nuanced interpretation of cluster membership. Instead of assigning a hard label to each point, GMM assigns probabilities, indicating the likelihood of a point belonging to each cluster. This probabilistic nature makes GMM robust to noise and outliers, as it can capture uncertainty in the data.

However, GMM does have some limitations, such as the sensitivity to the choice of the number of components (clusters) and the possibility of converging to local optima. Model selection techniques, such as the Bayesian Information Criterion (BIC) or cross-validation, can be used to determine the optimal number of components.

In summary, Gaussian Mixture Model clustering is a probabilistic approach that provides a flexible and powerful framework for capturing complex structures in data, making it a valuable tool in various applications, including image segmentation, speech recognition, and anomaly detection.

3 EXPERIMENTAL RESULTS

3.1 Datasets

3.1.1 Synthetic Datasets

We generated six distinct two-dimensional synthetic datasets, each characterized by unique shapes and distributions. These datasets were employed to assess the performance of clustering algorithms. The scatter plots corresponding to each dataset are illustrated in Figure 1.

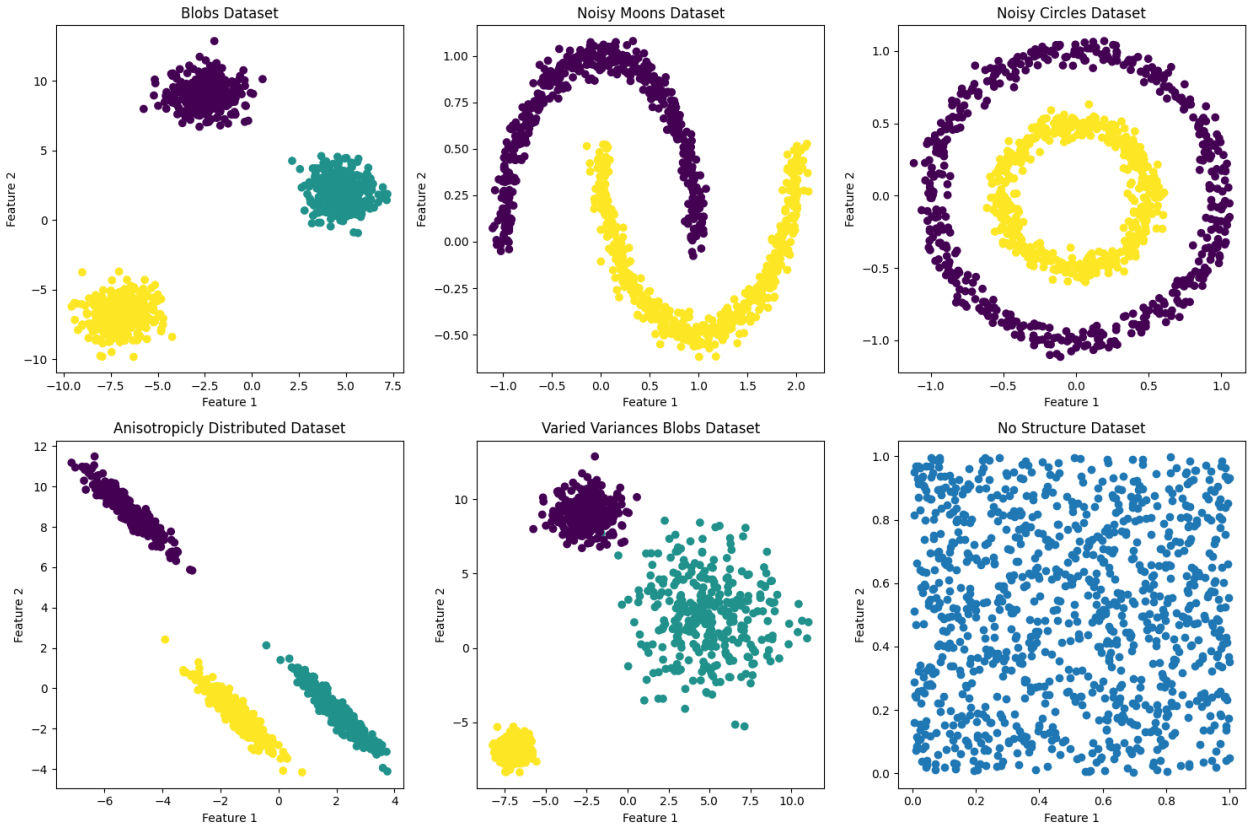

Figure 1: Synthetic Datasets

3.1.2 Real-world Dataset (Adult)

The Adult dataset, also recognized as the "Census Income" dataset, is a widely employed resource in the machine learning community, originating from the 1994 U.S. census data. It comprises 48,842 records, each representing an individual, with 14 features encompassing diverse demographic attributes. To address the challenge of visualizing a dataset with more than 3 features, we applied t-SNE to reduce the dimensions to 2. As illustrated in Figure 2, the scatter plot visually

represents the dimension-reduced version, providing a clearer insight into the relationships within the dataset. Notably, as part of our preprocessing efforts aimed at optimizing training efficiency for machine learning models, we have curated a modified version of the dataset. This processed iteration consists of 10,000 records, a deliberate reduction from the original size, and includes 12 features. This adjustment, undertaken to mitigate training time without compromising essential information, enhances the practicality and expedites the experimentation and development of predictive models on this dataset.

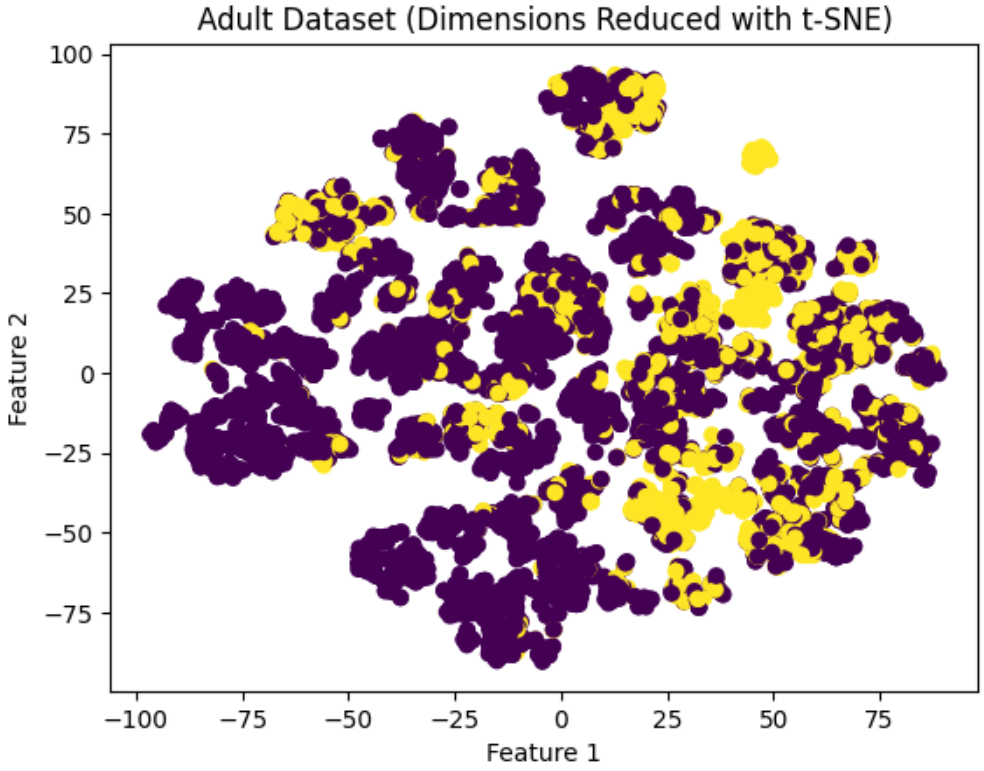

Figure 2: Adult Dataset (Dimensions Reduced with t-SNE)

3.2 Evaluation Metrics

3.2.1 Silhouette Score

The Silhouette Score is a metric used to quantify the goodness of a clustering technique on a dataset. It provides a measure of how well-defined the clusters are in relation to each other. This score takes into account both the cohesion within clusters and the separation between clusters.

For each data point in the dataset, the Silhouette Score is calculated based on the average distance from the other data points within the same cluster (a measure of cohesion) and the average distance from data points in the nearest neighboring cluster (a measure of separation). The

score ranges from -1 to 1, where a high value indicates that the object is well matched to its own cluster and poorly matched to neighboring clusters.

In practice, a Silhouette Score close to 1 suggests that the data point is appropriately clustered, while a score near -1 indicates that the point may be more suitable in a neighboring cluster. A score around 0 suggests overlapping clusters.

When applied to an entire dataset or clustering algorithm, the average Silhouette Score provides an overall assessment of the quality of the clustering solution. Higher average scores are generally indicative of better-defined clusters, making the Silhouette Score a valuable tool for evaluating and comparing different clustering algorithms and their parameter settings.

3.2.2 Adjusted Rand Index (ARI)

The Adjusted Rand Index (ARI) is a metric used to assess the similarity between the true class labels of a dataset and the labels assigned by a clustering algorithm, while considering the possibility of random chance. It is an extension of the Rand Index and adjusts for the expected similarity under random labeling.

The Rand Index measures the proportion of pairs of data points that are correctly classified by a clustering algorithm, whether into the same cluster or different clusters, compared to the true class labels. However, it does not account for the expected similarity due to random chance, which can be significant, especially when dealing with datasets where the number of clusters is not predefined.

The Adjusted Rand Index corrects for this chance agreement by normalizing the Rand Index based on the expected similarity between random clusterings. The index ranges from -1 to 1, where 1 indicates perfect similarity between the true and predicted clusterings, 0 suggests random labeling, and negative values indicate worse than random chance.

ARI is particularly useful when evaluating clustering algorithms in situations where the ground truth (true class labels) is available. It provides a more robust measure of clustering performance by accounting for the possibility of random agreement, making it a valuable metric for assessing the quality of clustering solutions.

3.2.3 Inertia

In the context of centroid-based clustering algorithms, inertia (also known as within-cluster sum of squares or WCSS) is a measure that quantifies how internally coherent clusters are. It is used to evaluate the compactness of clusters, indicating how close data points within the same cluster are to each other.

The inertia of a cluster is calculated as the sum of the squared distances between each data

point in the cluster and the centroid of that cluster. The centroid is typically the mean or median of the data points within the cluster. The idea is that in a well-defined cluster, the data points should be close to the centroid, resulting in a low inertia.

Inertia is a key parameter used in the k-means clustering algorithm. The algorithm works by iteratively assigning data points to clusters and updating cluster centroids to minimize the overall inertia. The final configuration of clusters is reached when further adjustments to the centroids do not significantly reduce the inertia.

When comparing different clustering solutions, a lower inertia generally indicates better-defined and more compact clusters. However, inertia should not be used in isolation for assessing the quality of clusters, as it tends to decrease as the number of clusters increases. The elbow method is often employed to determine an optimal number of clusters by looking for the point at which the rate of decrease in inertia slows down (the "elbow" point).

3.3 Results

We systematically applied each algorithm to all datasets, preceded by the crucial task of determining optimal hyperparameter values for each algorithm-dataset combination. While for k-means, the number of clusters was predefined for all datasets except the No-structure dataset, where labels were absent. To address this, we employed the elbow method, illustrated in Figure 3, to discern the optimal number of clusters. For other algorithms, an exhaustive grid search was conducted across various candidate values, resulting in the identification of the optimal hyperparameter values detailed in Table 1.

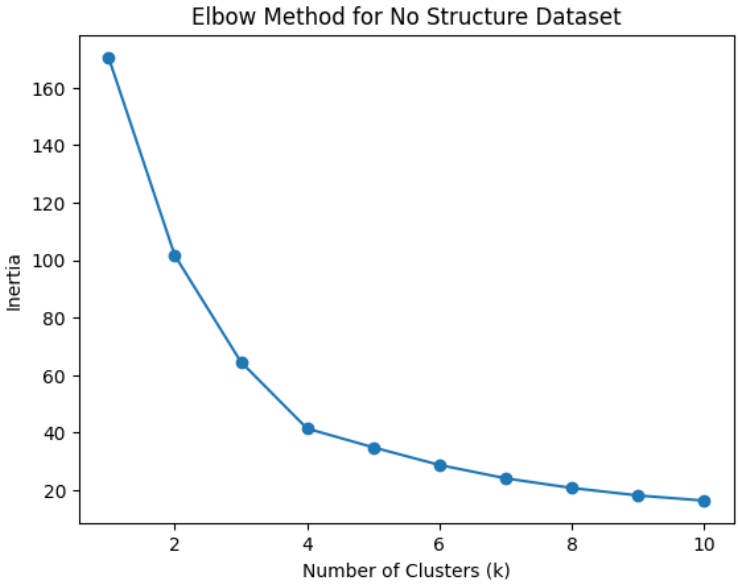

Figure 3: Elbow method applied.

Table 1: Hyperparameters values combination for all algorithms.

Dataset \Algorithm	DBSCAN	Hierarchical (Agglomerative)	Gaussian mixture model (GMM)
Blobs	eps=1.0 min_samples=10	linkage=ward	covariance_type=full init_params=kmeans max_iter=50 n_components=3
Noisy moons	eps=0.1 min_samples=3	linkage=single	covariance_type=full init_params=random max_iter=50 n_components=4
Noisy Circles	eps=0.1 min_samples=3	linkage=single	covariance_type=diag init_params=random max_iter=50 n_components=3
Anisotropicly Distributed	eps=0.5 min_samples=3	linkage=ward	covariance_type=full init_params=kmeans max_iter=50 n_components=2
Varied Variances Blobs	eps=1.5 min_samples=10	linkage=ward	covariance_type=tied init_params=kmeans max_iter=50 n_components=3
No Structure	eps=0.5 min_samples=5	linkage=complete	covariance_type=spherical init_params=kmeans max_iter=50 n_components=4
Real-world (Adult)	eps=1.5 min_samples=10	linkage=complete	covariance_type=full init_params=kmeans max_iter=50 n_components=2

The Agglomerative clustering algorithm did not undergo experimentation with different values for the number of clusters, as the optimal cluster count was predetermined. However, a visual aid, known as a dendrogram, was employed to precisely illustrate the optimal number of clusters. The dendrogram for the Noisy Moons dataset is presented in Figure 4.

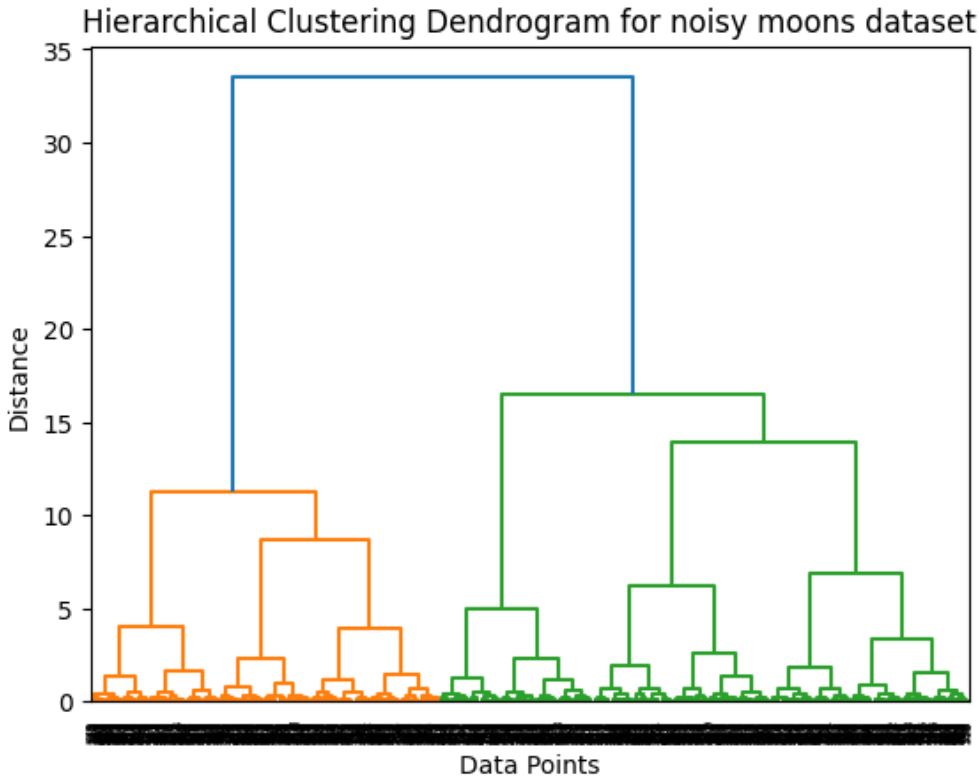

Figure 4: Hierarchical clustering dendrogram for noisy moons dataset.

Following the determination of optimal hyperparameters, each algorithm was executed on all datasets using these identified settings. Performance evaluation was carried out using the Silhouette score and Adjusted Rand Index metrics, as detailed in Table 2. Additionally, we visually represented the output of each algorithm for every dataset through scatterplots, showcased in Figure 5.

Table 2: Evaluation metrics values for all algorithms and datasets combinations.

Algorithm	**K-means**		**DBSCAN**		**Hierarchical (Agglomerative)**		**Gaussian mixture model (GMM)**	
Dataset \Metric	**Silhouette**	**ARI**	**Silhouette**	**ARI**	**Silhouette**	**ARI**	**Silhouette**	**ARI**
Blobs	0.844	1.000	0.823	0.992	0.844	1.000	0.844	1.000
Noisy moons	0.489	0.239	0.055	0.998	0.334	1.000	0.345	0.688
Noisy Circles	0.354	-0.001	0.112	1.000	0.112	1.000	0.105	0.111
Anisotropicly Distributed	0.702	0.991	0.696	0.988	0.701	1.000	0.806	0.572
Varied Variances Blobs	0.777	0.976	0.713	0.979	0.776	0.994	0.777	0.979
No Structure	0.413	-	-	-	0.344	-	0.413	-
Real-world (Adult)	0.162	0.036	0.089	-0.006	0.681	0.026	0.191	0.185

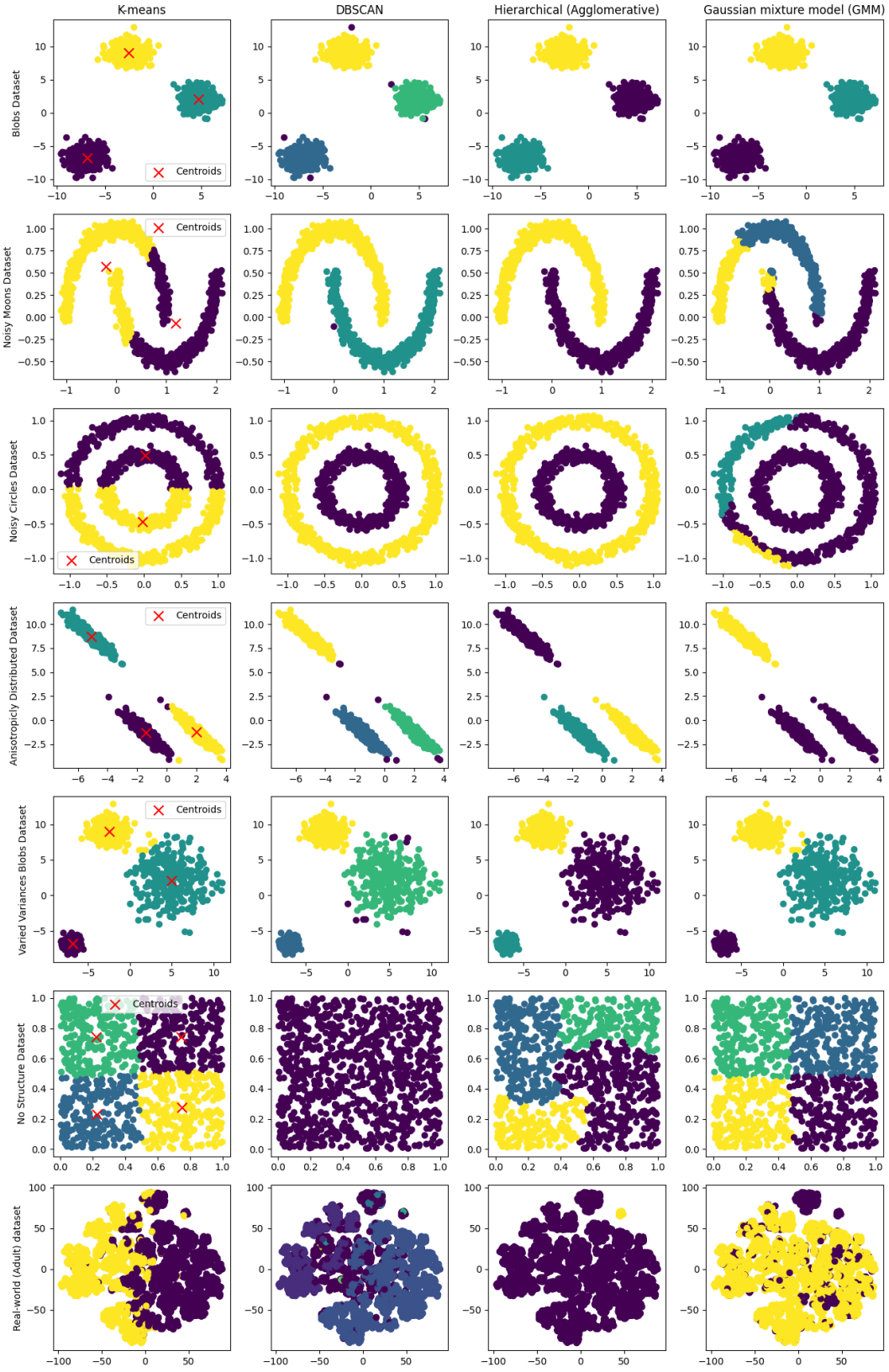

Figure 5: Overall comparison between the clustering algorithms

4 CONCLUSION

In this project, we conducted an in-depth examination of four prominent clustering algorithms: K-means, DBSCAN, Agglomerative clustering, and Gaussian Mixture Model (GMM). Through a rigorous analysis using synthetic datasets and a real-world dataset ('Adult'), we aimed to discern the strengths and weaknesses of each algorithm in capturing diverse patterns within data.

The experimental results revealed nuanced insights into the algorithms' performance across various scenarios. K-means, with its simplicity and efficiency, excelled in scenarios where clusters had spherical shapes and similar sizes. DBSCAN, leveraging density-based criteria, proved robust in detecting clusters with irregular shapes and handling noise effectively. Agglomerative clustering, with its hierarchical nature, provided a multi-scale view of the data's structure, emphasizing its utility in exploring relationships at different levels of abstraction. GMM, with its probabilistic modeling, demonstrated flexibility in capturing complex structures but required careful consideration of the number of components.

The evaluation metrics, including Silhouette Score, Adjusted Rand Index (ARI), and Inertia, provided quantitative measures of algorithm performance. These metrics allowed us to compare the algorithms based on criteria such as cluster cohesion, separation, and overall effectiveness.

As we conclude this study, it is essential to recognize that the choice of clustering algorithm depends on the characteristics of the dataset and the specific goals of the analysis. No single algorithm emerged as universally superior; instead, each showcased its strengths under certain conditions. Moreover, the findings underscore the importance of careful parameter tuning and model selection to optimize the performance of clustering algorithms.

Looking ahead, future research could delve deeper into algorithm hybridization, exploring ways to leverage the strengths of multiple algorithms to enhance overall performance. Additionally, further investigations into adapting these clustering techniques to dynamic datasets and exploring their scalability to larger datasets would be valuable contributions to the field.

In summary, this project contributes valuable insights into the comparative analysis of clustering algorithms, providing a foundation for informed decision-making in diverse data analysis scenarios. The dynamic nature of clustering research ensures that ongoing exploration and refinement of these techniques will continue to shape the landscape of unsupervised learning and data exploration.