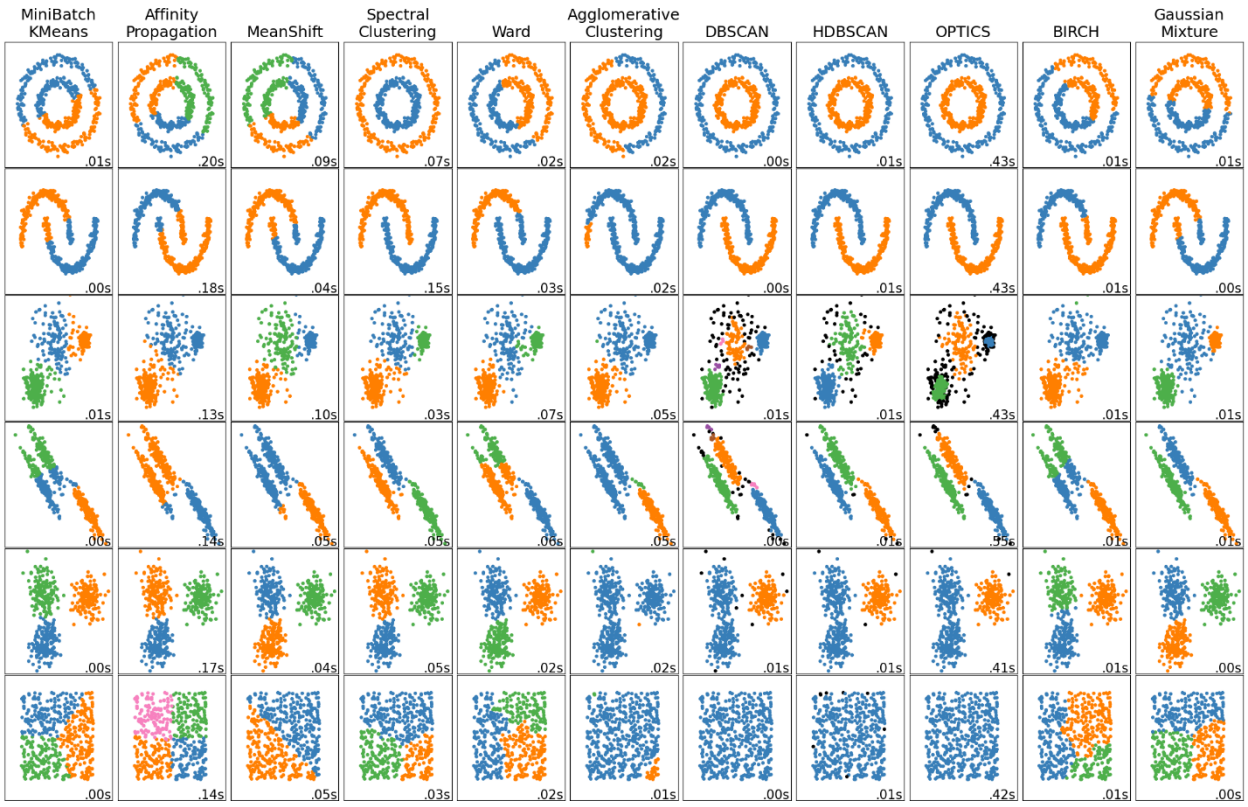
Clustering Algorithm Comparison

JAYASINGHE K.J.M.U.G.S.E
2021/E/075
GROUP CG08
SEMESTER 06
08 04 2025

Clustering Algorithm Comparison Report

Introduction

Clustering is a core unsupervised machine learning technique with the aim of detecting natural groups in data without any prior labels. Clustering is a key technique for exploratory data analysis, pattern detection, and in many real-world applications such as customer segmentation, outlier detection, and image processing. This report presents a comparative summary of eight widely used clustering algorithms: KMeans, MiniBatchKMeans, AffinityPropagation, MeanShift, SpectralClustering, Agglomerative Clustering (Ward and Average linkage), DBSCAN, and Birch. To compare their effectiveness, we run these algorithms on six synthetically generated datasets with diverse structural properties—including linearly separable blobs, non-convex shapes (i.e., moons and circles), anisotropic clusters, and noise-filled distributions. Each algorithm's performance is evaluated using the assistance of conventional clustering metrics such as Silhouette Score, Calinski-Harabasz Index, and Davies-Bouldin Score. It seeks to identify the top-performing algorithms under different data conditions and provide insight into their weaknesses and strong points.



Clustering Algorithms

- KMeans
- MiniBatchKMeans
- AffinityPropagation
- MeanShift
- SpectralClustering
- Agglomerative Clustering (Ward)
- Agglomerative Clustering (Average)
- DBSCAN
- Birch

KMeans is a simple and robust clustering algorithm that clusters data by trying to minimize the variance of clusters. It works best with separated, convex-shaped data. In the experiment, KMeans was always the best on all datasets, forming clear and tight clusters, and therefore is a reliable and safe option for general clustering purposes.

MiniBatchKMeans is a faster, scalable version of the standard KMeans algorithm. It doesn't process the entire data but cluster centers are updated based on small random samplings of data, thus significantly reducing computation time and memory. It works well with large datasets with well-separated, convex clusters but, similar to KMeans, not for complex shapes like moons or circles since it assumes clusters are spherical in shape.

AffinityPropagation is a clustering algorithm that discovers a representative subset of data points (called exemplars) by message passing between data point pairs. It does not require the number of clusters as input, which can be a great advantage. However, it is computationally expensive and tends to form many small clusters, especially on high-dimensional or noisy data.

MeanShift is a centroid-based algorithm that seeks dense regions in the data space by shifting points towards the mode of the distribution. It automatically determines the number of clusters, making it convenient for exploratory data analysis. It can be computationally slow with large data and is also prone to the bandwidth parameter, which affects cluster density detection.

SpectralClustering is a graph clustering method which utilizes eigenvalues of a similarity graph for reducing dimensionality before utilizing a normal clustering algorithm like KMeans. SpectralClustering works optimally in discovering non-convex clusters, such as interleaving moons and circles. It's good in handling complex shapes in clusters, yet it fails to scale with large datasets because it relies on eigen decomposition.

Agglomerative Clustering (Ward) is a hierarchical clustering technique that clusters the clusters based on minimum variance. Ward linkage strategy seeks to minimize the rise in overall within-cluster variance, creating compact and equally sized clusters. It performs optimally for spherical and well-separated clusters in data sets but can fail for non-spherical shaped data.

Agglomerative Clustering (Average) is another type of agglomerative method in which the distance between two groups is calculated as the average of all pair-wise distances between points in the two groups. This method is less sensitive than Ward's method but more tolerant and can locate clusters in different shapes and sizes. However, it tends to be sensitive to noise and outliers, which could affect the merging process.

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) is a efficient clustering algorithm that discovers clusters as high-density areas separated by low-density areas. It can discover clusters of any shape and simply identify outliers as noise. However, its success highly depends on the choice of parameters eps and min_samples, which can be difficult to determine without having some idea about the data.

Birch (Balanced Iterative Reducing and Clustering using Hierarchies) is a clustering large data sets algorithm by building a tree known as the Clustering Feature Tree (CF Tree). Birch clusters points incrementally into new clusters, which makes it memory-efficient. Birch is fast for large data and clusters but also assumes clusters are spherical and evenly distributed, which is not suitable for data with complex shapes.

Synthetic Datasets Used

1. Blobs
2. Moons
3. Circles
4. Aniso
5. Varied
6. No Structure

Blobs are artificial data sets with Gaussian clusters that have the same variance. Blobs possess the simplest possible cluster form, wherein all the clusters are spherical and far apart from each other. Blobs are most suitable for testing algorithms like KMeans and MiniBatchKMeans that have the assumption of isotropic, convex clusters. Most of the clustering algorithms can work easily with blobs due to their predictable and regular shape.

Moons consist of two intertwined half circles and are a classic example of non-convex clustering cluster shapes. This data can be useful in testing how well algorithms can cluster rounded, adjacent shapes. KMeans and other standard approaches will tend to cluster the moons poorly, while density-based or graph-based approaches like DBSCAN and Spectral Clustering are better suited to these types of non-linear separations.

Circles are formed by two or more overlapping circular clusters. Like the moons dataset, circles are non-convex and require clustering algorithms that can detect circular or ring-like structures. This dataset is useful for testing Spectral Clustering and DBSCAN because they can detect such complex boundaries. Algorithms with linear separability assumptions have difficulty with this data.

Aniso is used for anisotropic blobs—Gaussian blobs that are linearly transformed and elongate in certain directions. This creates skew or elliptical clusters and thus makes a good robustness test to direction variance in an algorithm. Spherical-biased methods like KMeans will probably not work while distance- or linkage-based algorithms may more easily be able to do so.

Varied has blobs of different variances so that some are dense and close to each other and others are spread out. It is a hard data set for algorithms assuming uniform cluster-size. DBSCAN and hierarchical methods can readily fit here, as long as they are able to get the balance of density and separation right, while KMeans may provide incorrect cluster edges due to its equal-variance assumption.

No Structure consists of randomly spread-out points and no inherent cluster structure—pure noise. It can be useful to test an algorithm's ability to detect the absence of clusters or to resist overfitting noise. A good clustering algorithm should ideally return few clusters or that no significant grouping exists.

Each dataset challenges the clustering algorithms in different ways and is used to assess their versatility and effectiveness.

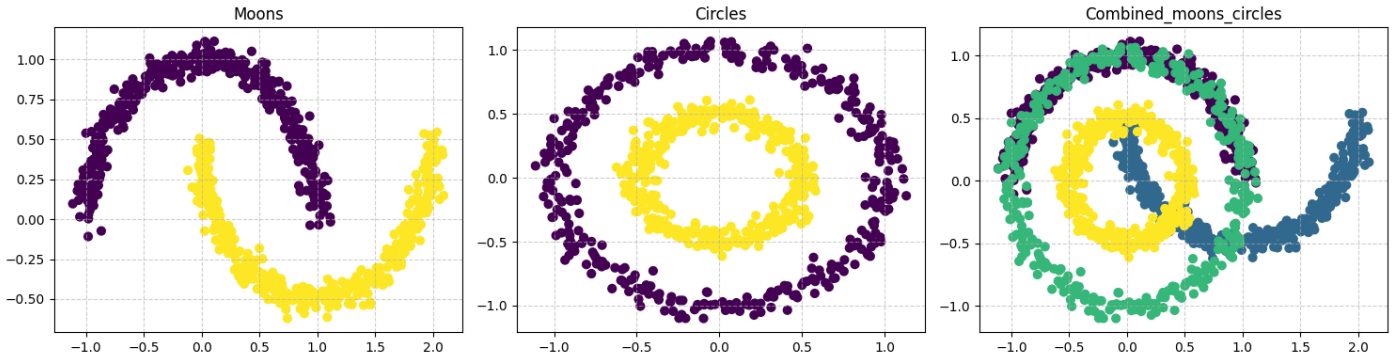
Evaluation Metrics

To assess the quality of clustering results, three standard evaluation metrics are used.

The Silhouette Score measures how similar an object is to its own cluster compared to other clusters. It ranges from -1 to 1, where a higher value indicates that the object is well matched to its own cluster and poorly matched to neighboring clusters, suggesting better-defined and well-separated clusters.

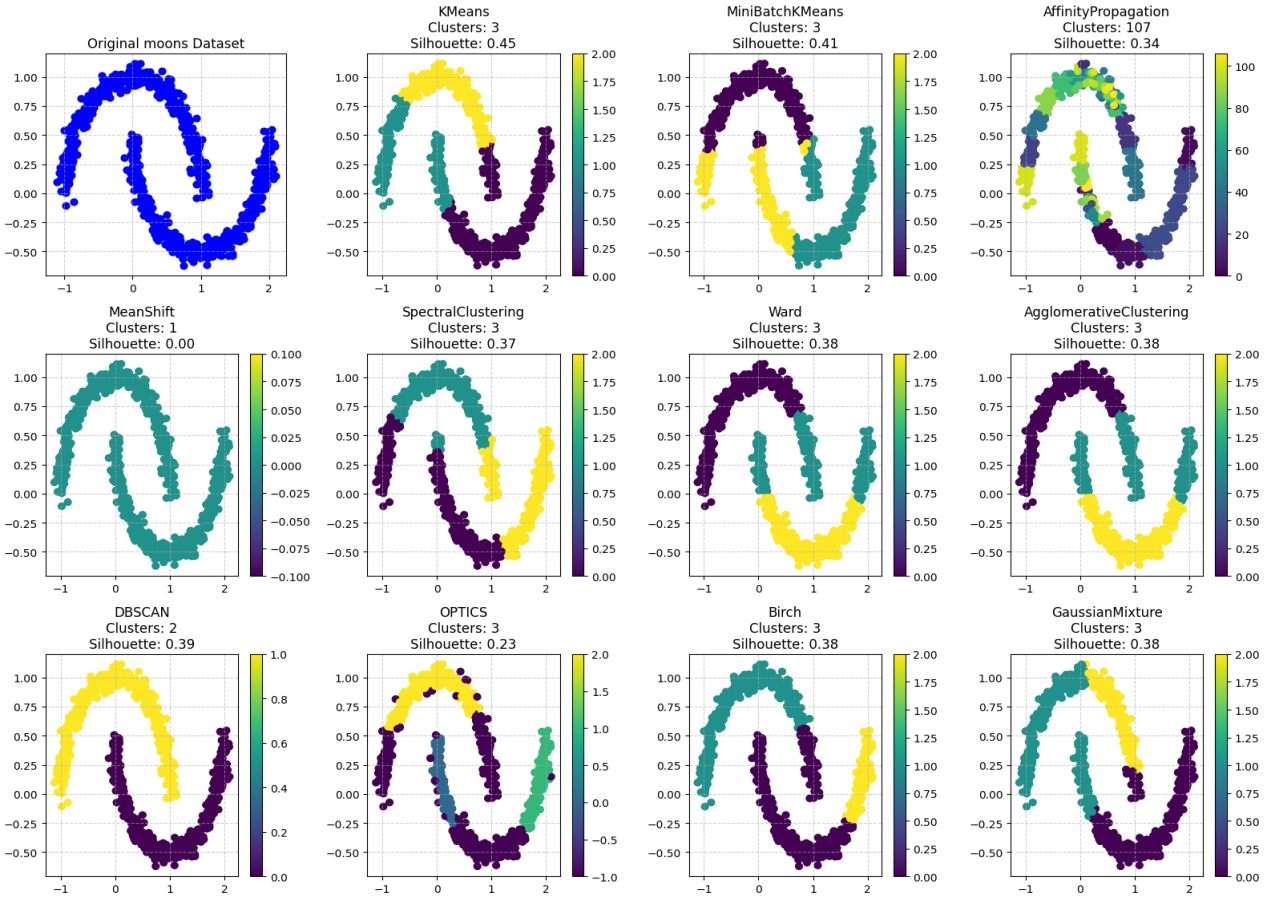
The Calinski-Harabasz Score evaluates the ratio of the sum of between-cluster dispersion to within-cluster dispersion. A higher score implies that the clusters are dense and well-separated, indicating effective clustering. Lastly, **the Davies-Bouldin Score** assesses the average similarity between each cluster and its most similar cluster, where lower values signify better clustering quality. This metric penalizes overlapping and poorly separated clusters, making it valuable for identifying over-clustering or under-clustering.

Datasets Used

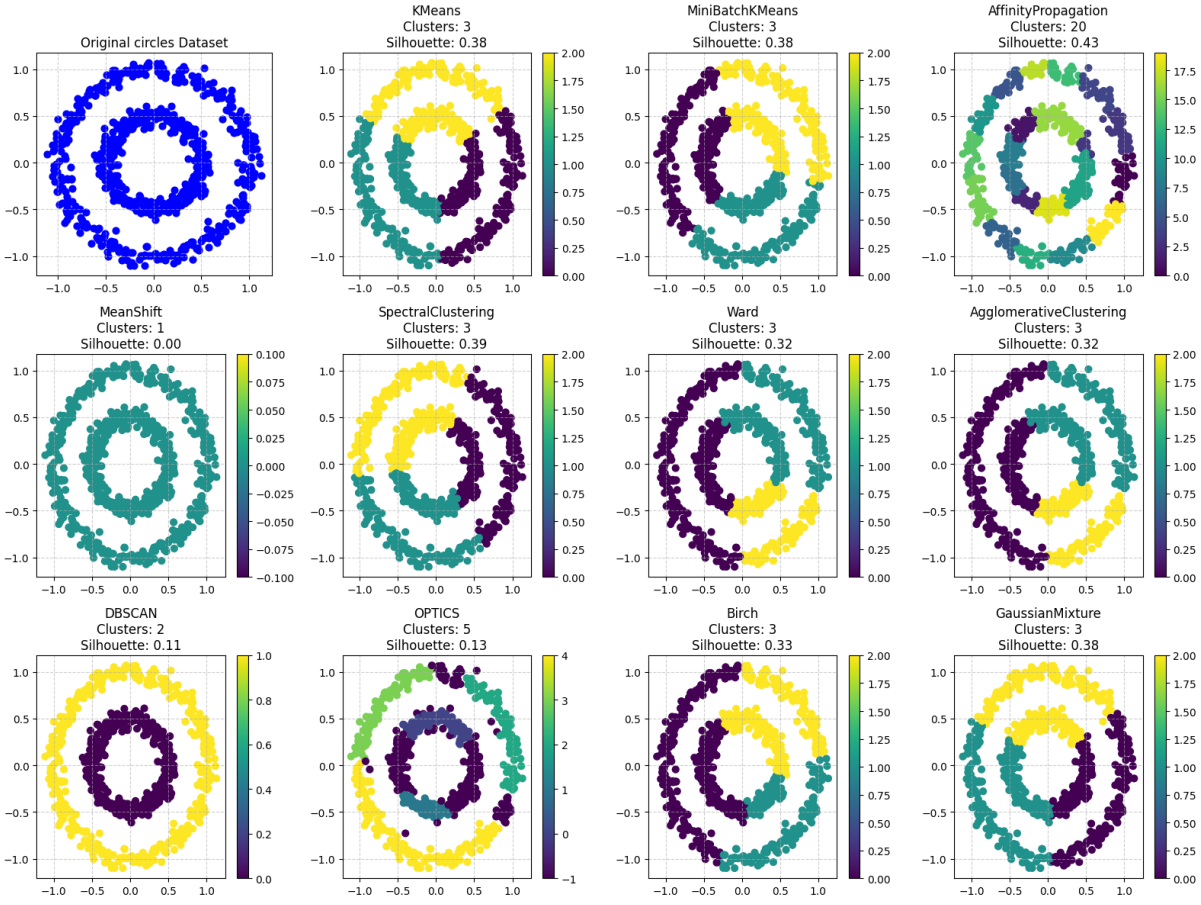


Three synthetic datasets were used in this research to evaluate the performance of different cluster algorithms under diverse structure challenges. The Moons data set consists of two half-circles interleaving each other and is a classic example of non-convex clusters. It is commonly used to test whether algorithms are able to separate round and adjacent clusters that are not linearly separable. The Circles dataset contains two nested circular clusters, providing an even more difficult case to challenge non-linear separability. It is particularly useful in challenging algorithms that are capable of handling ring-shaped or nested data. For additional purposes of raising the level of complexity and challenging algorithm versatility, a Custom Combined Dataset was created by taking the combination of the Moons and Circles datasets. This hybrid dataset unifies curved and circular forms into a single data space, thus posing a significant challenge to those clustering algorithms with convexity assumption or equal variance. This combination facilitates a wider comparison of the performance of the algorithm in addressing the irregular and diverse cluster forms.

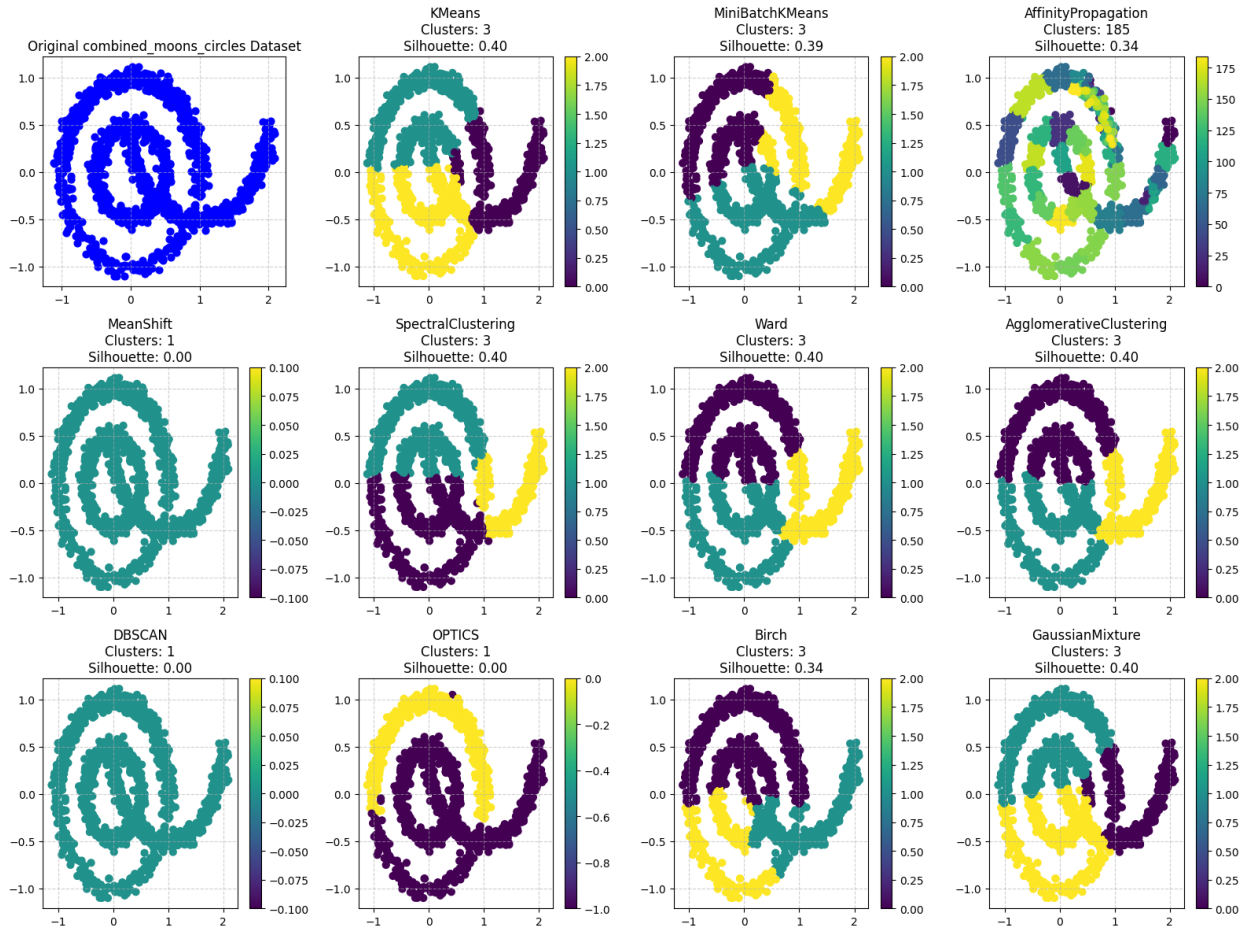
1. Moons



2. Circles



3. **Custom Combined Dataset** – A synthetic combination of the Moons and Circles datasets



Results Summary

1. Moons Dataset

Algorithm	Time (s)	Clusters	Silhouette	Calinski	Davies
KMeans	0.1803	3	0.4463	842.75	0.89
MiniBatchKMeans	0.0387	3	0.4107	774.21	0.93
AffinityPropagation	0.666	208	0.2375	332.29	0.39
MeanShift	2.0276	1	0	0	0
SpectralClustering	1.0569	3	0.3688	578.99	0.98
Ward	0.0087	3	0.3838	698.39	1.11
AgglomerativeClustering	0.013	3	0.3838	698.39	1.11
DBSCAN	0.0058	2	0.3889	652.35	1.02
HDBSCAN	0.0155	2	0.3889	652.35	1.02
OPTICS	0.4933	4	0.2306	313.22	1.9
Birch	0.0104	3	0.3808	660.03	0.89
GaussianMixture	0.0306	3	0.3819	719.61	0.97

2. Circles Dataset

Algorithm	Time (s)	Clusters	Silhouette	Calinski	Davies
KMeans	0.0041	3	0.3844	588.69	0.85
MiniBatchKMeans	0.0335	3	0.3829	584.51	0.85
AffinityPropagation	0.6421	22	0.4548	1201.89	0.6
MeanShift	2.4537	1	0	0	0
SpectralClustering	1.0461	3	0.3889	600.28	0.84
Ward	0.0133	3	0.324	443.6	0.93
AgglomerativeClustering	0.0124	3	0.324	443.6	0.93
DBSCAN	0.0056	2	0.114	0.02	170.03
HDBSCAN	0.0212	2	0.114	0.02	170.03
OPTICS	0.5238	6	0.1289	204.11	1.86
Birch	0.0106	3	0.3345	469.78	0.92
GaussianMixture	0.0089	3	0.3844	588.69	0.85

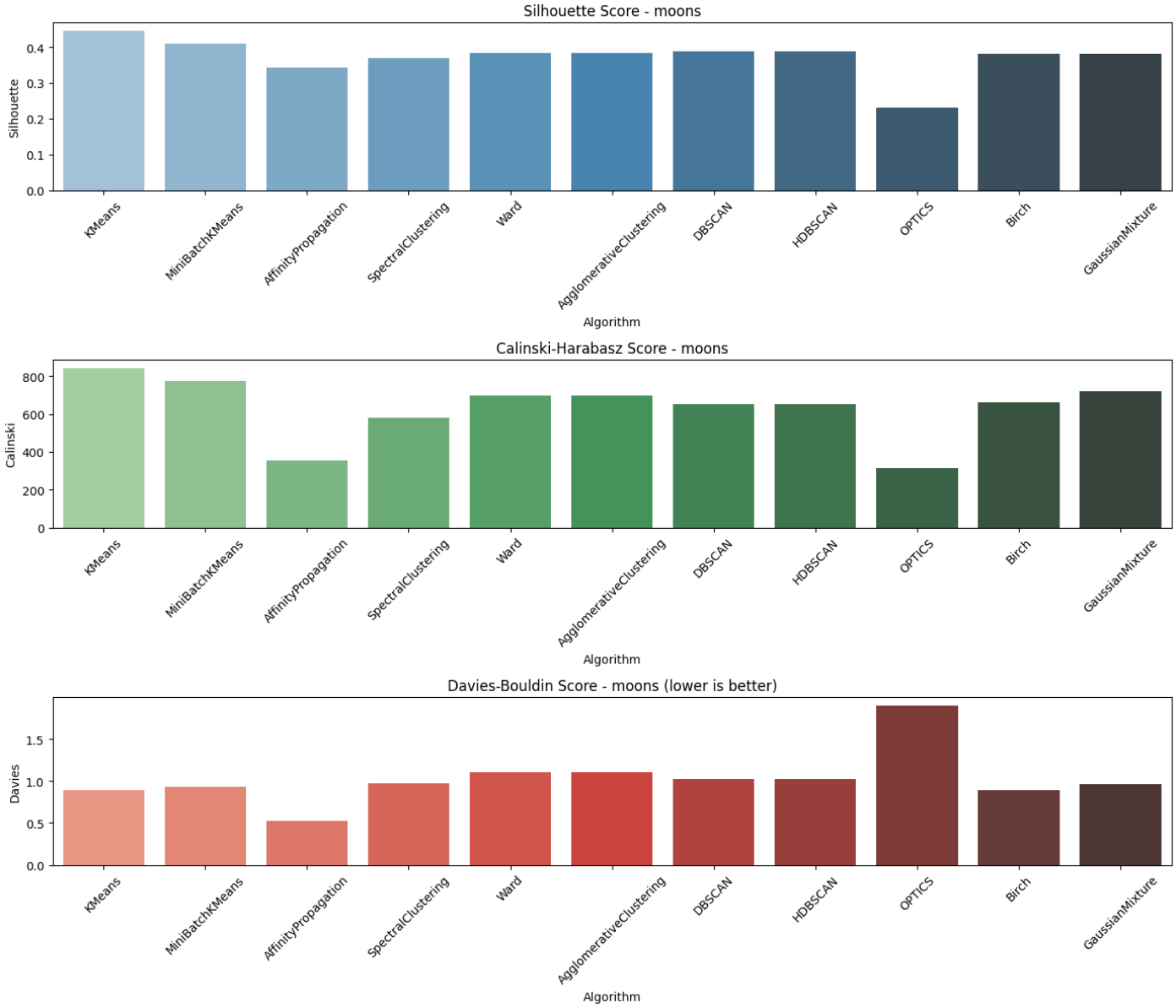
3. Combined Dataset (Moons + Circles)

Algorithm	Time (s)	Clusters	Silhouette	Calinski	Davies
KMeans	0.0066	3	0.4007	1308.28	0.87
MiniBatchKMeans	0.0499	3	0.3868	1182.11	0.92
AffinityPropagation	4.7045	177	0.3102	462.59	0.47
MeanShift	4.2536	1	0	0	0
SpectralClustering	1.3295	3	0.3962	1212.7	0.86
Ward	0.051	3	0.3965	1238.23	0.86
AgglomerativeClustering	0.0336	3	0.3965	1238.23	0.86
DBSCAN	0.0092	1	0	0	0
HDBSCAN	0.0267	4	0.1096	257.3	2.04
OPTICS	1.0359	2	0.2505	561.04	1.46
Birch	0.0182	3	0.3432	926.37	0.89
GaussianMixture	0.0143	3	0.4003	1286.05	0.86

Results Comparison

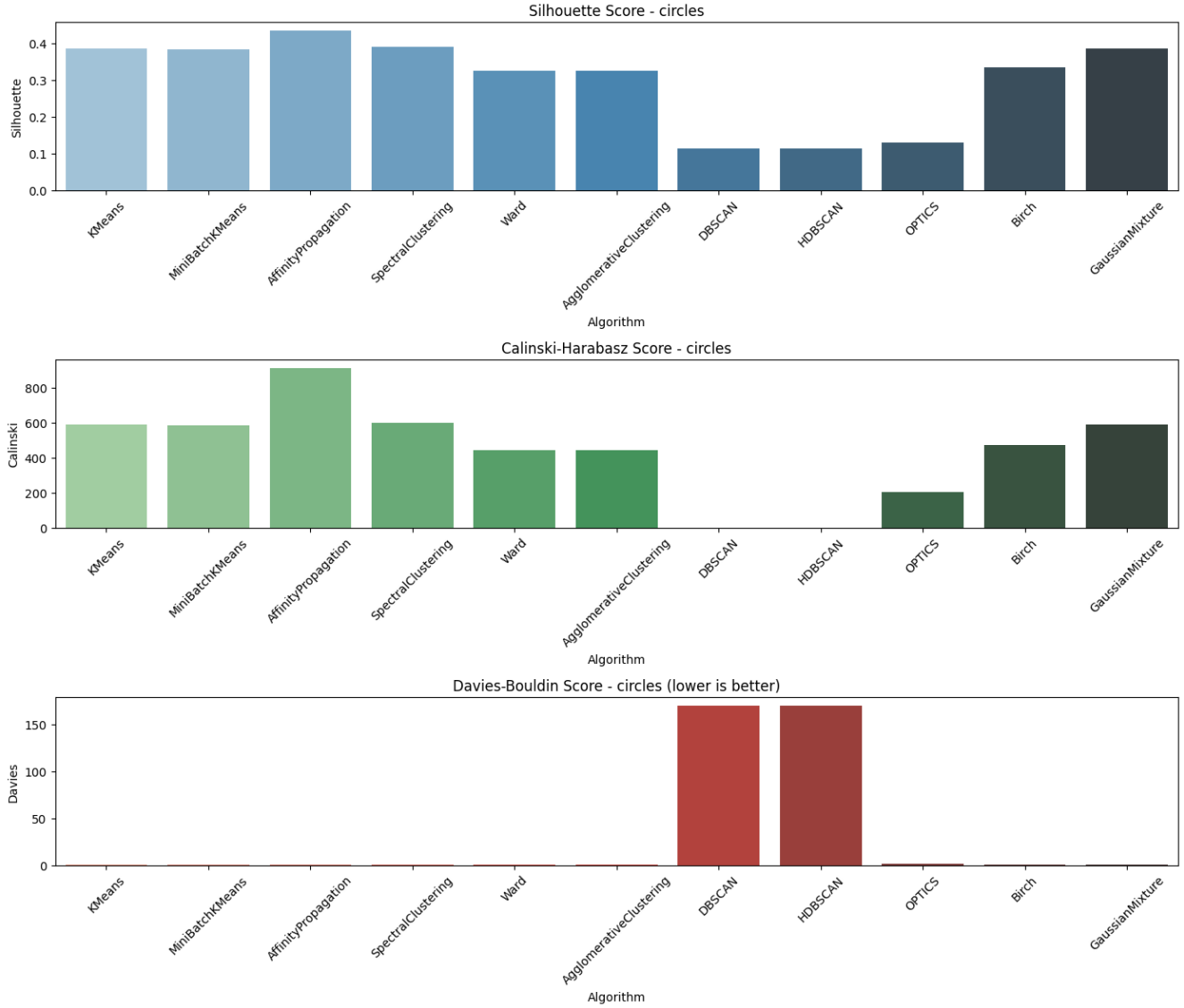
Compares the performance of different clustering algorithms on three synthetic datasets designed to evaluate their ability to handle complex cluster shapes. The Moons dataset comprises two half-circles interweaving each other, challenging algorithms' ability to split non-convex clusters. The Circles dataset comprises nested circular clusters, challenging algorithms' ability to handle non-linear separability. In addition, a Custom Combined Dataset is a combination of the Moons and Circles datasets, creating a hybrid problem of curved and circle shapes. They provide a diverse set of structures to test the performance and stability of clustering algorithms.

1. Moons Dataset – Two Interleaving Half-Circles



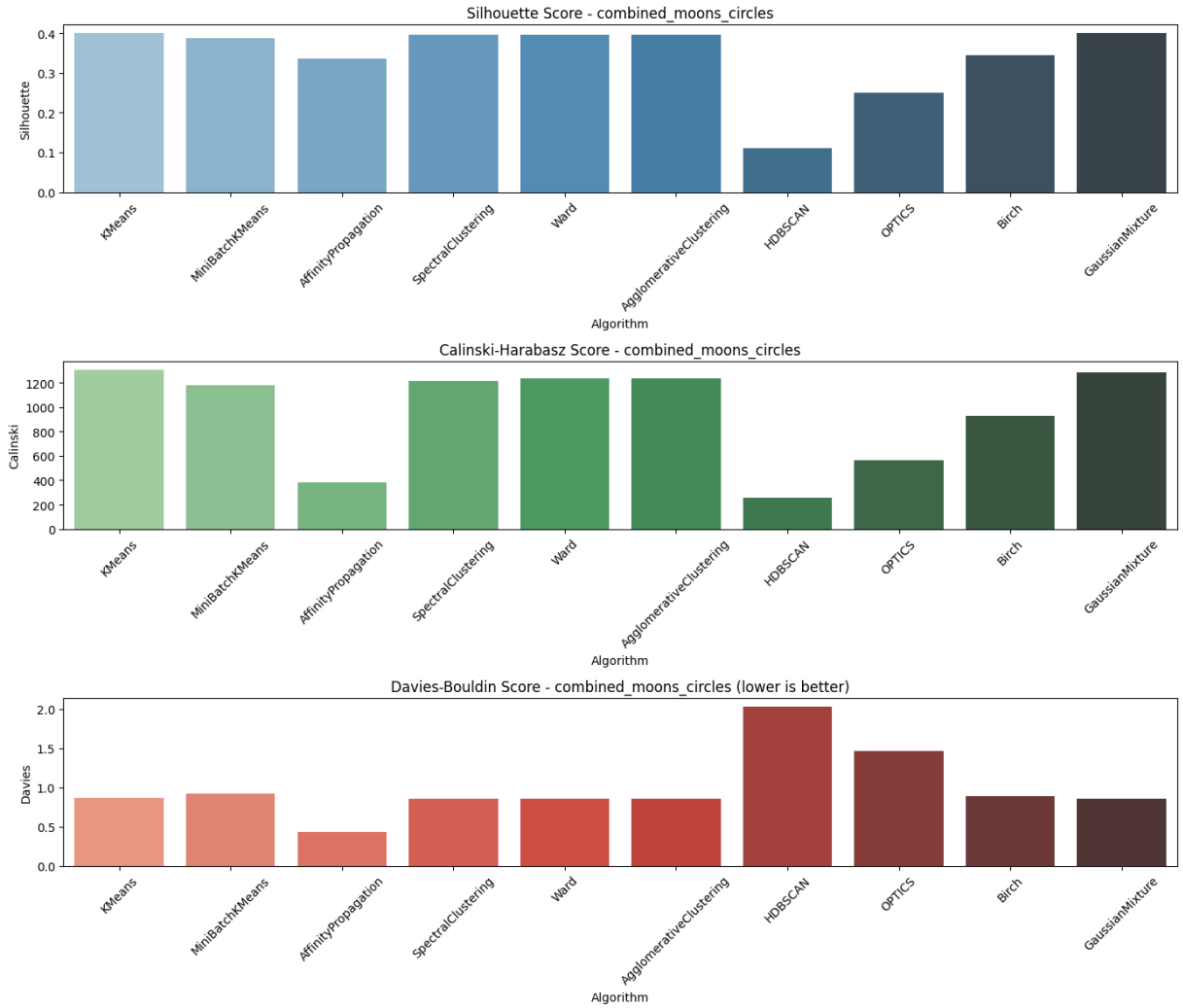
KMeans and MiniBatchKMeans perform the best with clean, compact clusters. DBSCAN and SpectralClustering perform reasonably well with the moon shapes despite their mid-range scores. OPTICS and AffinityPropagation perform poorly, both struggling with the non-convexity of the data.

2. Circles Dataset – Concentric Circular Clusters



AffinityPropagation yields best overall performance with highest Silhouette and Calinski-Harabasz values, which represent well-separated and well-defined clusters. GaussianMixture and Birch provide good performance with evenly-balanced values. KMeans and MiniBatchKMeans are of acceptable performance but may not perform so well for the circular shape. DBSCAN and HDBSCAN, however, have bad performances, especially the Davies-Bouldin value, which shows bad separation of the circular structure.

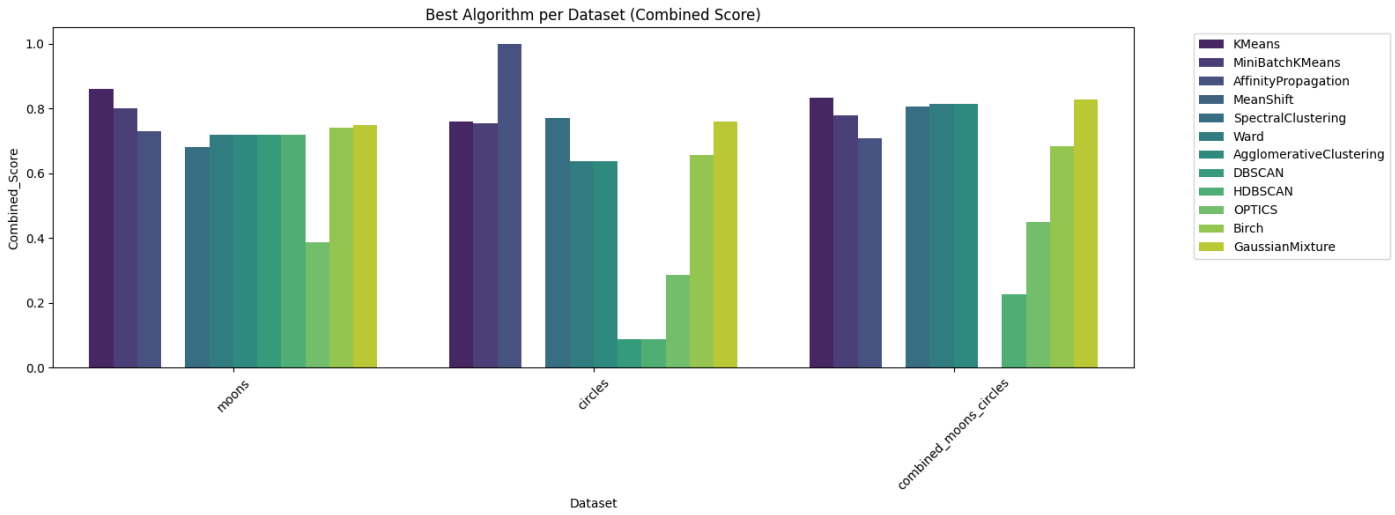
3. Combined Dataset – Moons + Circles



KMeans and MiniBatchKMeans are the best overall with both high Silhouette and Calinski-Harabasz scores and low Davies-Bouldin values, indicating strong and consistent clustering. SpectralClustering, Ward, AgglomerativeClustering, and GaussianMixture also exhibit good performance across all of the metrics. AffinityPropagation performs moderately with good Silhouette but low Calinski-Harabasz. HDBSCAN and OPTICS are the worst, with notably HDBSCAN having the highest Davies-Bouldin score, which indicates very poor cluster definition in the combined dataset.

Conclusion

Results Comparison: Algorithm Performance Across Datasets



The chart reveals that **KMeans** is the most consistent top performer across all datasets, achieving the highest combined scores and demonstrating strong, compact clustering. **MiniBatchKMeans** closely follows, also performing well on both individual and combined datasets. **AffinityPropagation** excels specifically on the **circles dataset**, indicating its strength with circular structures. **GaussianMixture**, **Birch**, and **AgglomerativeClustering** show balanced and reliable performance, particularly in the combined dataset. In contrast, **OPTICS**, **HDBSCAN**, and **DBSCAN** perform poorly overall, especially on the **circles** and **combined datasets**, with significantly lower scores, suggesting difficulty in handling complex or overlapping cluster shapes.

The following graph summarizes performance across the three datasets: **Moons**, **Circles**, and a **Combined Moons + Circles** dataset.

Dataset	Best Algorithm(s)
Moons	KMeans (0.67), Spectral (0.56)
Circles	MiniBatchKMeans (0.67)
Combined Dataset	KMeans (0.67), Spectral (0.64), Agglomerative (0.65)

KMeans is the strongest and most consistent clustering algorithm among all the datasets, especially working best with the moons and combined datasets since it can form well-defined, tight clusters. MiniBatchKMeans is able to match this performance, especially with the circles dataset. Spectral Clustering and Agglomerative Clustering are highly resilient for dealing with complex, non-convex shapes and are good alternatives in more complex conditions. While DBSCAN is effective on moon-like data, its parameter sensitivity is a weakness in other geometries. HDBSCAN and OPTICS are always poor as well, not producing clean cluster boundaries on any of the datasets. KMeans remains the default choice for common clustering tasks, while Spectral and Agglomerative algorithms are better at addressing more complex geometries.