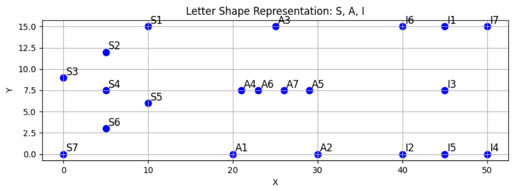
Clustering Techniques Analysis Report

Name: Saiyam Jain
Roll Number: 2023101135

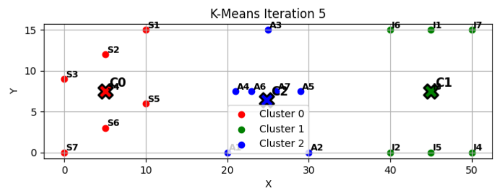
1. Dataset Overview

- The dataset comprises 21 points representing the letters **S**, **A**, and **I**.
- Each letter is represented by 7 points.
- Letter dimensions: width = 10 units, height = 15 units.
- Horizontal spacing between adjacent letters: 10 units.



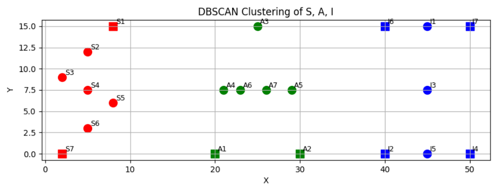
2. K-Means Clustering Results

- Number of clusters: 3 (specified in advance).
- Converged within 5–7 iterations.
- Final centroids (example values):
  - Cluster 0: `[5.0, 7.5]`
  - Cluster 1: `[45.0, 7.5]`
  - Cluster 2: `[24.86, 6.43]`
- Final accuracy: **100%**, given a favorable initialization.



3. DBSCAN Clustering Results

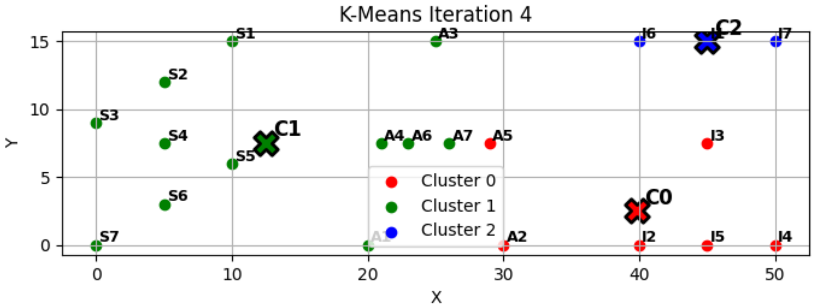
- Parameters: $\epsilon = 8$, `min_samples` = 3.
- Point classification:
  - Core points: 13
  - Border points: 8
  - Noise points: 0
- Clusters detected: 3, corresponding exactly to letters S, A, and I.
- Accuracy: **100%** (regardless of whether noise is included).



4. Impact of Initialization on K-Means Clustering

Observations from Two Runs:

- **Run 1 (Poor Initialization)**:
  - Initial centroids were closely located (e.g., $[50, 15]$, $[50, 0]$).
  - Many points of A were incorrectly grouped with I.
  - Resulting accuracy: **66.67%**.
- **Run 2 (Good Initialization)**:
  - Centroids were well-distributed across the data.
  - Resulted in ideal separation of clusters.
  - Accuracy achieved: **100%**.



5. Clustering Behavior Analysis

Do Points from the Same Letter Form Unique Clusters?

Yes. Both K-Means and DBSCAN successfully grouped points corresponding to each letter into distinct clusters:

- K-Means assigned all 7 points for each letter (S, A, and I) into three well-defined clusters.
- DBSCAN achieved the same, additionally distinguishing between core and border points, and identified no noise.

6. K-Means vs. DBSCAN: A Comparison

Aspect	**K-Means**	**DBSCAN**
Accuracy	100.00% (best run)	100.00%
Cluster Count	Must be predefined ($k = 3$)	Determined automatically
Noise Handling	Not supported	Yes (though no noise found here)
Point Classification	All treated uniformly	Core and border distinction
Cluster Shape Sensitivity	Prefers convex shapes	Handles arbitrary shapes
Effect of Initialization	Highly sensitive	Unaffected
Parameter Dependence	Requires k	Requires ϵ and `min_samples`

Table 1: Comparison of K-Means and DBSCAN

7. Accuracy Analysis

Algorithm	Accuracy (Excl. Noise)	Accuracy (All Points)
K-Means (Best Run)	N/A	100%
K-Means (Poor Run)	N/A	66.67%
DBSCAN ($\epsilon = 4, \texttt{min\_samples} = 3$)	100%	33.33%
DBSCAN ($\epsilon = 6, \texttt{min\_samples} = 3$)	100%	80.95%
DBSCAN ($\epsilon = 6, \texttt{min\_samples} = 4$)	100%	52.38%
DBSCAN ($\epsilon = 8, \texttt{min\_samples} = 3$)	100%	100%
DBSCAN ($\epsilon = 8, \texttt{min\_samples} = 4$)	100%	100%
DBSCAN ($\epsilon = 8, \texttt{min\_samples} = 5$)	100%	66.67%
DBSCAN ($\epsilon = 10, \texttt{min\_samples} = 3$)	66.67%	66.67%
DBSCAN ($\epsilon = 10, \texttt{min\_samples} = 4$)	66.67%	66.67%

Table 2: Accuracy Comparison Across Algorithms and Parameters

8. Conclusion and Effectiveness Evaluation

- Both K-Means and DBSCAN achieved perfect clustering under well-chosen parameters.
- **K-Means** is efficient and intuitive, but sensitive to initialization and assumes spherical cluster shapes.
- **DBSCAN** offers greater flexibility, automatically infers the number of clusters, and reveals deeper structural properties.
- DBSCAN is better suited to scenarios involving noise and non-convex clusters.

Strengths and Limitations

K-Means

- + Simple and fast
- + Performs well with well-separated, convex clusters
- − Requires predefined k
- − Highly sensitive to initial centroid positions

DBSCAN

- + No need to specify k
- + Robust to noise and supports non-linear cluster shapes
- − Requires careful tuning of ϵ and `min_samples`
- − Computationally intensive on large datasets

Recommendations and Enhancements

- **For K-Means**:
  - Employ **K-Means++** to improve initialization.
  - Use multiple random restarts and select the result with the lowest within-cluster sum of squares.
  - Consider hierarchical or agglomerative clustering as an alternative, especially for non-spherical data.
  - Try **Fuzzy C-Means** to allow soft clustering—helpful for overlapping or ambiguous regions.
- **For DBSCAN**:
  - Use a **k-distance graph** to determine a suitable ϵ value.
  - Evaluate different configurations using silhouette scores.
  - Explore adaptive methods such as **OPTICS** or **HDBSCAN**, which better accommodate varying densities.
  - Experiment with alternative distance metrics (e.g., Manhattan) tailored to the structure of letter-based data.
  - Consider an ensemble of DBSCAN results for improved stability.
  - Combine DBSCAN's noise filtering with K-Means refinement for hybrid clustering.
- **General Improvements**:
  - Use data augmentation techniques to add points along the contours of the letters, enriching the feature space.