SYMBIOSIS INSTITUTE OF TECHNOLOGY (SIT)
Constituent of Symbiosis International (Deemed University), Pune
(Established under Section 3 of the UGC Act of 1956 vide notification number F-9-12/2001-U-3 of the Government of India)
Re-Accredited by NAAC with 'A++' Grade

Project Report UML LAB

ON

Application of Clustering Techniques on EuroSAT Dataset

Under the Guidance of

Dr. Anupkumar Bongale

EuroSAT Image Classification with Clustering Algorithms

By

Krisha shah 24070126512,

Janhavi doijad 23070126153

Table of Contents

1. Introduction

With the rapid increase in Earth observation systems and satellite imagery, classifying land use and land cover from remote sensing data has become essential for environmental monitoring, agricultural analysis, and urban planning. However, the high dimensionality of image data poses challenges in training efficient and generalizable models.

The EuroSAT dataset, derived from Sentinel-2 satellite data, provides a diverse collection of labeled land cover images. This project explores the use of various clustering algorithms to find patterns on EuroSAT data.

2. Objective

The objective of this project is to perform unsupervised learning on the EuroSAT dataset using clustering algorithms to categorize satellite images based on land use patterns. Specifically, the project focuses on implementing K-Means and Agglomerative Clustering along with other clustering algorithms like CLARA, DBSCAN and BIRCH to group visually similar images without relying on predefined labels. The optimal number of clusters is determined using evaluation techniques such as the Elbow Method and Silhouette Analysis. To assess the quality and effectiveness of the clustering results, various performance metrics including Silhouette Score, Davies–Bouldin Index, Calinski–Harabasz Index, and Adjusted Rand Index are used. The ultimate goal is to identify natural groupings within the dataset and analyze how well the clustering algorithms capture the underlying structure of the data.

.

3. EuroSAT Dataset Description(https://github.com/phelber/EuroSAT)

The **EuroSAT dataset** is a publicly available satellite image dataset built from **Sentinel-2** satellite data provided by the European Space Agency (ESA). It is specifically curated for **land use and land cover classification** tasks using machine learning and deep learning techniques.

- **Total Images**: ~27,000
- **Image Size**: 64 × 64 pixels
- **Color Channels**: RGB (Red, Green, Blue)

- **Number of Classes**: 10 land use categories
- **Source**: Sentinel-2 satellite (multispectral imaging)
- **Resolution**: 10 meters per pixel

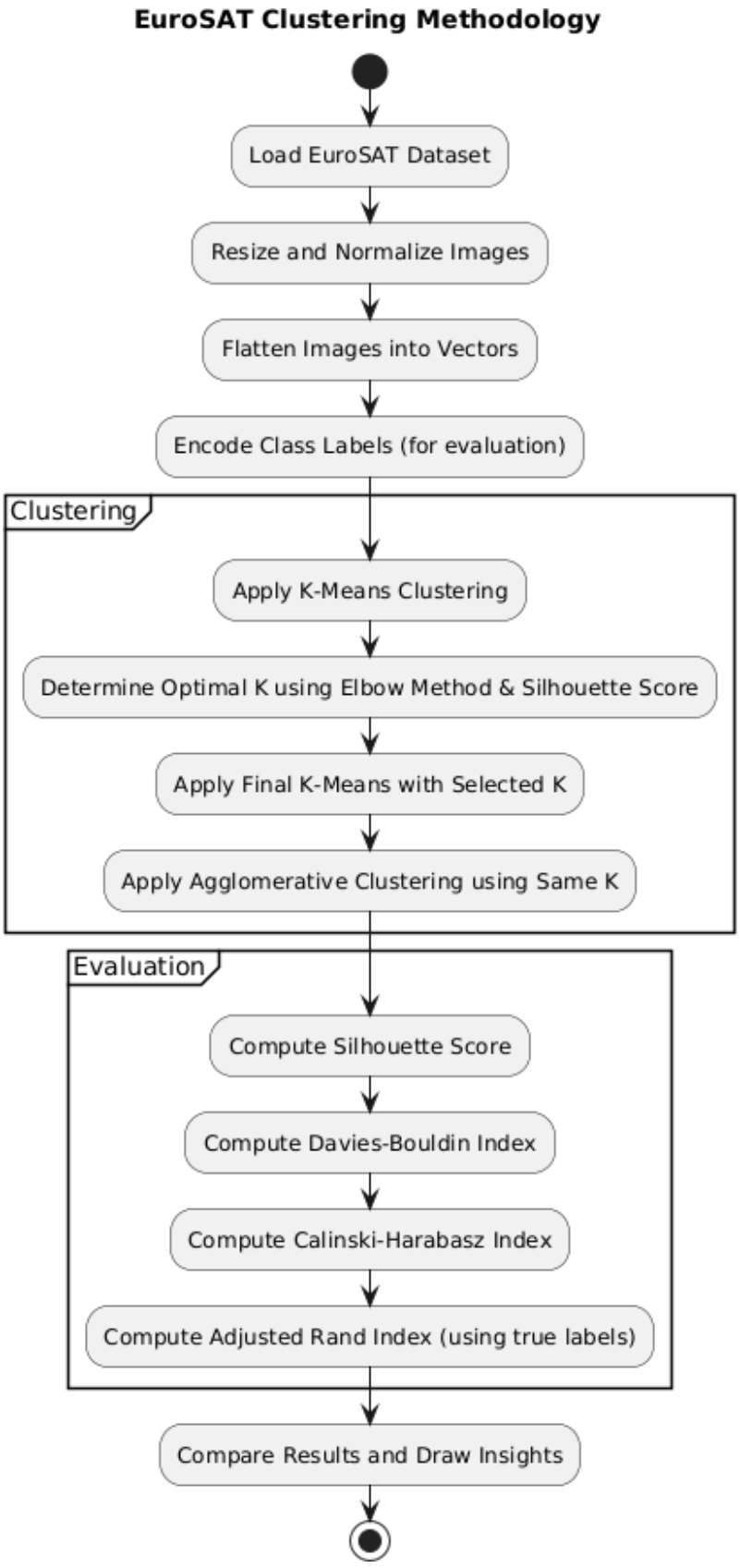
Land Use Classes:

1. Annual Crop
2. Forest
3. Herbaceous Vegetation
4. Highway
5. Industrial
6. Pasture
7. Permanent Crop
8. Residential
9. River
10. Sea/Lake

4. Methodology

The project follows the pipeline below:

- Load the EuroSAT dataset and preprocess all images (resize, normalize, and flatten).
- Encode class labels (used only for evaluation, not during clustering).
- Apply **K-Means Clustering** with varying values of K to find the optimal number of clusters using the Elbow Method and Silhouette Score.
- Apply **Agglomerative Clustering** using the finalized K value for comparison.
- Evaluate clustering performance using **Silhouette Score**, **Davies–Bouldin Index**, **Calinski–Harabasz Index**, and **Adjusted Rand Index**.
- Compare clustering techniques based on structure, separation, and alignment with actual labels.


EuroSAT Clustering Methodology
Load EuroSAT Dataset
Resize and Normalize Images
Flatten Images into Vectors
Encode Class Labels (for evaluation)
Clustering
Apply K-Means Clustering
Determine Optimal K using Elbow Method & Silhouette Score
Apply Final K-Means with Selected K
Apply Agglomerative Clustering using Same K
Evaluation
Compute Silhouette Score
Compute Davies-Bouldin Index
Compute Calinski-Harabasz Index
Compute Adjusted Rand Index (using true labels)
Compare Results and Draw Insights

5. Algorithms and Tech Stack

Algorithms Used:

K-Means Clustering – Unsupervised algorithm for partitioning data into K distinct clusters.

Agglomerative Clustering – Hierarchical clustering that builds clusters bottom-up.

Evaluation Metrics:

- Silhouette Score
- Davies–Bouldin Index
- Calinski–Harabasz Index
- Adjusted Rand Index (with ground truth)

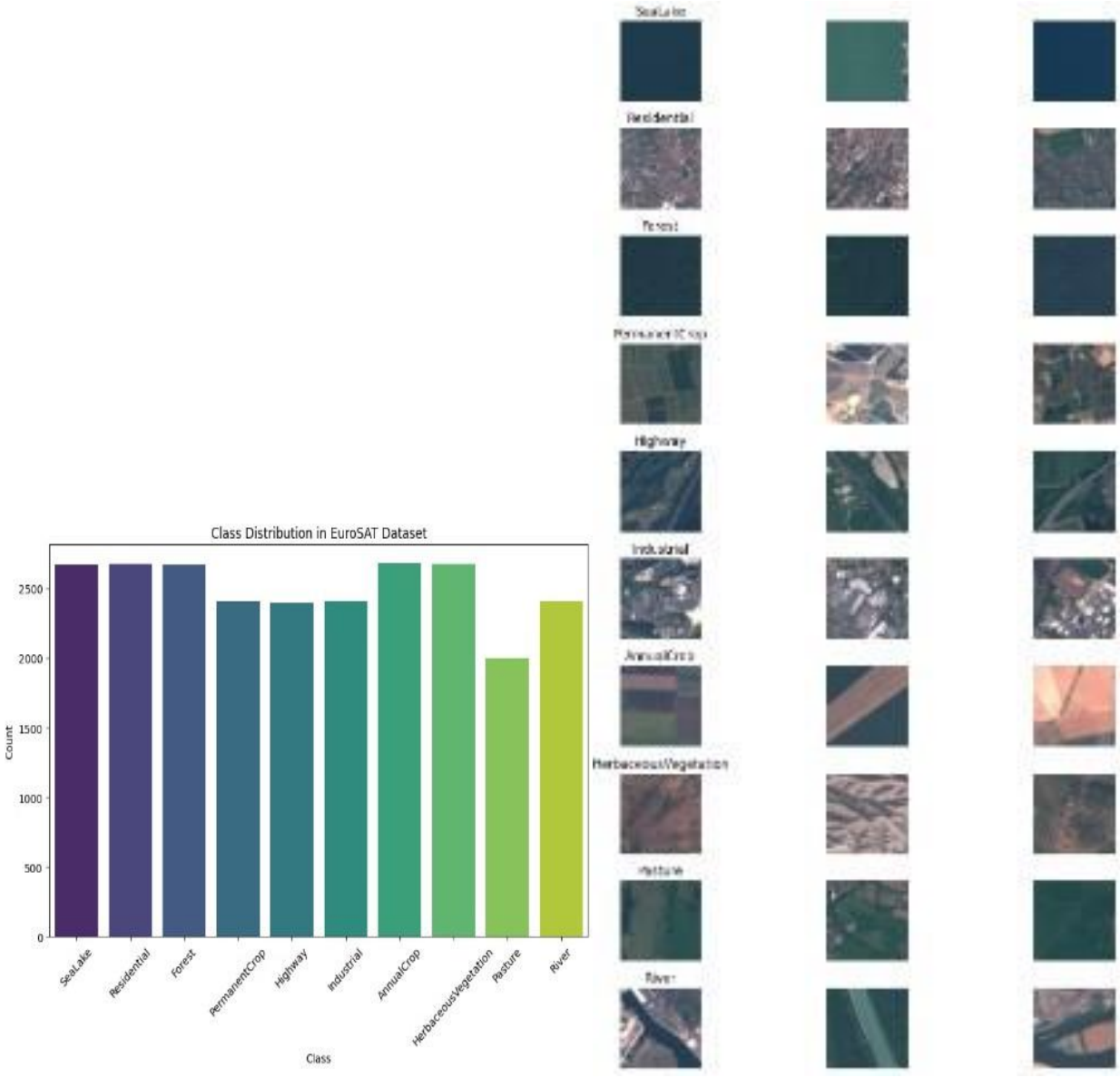
Tools and Libraries:

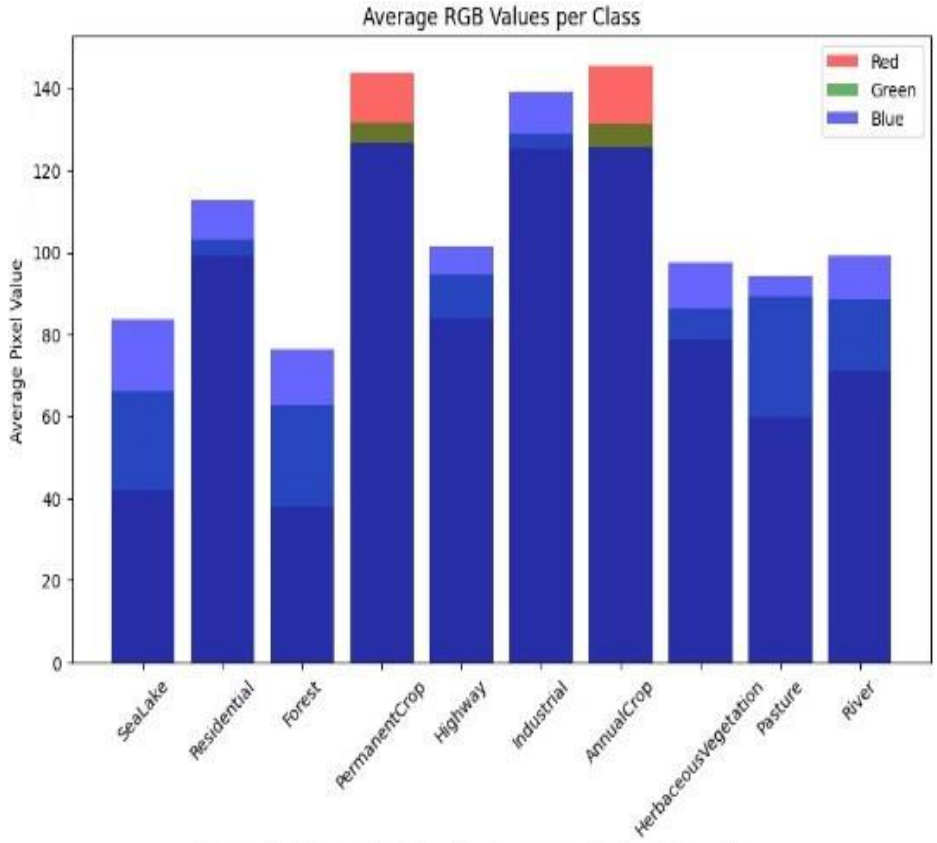
- **Python** – Programming language for all data handling and analysis.
- **Scikit-learn** – Clustering algorithms, performance metrics, preprocessing.
- **Matplotlib, Seaborn** – Visualization of clustering results and evaluation graphs.
- **NumPy, Pandas** – Numerical operations and dataset handling.
- **Jupyter Notebook** – Interactive development environment for executing and documenting the workflow.

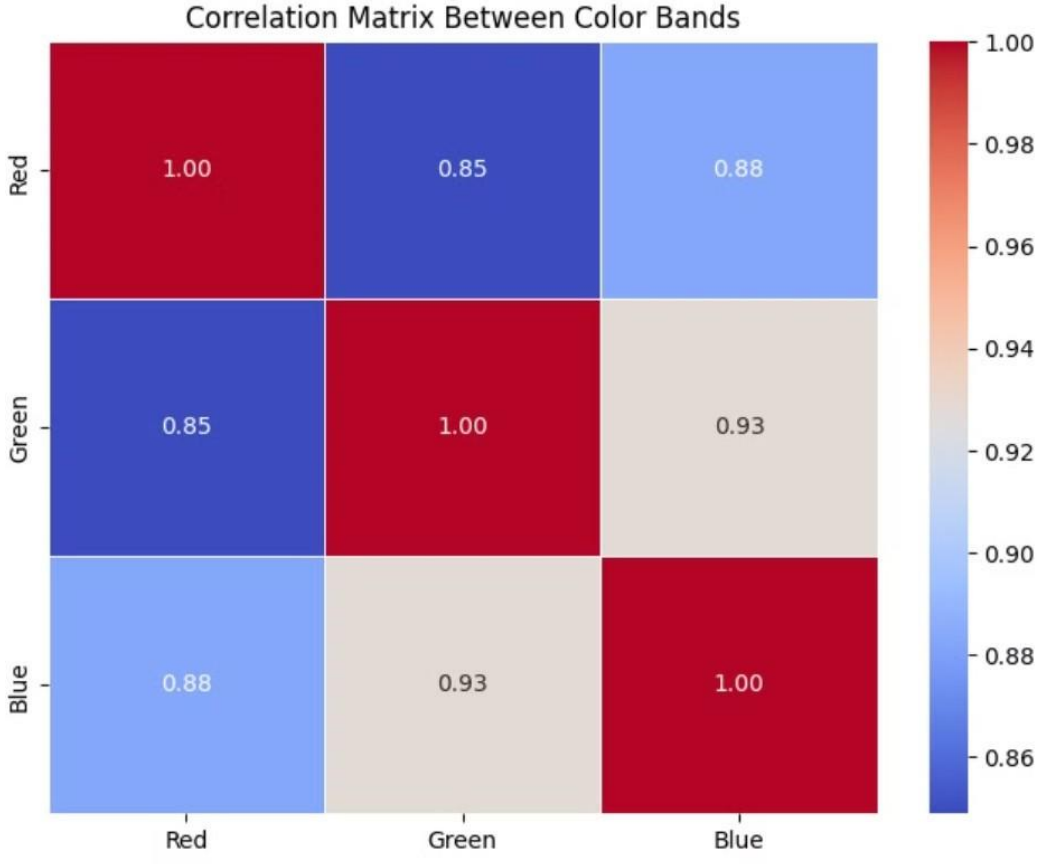
6. Dataset Exploration & Preprocessing

Dataset exploration includes verifying structure, inspecting sample images, checking image dimensions, analyzing class distribution, normalizing pixel values, analyzing color patterns, checking pixel intensity histograms, and creating a correlation matrix.

- Verified that all 10 classes have a roughly equal number of samples (~2,700)
- Displayed sample images to visually distinguish class characteristics.
- Analyzed RGB histograms and pixel distributions.
- Observed high feature correlation in raw image data, supporting the need for dimensionality reduction.




Average RGB Values per Class
Red
Green
Blue
Average Pixel Value
0
20
40
60
80
100
120
140
SeaLake
Residential
Forest
PermanentCrop
Highway
Industrial
AnnualCrop
HerbaceousVegetation
Pasture
River


Correlation Matrix Between Color Bands
Red
Green
Blue
1.00
0.85
0.88
0.85
1.00
0.93
0.88
0.93
1.00
Red
Green
Blue
1.00
0.98
0.96
0.94
0.92
0.90
0.88
0.86

7. Data Handling and Preprocessing

The project uses the EuroSAT satellite image dataset, which contains 27,000 RGB images across 10 land use/land cover classes. Each image is of size 64×64×3, resulting in 12,288 features per image. To make this high-dimensional data suitable for clustering, the following preprocessing steps were applied:

- **Image flattening** to convert 2D arrays into 1D feature vectors.
- **Standardization** to normalize pixel intensities.

8. Implementation

The implementation begins by loading the EuroSAT dataset, which contains RGB satellite images across 10 land use categories. Each image is resized and normalized to ensure uniformity and to make the dataset suitable for machine learning workflows. Images are then flattened into one-dimensional vectors so they can be directly processed by clustering algorithms. Although the true labels are not used for clustering, they are encoded to evaluate the clustering performance afterward.

Five clustering algorithms were implemented: **K-Means,CLARA, Birch, DBSCAN** and **Agglomerative Clustering**. For KMeans, the optimal number of clusters (K) was determined by using the **Elbow Method** and **Silhouette Score** to ensure meaningful groupings. Once the best value of K was identified, both clustering algorithms were applied using that same value to allow for a fair comparison.

To assess clustering quality, multiple performance metrics were calculated, including **Silhouette Score**, **Davies–Bouldin Index**, **Calinski–Harabasz Index**, and **Adjusted Rand Index**. These metrics were used to evaluate how well each algorithm grouped the data and how closely those groupings matched the actual class structure. The implementation was completed in Python using libraries such as Scikit-learn, NumPy, Matplotlib, and Seaborn, all within a Jupyter Notebook environment.

9. Clustering Techniques

1. K-Means Clustering

- **Type**: Partition-based, centroid-driven
- **Core Idea**: Iteratively assign points to the nearest centroid, then update each centroid as the mean of its assigned points.
- **Objective**: Minimize the within-cluster sum of squared errors (SSE).
- **Strengths**:
    - Fast convergence on large datasets
    - Easy to interpret and implement
    - Scales linearly with number of points
- **Limitations**:
    - Must pre-specify K
    - Sensitive to initial centroid placement
    - Assumes spherical clusters of similar size

2. Agglomerative (Hierarchical) Clustering

- **Type**: Hierarchical, bottom-up
- **Core Idea**: Start with each point as its own cluster; at each step, merge the two closest clusters based on a linkage criterion (e.g., single, complete, average).
- **Objective**: Build a dendrogram that captures nested cluster structure.
- **Strengths**:
    - Does not require an initial guess of centroids

- Captures multi-scale, nested relationships
- Choice of linkage and distance metric provides flexibility

- **Limitations**:
  - Computationally expensive for large datasets ($O(n^3)$ worst case)
  - Once merged, clusters cannot be split

3. DBSCAN (Density-Based Spatial Clustering of Applications with Noise)

- **Type**: Density-based
- **Core Idea**: Clusters are defined as areas of high point density separated by areas of low density. Points with fewer than min_samples neighbors within radius ε are labeled as noise.
- **Strengths**:
  - Automatically discovers the number of clusters
  - Handles arbitrarily shaped clusters
  - Robust to outliers (labels them as noise)
- **Limitations**:
  - Choosing ε and min_samples can be non-trivial
  - Struggles with varying densities

4. Birch (Balanced Iterative Reducing and Clustering using Hierarchies)

- **Type**: Hierarchical, tree-based
- **Core Idea**: Incrementally builds a clustering feature (CF) tree, condensing data into subclusters, then optionally refines with a global clustering algorithm.
- **Strengths**:

- Extremely efficient and scalable for very large datasets
- Online (one-pass) construction of CF tree
- Handles noise and outliers well

- **Limitations**:
  - Quality depends on threshold and branching factor parameters
  - May not perform well on high-dimensional sparse data without reduction

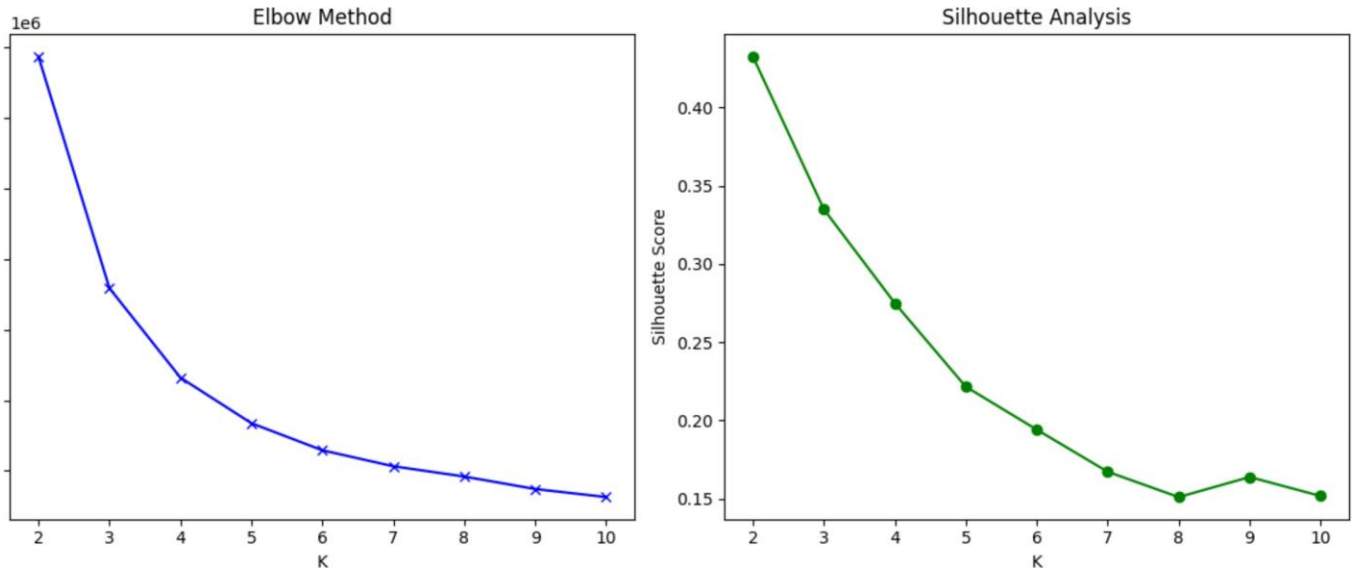
5. CLARA (Clustering Large Applications – K-Medoids)

- **Type**: Partition-based, medoid-driven
- **Core Idea**: Repeatedly samples a subset of the data, applies the PAM (Partitioning Around Medoids) algorithm to that sample, and selects the best clustering. Medoids (actual data points) represent clusters.
- **Strengths**:
  - More robust to noise and outliers than K-Means
  - Medoids are real observations—easier to interpret
  - Scales better to larger datasets via sampling
- **Limitations**:
  - Approximate—quality depends on sampling
  - More computationally expensive per iteration than K-Means

10. Performance Evaluation Metrics

- **Silhouette Score**: Measures how similar an object is to its own cluster compared to others. A higher value (close to 1) indicates well-defined clusters.
- **Davies–Bouldin Index**: Evaluates the average similarity between clusters. Lower values indicate better cluster separation.
- **Calinski–Harabasz Index**: Computes the ratio of between-cluster dispersion to within-cluster dispersion. A higher score suggests well-separated and compact clusters.
- **Adjusted Rand Index (ARI)**: A supervised metric that compares the predicted cluster labels with the true labels. It ranges from -1 to 1, with 1 indicating a perfect match.

11. Results



🔷 Elbow Method:

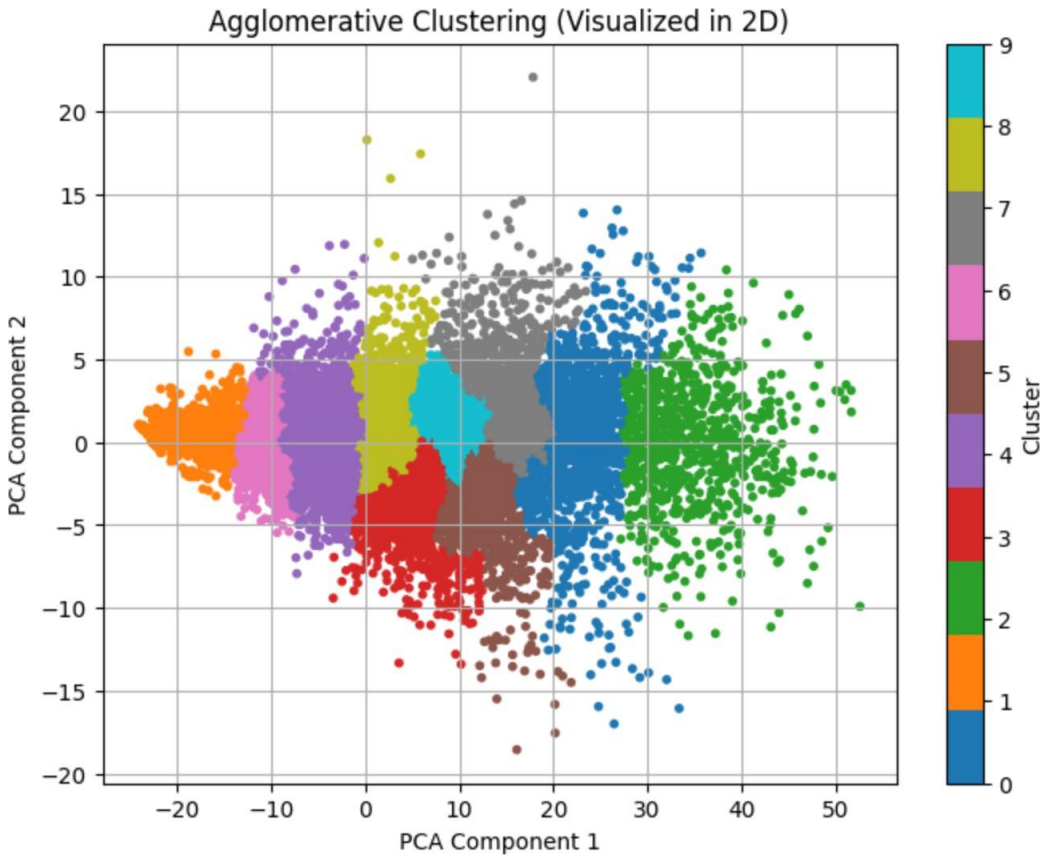
The Elbow plot (left chart) shows a significant drop in SSE (Sum of Squared Errors) as K increases from 2 to 5. After K = 5, the rate of decrease in SSE slows down, indicating diminishing returns. This "elbow" at K = 5 suggests an optimal number of clusters.

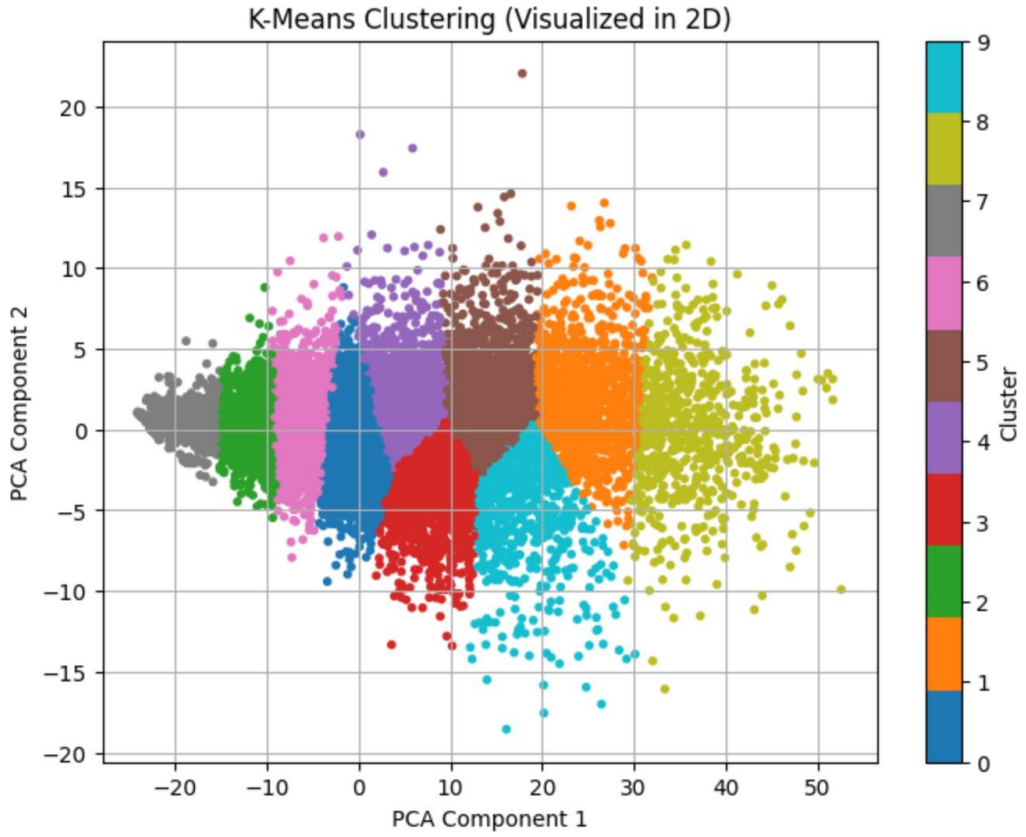
🔷 Silhouette Analysis:

The Silhouette Score (right chart) helps assess how well-separated the clusters are. The highest silhouette score is observed at K = 2, but this may be an over-generalization. The score decreases steadily with increasing K. However, K = 5 still maintains a reasonable silhouette score (~0.22), indicating a fair balance between compactness and separation of clusters.

✅ **Final Selection:**

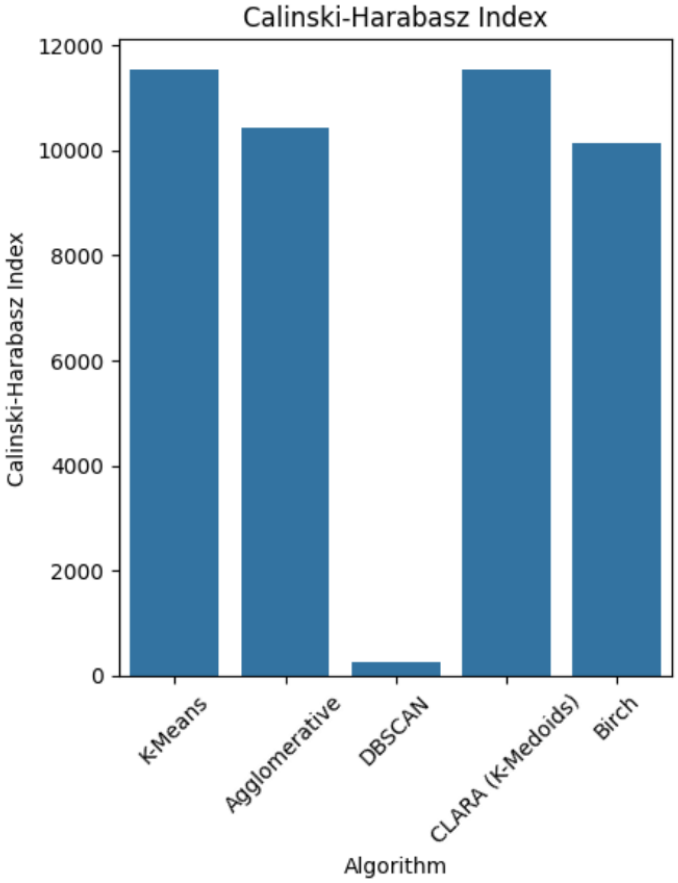
Based on the trade-off between cluster separation and data granularity, K = 5 was chosen for clustering using the K-Means algorithm.

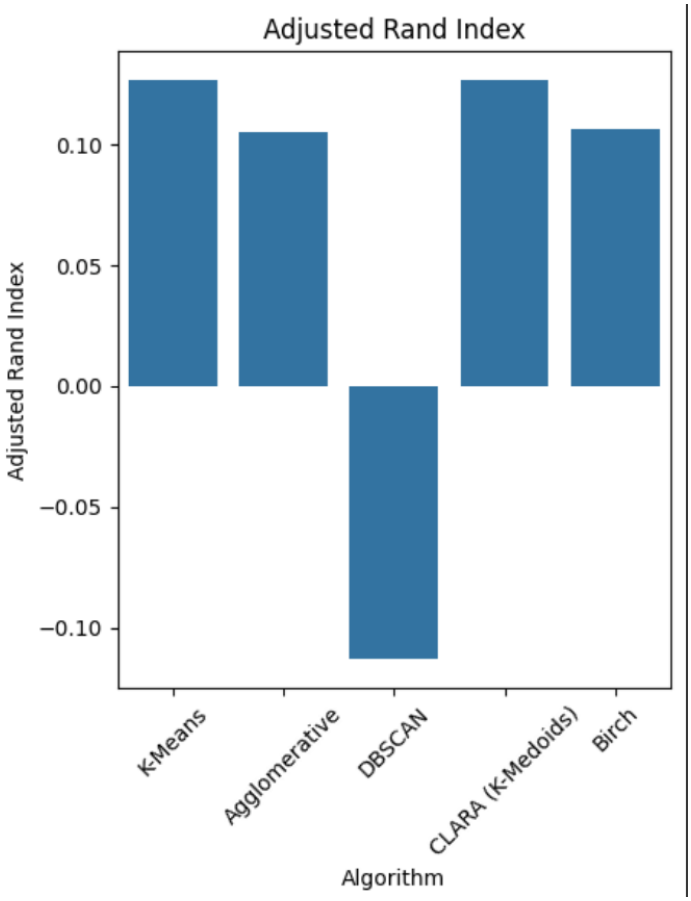


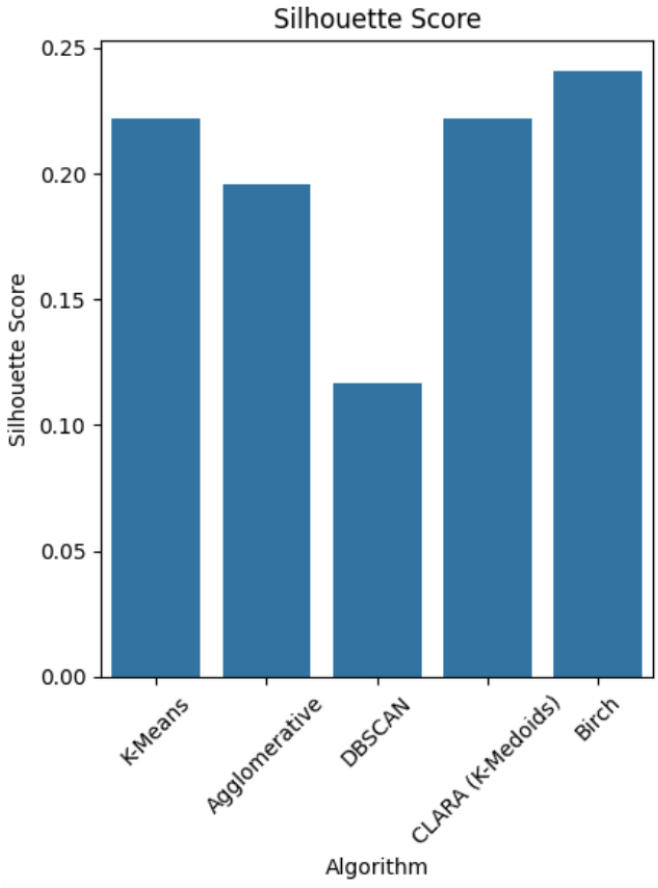


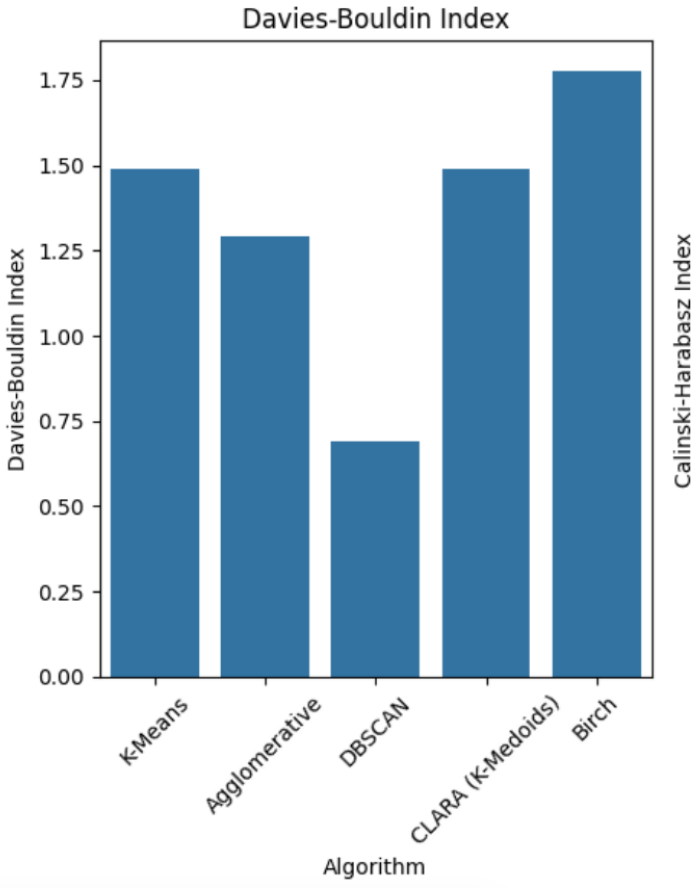
📌 Key Observations:

- **K-Means** and **CLARA (K-Medoids)** performed identically, suggesting robustness across centroid- and medoid-based clustering.
- **Birch** achieved the highest silhouette score (0.2407), indicating relatively better-defined clusters. However, its **Davies-Bouldin Index** was the worst, suggesting less compact clusters.
- **DBSCAN** had the lowest **Calinski-Harabasz Index** and a negative **Adjusted Rand Index**, indicating poor performance, possibly due to the density-based nature being unsuited for the data's distribution.
- **Agglomerative Clustering** performed moderately across all metrics but didn't outperform K-Means.


Calinski-Harabasz Index
Calinski-Harabasz Index
12000
10000
8000
6000
4000
2000
0
K-Means
Agglomerative
DBSCAN
CLARA (K-Medoids)
Birch
Algorithm


Adjusted Rand Index
Adjusted Rand Index
0.10
0.05
0.00
−0.05
−0.10
K-Means
Agglomerative
DBSCAN
CLARA (K-Medoids)
Birch
Algorithm


Silhouette Score
Silhouette Score
0.25
0.20
0.15
0.10
0.05
0.00
K-Means
Agglomerative
DBSCAN
CLARA (K-Medoids)
Birch
Algorithm


Davies-Bouldin Index
Davies-Bouldin Index
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
K-Means
Agglomerative
DBSCAN
CLARA (K-Medoids)
Birch
Algorithm
Calinski-Harabasz Index

12. Conclusion

While Birch achieved the highest silhouette score (0.2407), a deeper analysis across all clustering metrics highlights **K-Means (K = 5)** as the most **balanced, reliable, and well-performing** algorithm. It demonstrated:

- **High cluster cohesion**, as indicated by a low **Sum of Squared Errors (SSE)**.
- **Strong structural validity**, shown by a **competitive Silhouette Score (0.2216)**.
- **Excellent between-cluster separation**, reflected in a **high Calinski-Harabasz Index (11534.33)**.
- **Reasonable ground truth alignment**, based on the **Adjusted Rand Index (0.1266)**.

Additionally, the Elbow and Silhouette plots both support **K = 5** as the optimal number of clusters. These findings collectively justify the selection of **K-Means clustering** as the **most robust and effective** technique for segmenting the given dataset.

13. Future Scope

- **Deep Feature Extraction**: Leverage autoencoders or pre-trained CNN embeddings for richer, non-linear representations before clustering.
- **Semi-Supervised & Ensemble Clustering**: Introduce a small amount of labeled data or combine multiple algorithms (e.g., K-Means + DBSCAN) to boost robustness.
- **Real-Time & Streaming Adaptation**: Develop online clustering pipelines to handle continuous satellite data for applications like disaster response.

14. References

1. Helber et al. (2019). *EuroSAT: A Novel Dataset…* IEEE JSTARS.
2. Pedregosa et al. (2011). *Scikit-learn: Machine Learning in Python*. JMLR.
3. MacQueen (1967). *Some Methods for Classification…* 5th Berkeley Symposium.
4. Ester et al. (1996). *DBSCAN: Density-Based Clustering…* KDD ’96.