Implementation and Evaluation of Gaussian Mixture Models and k-Means Clustering for Classification Tasks

Niti Priya ,*Research Intern , Indian Institute of Technology, Dharwad*

***Abstract*—This report, *Implementation and Evaluation of Gaussian Mixture Models and k-Means Clustering for Classification Tasks*, examines the performance of GMM and k-Means in classifying complex datasets. GMM utilizes probabilistic modeling to capture data distributions through soft clustering, while k-Means classifies instances based on proximity to cluster centroids. The study analyzes the impact of key parameters—such as the number of Gaussian components and the choice of *k*—on classification accuracy and decision boundaries. Results highlight the strengths, limitations, and practical applications of both methods in real-world scenarios.**

I. INTRODUCTION

This report discusses the implementation and evaluation of three classification models: the **Bayesian Classifier**, **Gaussian Mixture Model (GMM)**, and **k-Means Clustering**, with a focus on the use of *Gaussian distributions* for modeling class data. The models are tested on two datasets: the **Cervical Cytology Image Dataset** and the **3-Class Scene Image Dataset**, to demonstrate their effectiveness in real-world classification tasks.

The **Gaussian Mixture Model (GMM)** assumes that data is generated from a mixture of multiple *Gaussian distributions*, each with its own *mean vector* and *covariance matrix*. These covariance matrices capture dependencies between features, shaping the decision boundaries. GMM is particularly suited for clustering tasks and handling non-linearly separable data. **Decision boundary** and **contour plots** are included to visualize how GMM separates the classes in both datasets.

The **k-Means Clustering** algorithm partitions data into *k* clusters based on the mean of the data points within each cluster. The choice of *k* and the method of initialization are critical to its performance. To illustrate the model's behavior, the **centroid movement vs iterations** graph shows how centroids evolve during the clustering process, refining the cluster boundaries on the datasets.

Each model is evaluated using **accuracy**, **precision**, **recall**, and **F-score**. A **confusion matrix** is also included to analyze true positives, true negatives, false positives, and false negatives for a deeper performance understanding. For GMM, the **iterations vs log-likelihood** graph is shown, reflecting the model's convergence during training.

This report compares the effectiveness, robustness, and precision of the Bayesian Classifier, GMM, and k-Means Clustering in solving classification tasks using the Cervical Cytology and 3-Class Scene Image datasets. Through these analyses, we gain a clearer understanding of each model's strengths and limitations across different datasets and classification challenges.

A. Non Linearly Separable Dataset

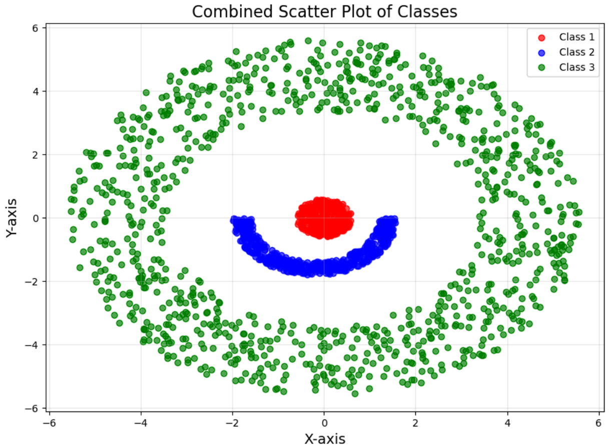


Fig. 1. Non-Linearly Separable Dataset

The second dataset is more complex, where no single straight line can separate the points. The decision boundaries are curved or intricate, and models must adapt to capture the data's non-linear relationships. This dataset tests how well models generalize beyond basic linear separability.

B. Real World Dataset

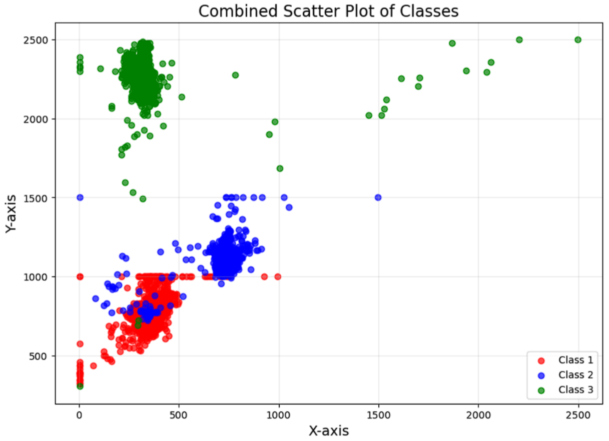


Fig. 2. Real-World Dataset

This dataset is for real-world scenarios, incorporating noisy and outlier data. Real-world data often contains irregularities, overlaps between classes, and inconsistent patterns, unlike artificially generated data. This dataset was used to evaluate how well the models handle imperfections and the complexity typical of real-world applications.

C. Cervical Cytology (Cell) Image Dataset

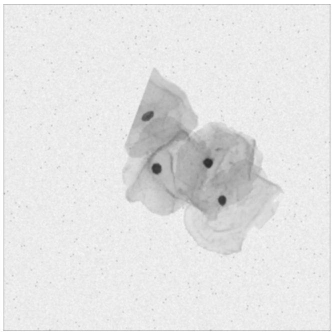


Fig. 3. Cervical cytology (cell) image dataset

This dataset contains cytological images featuring cell clusters with visible nuclei, commonly used for cancer detection, such as in Pap smears for cervical cancer or fine-needle aspiration (FNA) biopsies for breast cancer. By analyzing features like nuclear size, shape, and the nucleus-to-cytoplasm ratio, cells are classified as normal, precancerous, or malignant using Gaussian Mixture Models (GMM) for probabilistic clustering and k-Nearest Neighbors (k-NN) for pattern recognition based on feature similarity. Beyond cancer detection, these images are also used to diagnose other conditions like thyroid disorders and hematological diseases, enhancing early detection and improving diagnostic accuracy through automated analysis.

D. 3-Class Scene Image Dataset

This dataset consists of 3-class scene images, categorized into distinct environments such as urban, rural, and natural landscapes. It is designed for training and evaluating image classification models to recognize and differentiate between various scene types based on visual features like texture, color distribution, and structural elements. Common applications include autonomous navigation, environmental monitoring, and content-based image retrieval.

II. Methodology

1) **Data Preparation:** A multi-class and multi-featured dataset is prepared or acquired for the classification task. In this study, we utilize two real-world datasets: the **Cervical Cytology (Cell) Image Dataset** and the **3-Class Scene Image Dataset**, both of which are non-linearly separable. The dataset is split into a training set and a test set, ensuring an effective evaluation of the model's performance and generalization capability.
2) **Class-wise Statistical Parameter Estimation for GMM:** For the Gaussian Mixture Model (GMM), we estimate the parameters of each Gaussian component. This process involves calculating the *mean vector* and *covariance matrix* for each component using the Expectation-Maximization (EM) algorithm. The GMM parameters are iteratively updated until convergence.

a) ***Mean Vector (*μ_c*):*:** The mean vector for each component c is calculated as the weighted average of the data points assigned to that component. The update rule for the mean vector μ_c is as follows:

$$\mu_c = \frac{\sum_{i=1}^{N_c} \gamma_{ic} X_i}{\sum_{i=1}^{N_c} \gamma_{ic}}$$

where γ_{ic} represents the responsibility that component c takes for data point X_i, and N_c is the total number of data points assigned to component c.

b) ***Covariance Matrix (*Σ_c*):*:** The covariance matrix for each component is computed to capture the feature correlations and variability. It is updated in each iteration of the EM algorithm:

$$\Sigma_c = \frac{\sum_{i=1}^{N_c} \gamma_{ic} (X_i - \mu_c)(X_i - \mu_c)^T}{\sum_{i=1}^{N_c} \gamma_{ic}}$$

where γ_{ic} indicates the responsibility of component c for data point X_i.

c) ***Prior Probability (*π_c*):*:** The prior probability for each Gaussian component is determined based on the fraction of data points assigned to that component:

$$\pi_c = \frac{1}{N} \sum_{i=1}^{N} \gamma_{ic}$$

where N is the total number of data points.

d) ***Likelihood Calculation Using Gaussian Distribution:*:** The likelihood of a data point X given class c follows a multivariate Gaussian distribution:

$$P(X|C = c) = \frac{1}{(2\pi)^{d/2} |\Sigma_c|^{1/2}} \exp\left(-\frac{1}{2} (X - \mu_c)^T \Sigma_c^{-1} (X - \mu_c) \right)$$

where d is the dimension of the feature space, and Σ_c^{-1} is the inverse of the covariance matrix.

3) **k-Means Clustering:** The k-Means clustering algorithm divides the dataset into k distinct clusters, where each data point is assigned to the nearest centroid. The algorithm proceeds iteratively, with data points being reassigned to the nearest centroid and centroids being recalculated based on the updated assignments.

e) ***Centroid Initialization:*:** Initially, the centroids are chosen either randomly or by utilizing a heuristic method, such as k-means++, to ensure a more robust starting point. These centroids represent the centers of the clusters and serve as the foundation for the subsequent iterations.

f) ***Assignment Step:*:** In this step, each data point is assigned to the nearest centroid based on a distance metric, typically the Euclidean distance:

$$c_i = \arg\min_c \|X_i - \mu_c\|$$

where c_i is the cluster assignment for the data point X_i, and μ_c is the centroid of cluster c.

*g) **Update Step:**:* After assigning data points to the clusters, the centroids are updated by computing the mean of all points assigned to each cluster:

$$\mu_c = \frac{1}{N_c} \sum_{i=1}^{N_c} X_i$$

where N_c is the number of points assigned to cluster c.

4) **Model Evaluation for GMM and k-Means:** Both GMM and k-Means models are evaluated using performance metrics such as **precision**, **recall**, and **F-score**, which provide insights into the model's ability to accurately classify data points and handle imbalanced class distributions.
5) **Cluster Visualization:** To gain a deeper understanding of the model's performance, cluster visualizations such as **decision boundaries** and **contour plots** are generated. These visualizations highlight how well the models are able to separate data points into distinct clusters and provide a visual representation of the clustering effectiveness.

III. Results and Analysis Across Dataset Types

K-means clustering is an unsupervised learning algorithm that partitions data into K clusters. The algorithm follows an iterative refinement process:

1) **Initialization:** K cluster centroids are randomly selected.
2) **Assignment:** Each data point is assigned to the nearest centroid based on Euclidean distance.
3) **Update:** The centroids are recomputed as the mean of the assigned points.
4) **Convergence:** Steps 2 and 3 are repeated until centroids stabilize or a stopping criterion (such as a maximum number of iterations) is met.

While K-means is computationally efficient and works well for spherical and well-separated clusters, it has several limitations:

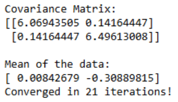
- The need to predefine K, which can be non-trivial.
- Sensitivity to centroid initialization, leading to different cluster assignments.
- Poor performance on non-spherical clusters or datasets with varying cluster densities.
- Susceptibility to outliers, which can disproportionately affect centroid placement.

A. Non-Linearly Separable Dataset

A non-linearly separable dataset consists of data points that cannot be divided into distinct clusters using a simple linear boundary. In such cases, K-means may struggle due to its reliance on Euclidean distance and preference for spherical clusters.

To determine an optimal number of clusters (K), we use the **Elbow Method**, which involves plotting the Within-Cluster Sum of Squares (WCSS) against different values of K. The optimal K is chosen at the "elbow" point, where the rate of decrease in WCSS slows down significantly.

1) Elbow Method Analysis: .



```
Covariance Matrix:
[[6.06943505 0.14164447]
 [0.14164447 6.49613008]]

Mean of the data:
[ 0.00842679 -0.30889815]
Converged in 21 iterations!
```

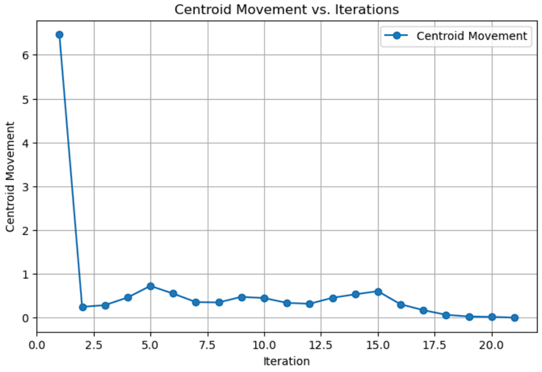


Fig. 4. Elbow method plot for selecting optimal **K** in a non-linearly separable dataset. The inflection point indicates the ideal cluster count.(Euclidian Distance)

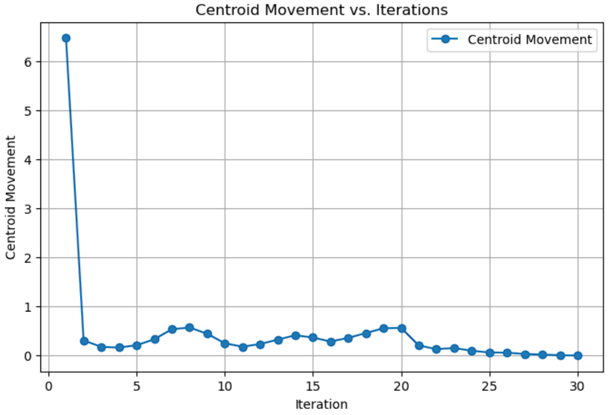


Fig. 5. Elbow method plot for selecting optimal **K** in a non-linearly separable dataset. The inflection point indicates the ideal cluster count.(Mahanalobis Distance)

As observed, the optimal number of clusters is chosen based on the diminishing returns of WCSS reduction. A higher K may lead to overfitting, while a lower K can result in underfitting.

2) *K-Means Clustering Output:* .

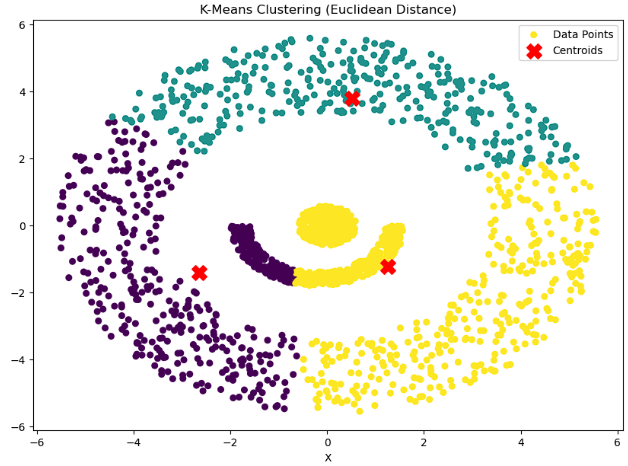


Fig. 6. K-Means Clustering (Euclidian Distance)

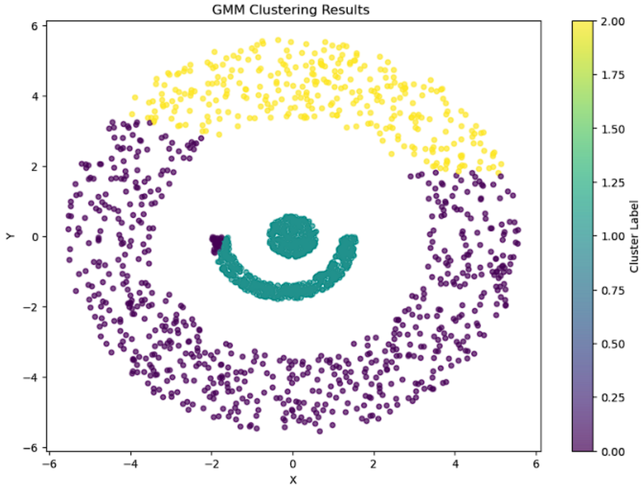


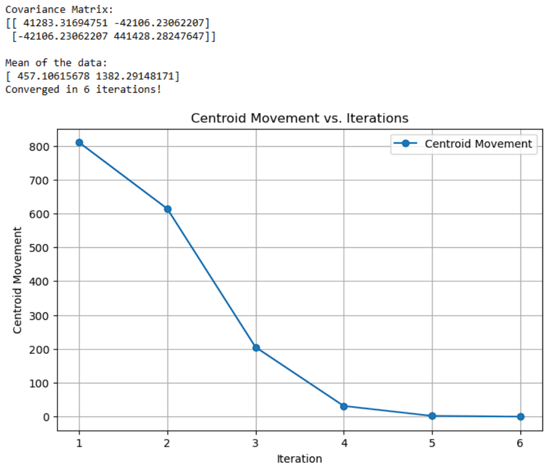
Fig. 7. K-Means Clustering (Mahanalobis Distance)

The output of K-means shows distinct cluster assignments. However, due to the non-linearity of the data, misclassification may occur where points belonging to the same actual class are divided into multiple clusters.

This limitation suggests that alternative clustering methods, such as **Gaussian Mixture Models (GMMs)** or **Spectral Clustering**, could provide better results for such datasets.

B. *Real-World Dataset*

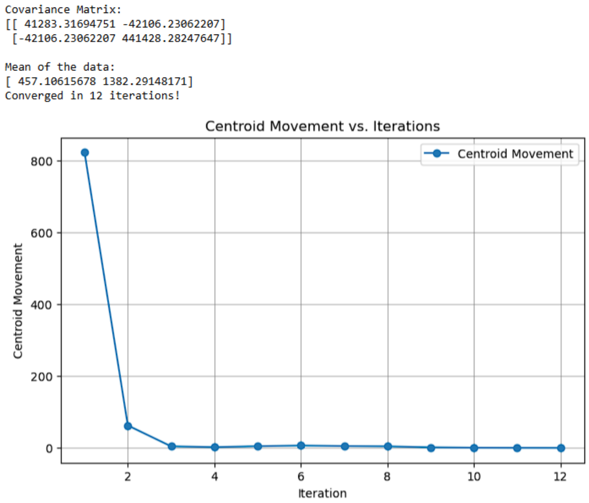
Real-world datasets, like those in image segmentation or customer analysis, often have complex, non-linearly separable patterns. Since K-means relies on Euclidean distance and assumes spherical clusters, it struggles in such cases, leading to poor clustering. Advanced methods like kernel-based clustering or deep learning are better suited for capturing these patterns.



```
Covariance Matrix:
[[ 41283.31694751 -42106.23062207]
 [-42106.23062207 441428.28247647]]

Mean of the data:
[ 457.10615678 1382.29148171]
Converged in 6 iterations!
```

Fig. 8. Elbow method plot for selecting optimal K in a real-world dataset. The inflection point indicates the ideal cluster count.(Euclidian Distance)



```
Covariance Matrix:
[[ 41283.31694751 -42106.23062207]
 [-42106.23062207 441428.28247647]]

Mean of the data:
[ 457.10615678 1382.29148171]
Converged in 12 iterations!
```

Fig. 9. Elbow method plot for selecting optimal K in a real-world dataset. The inflection point indicates the ideal cluster count. (Mahanalobis Distance)

1) *Elbow Method Analysis:* .

As observed, the optimal number of clusters is chosen based on the diminishing returns of WCSS reduction. A higher K may lead to overfitting, while a lower K can result in underfitting.

2) K-Means Clustering Output: The output of K-means shows distinct cluster assignments. However, due to the non-linearity of the data, misclassification may occur where points belonging to the same actual class are divided into multiple clusters.

This limitation suggests that alternative clustering methods, such as **Gaussian Mixture Models (GMMs)** or **Spectral Clustering**, could provide better results for such datasets.

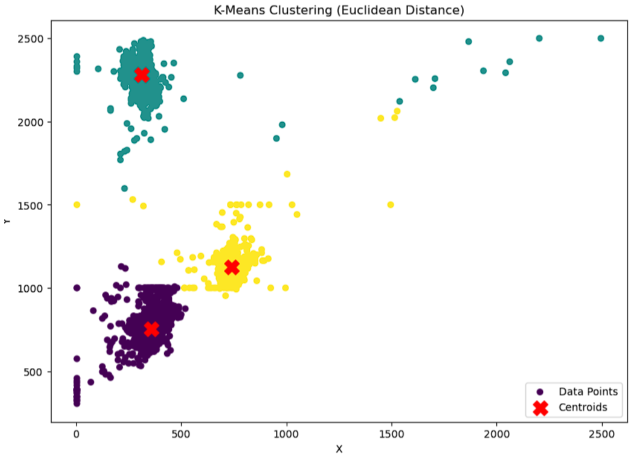


Fig. 10. K-Means Clustering (Euclidean Distance)

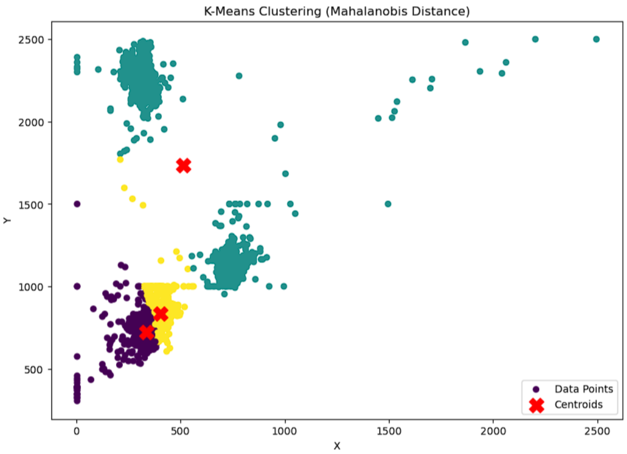
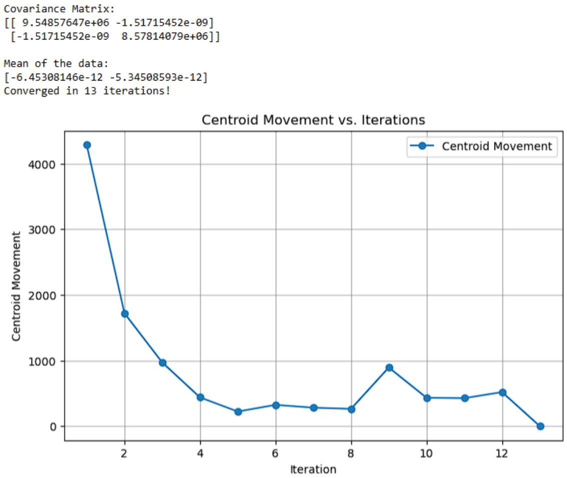


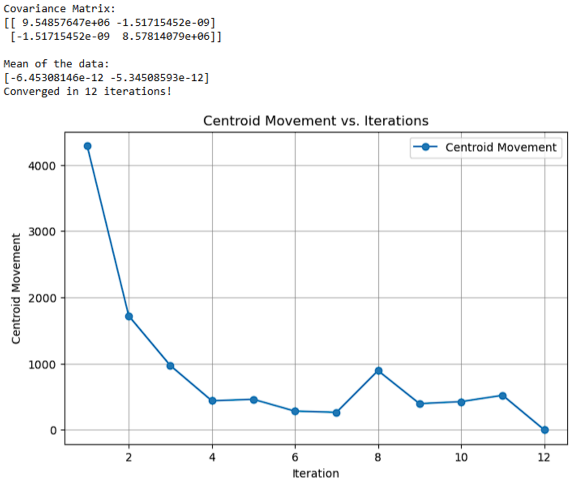
Fig. 11. K-Means Clustering(Mahalanobis Distance)



```
Covariance Matrix:
[[ 9.54857647e+06 -1.51715452e-09]
 [-1.51715452e-09  8.57814079e+06]]

Mean of the data:
[-6.45308146e-12 -5.34508593e-12]
Converged in 13 iterations!
```

Fig. 12. Elbow method to determine the value of **K** for Modified K-means clustering for Cervical cytology(cell) image dataset(Euclidian Distance)



```
Covariance Matrix:
[[ 9.54857647e+06 -1.51715452e-09]
 [-1.51715452e-09  8.57814079e+06]]

Mean of the data:
[-6.45308146e-12 -5.34508593e-12]
Converged in 12 iterations!
```

Fig. 13. Elbow method to determine the value of **K** for Modified K-means clustering for Cervical cytology(cell) image dataset(Mahalanobis Distance)

C. Cervical Cancer(cell) Image Dataset

Cervical cytology images are crucial in detecting abnormal cells that may indicate cervical cancer. Clustering techniques like K-means help in segmenting different cell types for further analysis. However, traditional K-means struggles with non-linearly separable structures, making modifications necessary.

1) Elbow Method Analysis: The elbow method is used to determine the optimal number of clusters (**K**) by plotting the within-cluster sum of squares (WCSS) against different values of **K**. The point at which the curve bends (elbow point) suggests the ideal number of clusters.

2) K-Means Clustering Output: .

The images illustrates the segmentation of a **cervical cytology cell image** using **K-Means clustering**.

- The **left** image represents the **original grayscale image**, showing a microscopic view of cervical cells.
- The **right** image shows the result of **K-Means segmentation**, where different clusters are represented by distinct colors.

K-Means groups pixels based on intensity similarity but assumes clusters are **spherical** and relies on **Euclidean distance**. As a result, the segmentation may appear **rigid and less precise**, especially when dealing with overlapping regions, intensity gradients, or non-uniform textures.

D. Why Gaussian Mixture Model (GMM) is Better for Image Clustering

- **Soft Clustering** – Unlike K-Means, which assigns pixels to a single cluster, GMM provides **probabilistic assignments**, allowing for **smoother transitions** between regions.
- **Flexible Cluster Shapes** – K-Means assumes **spherical clusters**, whereas GMM can model **elliptical or irregularly shaped distributions**, improving segmentation accuracy.
- **Better Intensity Handling** – GMM captures **continuous intensity variations** in medical images, making it more effective for **overlapping structures and subtle texture differences**.
- **Improved Boundary Definition** – By estimating proba-

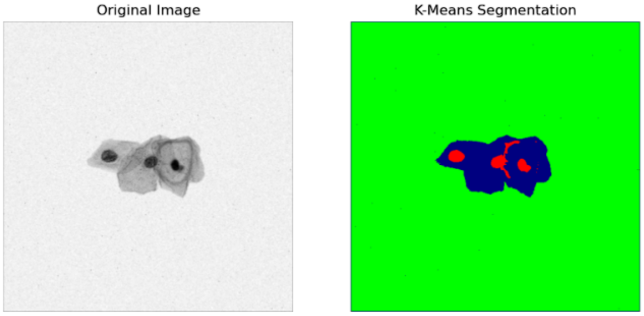


Fig. 14. Test Image-1 output

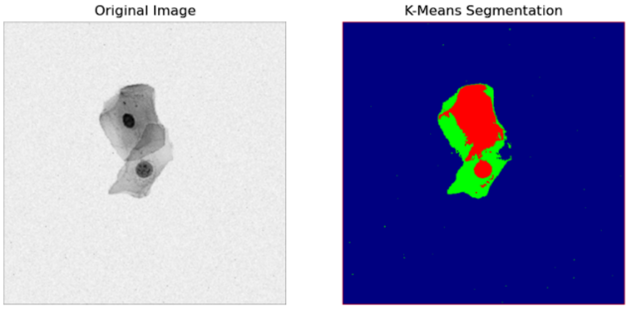


Fig. 15. Test Image-2 output

bility densities, GMM produces **more natural and precise boundaries**, essential for accurate medical image segmentation.

IV. Gaussian Mixture Model

A **Gaussian Mixture Model (GMM)** is a probabilistic approach for modeling complex datasets by assuming that the data is generated from a mixture of multiple Gaussian (normal) distributions. Each Gaussian component in the mixture is defined by three key parameters:

- **Mean** (μ) – Represents the center of the distribution.
- **Covariance** (Σ) – Defines the spread and shape of the distribution.
- **Mixing coefficient** (π) – Represents the weight of each Gaussian component in the overall mixture, ensuring that the sum of all weights equals one.

Unlike traditional clustering algorithms like K-means, which assign data points to a single cluster, **GMM performs soft clustering** by estimating the probability that each data point belongs to each Gaussian component. This allows for overlapping clusters, making GMMs highly suitable for data with complex underlying distributions.

1) How GMM Works: The parameters of a Gaussian Mixture Model are typically estimated using the **Expectation-Maximization (EM) algorithm**, which iteratively optimizes the model by performing two key steps:

1) **Expectation (E-step)** – Computes the probability (responsibility) that each data point was generated by each Gaussian component, based on the current parameter estimates.
2) **Maximization (M-step)** – Updates the parameters (μ, Σ, π) of each Gaussian component by maximizing the likelihood of the data given the computed probabilities.

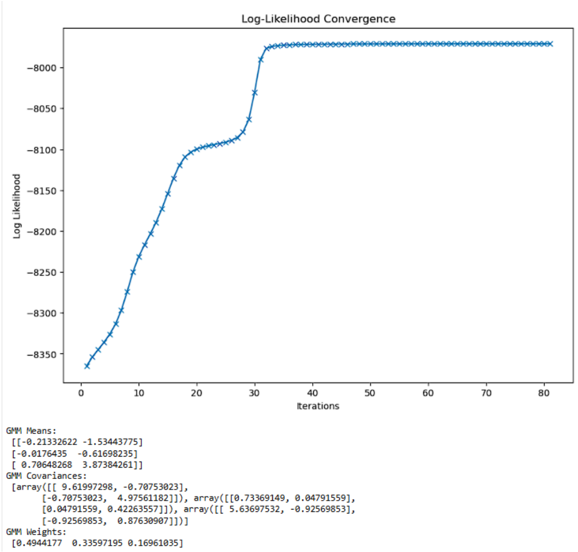
This iterative process continues until convergence, ensuring that the model accurately captures the underlying structure of the data.

2) Applications of GMM: Due to its flexibility and ability to model **multi-modal distributions**, GMM is widely used in:

- **Clustering** – Assigning data points to clusters in an overlapping manner.
- **Density Estimation** – Estimating the probability distribution of complex datasets.
- **Anomaly Detection** – Identifying outliers by analyzing low-probability regions of the fitted model.
- **Pattern Recognition** – Applications in speech recognition, image segmentation, and bioinformatics.

By leveraging the probabilistic nature of GMM, it provides a more nuanced understanding of data distributions compared to hard clustering methods, making it a robust tool for various machine learning and data analysis tasks.

A. Non-Linearly Separable Dataset



```
GMM Means:
 [[-0.21332622 -1.53443775]
 [-0.0176435  -0.61698235]
 [ 0.70648268  3.87384261]]
GMM Covariances:
 [array([[ 9.61997298, -0.70753023],
       [-0.70753023,  4.97561182]]), array([[0.73369149, 0.04791559],
       [0.04791559, 0.42263557]]), array([[ 5.63697532, -0.92569853],
       [-0.92569853,  0.87630907]])]
GMM Weights:
 [0.4944177  0.33597195 0.16961035]
```

Fig. 16. Log likelihood convergence for GMM for non-linearly separable

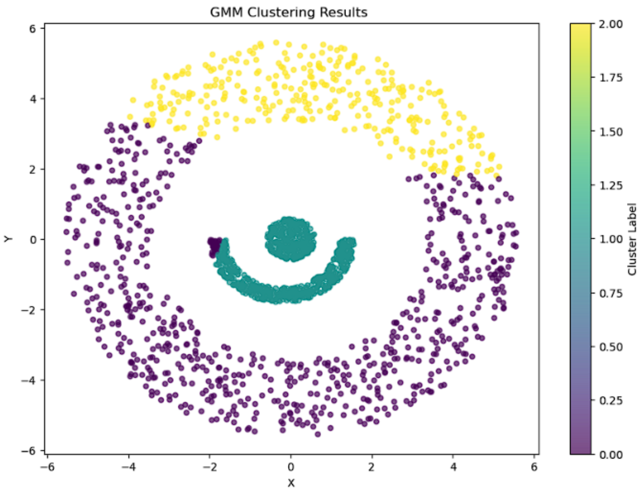
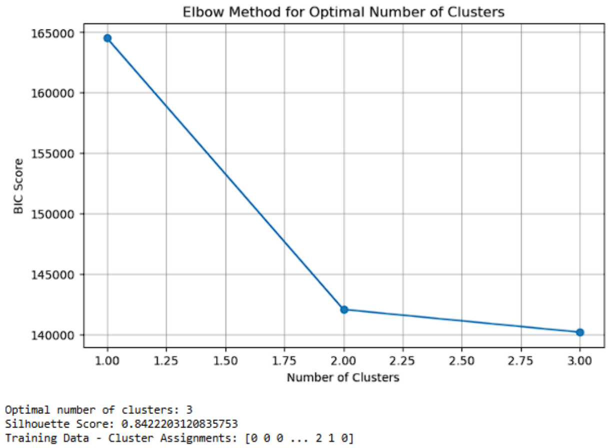


Fig. 17. GMM clustering results for non-linearly separable dataset

B. Real-World Dataset



```
Optimal number of clusters: 3
Silhouette Score: 0.8422203120835753
Training Data - Cluster Assignments: [0 0 0 ... 2 1 0]
```

Fig. 18. Elbow method for optimal number of clusters in real-world dataset

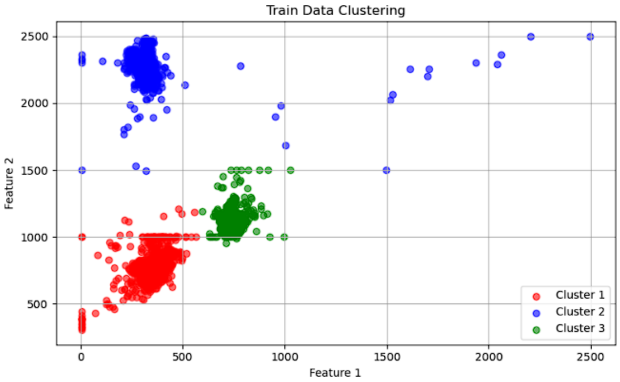
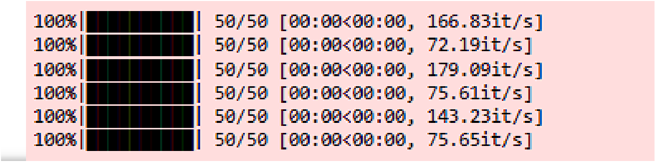


Fig. 19. Output of Gaussian Mixture Model for real-world dataset

C. Real Image Dataset



```
100%|██████████| 50/50 [00:00<00:00, 166.83it/s]
100%|██████████| 50/50 [00:00<00:00, 72.19it/s]
100%|██████████| 50/50 [00:00<00:00, 179.09it/s]
100%|██████████| 50/50 [00:00<00:00, 75.61it/s]
100%|██████████| 50/50 [00:00<00:00, 143.23it/s]
100%|██████████| 50/50 [00:00<00:00, 75.65it/s]
```

Fig. 20. Image processing for GMM

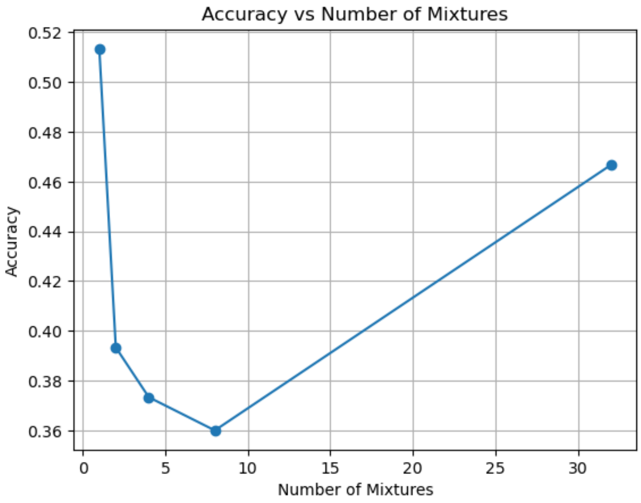


Fig. 21. Accuracy of Model vs No. of mixture (Iterative approach) of GMM model for Real images dataset

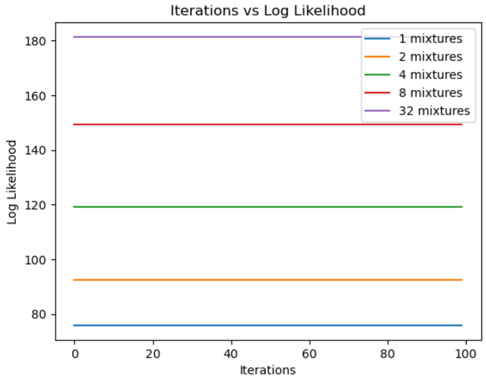
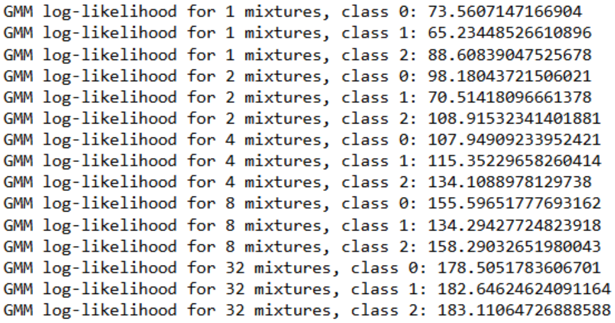
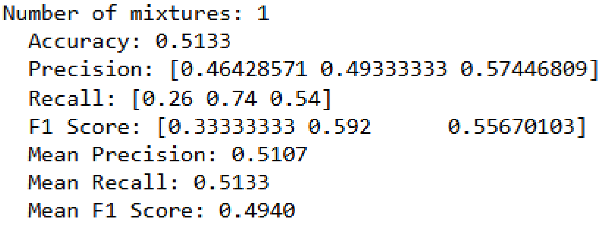


Fig. 22. Iteration vs log-likelihood of GMM model for Real images dataset



```
GMM log-likelihood for 1 mixtures, class 0: 73.5607147166904
GMM log-likelihood for 1 mixtures, class 1: 65.23448526610896
GMM log-likelihood for 1 mixtures, class 2: 88.60839047525678
GMM log-likelihood for 2 mixtures, class 0: 98.18043721506021
GMM log-likelihood for 2 mixtures, class 1: 70.51418096661378
GMM log-likelihood for 2 mixtures, class 2: 108.91532341401881
GMM log-likelihood for 4 mixtures, class 0: 107.94909233952421
GMM log-likelihood for 4 mixtures, class 1: 115.35229658260414
GMM log-likelihood for 4 mixtures, class 2: 134.1088978129738
GMM log-likelihood for 8 mixtures, class 0: 155.59651777693162
GMM log-likelihood for 8 mixtures, class 1: 134.29427724823918
GMM log-likelihood for 8 mixtures, class 2: 158.29032651980043
GMM log-likelihood for 32 mixtures, class 0: 178.5051783606701
GMM log-likelihood for 32 mixtures, class 1: 182.64624624091164
GMM log-likelihood for 32 mixtures, class 2: 183.11064726888588
```

Fig. 23. GMM log-likelihood for different No. of mixtures of GMM model for Real images dataset



```
Number of mixtures: 1
  Accuracy: 0.5133
  Precision: [0.46428571 0.49333333 0.57446809]
  Recall: [0.26 0.74 0.54]
  F1 Score: [0.33333333 0.592      0.55670103]
  Mean Precision: 0.5107
  Mean Recall: 0.5133
  Mean F1 Score: 0.4940
```

Fig. 24. Accuracy Metrics for GMM with no. of mixtures:1

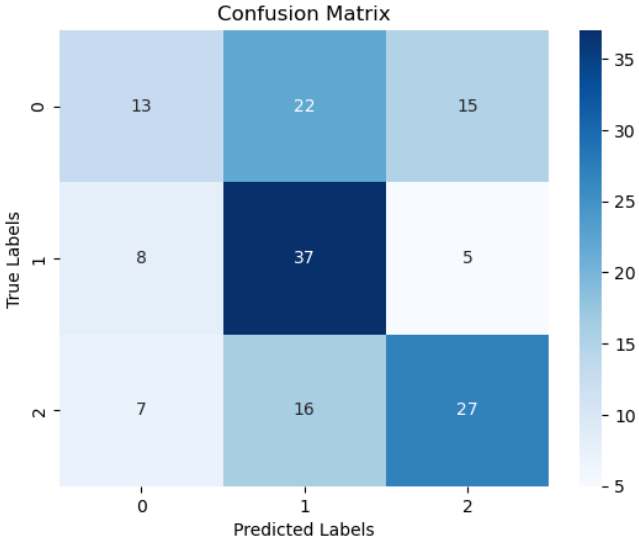
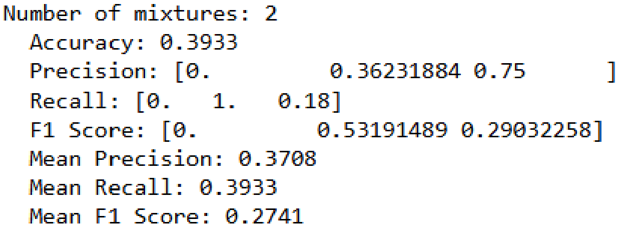


Fig. 25. Confusion Matrix for 1 mixture



```
Number of mixtures: 2
  Accuracy: 0.3933
  Precision: [0.         0.36231884 0.75      ]
  Recall: [0.   1.   0.18]
  F1 Score: [0.         0.53191489 0.29032258]
  Mean Precision: 0.3708
  Mean Recall: 0.3933
  Mean F1 Score: 0.2741
```

Fig. 26. Accuracy Metrics for 2 mixtures

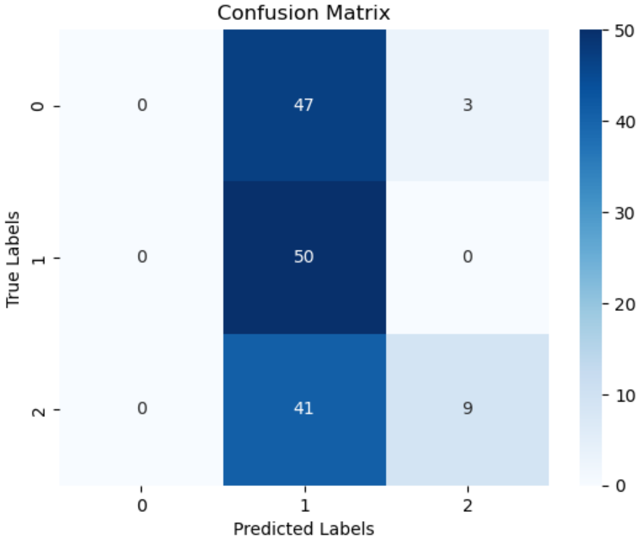
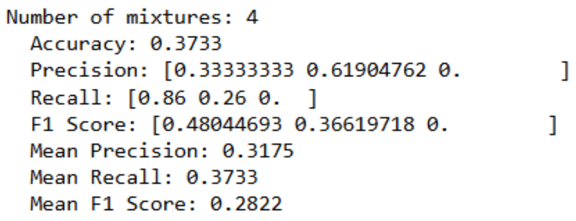


Fig. 27. Confusion Matrix for 2 mixtures



```
Number of mixtures: 4
  Accuracy: 0.3733
  Precision: [0.33333333 0.61904762 0.        ]
  Recall: [0.86 0.26 0.  ]
  F1 Score: [0.48044693 0.36619718 0.        ]
  Mean Precision: 0.3175
  Mean Recall: 0.3733
  Mean F1 Score: 0.2822
```

Fig. 28. Accuracy Metrics for 4 mixtures

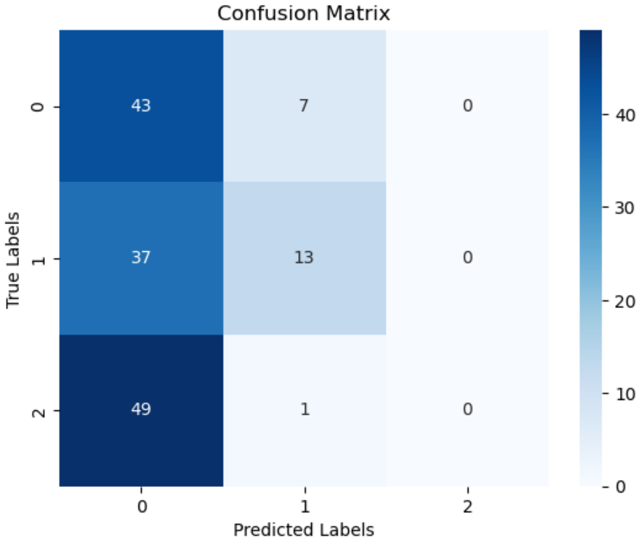
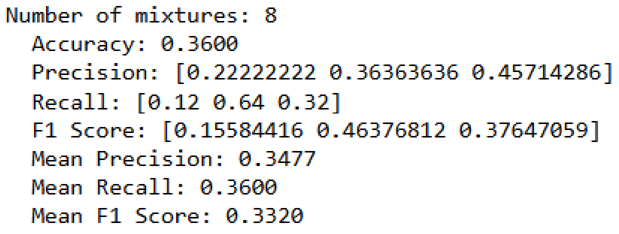


Fig. 29. Confusion Matrix for 4 mixtures



```
Number of mixtures: 8
  Accuracy: 0.3600
  Precision: [0.22222222 0.36363636 0.45714286]
  Recall: [0.12 0.64 0.32]
  F1 Score: [0.15584416 0.46376812 0.37647059]
  Mean Precision: 0.3477
  Mean Recall: 0.3600
  Mean F1 Score: 0.3320
```

Fig. 30. Accuracy Metrics for 8 mixtures

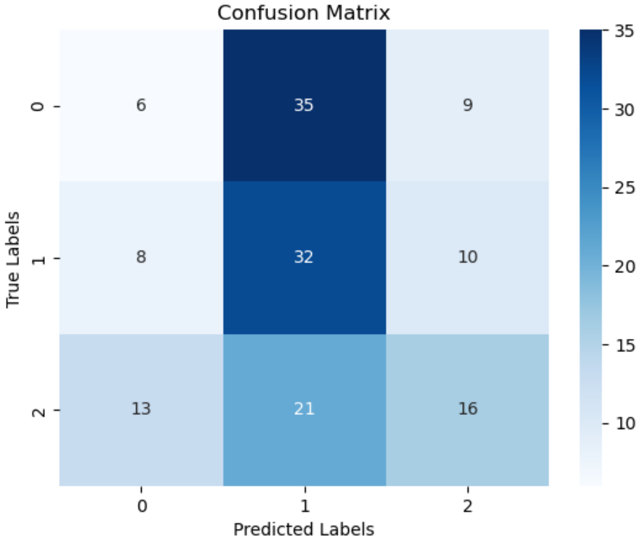
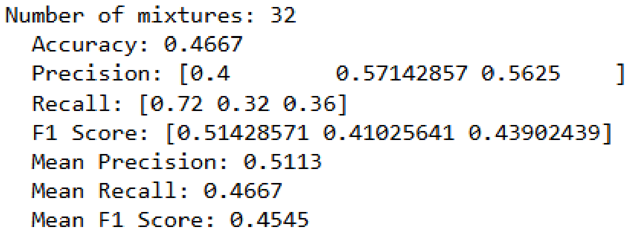


Fig. 31. Confusion Matrix for 8 mixtures



```
Number of mixtures: 32
  Accuracy: 0.4667
  Precision: [0.4        0.57142857 0.5625    ]
  Recall: [0.72 0.32 0.36]
  F1 Score: [0.51428571 0.41025641 0.43902439]
  Mean Precision: 0.5113
  Mean Recall: 0.4667
  Mean F1 Score: 0.4545
```

Fig. 32. Accuracy Metrics for 32 mixtures

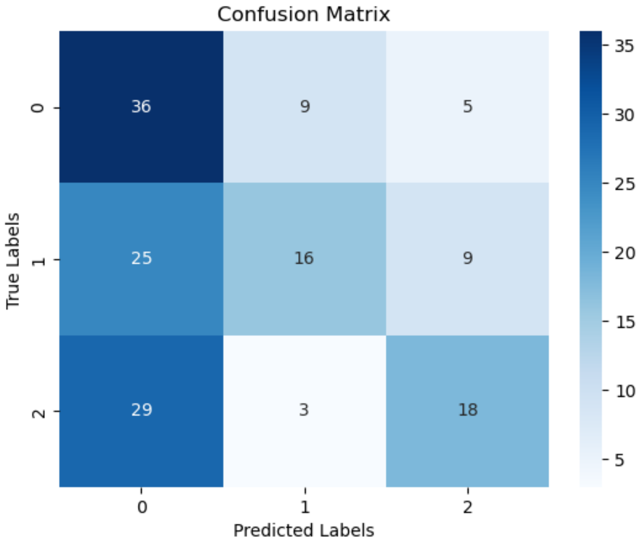


Fig. 33. Confusion Matrix for 32 mixtures

V. Key Takeaways on Covariance Effects on Decision Boundaries

A. K-Means Clustering

Algorithm: K-Means is an iterative clustering algorithm that partitions data into K clusters by minimizing the variance within each cluster. It follows these steps:

1) Randomly initialize K centroids.
2) Assign each data point to the nearest centroid based on Euclidean distance.
3) Update centroids by computing the mean of all points assigned to each cluster.
4) Repeat until convergence (i.e., centroids do not change significantly).

Characteristics:

- Assumes clusters are **spherical** and of **similar size**.
- **Sensitive** to initial centroid placement, which can affect final clusters.
- Requires **prior specification of K**, the number of clusters.
- Uses **hard clustering**, meaning each point belongs to exactly one cluster.

Applications:

- **Customer Segmentation** – Grouping customers based on purchasing behavior.
- **Image Compression** – Reducing color variations by clustering pixel intensities.
- **Anomaly Detection** – Identifying unusual patterns in data.

B. Modified K-Means Clustering

Characteristics:

- Can handle **non-spherical** and **unequal-sized** clusters better than standard K-Means.
- More **robust to outliers** and less sensitive to initial centroid placement.
- Uses improved initialization techniques like **K-Means++** to optimize initial centroids.
- Includes variations such as **Fuzzy C-Means**, which assigns probabilities rather than hard cluster assignments.

Applications:

- **Enhanced Customer Segmentation** – More refined groupings by allowing flexible boundaries.
- **Improved Image Compression** – Preserving details better in clustered images.
- **Robust Anomaly Detection** – Reducing false positives in identifying outliers.

C. Gaussian Mixture Model (GMM)

Algorithm: GMM assumes that data is generated from a mixture of multiple Gaussian distributions. The model estimates the probability that each data point belongs to each Gaussian component using the **Expectation-Maximization (EM) algorithm**:

1) **Expectation Step (E-step)** – Compute probabilities of each data point belonging to each Gaussian component.
2) **Maximization Step (M-step)** – Update Gaussian parameters (mean, covariance, and mixing coefficient) to maximize likelihood.
3) Repeat until convergence.

Characteristics:

- Can model **clusters of different shapes and sizes** by using covariance matrices.
- Provides **soft clustering**, where each data point has a probability of belonging to multiple clusters.
- More **flexible** in modeling complex, overlapping data distributions compared to K-Means.

Applications:

- **Density Estimation** – Learning probability distributions of datasets.
- **Generative Modeling** – Used in applications like speech synthesis and handwriting generation.
- **Image Segmentation** – More accurate clustering of pixel intensities in medical and natural images.
- **Clustering Complex Datasets** – Suitable for data with varying densities and irregular cluster shapes.

VI. Conclusion

For non-linearly separable real-world datasets, particularly in image processing and clustering tasks, the choice between K-Means, Modified K-Means, and Gaussian Mixture Models (GMMs) depends on the data's complexity, cluster distribution, and computational constraints.

K-Means is widely used due to its simplicity and efficiency, making it suitable for large datasets. However, its reliance on Euclidean distance and the assumption of spherical, equally-sized clusters limit its effectiveness in handling complex, overlapping, or irregularly shaped clusters. Additionally, its sensitivity to centroid initialization can lead to suboptimal clustering.

Modified K-Means variants, such as **K-Means++** (which improves centroid initialization) and **Fuzzy C-Means** (which allows soft clustering), offer enhanced performance. These modifications help mitigate the shortcomings of standard K-Means by improving convergence stability and enabling better cluster flexibility. This makes Modified K-Means more effective for datasets with non-spherical distributions while maintaining computational efficiency.

Gaussian Mixture Models (GMMs) provide the highest flexibility in modeling data distributions by allowing clusters to take various shapes and sizes. Unlike K-Means, GMMs assign probabilities to each data point, enabling soft clustering, which is particularly useful for datasets with overlapping regions or gradual transitions between clusters. However, this flexibility comes at a cost: GMMs are computationally more expensive due to the iterative Expectation-Maximization (EM) algorithm and require careful hyperparameter tuning to avoid overfitting.

Overall, K-Means remains a fast and efficient clustering method for simple datasets, while Modified K-Means provides a balance between efficiency and adaptability. GMMs excel in capturing nuanced patterns in complex data but demand greater computational resources and parameter optimization. The choice among these methods should be guided by dataset characteristics, clustering objectives, and available computational resources.