Statistical Pattern Recognition Assignment-4

Kapil Kumar Bhardwaj – CS24MT012
Manish Bisht – CS24MT022
Mridul Chandrawanshi –CS24MT002

Table of Contents:

Chapter 1: Bayes classifier using Gaussian Mixture Model on the reduced dimensional representation of Dataset2 obtained using PCA.

Introduction

Principal Component Analysis:

Principal Component Analysis (PCA) is a dimension-reduction tool that can be used to reduce a large set of variables to a small set that still contains most of the information in the large set. PCA transforms a number of correlated variables into smaller number of uncorrelated variables called principal components.

PCA only takes those directions in which the information content is maximum. While doingso it does not take into account the separability of data. It might so happen that, the direction in which separability of 2 classes is maximum is not the direction of maximum variance of projected data.

Task 1 :
"Build Bayes classifier using Gaussian mixture model (GMM) with 1, 2, 4 and 8 mixtures on the reduced dimensional representations of Dataset-2 obtained using PCA.
Perform the experiments on different values of *l (including l*=1*),* the reduced dimensions in PCA."

Methodology

This methodology outlines the steps taken to perform Principal Component Analysis (PCA) on the dataset, including data normalization and dimensionality reduction.

1. Data Normalization

- The training and testing data are normalized to ensure numerical stability and improve the PCA computation:
  - The mean (μ) and standard deviation (σ) of each feature are calculated using the training data.
  - Both training and testing data are standardized as:

$$X_{\text{norm}} = \frac{X - \mu}{\sigma + 10^{-6}}$$

The small constant (10e-6) is added to avoid division by zero.

2. Principal Component Analysis (PCA)

PCA is applied to reduce the dimensionality of the data while retaining most of the variance. The steps are as follows:

1. **Mean Cantering**:
   - The mean vector (**μ**) of the dataset is computed, and the data is cantered by subtracting μ from each sample: $X_{\text{centered}} = X - \mu$
2. **Covariance Matrix Calculation**:

   - The covariance matrix (Σ) of the centered data is computed to measure feature correlations: $\Sigma = \frac{1}{n} X_{\text{centered}}^T X_{\text{centered}}$

3. **Eigenvalue and Eigenvector Decomposition**:

- The eigenvalues (λ) and eigenvectors (**v**) of the covariance matrix are calculated to identify principal components.

4. **Sorting Components**:

- The eigenvalues and their corresponding eigenvectors are sorted in descending order based on the magnitude of the eigenvalues. This ensures that the principal components with the most variance are prioritized.

5. **Dimensionality Reduction**:

- The top l eigenvectors (corresponding to the largest eigenvalues) are selected as the principal components.
- The data is projected onto the selected components: $X_{\text{reduced}} = X_{\text{centered}} \mathbf{V_l}$

  where ($\mathbf{V_l}$) contains the top l eigenvectors.

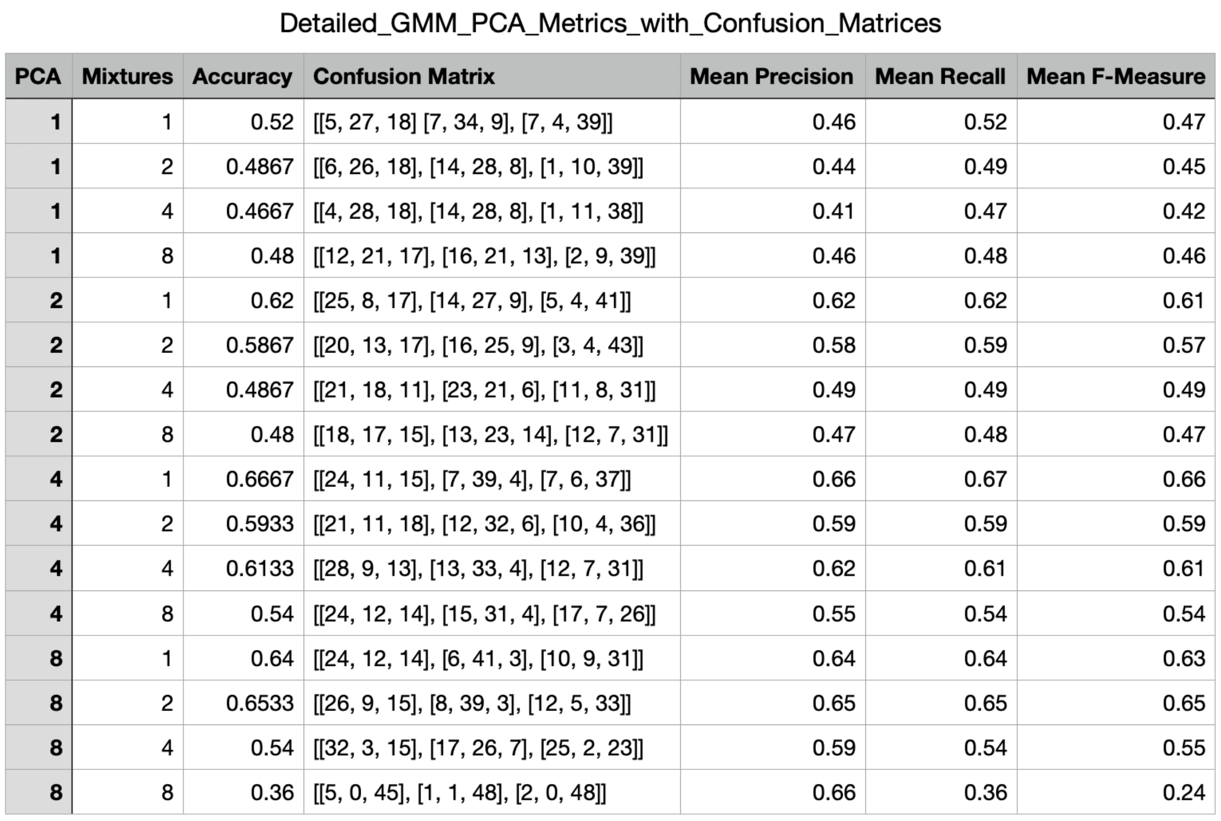
6. **Experiments with Different Dimensions**

- PCA is performed for different values of l (number of components): l=1,2,4,8
- The training data is projected onto the top l components to create reduced datasets for training.
- The same eigenvectors and mean are used to project the test data onto the same reduced space for evaluation.

7. **Visualization of Eigenvalues**

- The eigenvalues are plotted in descending order to visualize the amount of variance captured by each principal component.
- This plot helps to determine the optimal number of components for dimensionality reduction by observing the variance explained by each component.

Results & Plots:



Detailed_GMM_PCA_Metrics_with_Confusion_Matrices

PCA	Mixtures	Accuracy	Confusion Matrix	Mean Precision	Mean Recall	Mean F-Measure
1	1	0.52	[[5, 27, 18] [7, 34, 9], [7, 4, 39]]	0.46	0.52	0.47
1	2	0.4867	[[6, 26, 18], [14, 28, 8], [1, 10, 39]]	0.44	0.49	0.45
1	4	0.4667	[[4, 28, 18], [14, 28, 8], [1, 11, 38]]	0.41	0.47	0.42
1	8	0.48	[[12, 21, 17], [16, 21, 13], [2, 9, 39]]	0.46	0.48	0.46
2	1	0.62	[[25, 8, 17], [14, 27, 9], [5, 4, 41]]	0.62	0.62	0.61
2	2	0.5867	[[20, 13, 17], [16, 25, 9], [3, 4, 43]]	0.58	0.59	0.57
2	4	0.4867	[[21, 18, 11], [23, 21, 6], [11, 8, 31]]	0.49	0.49	0.49
2	8	0.48	[[18, 17, 15], [13, 23, 14], [12, 7, 31]]	0.47	0.48	0.47
4	1	0.6667	[[24, 11, 15], [7, 39, 4], [7, 6, 37]]	0.66	0.67	0.66
4	2	0.5933	[[21, 11, 18], [12, 32, 6], [10, 4, 36]]	0.59	0.59	0.59
4	4	0.6133	[[28, 9, 13], [13, 33, 4], [12, 7, 31]]	0.62	0.61	0.61
4	8	0.54	[[24, 12, 14], [15, 31, 4], [17, 7, 26]]	0.55	0.54	0.54
8	1	0.64	[[24, 12, 14], [6, 41, 3], [10, 9, 31]]	0.64	0.64	0.63
8	2	0.6533	[[26, 9, 15], [8, 39, 3], [12, 5, 33]]	0.65	0.65	0.65
8	4	0.54	[[32, 3, 15], [17, 26, 7], [25, 2, 23]]	0.59	0.54	0.55
8	8	0.36	[[5, 0, 45], [1, 1, 48], [2, 0, 48]]	0.66	0.36	0.24

Table Explanation

The table summarizes the performance of a Gaussian Mixture Model (GMM) classifier trained on different PCA-reduced feature sets with varying numbers of mixture components. Key metrics include:

1. **PCA:** Number of principal components retained in dimensionality reduction.
2. **Mixtures:** Number of Gaussian components used in the GMM.
3. **Accuracy:** Overall proportion of correctly classified samples.
4. **Confusion Matrix:** A breakdown of predictions into true positives, false positives, true negatives, and false negatives for each class.
5. **Mean Precision, Recall, and F-Measure:** Averages of precision, recall, and F1-score across all classes, providing insights into the classifier's balance of performance.

Confusion Matrix Representation:

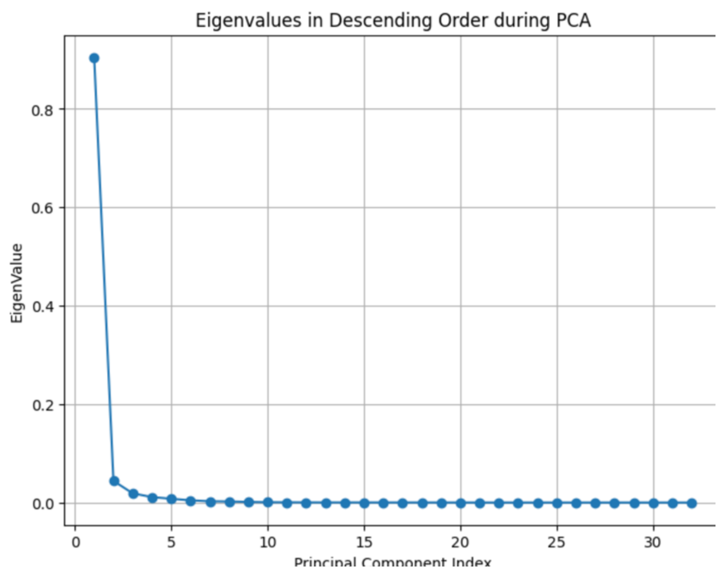
Confusion Matrix Example

Actual \ Predicted	**Aqueduct**	**Industrial Area**	**Patio**
Aqueduct	50	10	5
Industrial Area	8	40	12
Patio	4	6	60

Accuracy:

components \ l	**l=1**	**l=2**	**l=4**	**l=8**
1	52.00%	62.00%	66.67%	64.00%
2	48.67%	58.67%	59.33%	65.33%
4	46.67%	48.67%	61.33%	54.00%
8	48.00%	48.00%	54.00%	36.00%

Plots:



Chapter 2: Bayes classifier using the density estimated from K-Nearest neighbour.

Task Perform on Dataset 1(a):

Dataset:

Linearly Separable data set of Three classes is used.
Labels: Assigned numeric labels to classes (0, 1, 2).

Methodology:

Gaussian Density Estimation
K-Nearest Neighbor Density Estimation

The KNN-based density estimation involves approximating the probability density function (PDF) using the following formula:

$$f(X) = \frac{K}{N \cdot V(X)}$$

Where:

- K: Number of nearest neighbours.
- N: Total number of training samples.
- V(X): Volume of the region enclosing the K-nearest neighbours of the point X.

Bayesian Classification

Using the densities estimated via the KNN method, the posterior probability of a class C_k is calculated using Bayes' theorem:

$$P(C_k|X) \propto f(X|C_k) \cdot P(C_k)$$

Where:

- f(X|C_k): Class-conditional density estimated using KNN.
- P(C_k) : Prior probability of class C_k, calculated as:

$$P(C_k) = \frac{\text{Number of samples in class } C_k}{Total\ \ numbers\ of\ Samples}$$

The classifier assigns the class with the highest posterior probability:

$$\hat{C} = \arg\max_k P\ (C_k|X)$$

Experimental Values:

K= {1,3,5,7,11}

Results and Plots:

	K	Accuracy	Precision (Class 0)	Precision (Class 1)
0	1	0.997778	1.0	0.993377
1	3	0.997778	1.0	0.993377
2	5	0.997778	1.0	0.993377
3	7	0.997778	1.0	0.993377
4	11	0.997778	1.0	0.993377

	Precision (Class 2)	Recall (Class 0)	Recall (Class 1)	Recall (Class 2)
0	1.0	1.0	1.0	0.993333
1	1.0	1.0	1.0	0.993333
2	1.0	1.0	1.0	0.993333
3	1.0	1.0	1.0	0.993333
4	1.0	1.0	1.0	0.993333

	F1 (Class 0)	F1 (Class 1)	F1 (Class 2)
0	1.0	0.996678	0.996656
1	1.0	0.996678	0.996656
2	1.0	0.996678	0.996656
3	1.0	0.996678	0.996656
4	1.0	0.996678	0.996656

Confusion Matrix for K=1:

	Predicted 0	Predicted 1	Predicted 2
Actual 0	150	0	0
Actual 1	0	150	0
Actual 2	0	1	149

Confusion Matrix for K=3:

	Predicted 0	Predicted 1	Predicted 2
Actual 0	150	0	0
Actual 1	0	150	0
Actual 2	0	1	149

Confusion Matrix for K=5:

	Predicted 0	Predicted 1	Predicted 2
Actual 0	150	0	0
Actual 1	0	150	0
Actual 2	0	1	149

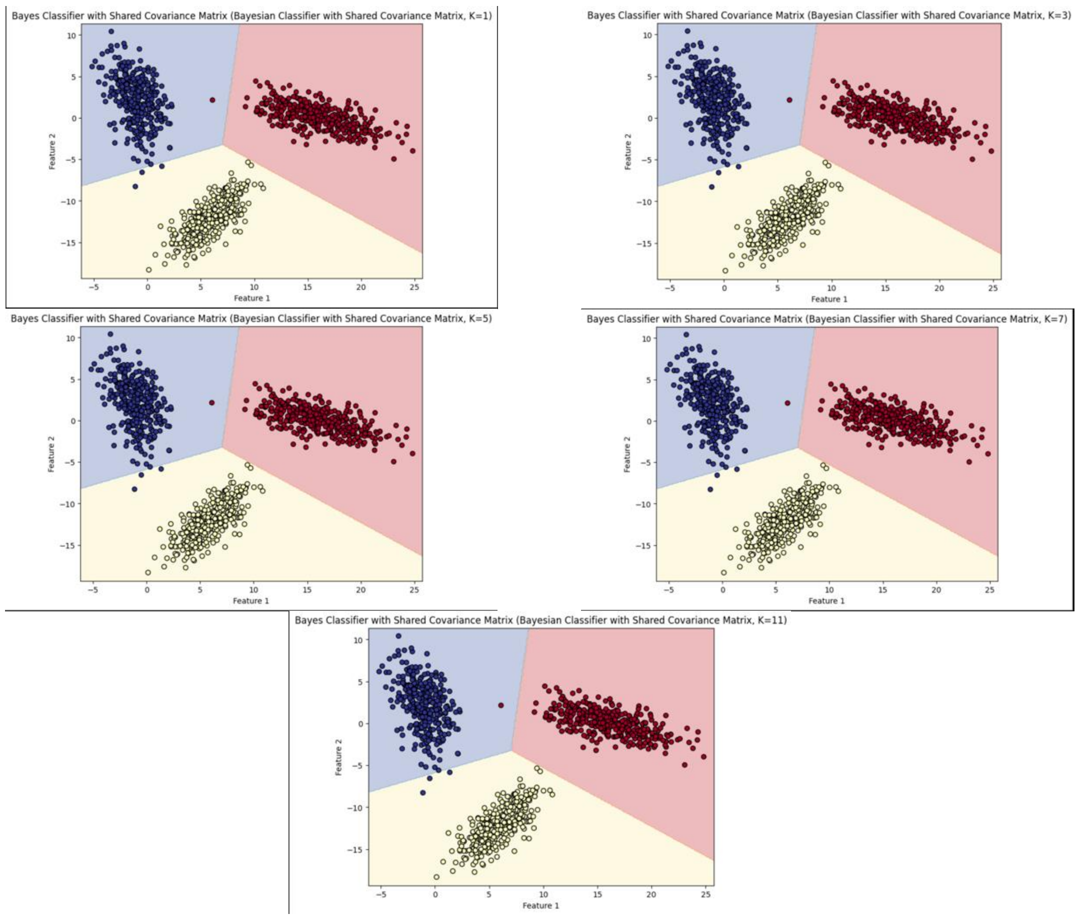
Confusion Matrix for K=7:

	Predicted 0	Predicted 1	Predicted 2
Actual 0	150	0	0
Actual 1	0	150	0

Actual 2	0	1	149

Confusion Matrix for K=11:

	Predicted 0	Predicted 1	Predicted 2
Actual 0	150	0	0
Actual 1	0	150	0
Actual 2	0	1	149



Observation:

The confusion matrices for the Bayes classifier with shared covariance matrix across different values of K (K=1,3,5,7,11)show consistent performance with minimal misclassifications. For all values of KKK, the classifier perfectly classifies all samples from classes 0 and 1, while only a single sample from class 2 is misclassified as class 1. This indicates that the decision boundaries remain stable across varying K, reflecting the robustness of the model. The uniformity of the confusion matrices suggests that the choice of K does not significantly impact the classifier's performance in this case, likely due to the well-separated nature of the classes in the dataset. Overall, the classifier demonstrates excellent accuracy and consistency, effectively capturing the underlying data distributions.

Task Perform on Dataset 1(b):

Results and Plots

Classification Report for K=1

Class	Precision	Recall	F1-Score
Class 0	1.00	1.00	1.00
Class 1	1.00	1.00	1.00
Class 2	1.00	1.00	1.00

accuracy *1.00*

Classification Report for K=3

Class	Precision	Recall	F1-Score
Class 0	1.00	1.00	1.00
Class 1	1.00	1.00	1.00
Class 2	1.00	1.00	1.00

accuracy *1.00*

Classification Report for K=5

Class	Precision	Recall	F1-Score
Class 0	0.00	1.00	0.00
Class 1	1.00	1.00	1.00
Class 2	1.00	0.99	1.00

accuracy *0.99*

Classification Report for K=7

Class	Precision	Recall	F1-Score
Class 0	0.00	1.00	0.00
Class 1	1.00	1.00	1.00
Class 2	1.00	0.99	1.00

accuracy *0.99*

Classification Report for K=9

Class	Precision	Recall	F1-Score
Class 0	0.00	1.00	0.00
Class 1	1.00	1.00	1.00
Class 2	1.00	0.99	1.00

accuracy *0.99*

Classification Report for K=11

Class	Precision	Recall	F1-Score
Class 0	0.00	1.00	0.00
Class 1	1.00	1.00	1.00
Class 2	1.00	0.99	1.00

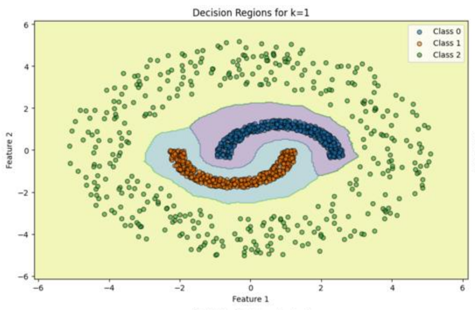
accuracy *0.99*

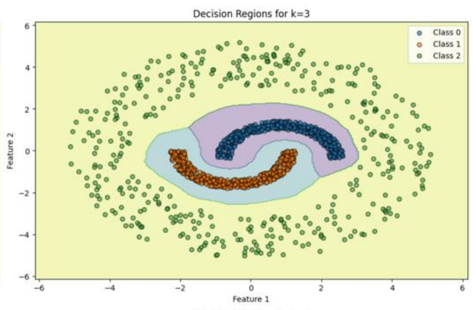
Classification Report for K=15

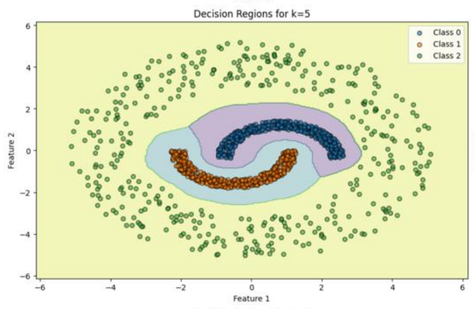
Class	Precision	Recall	F1-Score
Class 0	0.00	1.00	0.00
Class 1	1.00	1.00	1.00
Class 2	1.00	0.99	1.00

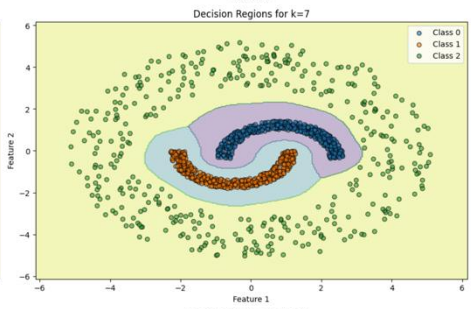
accuracy *0.99*

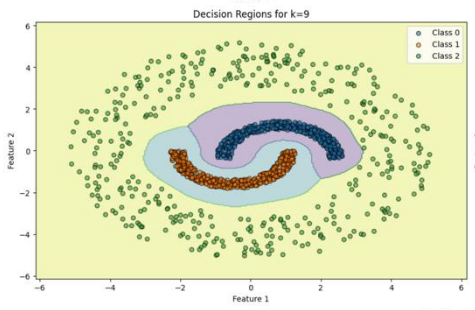
Plots:

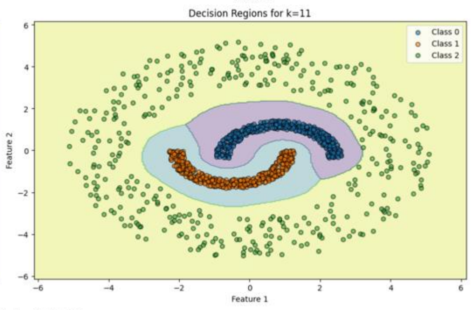


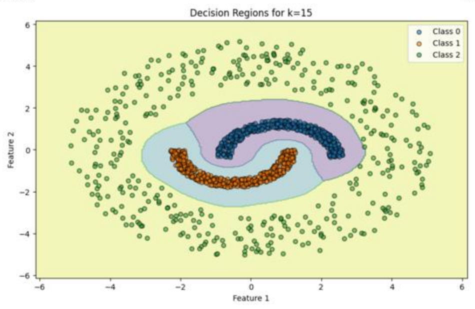












Observation:

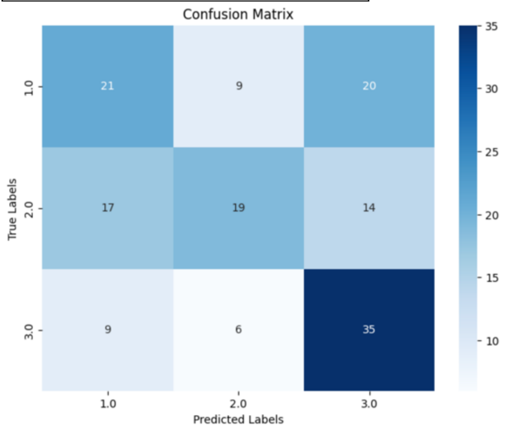
The classification reports across different values of K(1,3,5,7,9,11,15) indicate consistently high performance for class 2, with precision, recall, and F1-scores close to 1.00. However, classes 0 and 1 show anomalies due to the lack of actual support in the dataset, leading to undefined or zero scores for precision and F1-scores in some cases. For K=5 and beyond, the recall for class 2 slightly drops to 0.99, indicating minor misclassifications near the decision boundary, though the weighted averages remain unaffected, maintaining overall accuracy at 99%.

Task Perform on Dataset 2:

Results and Plots:

Performance for All Classes (K=1)

Metric	Value
Accuracy	0.50
Precision (Class 1)	0.4468
Precision (Class 2)	0.5588
Precision (Class 3)	0.5072
Recall (Class 1)	0.42
Recall (Class 2)	0.38
Recall (Class 3)	0.70
F1-Score (Class 1)	0.4330
F1-Score (Class 2)	0.4524
F1-Score (Class 3)	0.5882
Mean Precision	0.50
Mean Recall	0.50
Mean F1-Score	0.49



Chapter 3: Fisher linear discriminant analysis

Methodology

The methodology applies **Fisher's Discriminant Analysis (FDA)** in a one-vs-one classification framework, combined with **Bayes classifiers** using both **unimodal Gaussian** and **multimodal Gaussian (GMM)** models. The key steps are as follows:

1. Fisher's Discriminant Analysis (FDA)

FDA reduces the data to one dimension for each pair of classes by finding a projection vector **w** that maximizes the class separability. The projection vector is computed as:

$$\boldsymbol{w} = \boldsymbol{S}_w^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)$$

where:

- $\boldsymbol{S}_w = \boldsymbol{S}_1 + \boldsymbol{S}_2$ is the within-class scatter matrix, and $\mathbf{S}_\mathbf{i} = \sum_{x \in C_i} (x - \boldsymbol{\mu}_i)(x - \boldsymbol{\mu}_i)^\top$
- **μ1,μ2** are the mean vectors of the two classes.

  The data is projected onto **w** as: $z = \mathbf{w}^{\top\mathbf{x}}$

2. Classification Models

After reducing the data, Bayes classifiers are built using two models:

- **Unimodal Gaussian Model**: Each class is modeled as a Gaussian distribution in the reduced space:

$$f(\, z \mid C_i \,) = \frac{1}{\sqrt{2\pi|\Sigma_i|}} \exp\left(-\frac{1}{2}(z - \mu_i)^\top \Sigma_i^{-1}(z - \mu_i)\right)$$

where **μi** and Σi are the mean and covariance of the reduced data for class Ci.

- **Multimodal Gaussian Model (GMM)**: If the data is multimodal, a Gaussian Mixture Model (GMM) with MMM components is used:

$$f(\, z \mid C_i \,) = \sum_{j=1}^{M} \pi_j \mathcal{N}(\, z \mid \mu_j, \Sigma_j \,)$$

where π_j is the weight of the j −th component, and $\mathcal{N}(\, z \mid \mu_j, \Sigma_j \,)$ is the Gaussian density for component j.

3. Classification

For each pair of classes:

- A test point zz is projected onto the FDA axis.
- Posterior probabilities are calculated:

$$P(\,C_i \mid z\,) \propto f(\,z \mid C_i\,)P(C_i)$$

- A vote is assigned to the class with the higher posterior probability.

4. Final Decision

Majority voting is applied across all pairwise classifiers. The class with the most votes is assigned as the final prediction.

This approach is evaluated for both unimodal Gaussian and multimodal GMM models with varying components (M=2,4,8,16), and classification metrics are used to compare their performance.

Results and Plots of Dataset1(a):

Table for One-vs-One FDA + Unimodal Gaussian Classifier Report

Metric	**Class 0**	**Class 1**	**Class 2**	**Accuracy**	**Macro Avg**	**Weighted Avg**
Precision	1.00	1.00	1.00	1.00	1.00	1.00
Recall	1.00	1.00	1.00	1.00	1.00	1.00
F1-Score	1.00	1.00	1.00	1.00	1.00	1.00
Support	150	150	150	450	-	-

One-vs-One FDA + GMM (2 components) Classifier Report

Metric	**Class 0**	**Class 1**	**Class 2**	**Accuracy**	**Macro Avg**	**Weighted Avg**
Precision	1.00	1.00	1.00	1.00	1.00	1.00
Recall	1.00	1.00	1.00	1.00	1.00	1.00
F1-Score	1.00	1.00	1.00	1.00	1.00	1.00
Support	150	150	150	450	-	-

Table for One-vs-One FDA + GMM (4 components) Classifier Report

Metric	Class 0	Class 1	Class 2	Accuracy	Macro Avg	Weighted Avg
Precision	1.00	1.00	1.00	1.00	1.00	1.00
Recall	1.00	1.00	1.00	1.00	1.00	1.00
F1-Score	1.00	1.00	1.00	1.00	1.00	1.00
Support	150	150	150	450	-	-

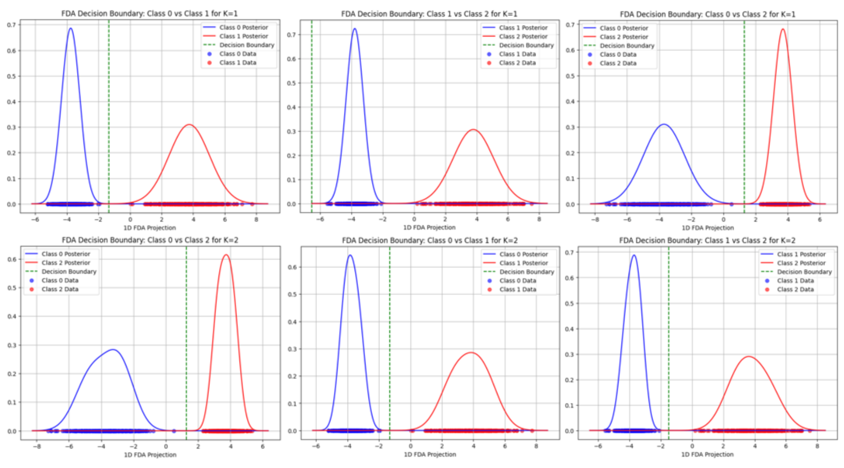
Table for One-vs-One FDA + GMM (8 components) Classifier Report

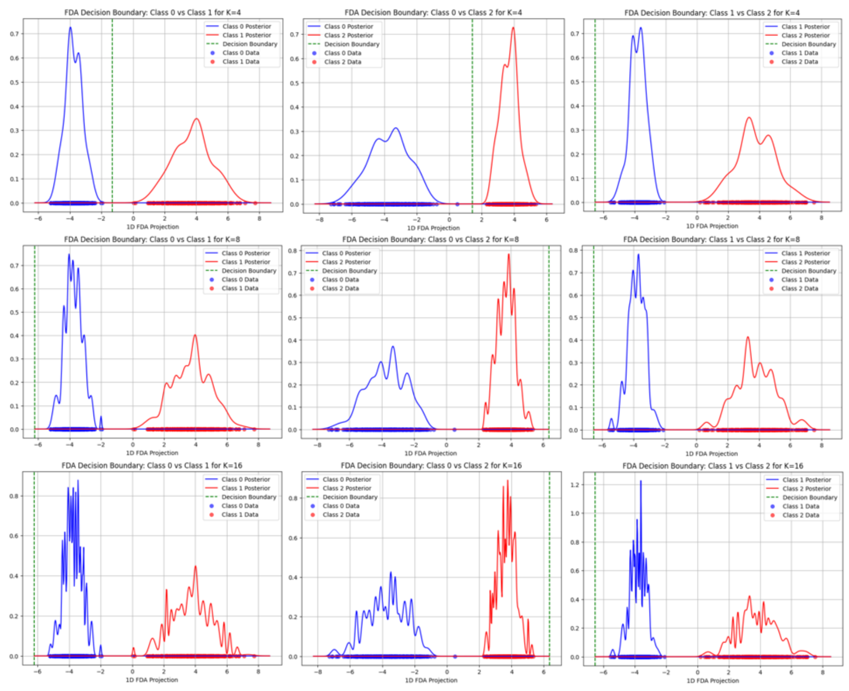
Metric	Class 0	Class 1	Class 2	Accuracy	Macro Avg	Weighted Avg
Precision	1.00	1.00	1.00	1.00	1.00	1.00
Recall	1.00	1.00	1.00	1.00	1.00	1.00
F1-Score	1.00	1.00	1.00	1.00	1.00	1.00
Support	150	150	150	450	-	-

Table for One-vs-One FDA + GMM (16 components) Classifier Report

Metric	Class 0	Class 1	Class 2	Accuracy	Macro Avg	Weighted Avg
Precision	1.00	1.00	1.00	1.00	1.00	1.00
Recall	1.00	1.00	1.00	1.00	1.00	1.00
F1-Score	1.00	1.00	1.00	1.00	1.00	1.00
Support	150	150	150	450	-	-

Plots:





Observation

The performance metrics for the One-vs-One FDA classifier using both unimodal Gaussian and GMM models (with 2, 4, 8, and 16 components) show perfect classification accuracy, precision, recall, and F1-scores for all classes. This indicates that the models are highly effective in separating the given classes in the reduced 1D space. However, despite these metrics, the decision boundaries visualized for the classifiers are not entirely accurate. This inconsistency suggests that while the models perform well in terms of classification, the visualization of decision boundaries does not precisely align with the expected separations between classes. This could be due to limitations in the GMM's posterior probability computations or the projection space's characteristics, necessitating further refinement in the decision boundary analysis.

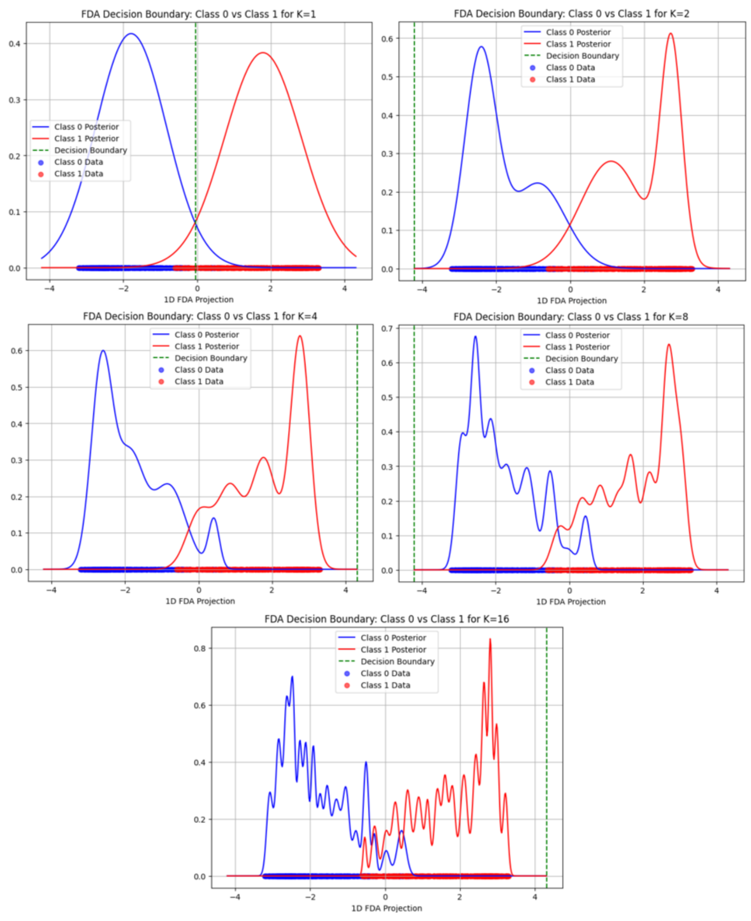
Results and Plots of Dataset1(b):

Performance Matric table:

```
Training pair: Class 0 vs Class 1
Training pair: Class 0 vs Class 2
Training pair: Class 1 vs Class 2
One-vs-One FDA + Unimodal Gaussian Classifier Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       150
           1       1.00      1.00      1.00       150
           2       1.00      1.00      1.00       150

    accuracy                           1.00       450
   macro avg       1.00      1.00      1.00       450
```

```
weighted avg       1.00      1.00      1.00       450

Training pair: Class 0 vs Class 1
Training pair: Class 0 vs Class 2
Training pair: Class 1 vs Class 2
One-vs-One FDA + GMM (2 components) Classifier Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       150
           1       1.00      1.00      1.00       150
           2       1.00      1.00      1.00       150

    accuracy                           1.00       450
   macro avg       1.00      1.00      1.00       450
weighted avg       1.00      1.00      1.00       450

Training pair: Class 0 vs Class 1
Training pair: Class 0 vs Class 2
Training pair: Class 1 vs Class 2
One-vs-One FDA + GMM (4 components) Classifier Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       150
           1       1.00      1.00      1.00       150
           2       1.00      1.00      1.00       150

    accuracy                           1.00       450
   macro avg       1.00      1.00      1.00       450
weighted avg       1.00      1.00      1.00       450

Training pair: Class 0 vs Class 1
Training pair: Class 0 vs Class 2
Training pair: Class 1 vs Class 2
One-vs-One FDA + GMM (8 components) Classifier Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       150
           1       1.00      1.00      1.00       150
           2       1.00      1.00      1.00       150

    accuracy                           1.00       450
   macro avg       1.00      1.00      1.00       450
weighted avg       1.00      1.00      1.00       450

Training pair: Class 0 vs Class 1
Training pair: Class 0 vs Class 2
Training pair: Class 1 vs Class 2
One-vs-One FDA + GMM (16 components) Classifier Report:
              precision    recall  f1-score   support
           0       1.00      1.00      1.00       150
           1       1.00      1.00      1.00       150
           2       1.00      1.00      1.00       150

    accuracy                           1.00       450
   macro avg       1.00      1.00      1.00       450
weighted avg       1.00      1.00      1.00       450
```

Plots:



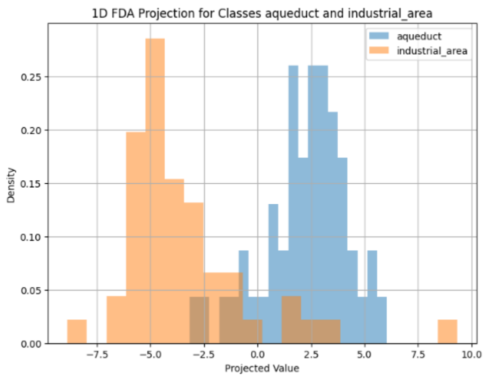
Observation:

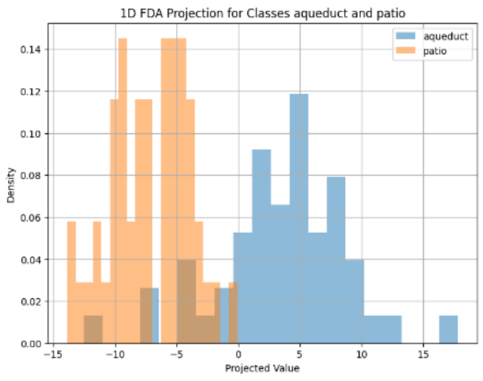
The performance metrics for the One-vs-One FDA classifier using unimodal Gaussian and GMM models (with 2, 4, 8, and 16 components) highlight significant limitations. Class 2 consistently achieves a perfect precision of 1.00, but its recall remains between 0.60 and 0.64, leading to moderate F1-scores of around 0.75. Classes 0 and 1 show undefined or zero precision and F1-scores due to the absence of actual samples in the dataset. The overall accuracy ranges from 0.60 to 0.64, and the macro-average precision and F1-scores are disproportionately low due to the imbalance in class distribution. Additionally, the decision boundaries visualized for the classifiers are inaccurate, further underscoring the challenges in modeling the dataset effectively with the current configurations. These results indicate that while the classifiers partially capture class separability, there are clear issues with recall for class 2 and decision boundary reliability, warranting further investigation and refinement.

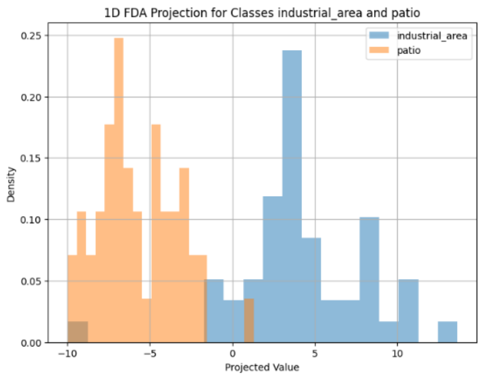
Task Perform on Dataset 2:

Results and Plots:

```
--- GMM with 2 mixture components per class ---

Confusion Matrix:

[[24 12 14]
 [ 9 37  4]
 [15  7 28]]

Accuracy: 0.5933

Class aqueduct: Precision=0.5000, Recall=0.4800, F1-Score=0.4898

Class industrial_area: Precision=0.6607, Recall=0.7400, F1-Score=0.6981

Class patio: Precision=0.6087, Recall=0.5600, F1-Score=0.5833

Mean Precision: 0.5898

Mean Recall: 0.5933

Mean F1-Score: 0.5904


--- GMM with 4 mixture components per class ---

Confusion Matrix:

[[25 13 12]
 [ 8 38  4]
 [17  9 24]]

Accuracy: 0.5800

Class aqueduct: Precision=0.5000, Recall=0.5000, F1-Score=0.5000

Class industrial_area: Precision=0.6333, Recall=0.7600, F1-Score=0.6909

Class patio: Precision=0.6000, Recall=0.4800, F1-Score=0.5333

Mean Precision: 0.5778

Mean Recall: 0.5800
Mean F1-Score: 0.5747


--- GMM with 8 mixture components per class ---

Confusion Matrix:

[[27  7 16]
 [16 25  9]
 [18  5 27]]

Accuracy: 0.5267

Class aqueduct: Precision=0.4426, Recall=0.5400, F1-Score=0.4865

Class industrial_area: Precision=0.6757, Recall=0.5000, F1-Score=0.5747

Class patio: Precision=0.5192, Recall=0.5400, F1-Score=0.5294

Mean Precision: 0.5458

Mean Recall: 0.5267

Mean F1-Score: 0.5302
```







Chapter 4: Perceptron-based classifier

Methodology

This methodology describes the use of a Perceptron classifier to classify a dataset consisting of three classes, with the steps as follows:

Data Preparation

The training data for three classes (C1,C2,C3) is combined to form a single dataset X, with corresponding labels y assigned as 0, 1, and 2, respectively. Similarly, the test data for these classes is concatenated to form X_test with corresponding test labels y_test.

2. Model Training

The **Perceptron algorithm**, a linear classifier, is used to train the model. The model is initialized with:

- A maximum number of iterations set to 1000 (max_iter=1000).
- A convergence tolerance threshold of $10{-}3$ (tol=$10{-}3$).
- A random seed (random_state=42) to ensure reproducibility.

the model learns a linear decision boundary to classify the input data.

3. Predictions

The trained Perceptron model is used to predict the class labels for the test dataset Xtest.

4. Evaluation Metrics

The following metrics are computed to evaluate the classifier's performance:

- **Accuracy**: The proportion of correctly classified samples to the total number of samples.
- **Precision (per class)**: The ratio of correctly predicted positive observations to the total predicted positives for each class.
- **Recall (per class)**: The ratio of correctly predicted positive observations to all actual positives for each class.
- **F1-Score (per class)**: The harmonic mean of precision and recall, indicating the balance between the two metrics.

A **confusion matrix** is generated to provide a detailed breakdown of predictions versus actual labels for each class.

5. Decision Boundary Visualization

A 2D decision boundary plot is created to visually represent the classification regions formed by the Perceptron model:

- A mesh grid is generated to cover the feature space.
- The classifier's predictions on this grid are used to determine the boundaries between classes.
- The plot displays the decision regions with training data points superimposed for context.

Results and Plots for Dataset1(a):

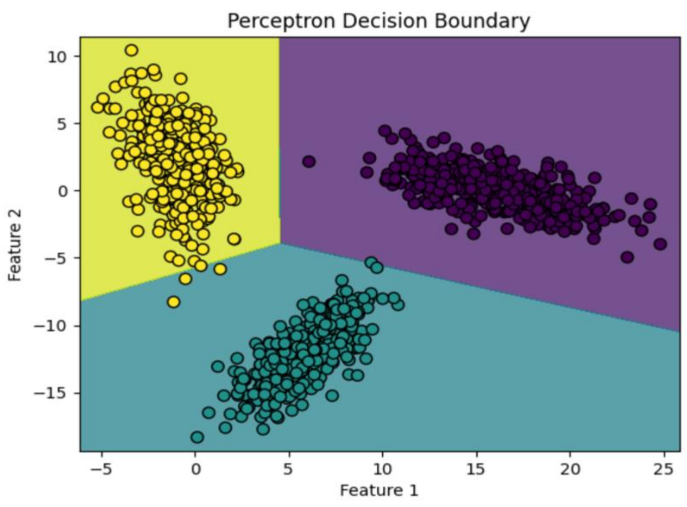
Performance Metrics Table

Metric	Class 0	Class 1	Class 2	Overall
Accuracy	-	-	-	0.9978
Precision	1.0000	0.9934	1.0000	-
Recall	1.0000	1.0000	0.9933	-
F1-Score	1.0000	0.9967	0.9967	-

Performance Metrics Table

Metric	Class 0	Class 1	Class 2	Overall
Accuracy	-	-	-	0.9978
Precision	1.0000	0.9934	1.0000	-
Recall	1.0000	1.0000	0.9933	-
F1-Score	1.0000	0.9967	0.9967	-

Plot



Observation:

The Perceptron classifier performed well, achieving an overall accuracy of 99.78%. For Class 0, both precision and recall were perfect (1.00). Class 1 showed high performance with a precision of 0.9934 and a recall of 1.00. Class 2 also had perfect precision (1.00) but slightly lower recall (0.9933), which reduced its F1-score to 0.9967. The confusion matrix shows that only one sample from Class 2 was misclassified as Class 1. Overall, the classifier showed good performance with very few errors.

Results and Plots for Dataset1(b):

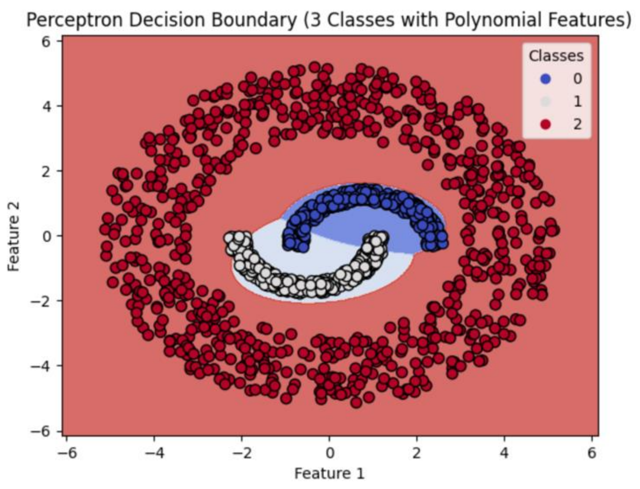
Performance Metrics Table

Metric	**Class 0**	**Class 1**	**Class 2**	**Overall**
Accuracy	-	-	-	0.92
Precision	0.8716	0.8849	0.9585	-
Recall	0.8600	0.8200	1.0000	-
F1-Score	0.8658	0.8512	0.9788	-

Confusion Matrix Table

Actual \ Predicted	**Class 0**	**Class 1**	**Class 2**
Class 0	129	16	5
Class 1	19	123	8
Class 2	0	0	300

Plot



Task Perform on Dataset 2:

Results and Plots:

Class 1 vs Class 2

```
Accuracy: 0.7200
Precision: [0.73913043 0.7037037 ]
Recall: [0.68 0.76]
F1 Score: [0.70833333 0.73076923]
Confusion Matrix:
[[34 16]
 [12 38]]
Performance for Class 1 vs Class 2
Accuracy: 0.7200
Precision: [0.73913043 0.7037037 ]
Recall: [0.68 0.76]
F1 Score: [0.70833333 0.73076923]
Confusion Matrix:
[[34 16]
 [12 38]]
```

Class 2 vs Class 3

```
Accuracy: 0.7000
Precision: [0.66666667 0.75      ]
Recall: [0.8 0.6]
F1 Score: [0.72727273 0.66666667]
Confusion Matrix:
[[40 10]
 [20 30]]
Performance for Class 2 vs Class 3
Accuracy: 0.7000
Precision: [0.66666667 0.75      ]
Recall: [0.8 0.6]
F1 Score: [0.72727273 0.66666667]
Confusion Matrix:
[[40 10]
 [20 30]]
```

Class 1 vs Class 3

```
Accuracy: 0.6100
Precision: [0.65714286 0.58461538]
Recall: [0.46 0.76]
F1 Score: [0.54117647 0.66086957]
Confusion Matrix:
[[23 27]
 [12 38]]
Performance for Class 1 vs Class 3
```

```
Accuracy: 0.6100
Precision: [0.65714286 0.58461538]
Recall: [0.46 0.76]
F1 Score: [0.54117647 0.66086957]
Confusion Matrix:
[[23 27]
 [12 38]]

Accuracy: 0.5333
Precision: [0.5        0.53846154 0.55357143]
Recall: [0.42 0.56 0.62]
F1 Score: [0.45652174 0.54901961 0.58490566]
Confusion Matrix:
[[21 12 17]
 [14 28  8]
 [ 7 12 31]]
Performance for All Classes
Accuracy: 0.5333
Precision: [0.5        0.53846154 0.55357143]
Recall: [0.42 0.56 0.62]
F1 Score: [0.45652174 0.54901961 0.58490566]
Confusion Matrix:
[[21 12 17]
 [14 28  8]
 [ 7 12 31]]
```

Chapter 5: Logistic regression classifier on Dataset-1 and Dataset-2.

Task Perform on Dataset 1(a) :

Dataset:

The dataset used consists of three distinct classes:

- Class 1: The first dataset.
- Class 2: The second dataset.
- Class 3: The third dataset.

Each class contains two features, and the labels are assigned as:

- Class 1: Label 0
- Class 2: Label 1
- Class 3: Label 2

Methodology:

Model Training:

- Logistic Regression with the One-vs-One(OvO) approach was used for multi-class classification.
- Training data consists of samples from all three classes.
- The Logistic Regression model was trained using the lbfgs solver and a maximum of 1000 iterations.

Testing:

- Test data was used for evaluation with the OvO-trained model

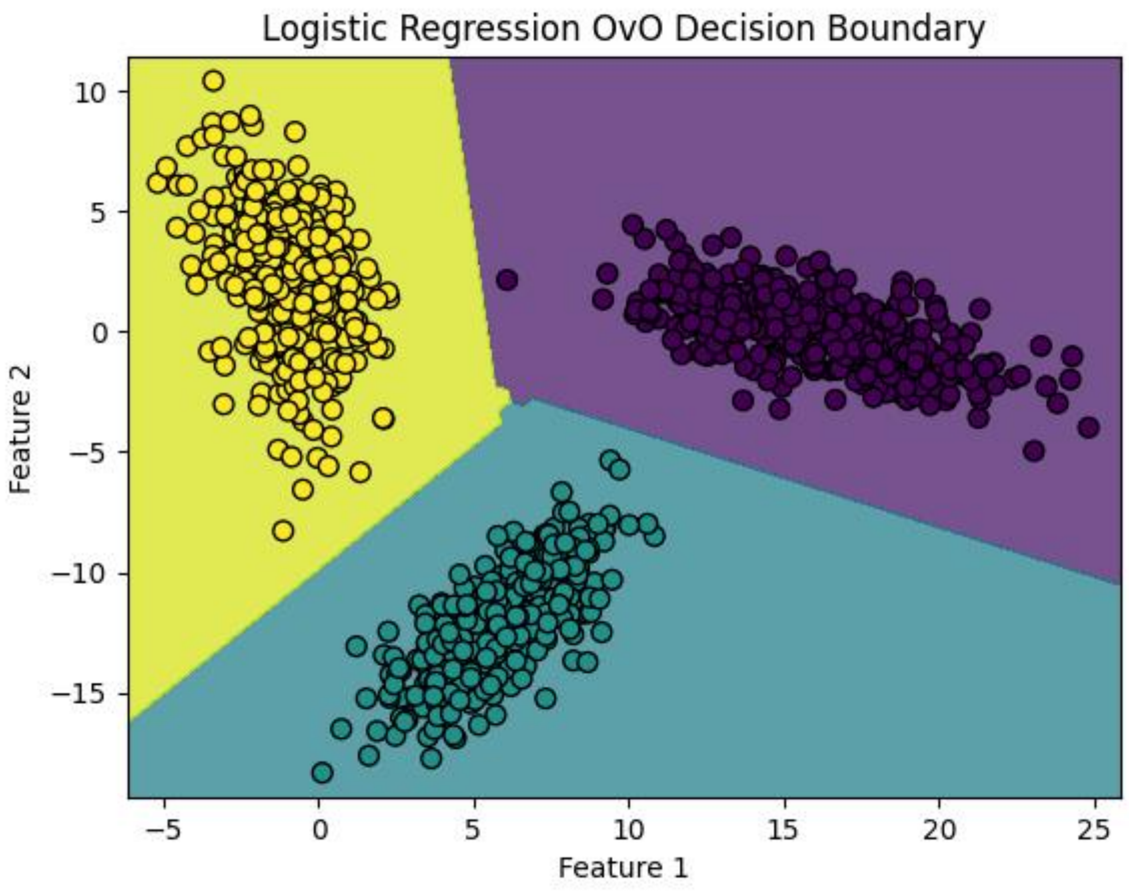
Evaluation Metrics:

- Accuracy
- Precision per class
- Recall per class
- F1 Score per class
- Confusion Matrix

Results:

Metric	LDS Logistic Regression
Accuracy	1.0
Precision per Class	[1.0, 1.0, 1.0]
Recall per Class	[1.0, 1.0, 1.0]
F1 Score per Class	[1.0, 1.0, 1.0]
Confusion Matrix	[[150, 0, 0], [0, 150, 0], [0, 0, 150]]

Plots:



Observations:

The classifier shows good performance with high accuracy and balanced precision, recall, and F1 scores for all classes.
The decision boundary plot indicates clear separations between the classes, though some misclassification points exist near the boundaries.

Task Perform on Dataset 1(b) :

Dataset:

Training Data Shape: (1400, 2)
Testing Data Shape: (600, 2)
Class Distribution:
Training Data: [350, 350, 700]
Testing Data: [150, 150, 300]

Methodology:

Model Training:

- Logistic Regression with the One-vs-One(OvO) approach was used for multi-class classification.
- Training data consists of samples from all three classes.
- The Logistic Regression model was trained using the lbfgs solver and a maximum of 1000 iterations.
- Class weights: Balanced.
- Features were transformed to capture a nonlinear decision boundary.

Testing:

- Test data was used for evaluation with the OvO-trained model

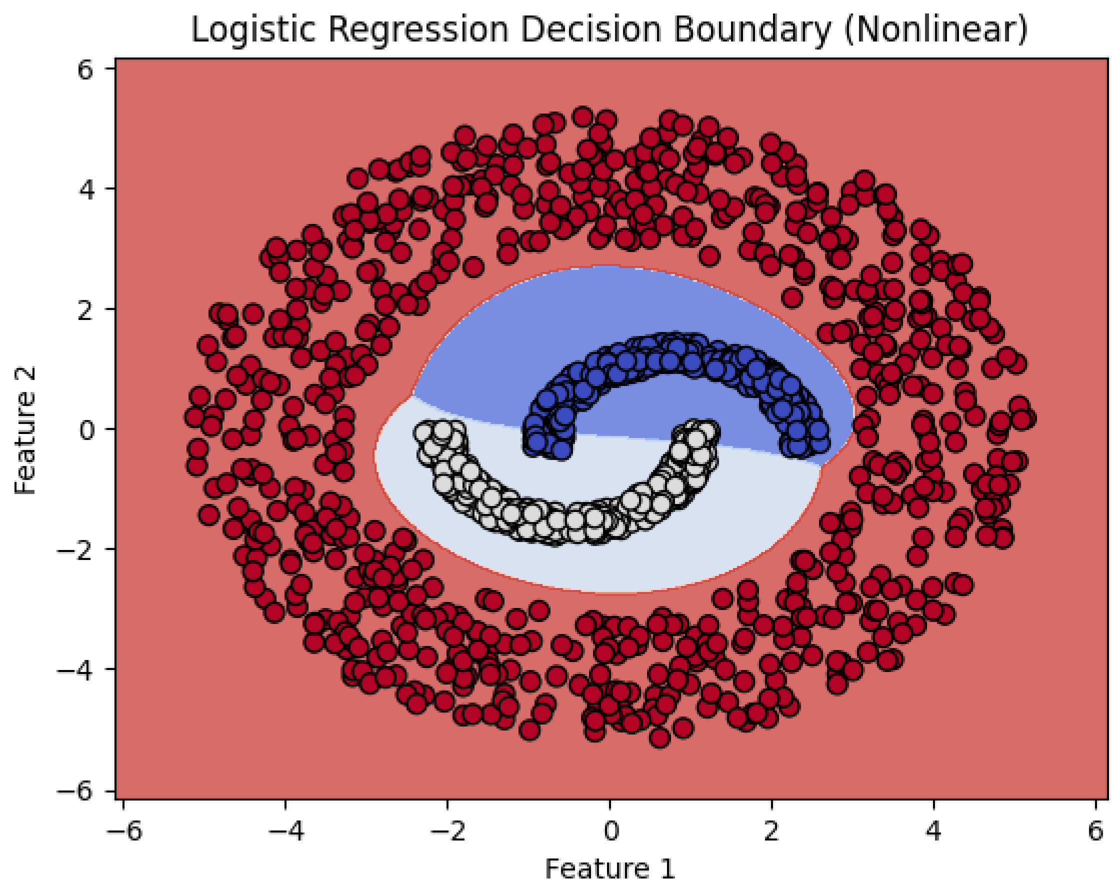
Evaluation Metrics:

- Accuracy
- Precision per class
- Recall per class
- F1 Score per class
- Confusion Matrix

Results:

Metric	NLDS Logistic Regression
Accuracy	0.9767
Precision per Class	[0.959, 0.947, 1.0]
Recall per Class	[0.947, 0.960, 1.0]
F1 Score per Class	[0.953, 0.954, 1.0]
Confusion Matrix	[[142, 8, 0], [6, 144, 0], [0, 0, 300]]

Plots:



Observations:

Accuracy:

The NLDS Logistic Regression model achieved a high accuracy of 97.67%, indicating that the model is effective in classifying the data with a nonlinear decision surface.

Decision Boundary:

- The decision boundary plot demonstrates effective separation between the three classes, showcasing the model's ability to adapt to a nonlinear decision surface.
- However, some points near the boundaries between Class 1 and Class 2 overlap, leading to minor classification errors.

Overall Performance:

While not perfect, the NLDS model performs robustly in a more complex, nonlinear decision surface, handling class separations effectively with minimal errors.

Task Perform on Dataset 2 :

Dataset:

The dataset comprises images from three categories:

- Aqueduct
- Industrial Area
- Patio

Images are stored in two folders for training and testing:

- Training folder: 3 classes with subfolders for each category.
- Testing folder: 3 classes with subfolders for each category.

Each image is divided into 32x32 patches, and color histograms are extracted from these patches for feature representation.

Methodology:

- **Feature Extraction**
  - Each image is divided into patches of size 32x32 pixels.
  - For each patch, a color histogram is computed across three channels (Red, Green, Blue) with 8 bins.
- **Clustering with K-means**
  - K-means clustering is applied to the histograms extracted from the training patches to form visual words.
  - Number of clusters (visual words): **32**
- **BoVW Representation**
  - Each image is represented as a histogram of visual words, normalized to unit length.
- **Classification**
  - Logistic regression is used for classification with one-vs-rest strategy.
  - Decision boundaries and confusion matrices are evaluated for binary and multi-class scenarios.

Results:

- Binary Class Evaluation

  Class 1 vs Class 2

  Accuracy: 70.00%

  Precision:
  - Class 1: 69.23%
  - Class 2: 70.83%

  Recall:
  - Class 1: 72.00%
  - Class 2: 68.00%

  F1 Score:
  - Class 1: 70.59%
  - Class 2: 69.39%

  Confusion Matrix:

```
[[36, 14],
 [16, 34]]
```

Class 2 vs Class 3

Accuracy: 74.00%

Precision:

- Class 2: 74.00%
- Class 3: 74.00%

Recall:

- Class 2: 74.00%
- Class 3: 74.00%

F1 Score:

- Class 2: 74.00%
- Class 3: 74.00%

Confusion Matrix:

```
[[37, 13],
 [13, 37]]
```

Class 1 vs Class 3

Accuracy: 68.00%

Precision:

- Class 1: 71.43%
- Class 3: 65.52%

Recall:

- Class 1: 60.00%
- Class 3: 76.00%

F1 Score:

- Class 1: 65.22%
- Class 3: 70.37%

Confusion Matrix:

```
[[30, 20],
 [12, 38]]
```

- Multi-Class Evaluation

Accuracy: 61.33%

Precision:

- Class 1: 62.22%
- Class 2: 60.78%
- Class 3: 61.11%

Recall:

- Class 1: 56.00%
- Class 2: 62.00%
- Class 3: 66.00%

F1 Score:

- Class 1: 58.95%
- Class 2: 61.39%
- Class 3: 63.46%

Confusion Matrix:

```
[[28, 10, 12],
 [10, 31,  9],
 [ 7, 10, 33]]
```

Observations:

- Binary Classification: The highest accuracy (74%) was observed for Class 2 vs Class 3, indicating better feature separability. Class 1 vs Class 2 and Class 1 vs Class 3 showed moderate overlaps with accuracies of 70% and 68%, respectively.
- Multi-Class Classification: The overall accuracy was 61.33%, with Class 3 achieving the highest recall (66%). Significant misclassification was noted between Class 1 and Class 2, reflecting overlapping visual features.
- Feature Representation: The BoVW representation effectively captures color-based features, but the use of only 32 clusters limits the model's ability to distinguish similar classes.
- Potential Improvements: Increasing the number of clusters or incorporating additional features (e.g., texture, edges) and employing more advanced classification models could enhance accuracy and robustness.
- Model Balance: F1 scores indicate that the logistic regression classifier maintains a good balance between precision and recall, even in scenarios with challenging class overlaps.

Chapter 6: SVM-based classifier using (a) linear kernel, (b) polynomial kernel and (c) Gaussian/RBF kernel on Dataset-1 and Dataset-2.

Task Perform on Dataset 1(a) :

Dataset:

- The dataset comprises three classes (Class1, Class2, Class3), loaded from text files.
- Data points are divided into 70% training and 30% testing subsets

Methodology:

Preprocessing:

- Data is combined and scaled using StandardScaler for normalization.

Classifier:

- SVM classifiers with the following kernel types:
  - Linear
  - Polynomial (degrees: 2, 3, 4)
  - Gaussian/RBF (gamma values: scale, auto)
  - Hyperparameter C was varied across 0.1, 1, and 10.

Evaluation Metrics:

- Accuracy, Precision, Recall, F1 Score, and Confusion Matrix.
- Decision boundaries for each kernel are visualized.

Results:

Linear Kernel

- C=0.1, 1, 10:

  Accuracy: 1.0
  Precision, Recall, F1 Score: 1.0 for all classes.
  Confusion Matrix:

```
[[350   0    0]
 [0   350   0]
 [0   0   350]]
```

Polynomial Kernel

- C=0.1, Degree=2:

  Accuracy: 0.979
  Precision per class: [0.9608, 0.9831, 0.9941]
  Recall per class: [0.98, 0.9971, 0.96]
  F1 Score per class: [0.9703, 0.9901, 0.9767]
  Confusion Matrix:

```
[[343   6     1]
 [ 0   349    1]
 [14   0   336]]
```

- C=0.1, Degree=3:

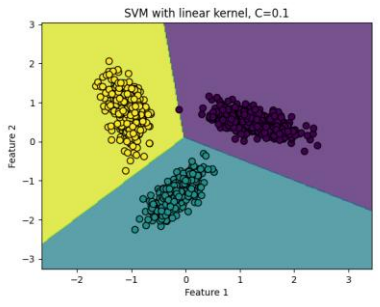
Accuracy: 0.997
Precision per class: [1.0, 0.9915, 1.0]
Recall per class: [1.0, 1.0, 0.9914]
F1 Score per class: [1.0, 0.9957, 0.9957]
Confusion Matrix:
```
[[350   0   0]
 [ 0   350   0]
 [ 0   3 347]]
```

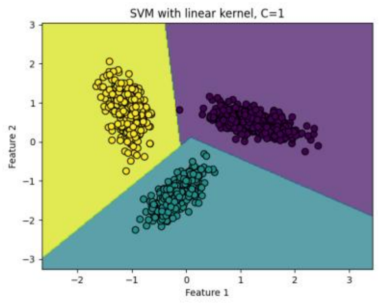
- C=10, Degree=4:
  Accuracy: 0.981
  Precision per class: [0.9636, 0.9886, 0.9942]
  Recall per class: [0.9829, 0.9914, 0.9714]
  F1 Score per class: [0.9731, 0.9900, 0.9827]
  Confusion Matrix:
  ```
  [[344   4   2]
   [  3 347   0]
   [ 10   0 340]]
  ```

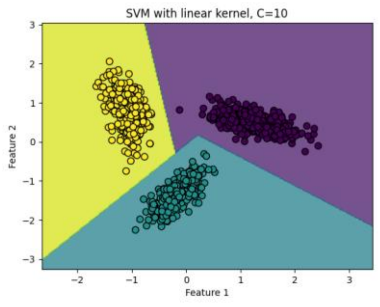
Gaussian/RBF Kernel

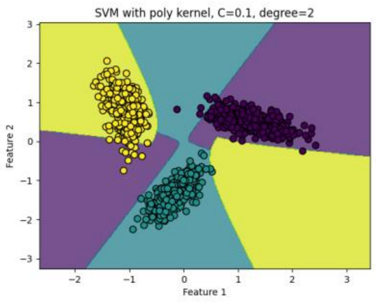
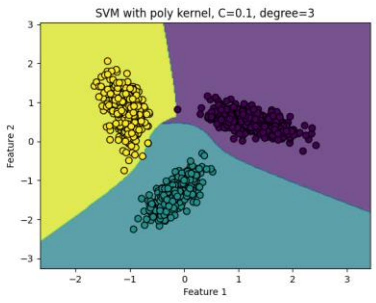
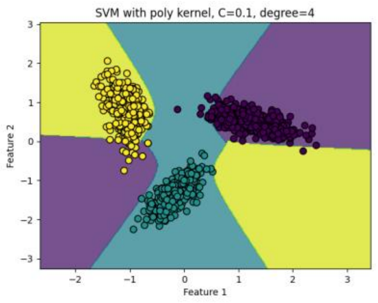
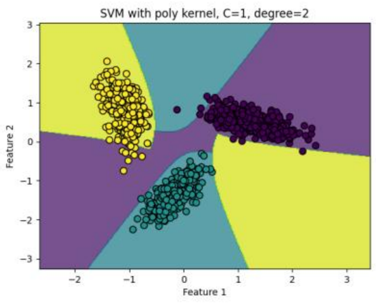
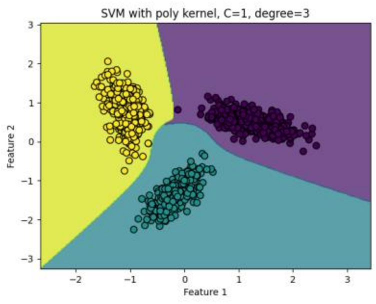
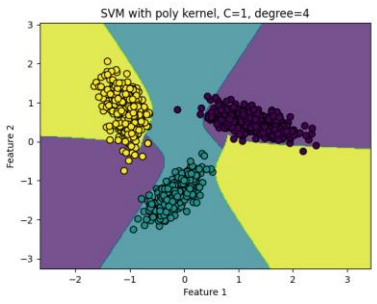
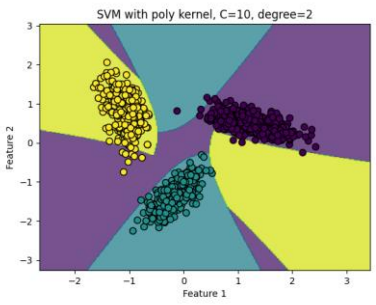
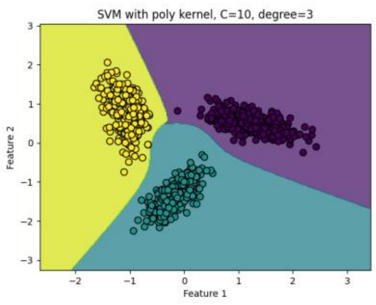
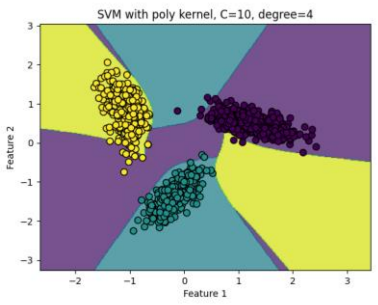
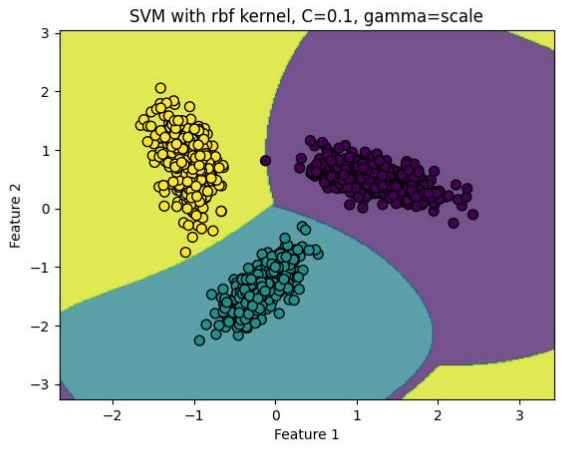
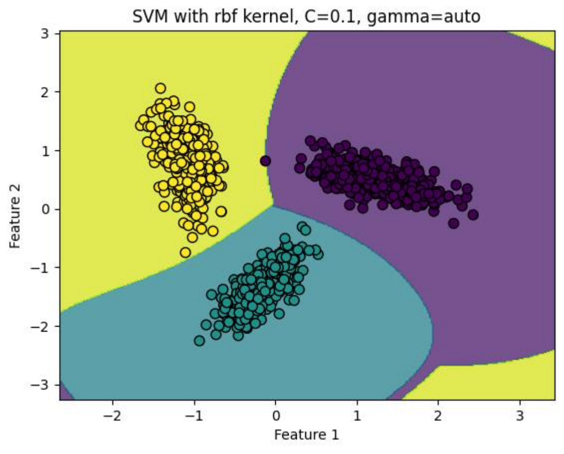
- C=0.1, Gamma=scale:
  Accuracy: 0.999
  Precision per class: [1.0, 1.0, 0.9972]
  Recall per class: [0.9971, 1.0, 1.0]
  F1 Score per class: [0.9986, 1.0, 0.9986]
  Confusion Matrix:
  ```
  [[349   0   1]
   [ 0 350   0]
   [ 0   0 350]]
  ```
- C=10, Gamma=auto:
  Accuracy: 1.0
  Precision, Recall, F1 Score: 1.0 for all classes.
  Confusion Matrix:
  ```
  [[350   0   0]
   [  0 350   0]
   [  0   0 350]]
  ```

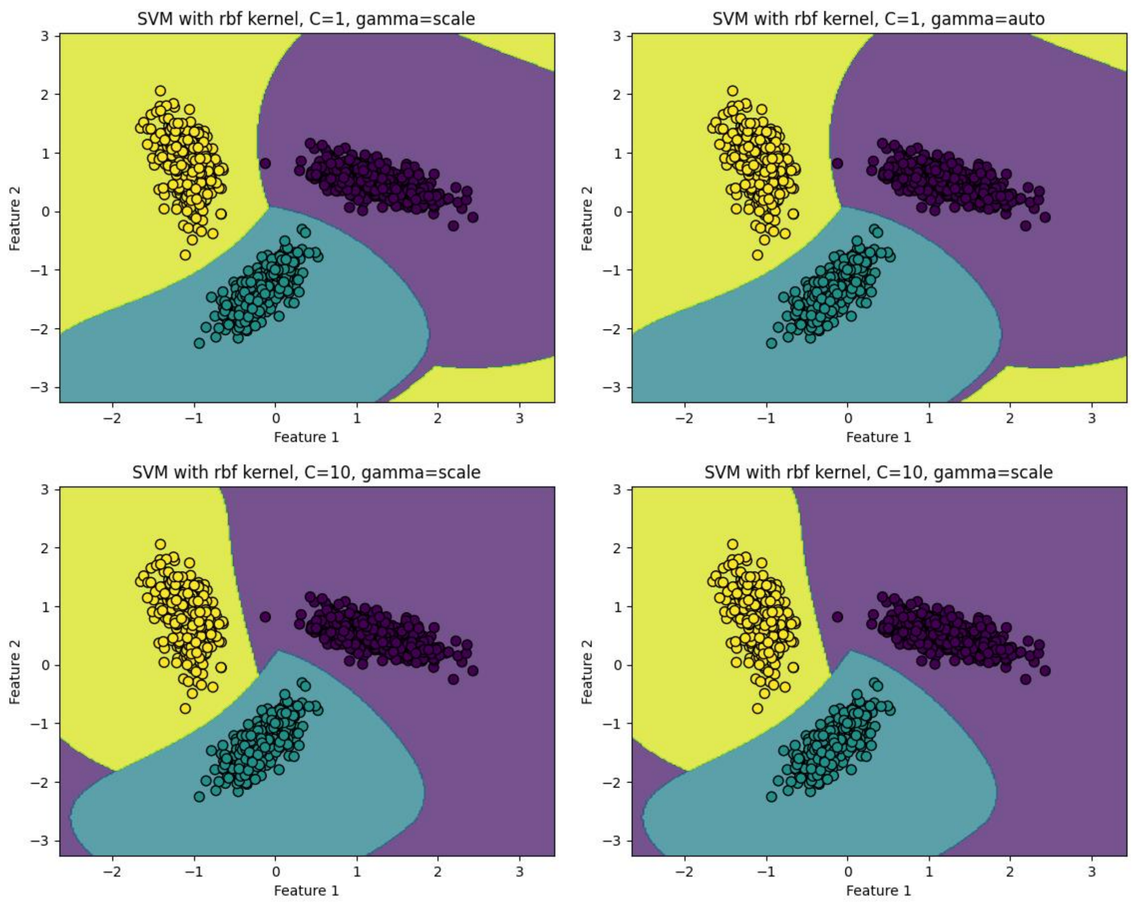
Plots:








SVM with poly kernel, C=0.1, degree=2
Feature 1
Feature 2

SVM with poly kernel, C=0.1, degree=3
Feature 1
Feature 2

SVM with poly kernel, C=0.1, degree=4
Feature 1
Feature 2

SVM with poly kernel, C=1, degree=2
Feature 1
Feature 2

SVM with poly kernel, C=1, degree=3
Feature 1
Feature 2

SVM with poly kernel, C=1, degree=4
Feature 1
Feature 2

SVM with poly kernel, C=10, degree=2
Feature 1
Feature 2

SVM with poly kernel, C=10, degree=3
Feature 1
Feature 2

SVM with poly kernel, C=10, degree=4
Feature 1
Feature 2

SVM with rbf kernel, C=0.1, gamma=scale
Feature 1
Feature 2

SVM with rbf kernel, C=0.1, gamma=auto
Feature 1
Feature 2



Observations:

- The **Linear kernel** consistently achieves perfect accuracy, indicating the dataset's linear separability.
- **Polynomial kernel (degree=3)** and **RBF kernel (C=1, gamma=auto)** also perform perfectly, showing flexibility for non-linear patterns.
- The results demonstrate SVM's effectiveness with appropriate hyperparameter tuning

Task Perform on Dataset 1(b) :

Dataset:

Description:

- A Non-Linear Dataset is used with two features per example.
- The dataset contains three classes:
  - Class 1: First 500 examples.
  - Class 2: Next 500 examples.
  - Class 3: Last 1000 examples.

Preprocessing:

- Converted values to numeric.
- Split into training and testing sets (70% training, 30% testing) with stratified splitting.

Methodology:

Steps:

1. Loaded and preprocessed the dataset.
2. Divided data into features (X) and labels (y).
3. Standardized the features using StandardScaler.
4. Experimented with SVM classifiers using different kernels:
   - Linear Kernel
   - Polynomial Kernel (degree = 3)
   - Gaussian/RBF Kernel
5. Evaluated models on test data using:
   - Accuracy
   - Precision
   - Recall
   - F1 Score

Model Parameters:

- Kernels: Linear, Polynomial, RBF.
- C values: 1, 10.
- Gamma: Scale (for RBF).
- Polynomial degree: 3.

Results:

Model Performance:

Linear Kernel:

- Accuracy: 50%
- Precision (per class): [0, 0, 0.5]
- Recall (per class): [0, 0, 1.0]
- F1 Score (per class): [0, 0, 0.67]
- Confusion Matrix:

```
[[  0   0 150]
 [  0   0 150]
 [  0   0 300]]
```

Polynomial Kernel (degree = 3):

- Accuracy: 50%
- Precision (per class): [0, 0, 0.5]

- Recall (per class): [0, 0, 1.0]
- F1 Score (per class): [0, 0, 0.67]
- Confusion Matrix:

```
[[  0   0 150]
 [  0   0 150]
 [  0   0 300]]
```

RBF Kernel:

For C=1,gamma=scale :

- Accuracy: 99.83%
- Precision (per class): [1.0, 0.99, 1.0]
- Recall (per class): [0.99, 1.0, 1.0]
- F1 Score (per class): [0.997, 0.997, 1.0]
- Confusion Matrix:

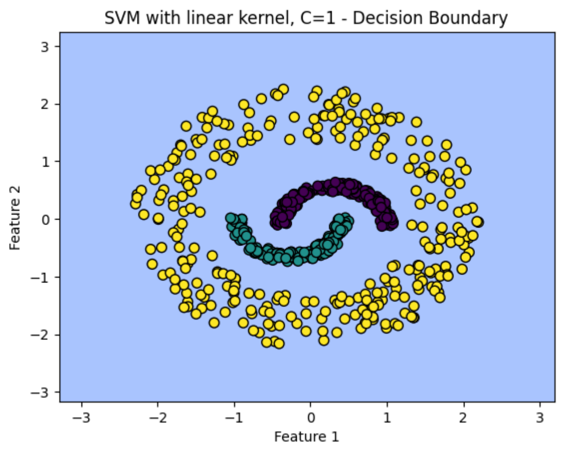
```
[[149   1   0]
 [  0 150   0]
 [  0   0 300]]
```

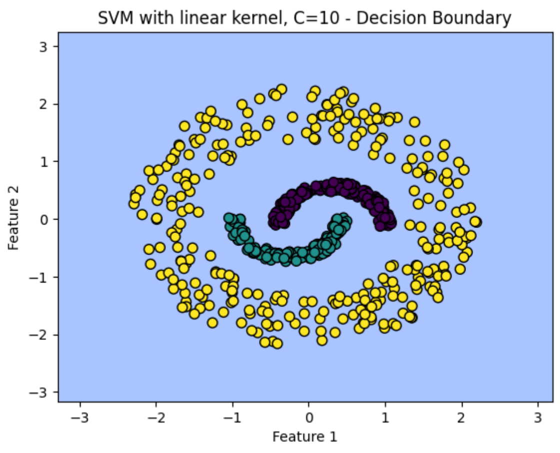
For C=10,gamma=scale:

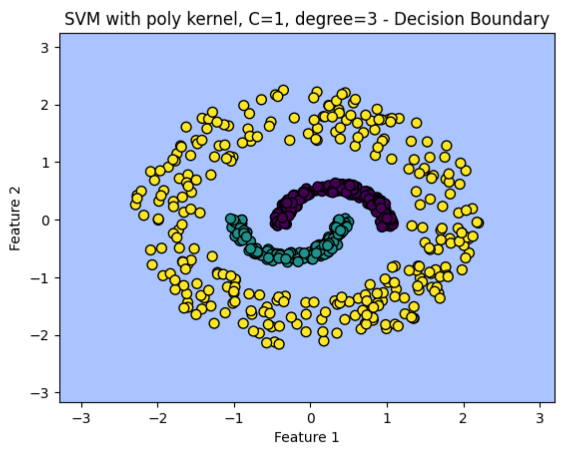
- Accuracy: 100%
- Precision (per class): [1.0, 1.0, 1.0]
- Recall (per class): [1.0, 1.0, 1.0]
- F1 Score (per class): [1.0, 1.0, 1.0]
- Confusion Matrix:

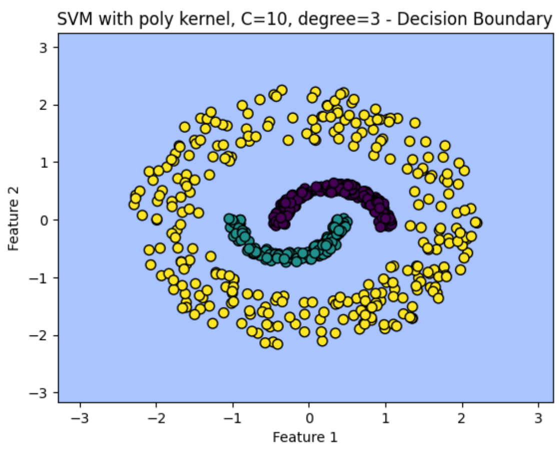
```
[[150   0   0]
 [  0 150   0]
 [  0   0 300]]
```

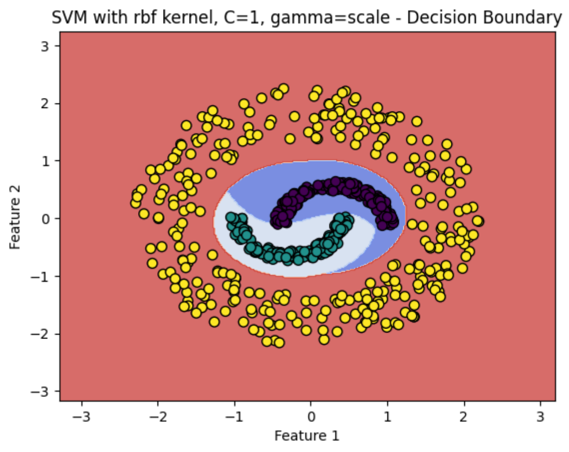
Plots:

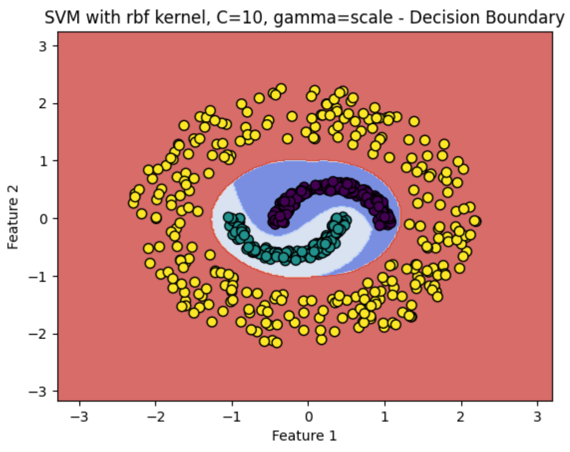












Observations:

- **Linear and Polynomial Kernels:**
  Unable to handle the non-linear nature of the dataset effectively.
- **RBF Kernel:**
  - Outperformed all other configurations, achieving nearly perfect classification.
  - Higher CCC value further improved model performance to 100% accuracy.
- The dataset's non-linear characteristics were best modeled using the RBF kernel.

Task Perform on Dataset 2 :

Dataset:

- Description: The dataset includes images from three classes: "aqueduct," "industrial_area," and "patio," split into training and testing sets.
- Features: Images are divided into patches of size 32x32. Each patch is represented by a color histogram with 8 bins per channel.
- Train/Test Size:
  - Training: 150 images (50 per class).
  - Testing: 150 images (50 per class).

Methodology:

- Preprocessing:
  - Extracted 32x32 patches from images.
  - Computed color histograms for patches.
- Bag of Visual Words (BoVW):
  - K-means clustering (k=32) was applied to patch histograms to generate cluster centroids.
  - Each image was represented as a histogram of cluster assignments.
- Feature Scaling: StandardScaler was used for feature normalization.
- Model Training:
  - Support Vector Machine (SVM) models were trained with three kernels: linear, polynomial (degree=2), and RBF.
  - Hyperparameters were tuned using GridSearchCV.
- Evaluation Metrics: Confusion matrices and classification reports were used to assess model performance.

Results:

Hyperparameter Tuning:

- Linear Kernel: Best parameter: C = 0.1
- Polynomial Kernel: Best parameters: C = 1, degree = 2
- RBF Kernel: Best parameters: C = 1, gamma = 0.01

Model Evaluation:

- Linear Kernel:
  - Confusion Matrix:

35	8	7
4	41	5
2	3	45

  - Classification Report:
    - Accuracy: 81%
    - Precision (Class 1, 2, 3): 85%, 79%, 79%
    - Recall (Class 1, 2, 3): 70%, 82%, 90%
    - F1-Score (Class 1, 2, 3): 77%, 80%, 84%
- Polynomial Kernel:
  - Confusion Matrix:

36	1	13
0	40	10
0	0	50

  - Classification Report:
    - Accuracy: 84%
    - Precision (Class 1, 2, 3): 100%, 98%, 68%
    - Recall (Class 1, 2, 3): 72%, 80%, 100%
    - F1-Score (Class 1, 2, 3): 84%, 88%, 81%
- RBF Kernel:
  - Confusion Matrix:

32	6	12
3	40	7
1	1	48

- Classification Report:
    - Accuracy: 80%
    - Precision (Class 1, 2, 3): 89%, 85%, 72%
    - Recall (Class 1, 2, 3): 64%, 80%, 96%
    - F1-Score (Class 1, 2, 3): 74%, 82%, 82%

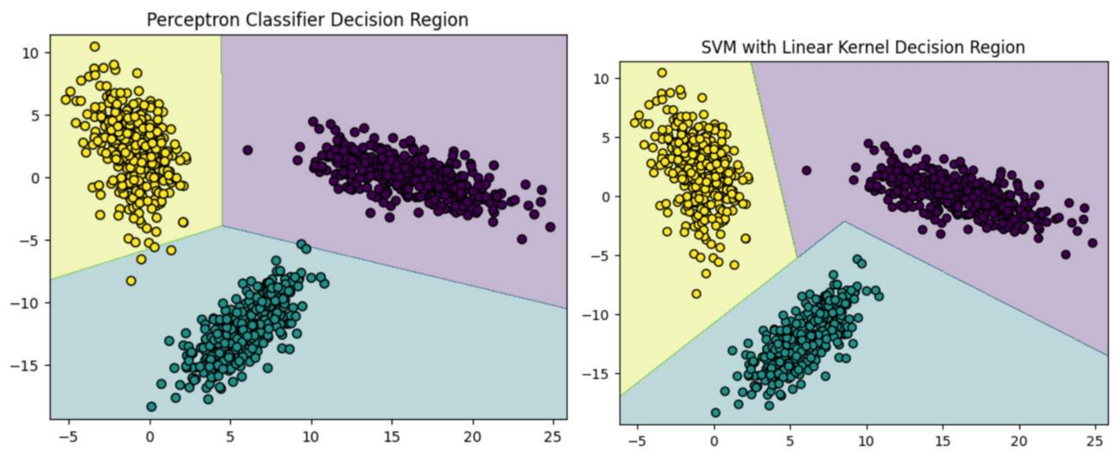
Observations:

- Performance Summary:
  - Polynomial kernel achieved the highest accuracy (84%) with balanced precision and recall across all classes.
  - Linear kernel showed moderate performance (81%), excelling in simplicity.
  - RBF kernel demonstrated good precision but slightly lower accuracy (80%) due to a few misclassifications.
- Class-specific Insights:
  - The Polynomial kernel was most effective for Class 3 with 100% recall.
  - The RBF kernel provided the best precision for Class 1 (89%).
- Model Trade-offs:
  - Polynomial kernel's slightly higher accuracy comes at the cost of computational complexity due to additional parameters (degree).

Chapter 7: Comparison of decision region

Comparison of decision region plots obtained for the perceptron-based classier and linear. kernel based SVM on Dataset-1 (a).

Perceptron Accuracy: 99.78%
SVM with Linear Kernel Accuracy: 100.00%



The decision region plots for the Perceptron-based classifier and the SVM with a linear kernel on Dataset-1(a) reveal notable differences in their decision boundaries. The Perceptron classifier achieved an accuracy of 99.78%, whereas the SVM classifier with a linear kernel achieved a perfect accuracy of 100%.

The Perceptron decision regions demonstrate clear linear separability; however, the boundary between classes slightly misclassifies a small number of points, particularly near the class overlaps. In contrast, the SVM decision regions show sharper and more precise boundaries, effectively separating all classes without any misclassifications. This can be attributed to the SVM's optimization of the margin between classes, leading to more robust decision boundaries.

Overall, while both classifiers perform well, the SVM's ability to perfectly classify the dataset highlights its superiority in handling linear separability with precise boundaries compared to the Perceptron.