ML Assignment-1

By Akshith Vinu(231EC105) and Tejhuram Ravichandran(231EC263)

1 Brief Description of the code

This code implements a Bayesian classifier to distinguish between two types of leaves (Quercus_Variabilis and Liriodendron_Tulipifera) using shape-based features. We have used uses area ratio and aspect ratio as primary features for classification.

1.1 Dependencies

The code uses os and PIL libraries for loading the images, while numpy and pandas handle numerical computations and data organization respectively. Matplotlib serves as the visualization tool for creating various plots and graphs to display results and analysis. Finally, scipy.stats provides statistical functions for Gaussian probability calculations, while sklearn.metrics is used for model evaluation through confusion matrix computation.

2 Data Loading and Preprocessing

2.1 Loading Class Information

- Loads class IDs and species information from CSV
- Creates a dictionary mapping species names to their corresponding image IDs

2.2 Image Loading Function

Loads the image IDs that match the classes we are looking for. Then goes through the images dataset and extracts images with the same name as the IDs and then converts the images to NumPy arrays. We then calculate the area and aspect ratios for each image and store them in an array.

3 Feature Extraction

3.1 Area Ratio Calculation

Calculates ratio of white pixels to total pixels. Returns normalized area ratio as a feature.

3.2 Aspect Ratio Calculation

We have computed the aspect ratio of each image by taking the ratio of the number of columns to the number of rows. This is a helpful feature for our class as the images differ clearly on this feature

4 Dataset Creation

4.1 Loading Class Data

Takes path and class name as input. Extracts features for all images in class. Returns IDs and features for entire class.

4.2 Creating Feature DataFrame

Creates DataFrame with features and labels. Class Quercus_Variabilis labelled as 1, Class Liriodendron_Tulipifera labelled as 0. Combines data from both classes. Returns structured DataFrame for classification.

5 Data Split and Model Training

5.1 Train-Test Split

Splits data into 70 percent to training, 30 percent to testing. Preserves feature order in split.

5.2 Computing Class Statistics

Calculates mean and covariance matrix for each class. Used for Gaussian likelihood computation.

6 Classification

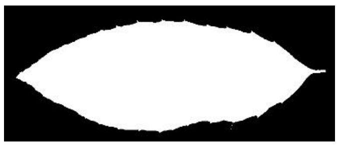
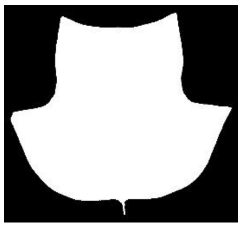
6.1 Gaussian Likelihood Calculation

Implements multivariate Gaussian probability density function. Used for computing class probabilities. Returns likelihood score for classification.

7 Visualization

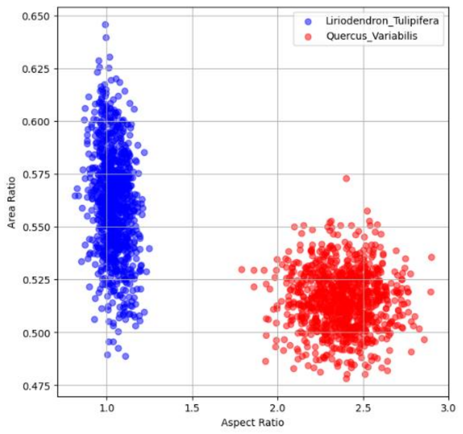
7.1 Feature Distribution Plot

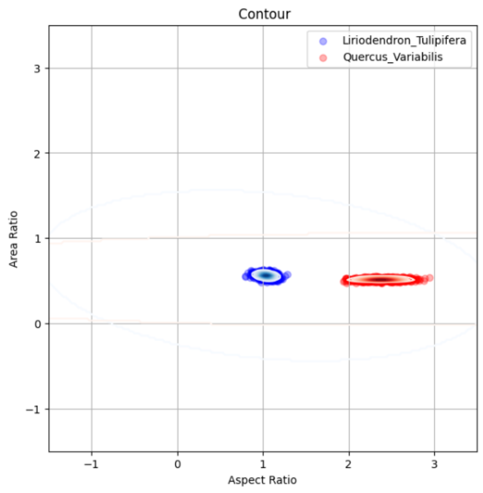
- Visualizes distribution of features for both classes
- Blue points: Liriodendron_Tulipifera
- Red points: Quercus_Variabilis

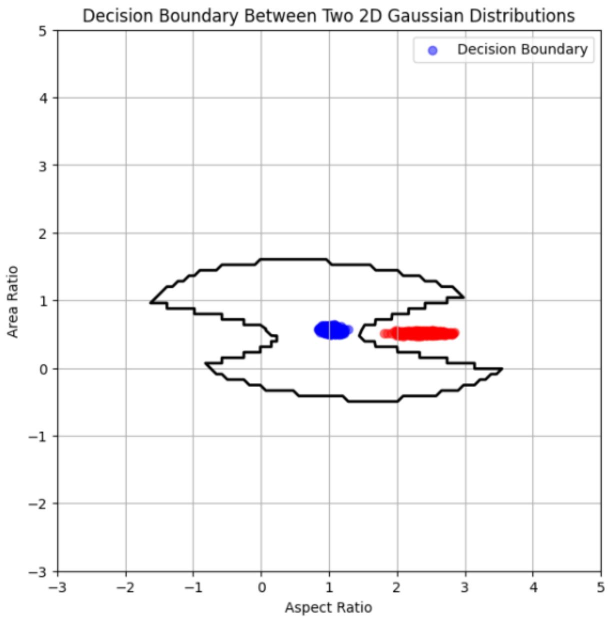



7.2 Gaussian Distribution Contours

Plots probability density contours. Shows decision boundaries between classes. Visualizes Gaussian distribution overlap.




Contour
Liriodendron_Tulipifera
Quercus_Variabilis
Area Ratio
Aspect Ratio


Decision Boundary Between Two 2D Gaussian Distributions
Decision Boundary
Area Ratio
Aspect Ratio

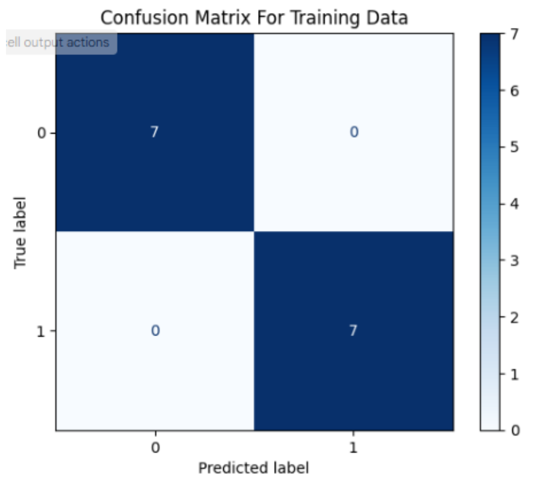
8 Model Evaluation

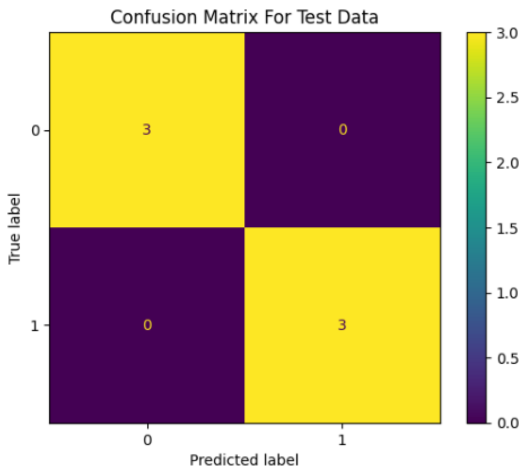
8.1 Performance Metrics

Calculates classification error rate. Computes confusion matrix for detailed performance analysis.

8.2 Confusion Matrix Visualization

Visualizes classification results. Shows true positives, false positives, true negatives, and false negatives.




Confusion Matrix For Test Data
True label
Predicted label
0
1
3
0
0
3
3.0
2.5
2.0
1.5
1.0
0.5
0.0