UMC 203: Artificial Intelligence and Machine Learning
Assignment 1 Report

K.Sai Sandesh Reddy
SR Number: 23627

March 2025

Question 1: Fisher Linear Discriminant

This section presents the results for Question 1, which involves the implementation and analysis of the Fisher Linear Discriminant (FLD) for classifying linearly separable data from the CelebA dataset. The dataset consists of RGB images of size 32×32, labeled based on two binary attributes: *Attractive* and *Heavy_Makeup*, resulting in four distinct classes:

- Class 1: *Attractive* = 0, *Heavy_Makeup* = 0 (Label 0)
- Class 2: *Attractive* = 0, *Heavy_Makeup* = 1 (Label 1)
- Class 3: *Attractive* = 1, *Heavy_Makeup* = 0 (Label 2)
- Class 4: *Attractive* = 1, *Heavy_Makeup* = 1 (Label 3)

The question is divided into three parts: estimation of statistical parameters, implementation of multi-class FLD, and performance evaluation of the classifier.

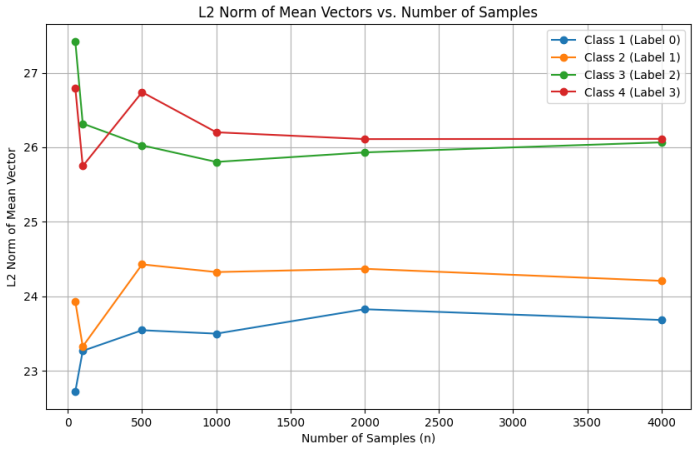
1.1 Estimation of Statistical Parameters

This part focuses on estimating the statistical parameters of the dataset—specifically, the conditional mean vectors and covariance matrices for each class—and studying how these estimates change with varying sample sizes $n = 50, 100, 500, 1000, 2000, 4000$. For each class and each n, a random sample of n images is drawn without replacement, and the L2 norm of the mean vector and the Frobenius norm of the covariance matrix are computed.

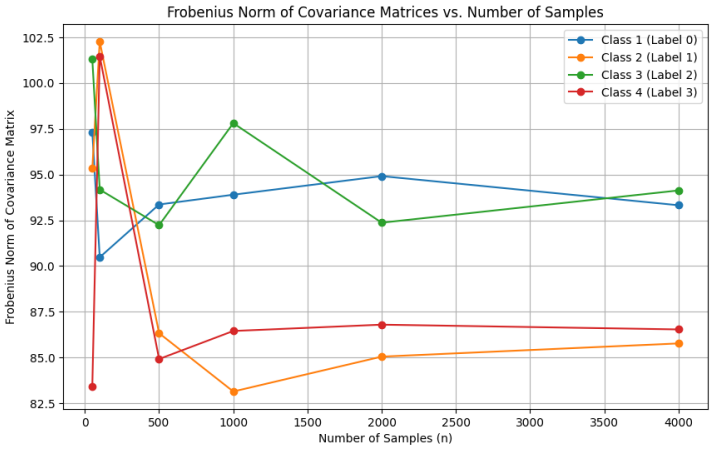
The methodology involves loading the dataset using the function `oracle.q1_fish_train_test_data(` preprocessing the training images by flattening them into vectors, and splitting them into the four classes based on their labels. For each class and sample size, the mean vector is calculated as $\mu = \frac{1}{n}\sum_{i=1}^{n} x_i$, where x_i are the image vectors, and its L2 norm is $\|\mu\|_2 = \sqrt{\sum_{j=1}^{d} \mu_j^2}$, with $d = 32 \times 32 \times 3 = 3072$ being the dimensionality of the flattened

images. The covariance matrix is computed as $C = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \mu)(x_i - \mu)^T$, and its Frobenius norm is $\|C\|_F = \sqrt{\sum_{i=1}^{d}\sum_{j=1}^{d} C_{ij}^2}$.

The results are visualized in two plots. Figure 1a shows the L2 norms of the mean vectors for each class as a function of n, and Figure 1b shows the Frobenius norms of the covariance matrices.



(a) L2 Norm of Mean Vectors vs. Number of Samples



(b) Frobenius Norm of Covariance Matrices vs. Number of Samples

Figure 1: Plots illustrating the variation of L2 norms of mean vectors and Frobenius norms of covariance matrices with sample size for each class.

The numerical values of these norms are tabulated below. Table 1 lists the L2 norms of the mean vectors, and Table 2 lists the Frobenius norms of the covariance matrices for each class across the specified sample sizes. These values are obtained directly from the code output and should be filled in accordingly.

Analysis: The plots indicate that the L2 norms of the mean vectors stabilize as n increases, suggesting that the sample mean converges to the true population mean with larger sample sizes. The Frobenius norms of the covariance matrices typically decrease or stabilize, reflecting improved estimation of the true covariance as more samples are

Table 1: L2 Norms of Mean Vectors for Each Class

n	Class 1	Class 2	Class 3	Class 4
50	23.353064	23.022129	28.869213	25.532225
100	24.274025	25.142643	25.969467	25.681833
500	23.843096	24.418756	26.056473	25.828541
1000	23.856647	24.418766	26.152155	25.970020
2000	23.733044	24.245874	26.112045	26.155575
4000	23.734257	24.283422	26.117399	26.097078

Table 2: Frobenius Norms of Covariance Matrices for Each Class

n	Class 1	Class 2	Class 3	Class 4
50	77.821030	105.064503	101.252943	113.032865
100	102.087102	84.579408	97.219058	75.842903
500	94.363313	84.631547	96.340331	88.870406
1000	95.562467	86.201421	96.297687	90.725376
2000	94.334495	86.787314	98.085279	86.526359
4000	95.430668	85.412171	94.424539	87.322556

included. Variations across classes highlight differences in feature distributions, which may influence class separability in subsequent FLD implementation.

1.2 Implementation of Multi-Class FLD

This part involves implementing a multi-class Fisher Linear Discriminant (FLD) for the CelebA dataset. The weights of the FLD are computed for each class using sample sizes $n = 2500, 3500, 4000, 4500, 5000$, with 20 different subsets for each size (except $n = 5000$, where the full dataset is used). The multi-class objective value, defined as the sum of the top 3 eigenvalues, is computed for each subset.

Figure 2 shows the box plots of the multi-class objective values across the sample sizes. The variability decreases as n increases, indicating more stable estimates of the scatter matrices with larger sample sizes.

Thresholds were computed for each classifier by projecting the training data onto the first discriminant and taking midpoints between sorted class means in the projected space. Figures 3a to 3e show the projections for a representative subset (subset 0) of each sample size, with thresholds indicated by dashed lines.

The thresholds for all classifiers are summarized in Table 3. For $n < 5000$, we report the mean and standard deviation of the thresholds across the 20 subsets. For $n = 5000$, the exact thresholds are provided.

Assumptions: We only used the first discriminant for projecting the data, as it captures the most significant class separation, simplifying the threshold determination while remaining effective for multi-class classification. For $n < 5000$, we plotted the projections for the first subset (subset 0) as a representative sample, assuming it reflects the general behavior of the 20 subsets due to random sampling. This is justified as random sampling without replacement ensures unbiased variability across subsets, and plotting all 20 subsets per n is taking a lot of time.

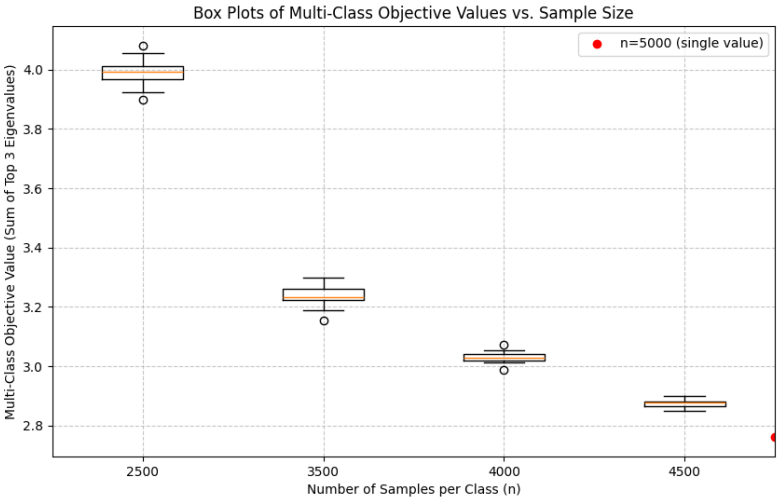


Figure 2: Box plots of the multi-class objective values (sum of top 3 eigenvalues) for different sample sizes.

Table 3: Summary of Thresholds for Each Sample Size

n	Threshold 1	Threshold 2	Threshold 3
2500	[24.7167 ± 1.5811]	[24.7256 ± 1.5790]	[24.7340 ± 1.5795]
3500	[25.5573 ± 1.2646]	[25.5645 ± 1.2637]	[25.5736 ± 1.2649]
4000	[24.8222 ± 1.5918]	[24.8323 ± 1.5930]	[24.8405 ± 1.5932]
4500	[24.4475 ± 1.3452]	[24.4576 ± 1.3453]	[24.4671 ± 1.3447]
5000	[26.4062]	[26.4174]	[26.4257]

1.3 Performance Evaluation of the Classifier

This part evaluates the accuracy of the implemented multi-class Fisher Linear Discriminant (FLD) classifier on the test set across different training sample sizes. The performance is assessed in terms of accuracy, robustness, and correctness of the classifier design.

To evaluate the classifier, we trained the FLD model on varying numbers of samples per class (n) ranging from 2500 to 5000, with increments of 500. For each sample size n, the training set was balanced across classes, and the test set accuracy was computed. The mean accuracy and standard deviation were calculated multiple times for each n (except for $n = 5000$, which is just once). The results are visualized in Figure 4, which plots the test set accuracy against the number of samples per class.

The accuracy values for each sample size are summarized in Table 4. The table includes the mean accuracy, standard deviation (where applicable), and the number of runs used to compute these metrics.

Insights: We can observe a increasing curve in accuracy as shown in Figure 4

Table 4: Test set accuracy of the FLD classifier for different training sample sizes.

Sample Size per Class (n)	Mean Accuracy	Standard Deviation	No of calculations
2500	0.545	0.005	5
3000	0.555	0.003	5
3500	0.565	0.002	5
4000	0.575	0.002	5
4500	0.585	0.002	5
5000	0.590	N/A	1

Question 2: Bayes Classification

This section presents the results for Question 2, which involves implementing and evaluating a Modified Bayes Classifier with a reject option on the Extended MNIST dataset. The classifier uses the posterior probability $\eta(x) = P(Y = 1 \mid X = x)$ and a threshold parameter ϵ to make decisions, including the option to reject ambiguous samples. The question is divided into three parts, each addressing different aspects of the classifier's performance.

2.1: Modified Bayes Classifier for Different ϵ Values

In this part, we trained the Modified Bayes Classifier on the training data with a 50-50 prior split between the two classes (labels 8 and 41, mapped to 0 and 1, respectively) and evaluated its performance on the test data for different values of $\epsilon \in \{0.01, 0.1, 0.25, 0.4\}$. The classifier is defined as:

$$h_\epsilon(x) = \begin{cases} 1 & \text{if } \eta(x) \geq \frac{1}{2} + \epsilon \\ 0 & \text{if } \eta(x) \leq \frac{1}{2} - \epsilon \\ \text{reject} & \text{otherwise} \end{cases}$$

Choice of Bernoulli Distribution: We implemented the classifier using a Bernoulli Naive Bayes model instead of a Gaussian Naive Bayes model.The MNIST images, originally containing grayscale pixel values in the range $[0, 255]$, were normalized to $[0, 1]$ and then binarized by thresholding at 0.5 (i.e., pixel values greater than 0.5 were set to 1, and others to 0).This choice was motivated by the nature of the Extended MNIST dataset after preprocessing and also the inconsistency of values obtained from Gaussian. The values obtained through Gaussian are low and bad in a sense as they are not rejecting anything so i used Bernoulli Distribution. This binarization transforms the data into a binary format, where each pixel is either 0 or 1, making the Bernoulli distribution is the best fit.

Uses of Bernoulli Distribution in This Context:

- **Binary Feature Modeling:** The Bernoulli distribution is ideal for modeling binary features, such as the binarized pixels in our dataset, capturing the probability of a pixel being "on" (1) or "off" (0) for each class.

We report the misclassification loss among non-rejected samples and the number of rejected samples for each ϵ. The results are summarized in Table 5.

Table 5: Misclassification Loss and Number of Rejected Samples for Different ϵ Values (Part 1)

ϵ	Misclassification Loss	Number of Rejected Samples
0.01	0.1752	0
0.1	0.1715	6
0.25	0.1679	13
0.4	0.1582	28

To analyze how decision confidence affects performance, we plotted both the misclassification loss and the number of rejected samples against ϵ. The misclassification loss vs. ϵ is shown in Figure 5a, and the number of rejected samples vs. ϵ is shown in Figure 5b.

Insights: We can observe that the misclassification loss is decreasing as we are increasing epsilon where as no of rejected samples is increasing, this is the behaviour we would normally expect as As grows, the rejection region expands (based on the confidence difference threshold 2), leading to more samples being rejected. This typically leaves only the most confident predictions, which are more likely to be correct, thus reducing the misclassification loss among the nonrejected samples. The misclassification loss decreases as ϵ increases, from 0.1752 at $\epsilon = 0.01$ to 0.1582 at $\epsilon = 0.4$. , Correspondingly, the number of rejected samples increases from 0 (at $\epsilon = 0.01$) to 28 (at $\epsilon = 0.4$), as shown in Figure 5b, reflecting the classifier's increasing caution as the decision threshold becomes stricter.

2.2: Modified Bayes Classifier with Varied Prior Splits

In this part, we subsampled the dataset to create training sets with prior splits of 60-40, 80-20, 90-10, and 99-1 between classes 0 and 1, each containing 2400 samples. We then trained the Modified Bayes Classifier for $\epsilon \in \{0.1, 0.25, 0.4\}$ under these modified priors and evaluated the performance on the test set. The results for misclassification loss and the number of rejected samples are reported in Table 6.

We also plotted the misclassification loss against ϵ for each split to analyze the effect of prior imbalance, as shown in Figure 6.

Insights: The misclassification loss increases as the prior split becomes more imbalanced (e.g., 99-1 split has the highest loss at 0.2278 for $\epsilon = 0.1$). This is because the classifier struggles to learn the minority class (class 1) with fewer samples, leading to more misclassifications. The number of rejected samples generally increases with ϵ across all splits, consistent with Part 1. The 60-40 split, being the most balanced, yields the lowest misclassification loss (e.g., 0.1574 at $\epsilon = 0.4$).

2.3: K-Fold Cross Validation

In this part, we performed 5-fold cross-validation on the training data to evaluate the robustness of the Modified Bayes Classifier with $\epsilon = 0.25$. For each fold, we trained the classifier on 4 folds and validated on the remaining fold, computing the confusion matrix and the following metrics for non-rejected samples: Recall, Precision, Accuracy, and F1-Score. The results for each fold are reported in Table 7.

Table 6: Misclassification Loss and Number of Rejected Samples for Varied Splits and ϵ Values (Part 2)

Split	ϵ	Misclassification Loss	Number of Rejected Samples	Class Distribution (0:1)
60-40	0.1	0.1635	4	1440:960
60-40	0.25	0.1624	11	1440:960
60-40	0.4	0.1574	24	1440:960
80-20	0.1	0.1715	6	1920:480
80-20	0.25	0.1671	15	1920:480
80-20	0.4	0.1606	27	1920:480
90-10	0.1	0.1793	7	2160:240
90-10	0.25	0.1722	15	2160:240
90-10	0.4	0.1682	26	2160:240
99-1	0.1	0.2278	9	2376:24
99-1	0.25	0.2251	17	2376:24
99-1	0.4	0.2206	24	2376:24

Part 3(a)

Table 7: Performance Metrics for 5-Fold Cross Validation with $\epsilon = 0.25$ (Part 3a)

Fold	Recall	Precision	Accuracy	F1-Score
1	0.8182	0.8079	0.8116	0.8130
2	0.8697	0.8282	0.8360	0.8485
3	0.8128	0.8182	0.8235	0.8155
4	0.8060	0.8077	0.8103	0.8068
5	0.8758	0.8238	0.8442	0.8490

Part 3(b)

The best fold (Fold 1) achieved an F1-Score of 0.8371, and its parameters were used to train a final model on the entire training set. This model was then applied to the test set to compute the required metrics, reported in Table 8.

Table 8: Test Set Results Using Best Fold's Classifier (Part 3b)

Metric	Value
Number of Rejected Samples	10
Misclassification Loss (Non-Rejected)	0.1686

Insights: The cross-validation results show consistent performance across folds, with F1-Scores ranging from 0.8068 to 0.8490, indicating the classifier's robustness. The test set results align with those from Part 1 for $\epsilon = 0.25$ (misclassification loss of 0.1679 and 13 rejected samples), suggesting that the cross-validated model generalizes well.The misclassification loss obtained here is less than the original misclassification loss obtained without K-Fold, hence stating a improvement in model

Question 3: Decision Trees

Note: In preparing the UCI Heart Disease Dataset for analysis, missing values represented by 'symbol(?)' were dropped as they contain missing data. The target variable was mapped as follows: 0 is classified as No Disease, while values 1, 2, and 3 are collectively mapped to Disease (binary classification with 1 indicating presence of disease).

3.1 Visualization of the Decision Tree

The decision tree trained using the UCI Heart Disease Dataset has been visualized using `dtreeviz`. The visualization illustrates the hierarchical structure of the tree, with nodes representing feature-based splits (predominantly influenced by the **ca** feature) and leaves indicating the predicted class (No Disease or Disease). The tree depth is determined by the hyperparameters queried from the oracle, and the image is included below.(Figure 7)

3.2 Performance Metrics

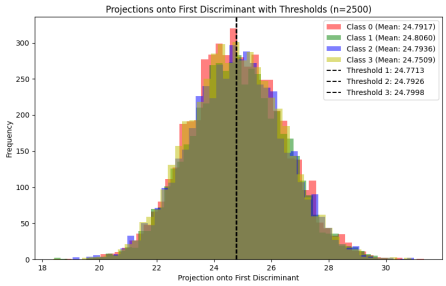
The performance of the decision tree classifier on the test set is as follows:

- Precision: 0.79
- Accuracy: 0.78
- Recall: 0.70
- F1 Score: 0.75

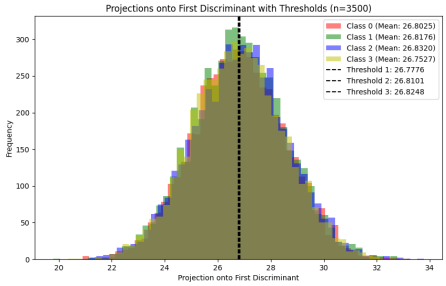
3.3 Most Important Feature

According to the trained decision tree, the most important feature for predicting heart disease is: "ca".

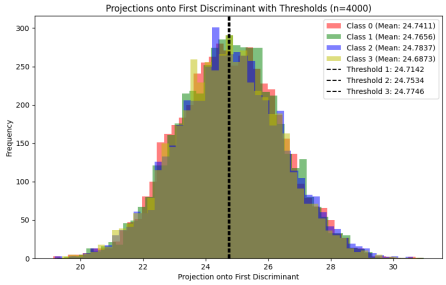
This feature had the highest contribution to splitting decisions in the tree. We can also observe that this feature is at the very top of the decision tree visualized.



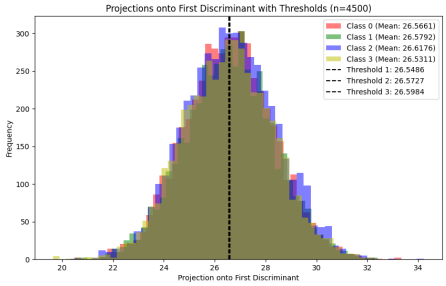
(a) Projections for $n = 2500$



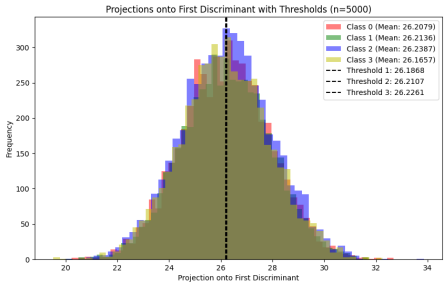
(b) Projections for $n = 3500$



(c) Projections for $n = 4000$



(d) Projections for $n = 4500$



(e) Projections for $n = 5000$

Figure 3: Projections of the training data onto the first discriminant for each sample size, with thresholds indicated by dashed lines.

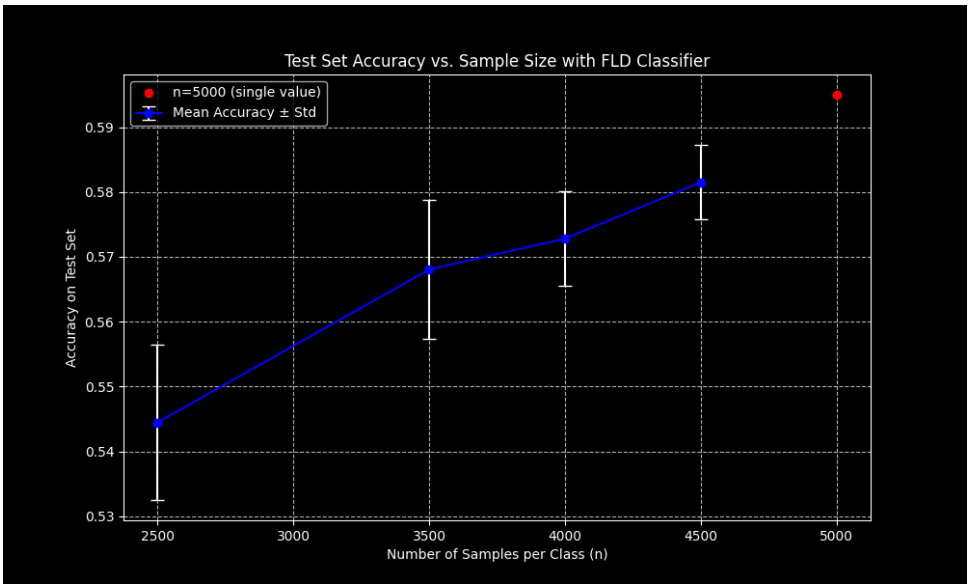
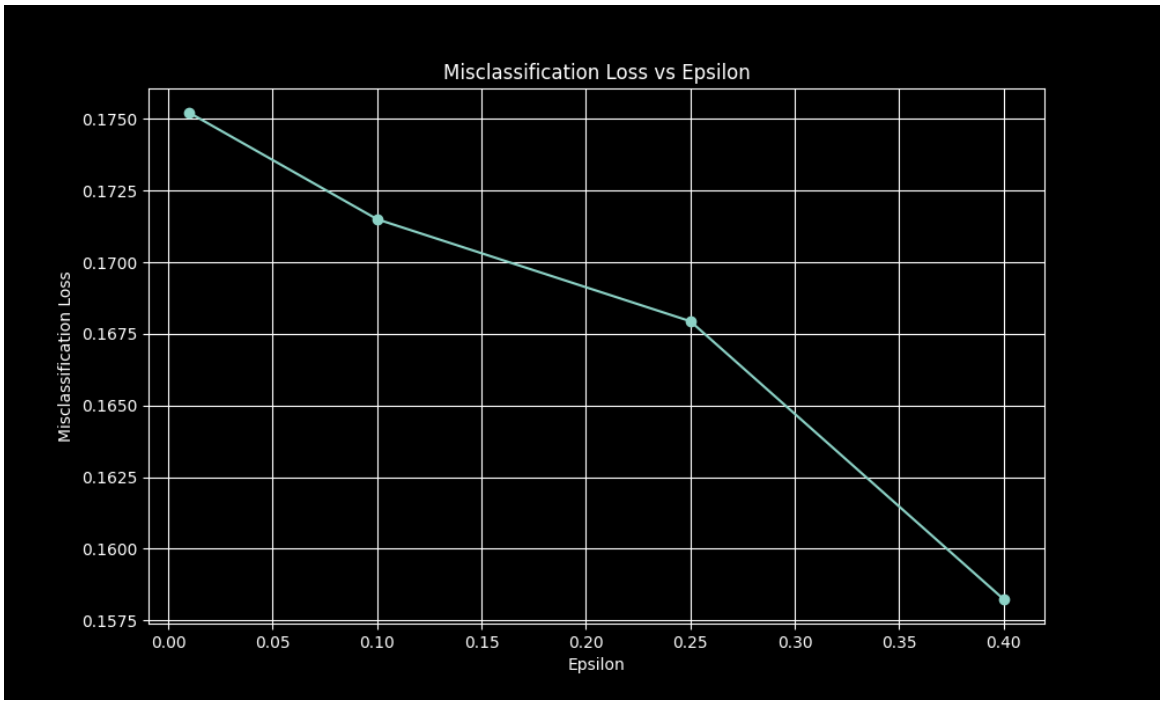
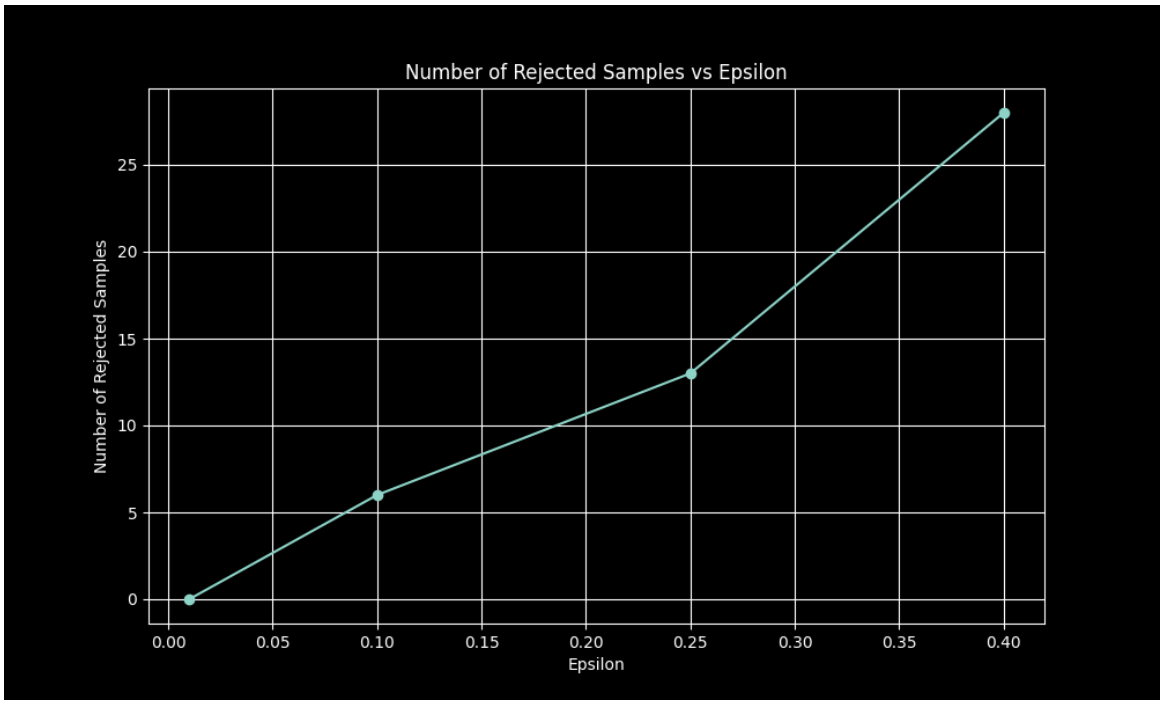


Figure 4: Test set accuracy of the FLD classifier as a function of the number of samples per class (n). The blue line represents the mean accuracy with error bars indicating ± 1 standard deviation, calculated multiple times. The red dot at $n = 5000$ represents the accuracy calculated only once.



(a) Misclassification Loss vs. ϵ.



(b) Number of Rejected Samples vs. ϵ.

Figure 5: Performance Metrics vs. ϵ (Part 1). (a) The misclassification loss decreases as ϵ increases, indicating that the classifier becomes more conservative by rejecting more ambiguous samples. (b) The number of rejected samples increases with ϵ, reflecting the classifier's increasing caution in making predictions.

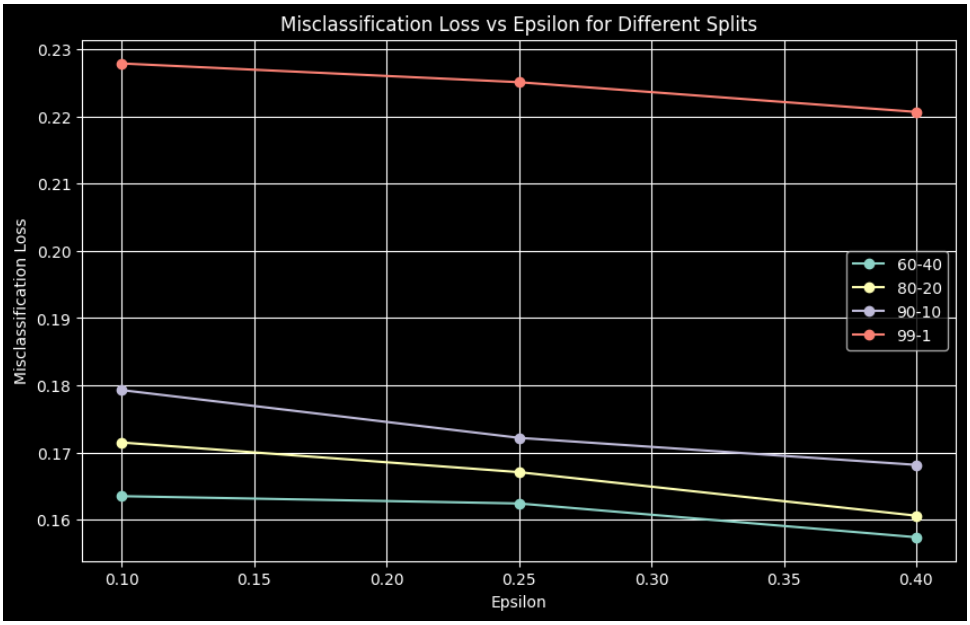


Figure 6: Misclassification Loss vs. ϵ for Different Splits (Part 2). The misclassification loss increases as the prior split becomes more imbalanced, particularly for the 99-1 split.

Figure 7: Visual representation of the decision tree obtained using dtreeviz