|  |
| --- |
| **Maximum Likelihood Estimation and Face Detection** |
| **Josh Gleason and Rod Pickens** |
| **Programming Assignment 2** |
| **Computer Science 679 – Pattern Recognition, UNR, Dr. Bebis** |
| **Due: Monday March 23, 2015** |
| **Submitted: Monday March 23, 2015** |

Contents

[1 Abstract 3](#_Toc414124669)

[2 Technical Discussion 3](#_Toc414124670)

[2.1 Maximum Likelihood Estimation 3](#_Toc414124671)

[2.1.1 Multivariate Normal Distribution 5](#_Toc414124672)

[2.2 Classifiers 7](#_Toc414124673)

[2.2.1 N-Class classifier (JG) 7](#_Toc414124674)

[2.2.2 One class classifier (JG) 8](#_Toc414124675)

[2.2.3 Comparison (JG) 9](#_Toc414124676)

[2.3 Face Detection (RP) 9](#_Toc414124677)

[3 Project 9](#_Toc414124678)

[3.1 Problem 1 & 2 (JG) 9](#_Toc414124679)

[3.1.1 Problem 1 10](#_Toc414124680)

[3.1.2 Problem 2 11](#_Toc414124681)

[3.2 Problem 3 (RP) 12](#_Toc414124682)

[4 Results 12](#_Toc414124683)

[4.1 Maximum Likelihood Estimation (JG) 12](#_Toc414124684)

[4.1.1 Problem 1 (JG) 12](#_Toc414124685)

[4.1.2 Problem 2 (JG) 17](#_Toc414124686)

[4.2 Face Detection (RP) 22](#_Toc414124687)

[4.2.1 Problem 3a (Chromatic Color Space) (RP) 22](#_Toc414124688)

[4.2.2 Problem 3b (YCbCr Color Space) (RP) 23](#_Toc414124689)

[4.2.3 Problem 3c (RGB Color Space) (RP) 23](#_Toc414124690)

[4.2.4 Conclusion 23](#_Toc414124691)

[5 Conclusion 24](#_Toc414124692)

[6 Contributors 24](#_Toc414124693)

**Bayesian Classification of Classes with Features that are Normally Distributed**

# Abstract

This paper describes our research regarding the second class project for the Computer Science (CS) pattern recognition class CS 679 taught by Dr. Bebis Department Chair of the Computer Science Department at the University of Nevada in Reno, Nevada.

The topics included in this main body of this paper are a description of the project, maximum likelihood estimate, face detection and the classifiers used face detection.

# Technical Discussion

In the CS 679 class, we have learned about maximum likelihood estimation and Bayesian estimation. For this project, we were to focus on estimation of classifier parameters using likelihood methods and face detection using color space features. For the maximum likelihood section, we discuss the theory and the approach used in this project to estimate the parameters of several Gaussian distributions. In the face detection section, we briefly discuss color spaces, but the authors are not experienced in color space theory, so the discussion is terse.

## Maximum Likelihood Estimation

Maximum Likelihood (ML) is one method of approximating the likelihood functions given that the parametric form of the distributions is known. Equation (1) describes the parametric form of the likelihood function with representing parameter number for class

|  |  |  |
| --- | --- | --- |
|  |  | (1) |

For simplicity, the parameters will be manipulated in vector form

|  |  |  |
| --- | --- | --- |
|  |  | (2) |

In order to find parameters which most accurately describe the likelihood function, training data is used. One set of training data is used for each class and that training data consists of independently drawn samples. In other words, the training data used to determine the likelihood function for class are samples drawn from a random variable with probability density function . The training data for class is represented as follows

|  |  |  |
| --- | --- | --- |
|  |  | (3) |

The premise of ML is to determine the parameters of the likelihood function that would be most likely to produce the training data. This leads to the following definition for the parameters estimated by maximum likelihood.

|  |  |  |
| --- | --- | --- |
|  |  | (4) |

Where is an estimate of the true parameters for the likelihood function.

Because the samples in the training data were drawn independently, the probability density function in the right hand side of Equation (4) may be expressed as

|  |  |  |
| --- | --- | --- |
|  |  | (5) |

Because Equation (4) is an optimization, and probability density functions are always positive, we can take the expression in the maximization and apply a log function (or any other monotonically increasing function). By also substituting in Equation (5), Equation (4) becomes

|  |  |  |
| --- | --- | --- |
|  |  | (6) |

The logarithm makes dealing with the maximization problem easier in many cases. For example, when the model for the likelihood function is a Normal distribution as we will see in Section 2.1.1.

In order to solve the Equation (6) we take the gradient of the optimization expression and find where it becomes zero.

|  |  |  |
| --- | --- | --- |
|  |  | (7) |

where

|  |  |  |
| --- | --- | --- |
|  |  | (8) |

Solving for all solutions to Equation (7) will give the position of all local maxima, local minima, and inflection points. If not apparent, it is important to test each solution as well as the bounds of the function to determine where the true global maxima lies.

### Multivariate Normal Distribution

One interesting case to consider for ML is when likelihood takes the form of a Multivariate Normal distribution with an unknown mean and unknown covariance .

The parameter vector then consists of the mean and covariance of the distribution.

To begin the derivation we substitute the equation for the Multivariate Normal distribution into Equation (7)

|  |  |  |
| --- | --- | --- |
|  |  | (9) |

Because is composed of and we express Equation (9) as the following scalar-matrix derivatives

|  |  |  |
| --- | --- | --- |
|  |  | (10) |
|  |  | (11) |

Equation (10) may be solved as follows

|  |  |  |
| --- | --- | --- |
|  |  | (12) |

Similarly, Equation (11) may be solved as follows (substituting for )

|  |  |  |
| --- | --- | --- |
|  |  | (13) |

Together, the results from Equations (12) and (13) express the ML estimates for the parameters of the Multivariate Normal distribution as

|  |  |  |
| --- | --- | --- |
|  |  | (14) |
|  |  | (15) |

which are the estimates that will be tested in this project.

## Classifiers

In this project we again make use of a Bayesian classifier so a brief recap of the theory is warranted.

In pattern recognition a classifier is a machine that makes a decision among candidate choices given the available evidence conditioned on the choice under consideration. Conditioned evidence may be described by an n-dimensional feature vector as follows:

|  |  |  |
| --- | --- | --- |
|  |  | (16) |

Each dimension of the vector is a measurement of one aspect of the object being evaluated prior to that object being classified.

### Multiclass classifier (JG)

The Multiclass Bayesian classifier is the general case of the Bayesian classifier in that it classifies conditioned evidence into one of the states nature with . In general, a state of nature is unpredictable and must therefore be described probabilistically.

The goal of a Bayesian Classifier is to classify conditioned evidence in such a way that overall risk is minimized. Risk is defined by Hart[[1]](#footnote-3) as

|  |  |  |
| --- | --- | --- |
|  |  | (17) |

Where the function is the conditional risk associated a decision rule given a piece of conditioned evidence.

The maximum error classifier is a special version of the minimum risk where

|  |  |  |
| --- | --- | --- |
|  |  | (18) |

And the decision corresponds to the action of classifying as class which can be expressed mathematically as .

This reduces to the following decision rule (the shorthand is used to represent

|  |  |  |
| --- | --- | --- |
|  |  | (19) |

which simply defines the decision rule as choosing the class which yields lowest a posteriori probability.

Because this is a maximization problem, the posteriori probability doesn’t need to be computed directly. Instead we can apply Bayes formula to Equation (19) and remove the scaling factor from the maximization yielding

|  |  |  |
| --- | --- | --- |
|  |  | (20) |

Furthermore, we can apply the logarithm function to the maximization portion of (20) which then becomes

|  |  |  |
| --- | --- | --- |
|  |  | (21) |

The functions which are being maximized is generally known as the set of discriminate functions which serve to simplify our discussion. While function which maintains the maximization may be used, a commonly used discriminate function is the one in Equation (20)

|  |  |  |
| --- | --- | --- |
|  |  | (22) |

When classifying a sample, the classifier assigns the state of nature which yields the lowest value for the corresponding discriminate function.

### One class classifier (JG)

### Comparison (JG)

## Face Detection (RP)

Face detection and recognition have been an important research area for decades, and a number of methods have been explored and developed for detecting and recognizing faces.

Face detection and recognition are related but different. Face detection is the task of finding the face of any person (without identifying the person) in either a black and white image or a color image. Face recognition is the task of identifying a person by their name in an image. We focus on face detection for this class project.

In this paper, we give a quick summary of face detection and recognition based on a survey paper by Jafri and Arabina[[2]](#footnote-4), and then we discuss the approach we use in this research project based on skin color.

Face detection and recognition fall into two general categories: feature-based and holistic. Feature-based methods rely on local geometric measurements across the face based local features that are expressed in terms of distance ratios between the eyes, nose, mouth, forehead, chin, ears, and other facial features to each other. Methods used to measure relative distances are to detect, for example, the eyes and nose and measure the distances between the eyes and nose.

Holistic methods rely on measurements across the whole face rather than local measurements throughout the face. Early approaches to holistic face detection were based on techniques related to principal component analysis (PCA) or Fisher linear discriminant analysis (LDA).[[3]](#footnote-5) The research continues in this area.

Skin color is considered a holistic approach to detecting faces and has proven to be very valuable because the algorithms and features are computationally inexpensive and give quite good results. Because of the success of the early experiments, researchers have experimented with numerous color spaces to determine which colors give best features for face detection, such as RGB and YCbCr.

Jie Yang and Alex Waibel were early proponents of detecting faces in video using color features, and they experimented with chromatic colors.[[4]](#footnote-6) The researchers chose chromatic colors because they wanted to remove brightness inherent in RGB imagery. The school project expands upon the paper by having the students consider YCbCr for comparison. In addition, the RGB color space is evaluated as a baseline even though it is not required for the class project.

Based on material from Pratt,[[5]](#footnote-7) additive colors can be defined by matching any color (C) of the additive colors to a blended set of calibrated primary color sources: white (W), red (P1), green (P2), and blue (P3). When the blended colors match the color (C), the color C has been defined by calibrated standards. The color sources (lights) for P1, P2, and P3 are calibrated with a calibrated white color source A(W), and the calibrated sources for P1, P2, and P3 are labeled A(P1), A(P2), and A(P3). With these a color C that is indistinguishable from the mixture of colors A(P1), A(P2), and A(P3) is defined with what are called the tristimulus values

|  |  |
| --- | --- |
|  | (23) |

Not demonstrated in this paper but noted by the author Pratt is that tristimulus values can be negative, which is a primary reason researchers develop and utilize other color schemes, such as RGB and YCbCr. Basically, color spaces based on all positive values is more intuitive.

The tristimulus values specify a color’s lightness, hue, and saturation. Another common color scheme is chromaticity which specifies the hue and saturation but not lightness. These are defined as follows:

|  |  |
| --- | --- |
|  | (24) |
|  | (25) |
|  | (21) |
|  |  |

The YCbCr colors are defined using the RGB colors as follows

|  |
| --- |
|  |
|  |
|  |

The luminance component Y is related to lightens, or brightness, and is not used as a feature in this project.

The project uses both a training data set to design the classifier and an independent testing data set to evaluate the classifier performance. All three classifiers (one for RGB features, one for chromatic features, and one for YCbCr features) are trained with the data from a single image, and all three classifier are tested on two other independent images. The training image is shown in Figures 1.

|  |
| --- |
| Figure 1: Training image |

The testing images are similar in composition to the training image in that the photographs are taken at the same location, same time of day, and with same lightening conditions. Most of the subjects are younger adults, mostly male, see Figures 2a and 2b.

|  |  |
| --- | --- |
| Figure 2a and 2b: Test images |  |

The We compare results between the classifiers and their features using receiver operator curves (ROC) that plot false positive rates against false negative rates and ROC charts that plot true positive rates versus false positive rates.

The classifier in all cases is a dichotomizer using as class human skin on the faces of people and as class other objects other than the skin of people’s faces such as clothing, sky, bridge structure, concrete, and some components of buildings. However, class also includes skin on the hands, arms, and necks of the photographed individuals. Consequently, there is overlap between the face skin examples and these other skin examples, which lowers the false positive performance of the algorithms. This is pointed out in each section on classifier results.

Face skin

Other

Other:

skin,

e.g. arm.

# Project

This project consists of 3 sections, all of which apply Maximum Likelihood estimation for learning models for Bayesian classification.

## Problem 1 & 2 (JG)

Parts 1 and 2 of the project use the random samples generated in the first project and attempt to learn the distributions using Maximum Likelihood estimation. Classification is then performed using the estimated models and the results are compared to results from project 1 where the actual models are known.

The following sequence will be used to implement the experiment.

1. Load two class test data used in Project 1
2. Estimate the probability density functions using Maximum Likelihood Estimation using a subset of the test samples.
3. Classify all test samples using a Bayesian Classifier designed using the estimated likelihood function with equal priors.
4. Count misclassified samples and compare with project 1 results.

As in project 1, the Bayesian classifier used will be a dichotomizer which examines the difference between discriminant functions for class and as follows

|  |  |  |
| --- | --- | --- |
|  |  | (23) |

Where the general discriminant function for class is defined as follows:

|  |  |  |
| --- | --- | --- |
|  |  | (24) |

Where

|  |  |  |
| --- | --- | --- |
|  |  | (25) |

and

|  |  |  |
| --- | --- | --- |
|  |  | (26) |

and

|  |  |  |
| --- | --- | --- |
|  |  | (27) |

where the parameters and are the Maximum Likelihood estimation parameters for class .For Multivariate Normal Distributions, the Maximum Likelihood estimation turns out to be the sample mean and sample covariance. The derivation of the Maximum Likelihood solution is given in Section 2.1.1.

Also, in this experiment we are no longer using the linear version of the discriminant function in problem 1 since the Maximum Likelihood estimate is not guaranteed to result in parameters with the same properties as the true distributions.

The required experiments for the project are to use 10,000 and 1,000 samples for the Maximum Likelihood estimation. However, we will perform experiments using both 100 and 10 training samples as well. The primary goal behind the extra experiments to gain a better understanding of the effect different sized training data sets have on a simple two dimensional, two-class classifier.

As an extra experiment, all the experiments for Problems 1 and 2 will be run 1000 times with different sets of random data. The classification error rate for each experiment will be recorded as the result of each experiment. The sample mean and variance of the classification error rates will then be visualized and examined in order to gain insight into how reliably a classifier may be built given a particular number of training samples.

### Problem 1

Problem 1 uses samples drawn from multivariate distributions with independent features and unity variance.

The true parameters for the normal distribution for class are:

|  |  |  |
| --- | --- | --- |
|  |  | (28) |

and the true parameters for the normal distribution for the class are:

|  |  |  |
| --- | --- | --- |
|  |  | (29) |

As a comparison, Figure 1 shows the classification results from Project 1 where the classifier was constructed with the true likelihood function known with equal priors. If the estimation is working correctly, the classification error for this project should be close to the 1.72% error rate achieved in project 1.

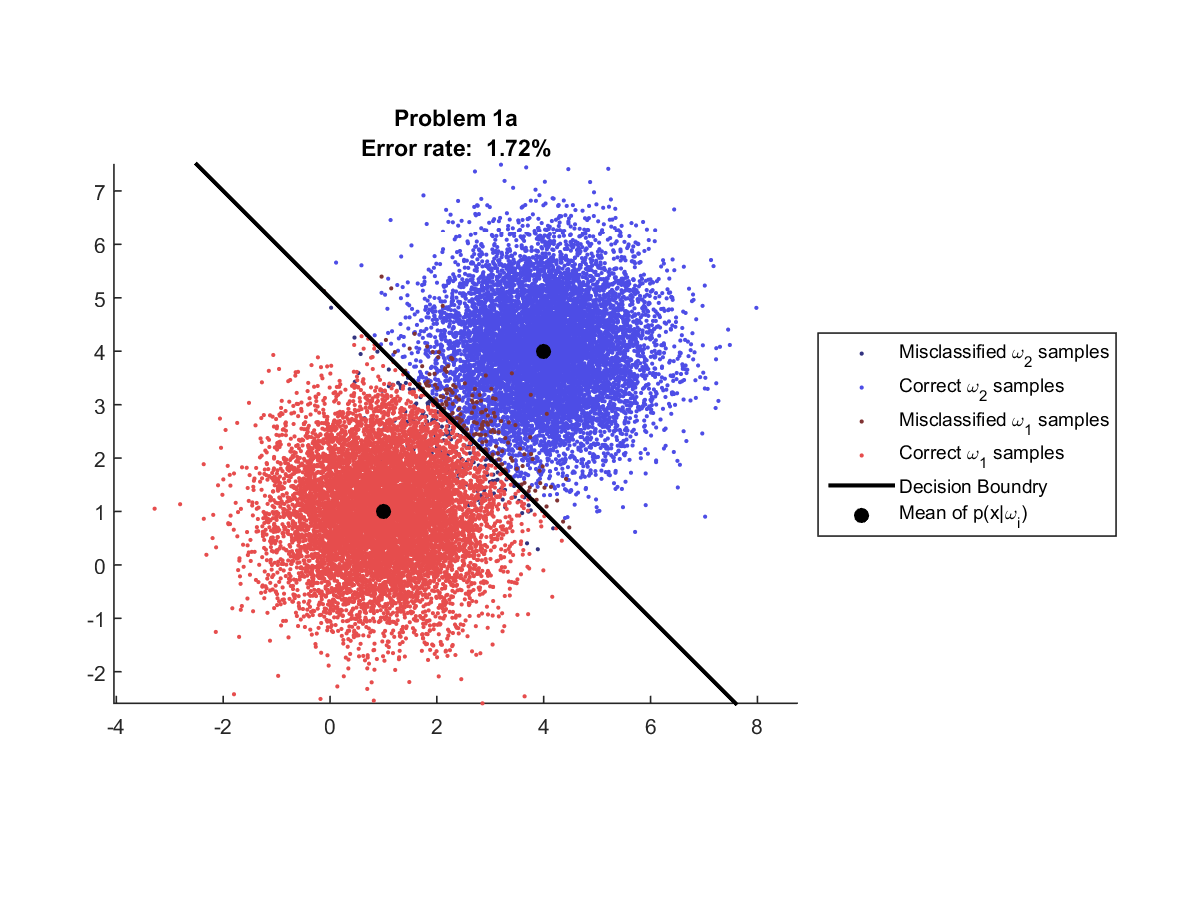


Figure 3: Project 1, Problem 1 classification results with equal priors

### Problem 2

Problem 2 uses samples drawn from multivariate normal distributions with independent features.

The true parameters for the normal distribution for class are:

|  |  |  |
| --- | --- | --- |
|  |  | (30) |

and the true parameters for the normal distribution for the class are:

|  |  |  |
| --- | --- | --- |
|  |  | (31) |

As a comparison, Figure 1 shows the classification results from Project 1 where the classifier was constructed with the true likelihood function known with equal priors. If the estimation is working correctly, the classification error for this project should be close to the 7.03% error rate achieved in project 1.

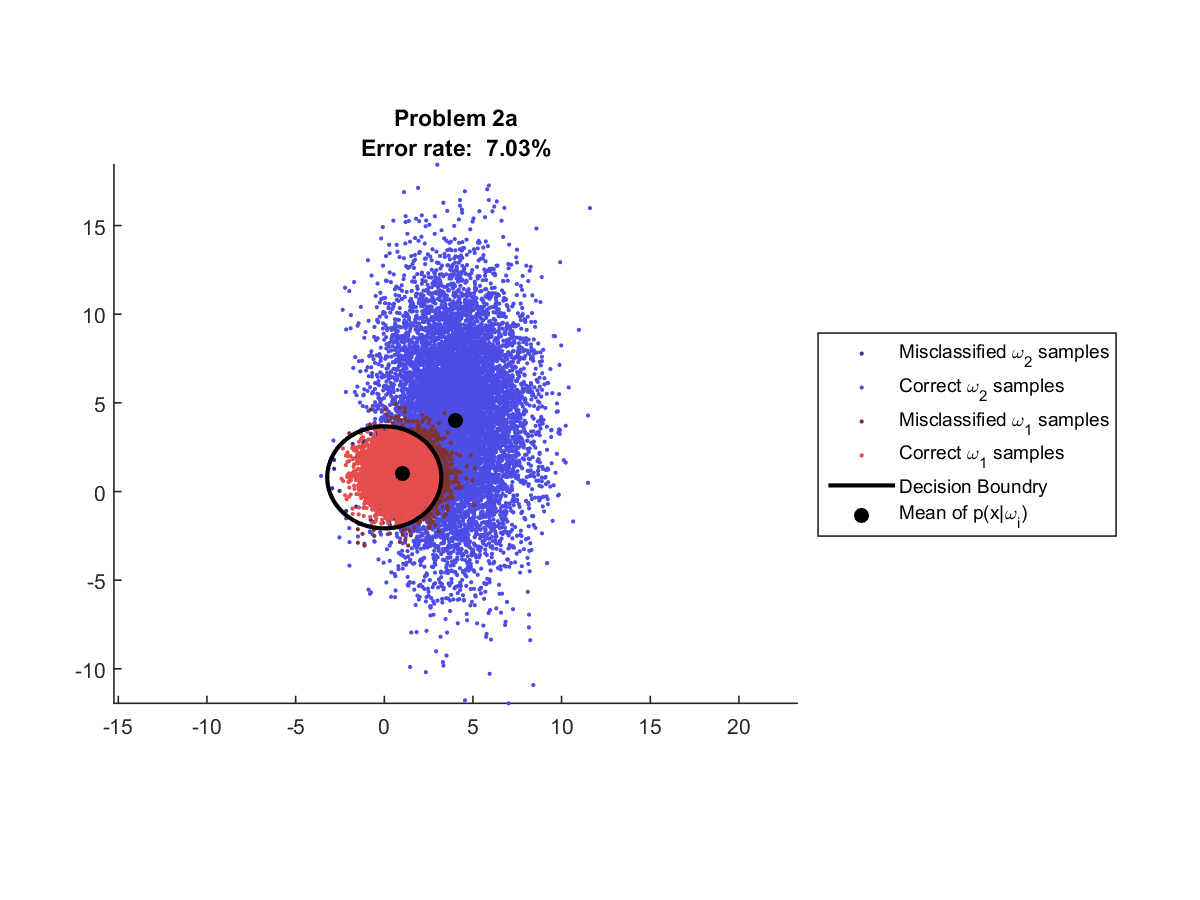


Figure 4: Project 1, Problem 2 classification results with equal priors

## Problem 3 (RP)

The task is to build a face detector using features in a classifier that are based on 1) the chromatic color space as in Yang96[[6]](#footnote-8) and 2) the YCbCr color space and to compare results between the feature sets. The advantage anticipated using the chromatic and YCbCr color spaces over RGB is that in RGB lightness is inherent in the data, and in contrast, in chromatic and YCbCr color spaces, lightness is normalized making the color based features more wide-sense stationary over varying lightening conditions.

# Results

## Maximum Likelihood Estimation (JG)

### Problem 1 (JG)

For Problem 1, we use the same samples used for each of the two classes from Project 1 problem 1. The true distributions which the samples were taken from are defined as follows:

|  |  |  |
| --- | --- | --- |
|  |  | (32) |
|  |  | (33) |

The a-priori probabilities are as follows

|  |  |  |
| --- | --- | --- |
|  |  | (34) |

In Figure 3, the two classes are colored differently: class 1 (red dots) is and class 2 (blue dots) is The decision boundary between the two classes is very close to the an affine function (a linear function with an additive intercept) which matches the expectation from Project 1 show in Figure 1.

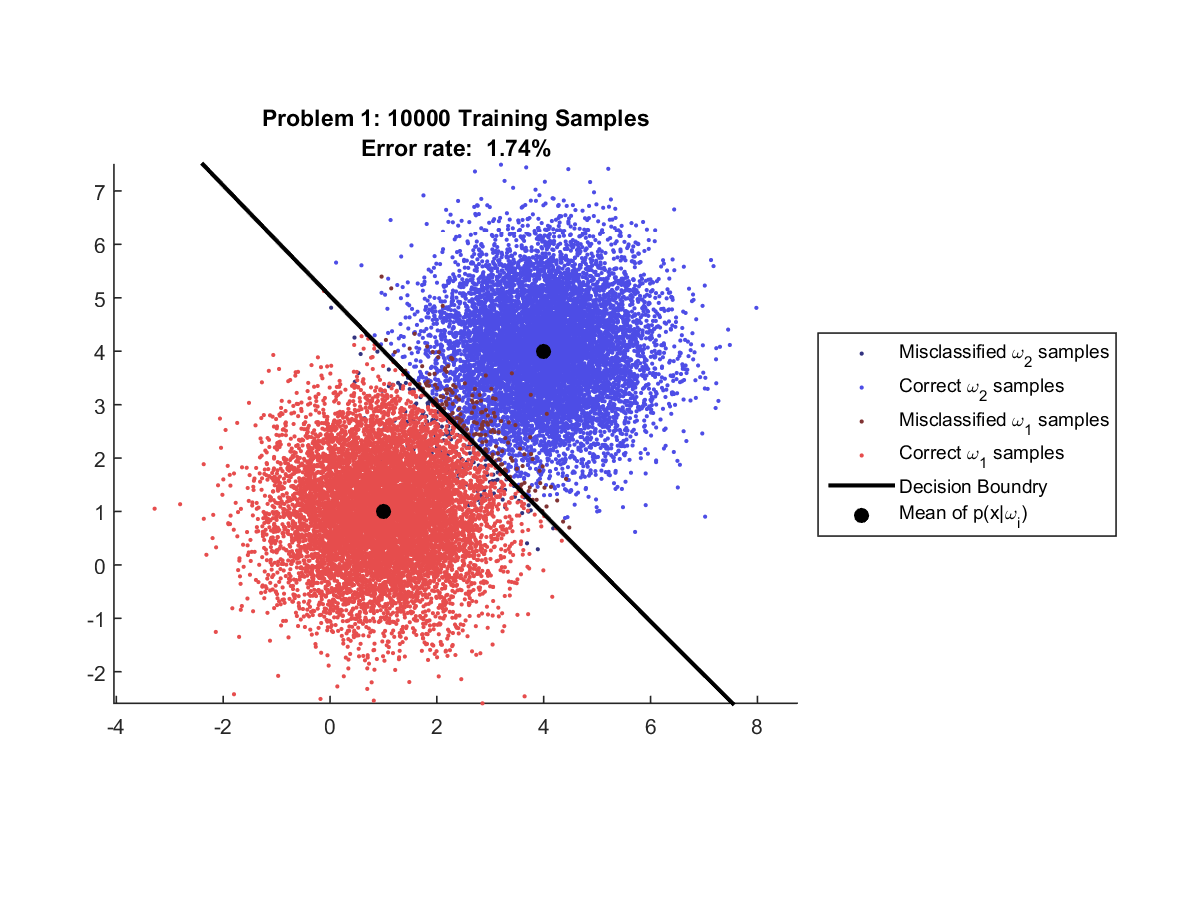


Figure 5: Classification with 10,000 training samples

The error rate is 1.74% which is very close to the 1.72% reported using the project 1 classifier. This indicates that the Maximum Likelihood estimate using 10,000 is fairly accurate.

The estimated Maximum Likelihood parameters for 10,000 training samples are:

|  |  |  |
| --- | --- | --- |
|  |  | (35) |
|  |  | (36) |

As expected from the low error rate, these estimates are very close to the true parameters.

Figure 4 shows the classification results using 1,000 training samples. This case appears nearly identical to the 10,000 sample experiment with an even better error rate of 1.73%. The fact that the error rate is very slightly better, even though less training samples were used is due to the random nature of the sample data. This result appears to indicate that 1,000 samples is sufficient to estimate the likelihood distributions and using 10,000 training samples is only marginally better; if at all.

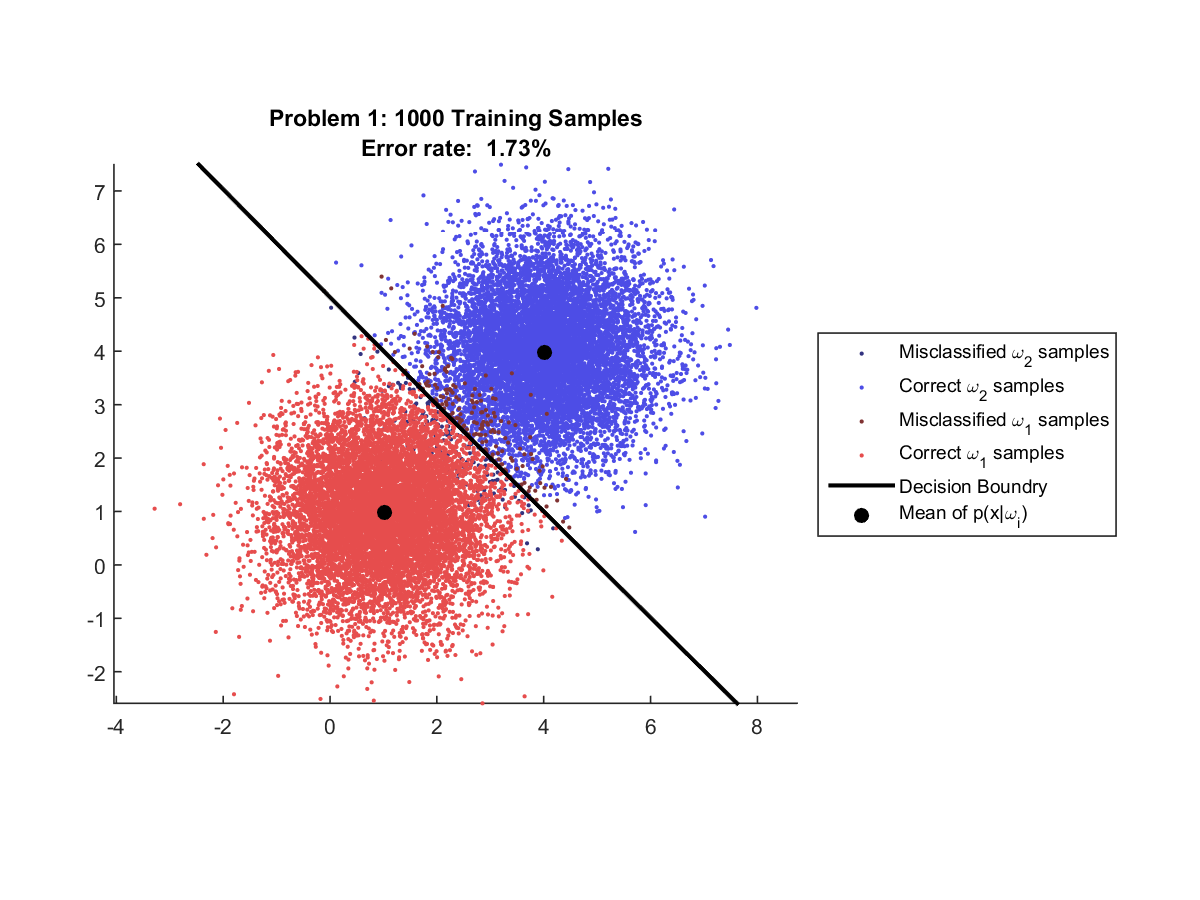


Figure 6: Classification with 1,000 training samples

The estimated Maximum Likelihood parameters for 1,000 training samples are:

|  |  |  |
| --- | --- | --- |
|  |  | (37) |
|  |  | (38) |

Surprisingly, even though the parameters estimated with 1,000 samples are further from the truth on average than the 10,000 sample experiment, the error rate is better. This again is probably due to the random nature of the data.

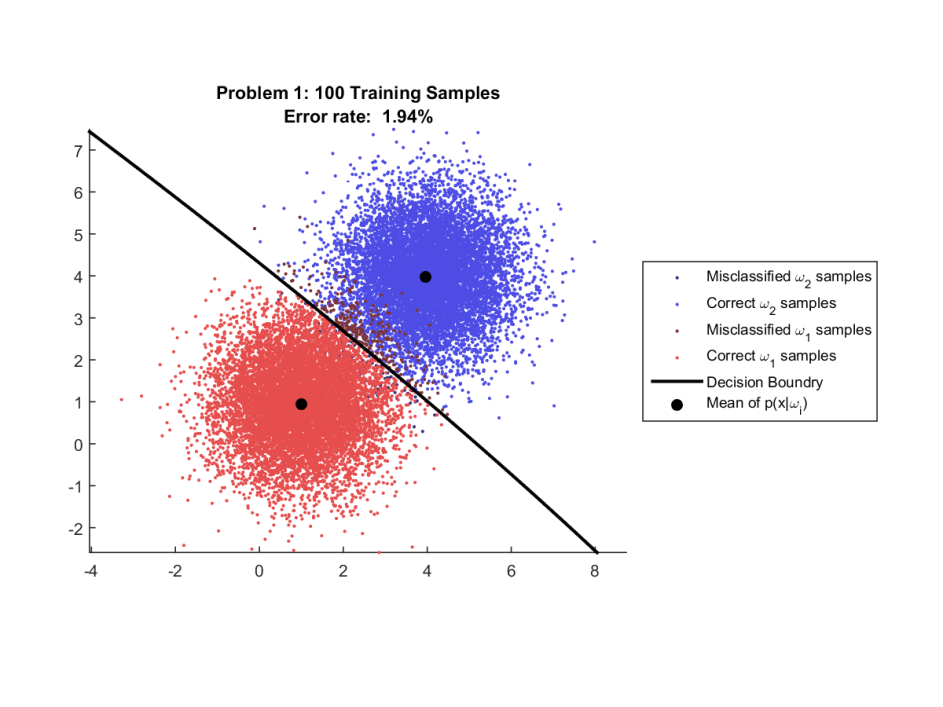
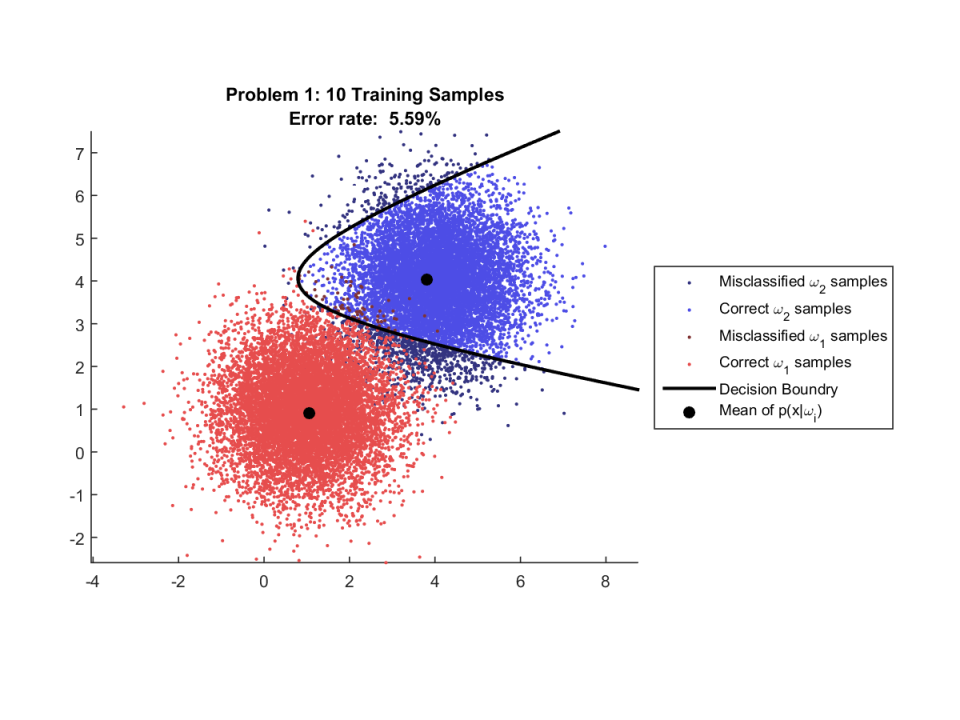
 

Figure 7: Classification with 100 (left) and 10 (right) training samples

In Figure 5 the classification results using 100 and 10 training samples can be seen. These experiments show the error rate clearly getting higher as the number of samples used for training decreases.

The estimated Maximum Likelihood parameters for 100 training samples are:

|  |  |  |
| --- | --- | --- |
|  |  | (39) |
|  |  | (40) |

In this experiment, the mean is fairly well estimated, but most of the error appears in the standard deviation estimate. This indicates that an accurate estimate of standard deviation requires more samples than the mean. This makes sense intuitively because the sample variance is calculated using the sample mean. Because of the dependency, errors in the mean will be compounded with the errors in the covariance estimate making the estimate less precise. Also, the covariance is composed of units squared which will make a direct comparison with errors in the less meaningful.

While 100 samples do not lead to a near-truth classifier, the results in this experiment are far from horrible and it is this authors opinion that 100 samples would be sufficient for certain real-world applications.

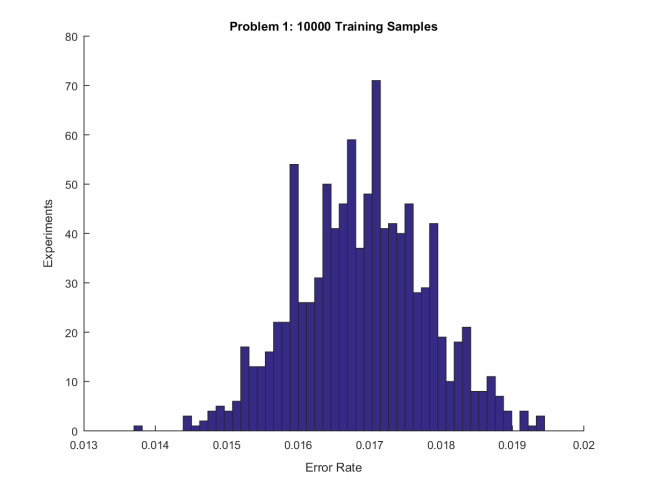
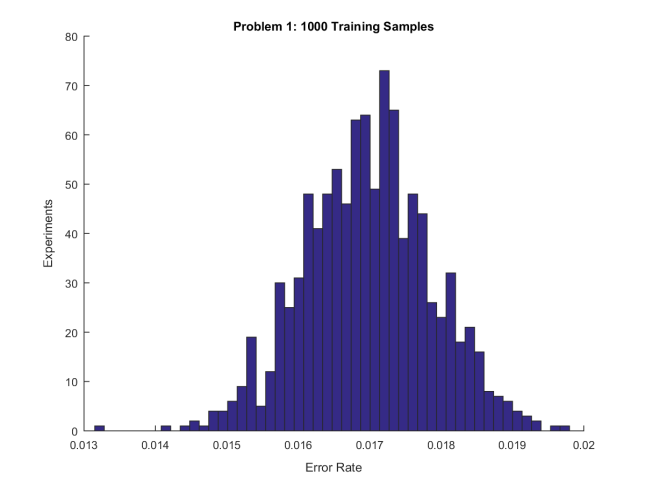
The estimated Maximum Likelihood parameters for 10 training samples are:

|  |  |  |
| --- | --- | --- |
|  |  | (41) |
|  |  | (42) |

The experiment using 10 training samples gives a very poor estimate of the distribution especially for the covariance estimate. This can be seen by the highly non-linear decision boundary constructed by the classifier. It appears that 10 samples is far from ideal and that generally more samples would be required to obtain an adequate estimate of the likelihood distribution.

#### Problem 1 Extra Experiments

Based on the experiments in Section 4.1.1 we can make certain assumptions about the necessary amount of training data for the simple two class classifier. However, it is not impossible to imagine that the results for the 10 sample experiment were unusually bad or unusually good. If we were to run the experiment again, how well would we expect the classifier to perform? The purpose of this experiment is to answer that question. In this section we run the entire training/classification problem 1000 times using different, randomly generated data sets consisting of 10,000 samples from each class.

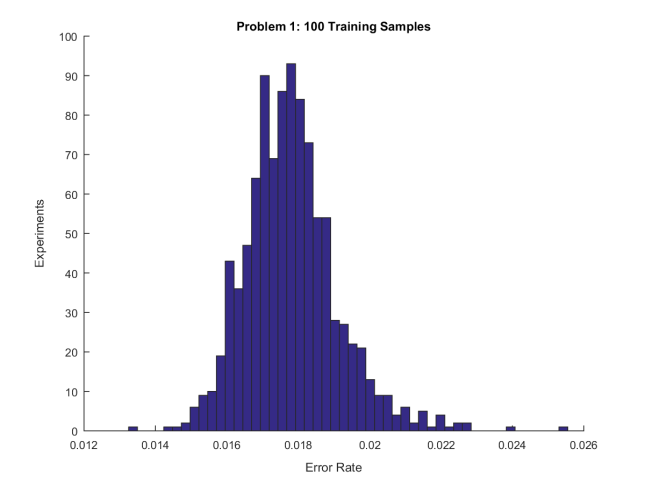
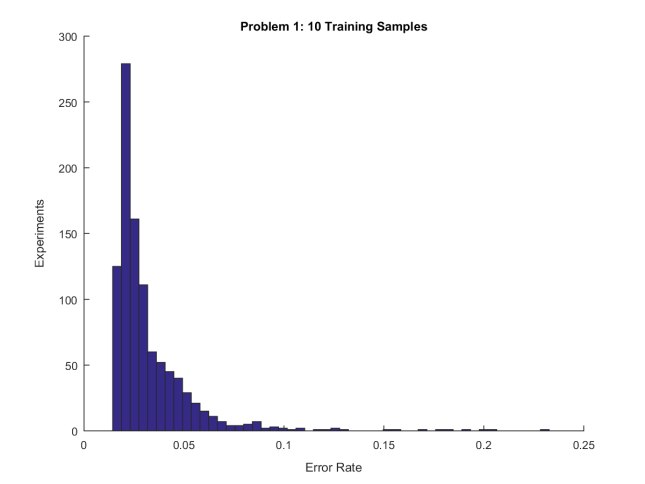
 

Figure 8: Distribution of Error Rates after 1000 experiments using 10,000 training samples (upper left), 1,000 training samples (upper right), 100 training samples (lower left), and 10 training samples (lower right).

From examining the histograms in Figure 6 we can see that the variance in the classification error when using only 10 training samples is very high, however the variance and expected error in the 100 training sample case is actually quite good. In fact the 100, 1,000, and 10,000 training sample cases appear to be nearly the same. The expected error in the 100 sample case is about 0.08% less than the 1,000 and 0.09% worse than the 10,000 training sample case but this seems to be marginal even with multiple orders of magnitude less in training data. Because this author was still curious about what a good amount of training data would be, we ran one more experiment.

In the final experiment we performed the classification experiment 1000 times but this time used only 3 training samples and worked our way up to 10,000 training samples, plotting the mean and standard deviation of the error rate for various amounts of training data.

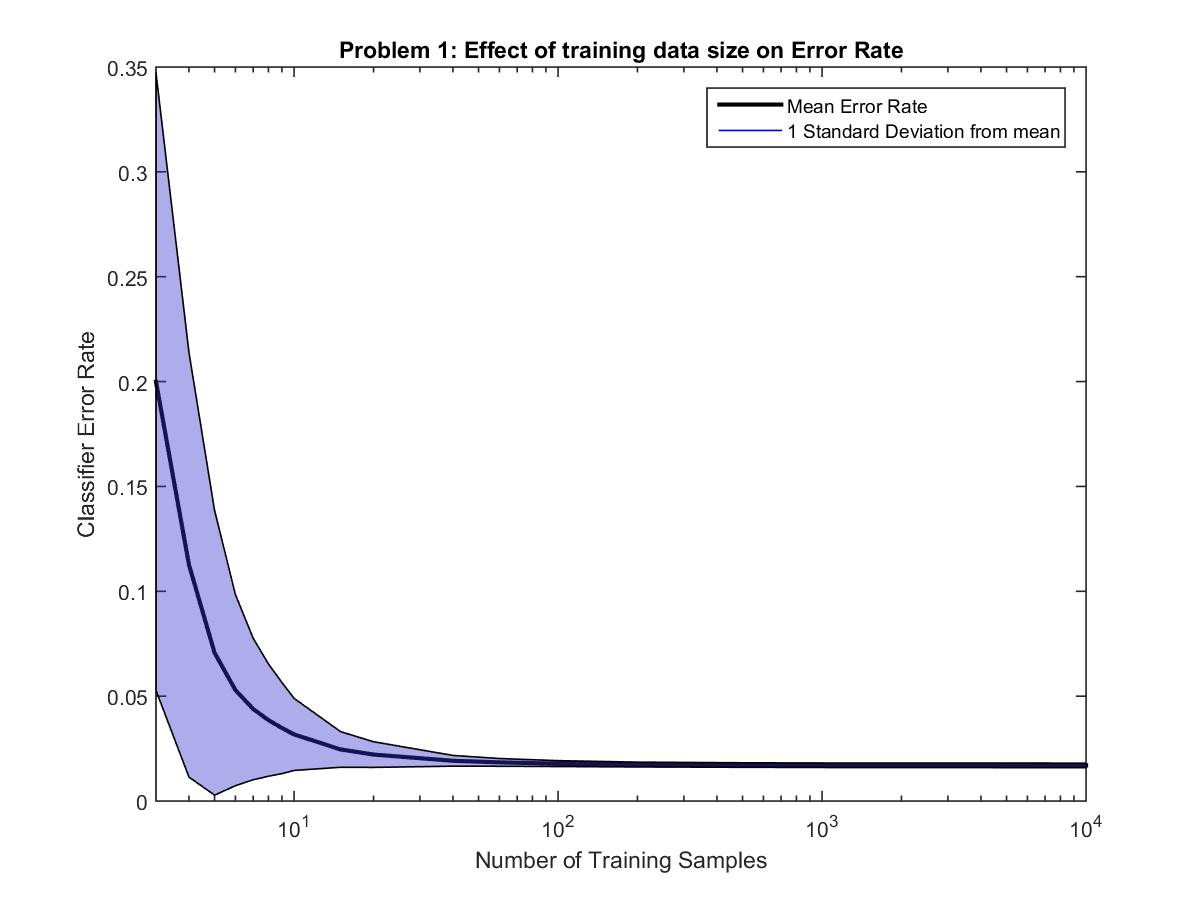


Figure 9: Effect of amount of training data on Error Rate

Figure 7 shows the results of the experiment. The black line indicates the mean error rate given some number of samples. The blue region is filled one standard deviation above and below the black line.

From the figure it is clear that 100, 1,000, and 10,000 training samples result in a very similar classification accuracy. However, from inspection we see that less than 10 training samples results in very poor performance, with 3 training samples having an average error rate of 20%. The accuracy of the classifier also seems to converge fairly quickly, with 20 or more samples giving surprisingly good results.

### Problem 2 (JG)

For Problem 2, we use the same samples used for each of the two classes from Project 1 Problem 2. The true distributions which the samples were taken from are defined as follows:

|  |  |  |
| --- | --- | --- |
|  |  | (43) |
|  |  | (44) |

The a-priori probabilities are as follows

|  |  |  |
| --- | --- | --- |
|  |  | (45) |

Figure 8 shows the classification results using 10,000 training samples. The decision boundary in this case appears to be nearly identical to the decision boundary from Project 1 show in Figure 2.

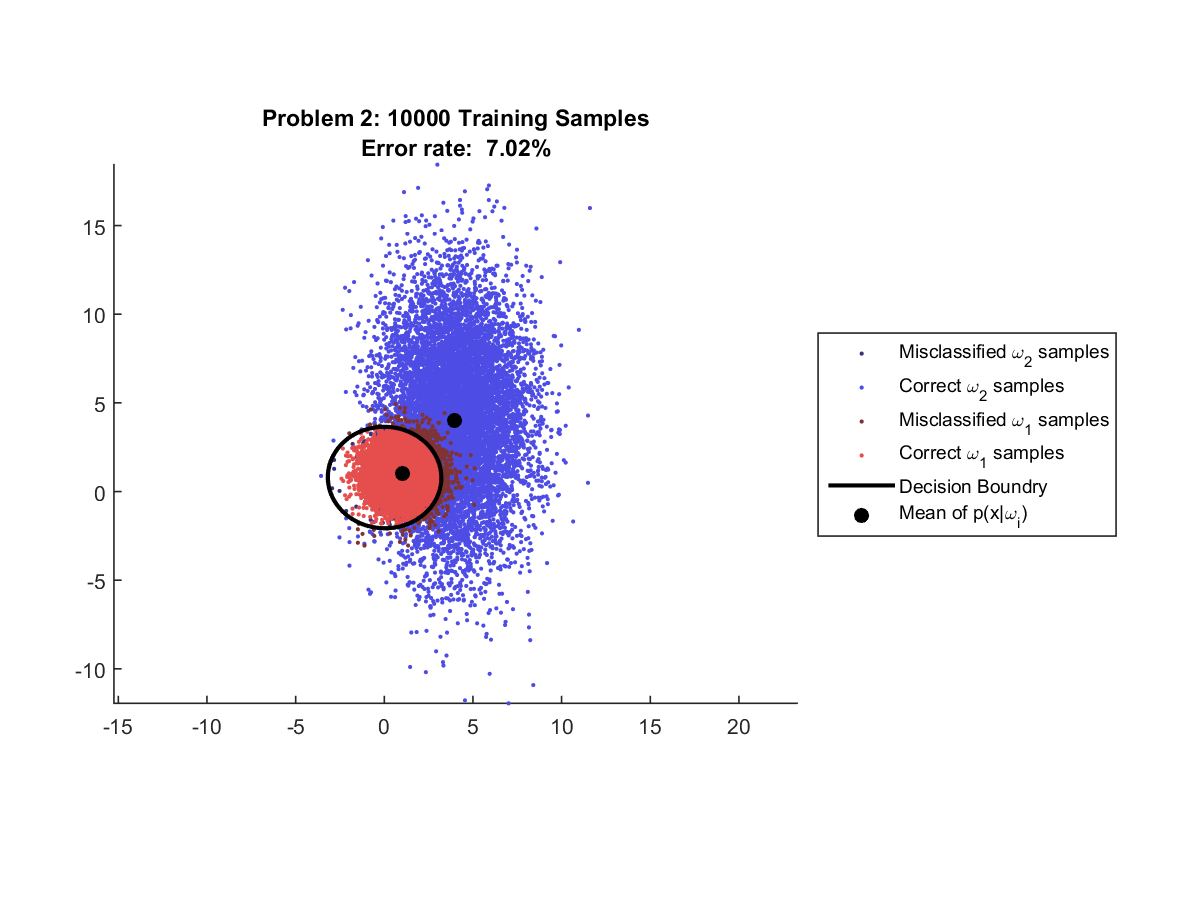


Figure 10: Classification with 10,000 training samples

The error rate is 7.02% which is actually better than the 7.03% reported using the project 1 classifier. This indicates that the Maximum Likelihood estimate using 10,000 is very accurate.

The estimated Maximum Likelihood parameters for 10,000 training samples are:

|  |  |  |
| --- | --- | --- |
|  |  | (46) |
|  |  | (47) |

As expected from the low error rate, these estimates are very close to the true parameters. It is worth noting that the mean for class two has more error than the mean for class 1. This is extra error is almost certainly due to the higher variance in class two.

Figure 9 shows the classification results using 1,000 training samples. Unlike the first experiment, the error rate here is somewhat larger than in the 10,000 sample experiment with an even better error rate of 7.22%. This behavior is discussed in more detail in Section 4.1.2.1.

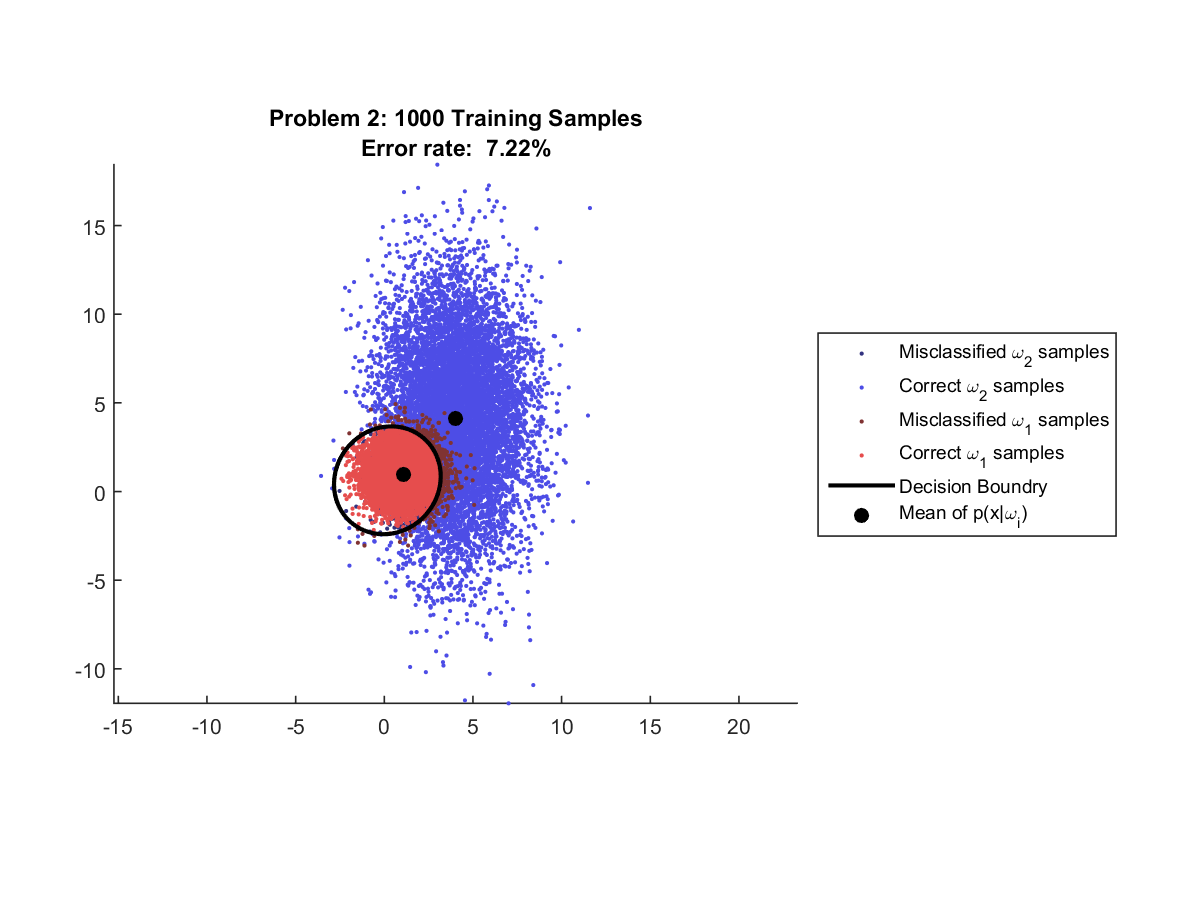


Figure 11: Classification with 1,000 training samples

The estimated Maximum Likelihood parameters for 1,000 training samples are:

|  |  |  |
| --- | --- | --- |
|  |  | (48) |
|  |  | (49) |

Similar to problem 1, the parameters estimated with 1,000 samples are further from the truth on average than the 10,000 sample experiment, however, in this case the error rate is noticeably worse.

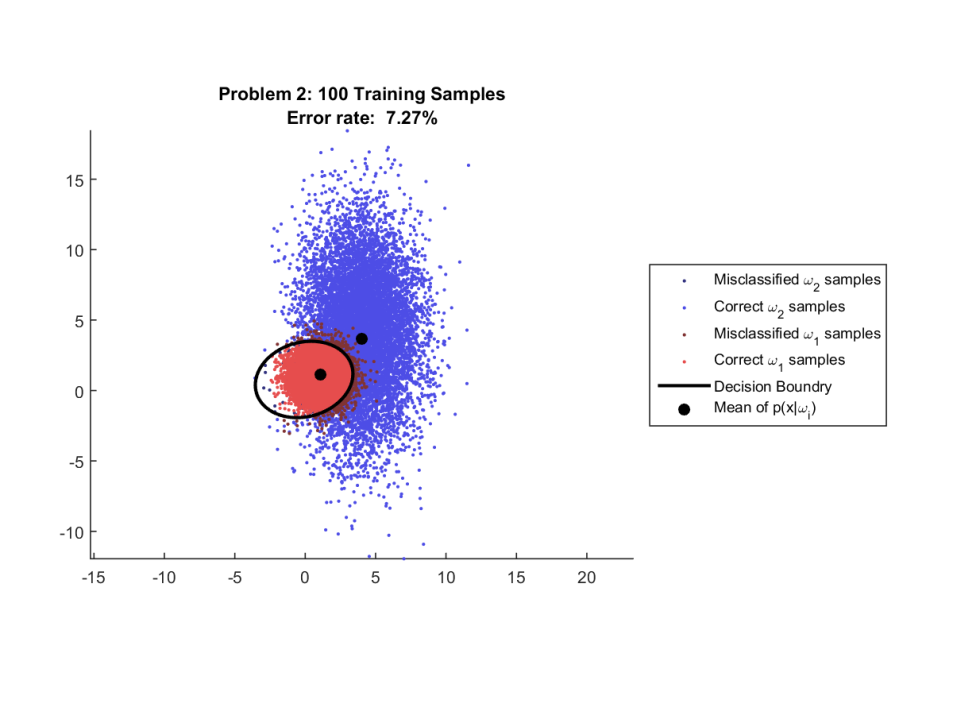
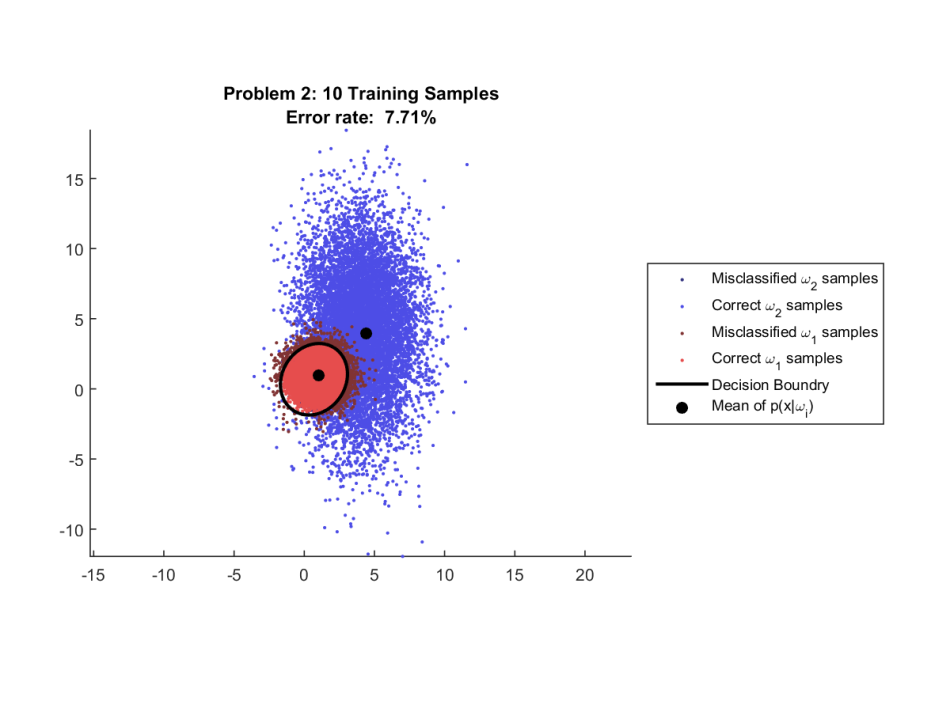
 

Figure 12: Classification with 100 (left) and 10 (right) training samples

In Figure 10 the classification results using 100 and 10 training samples can be seen. These experiments show the error rate clearly getting higher as the number of samples used for training decreases, however, the magnitude of the error rate increase is actually quite small compared to the drastic 3% increase in problem 1. This result seems to indicate that the number of training samples does not have as drastic of an affect on the classification results as in problem 1.

The estimated Maximum Likelihood parameters for 100 training samples are:

|  |  |  |
| --- | --- | --- |
|  |  | (50) |
|  |  | (51) |

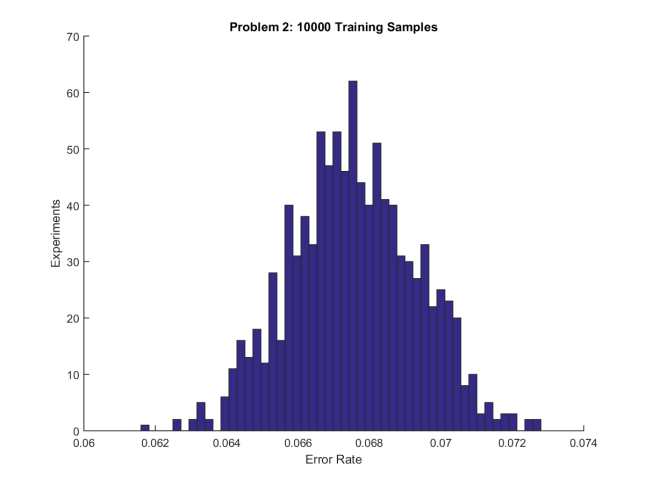
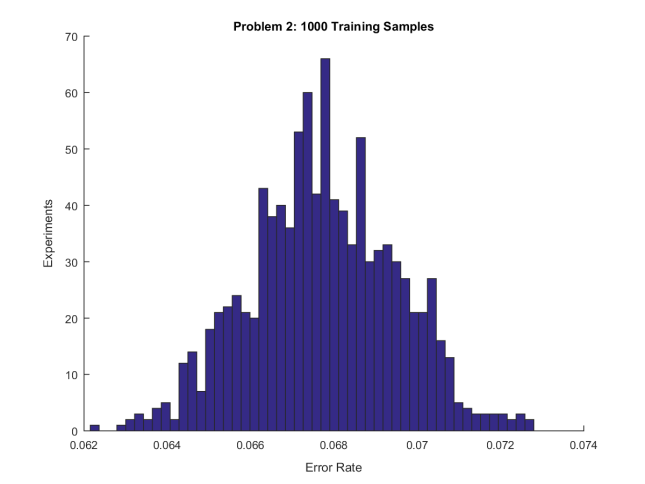
The estimated Maximum Likelihood parameters for 10 training samples are:

|  |  |  |
| --- | --- | --- |
|  |  | (52) |
|  |  | (53) |

The experiment using 10 training samples gives a very poor estimate of the distribution especially for the covariance estimate. However, unlike in problem 1, the decision boundary looks relatively sane. It appears that while 10 samples leads to a less accurate classifier, the difference is much less severe than the problem 1 case.

#### Problem 2 Extra Experiments

The same extra experiments performed in problem 1 (see 4.1.1.1) were also performed for problem 2. By running each of the training plus classification experiments 1000 times we can examine the stability of the classifier as a function of the amount of training data.

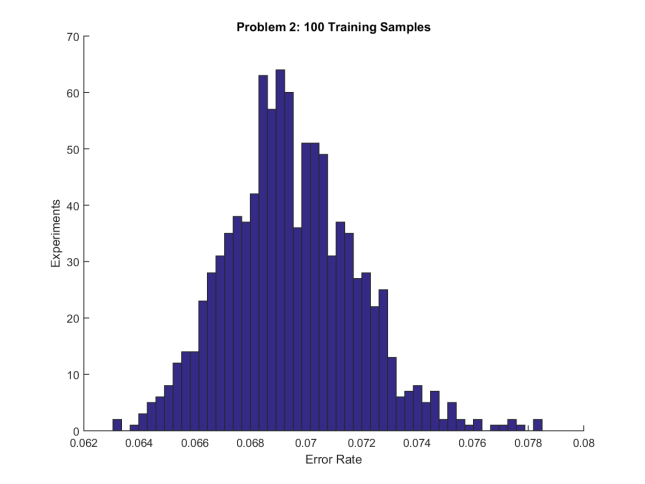
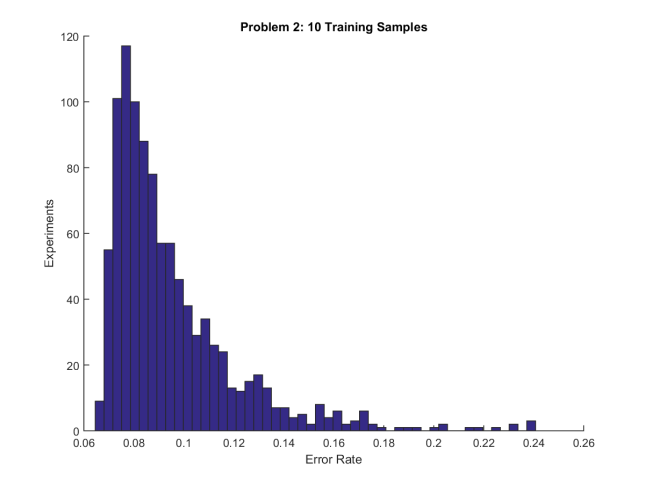
 

Figure 13: Distribution of Error Rates after 1000 experiments using 10,000 training samples (upper left), 1,000 training samples (upper right), 100 training samples (lower left), and 10 training samples (lower right).

From examining the histograms in Figure 11 we can see that the variance in the classification error when using only 10 training samples is very high, however the variance and expected error in the 100 training sample case is actually quite good. As in problem 1, the 100, 1,000, and 10,000 training sample cases appear to be nearly identical. The variance of the classifier error rate is double what it was in problem 1, even with 10,000 samples. This leads to the behavior of the 1,000 training sample experiment was noticeably worse than the 10,000 training sample experiment. From these histograms it would not be unexpected for the reverse to be true and the 1,000 training sample experiment could have noticeably outperformed the 10,000 training sample experiment. The expected error in the 100 sample case is about 0.18% less than the 1,000 and 0.19% worse than the 10,000 training sample case but this seems to be marginal even with multiple orders of magnitude less in training data.

In the final experiment we performed the classification experiment 1000 times but this time used only 3 training samples and worked our way up to 10,000 training samples, plotting the mean and standard deviation of the error rate for various amounts of training data.

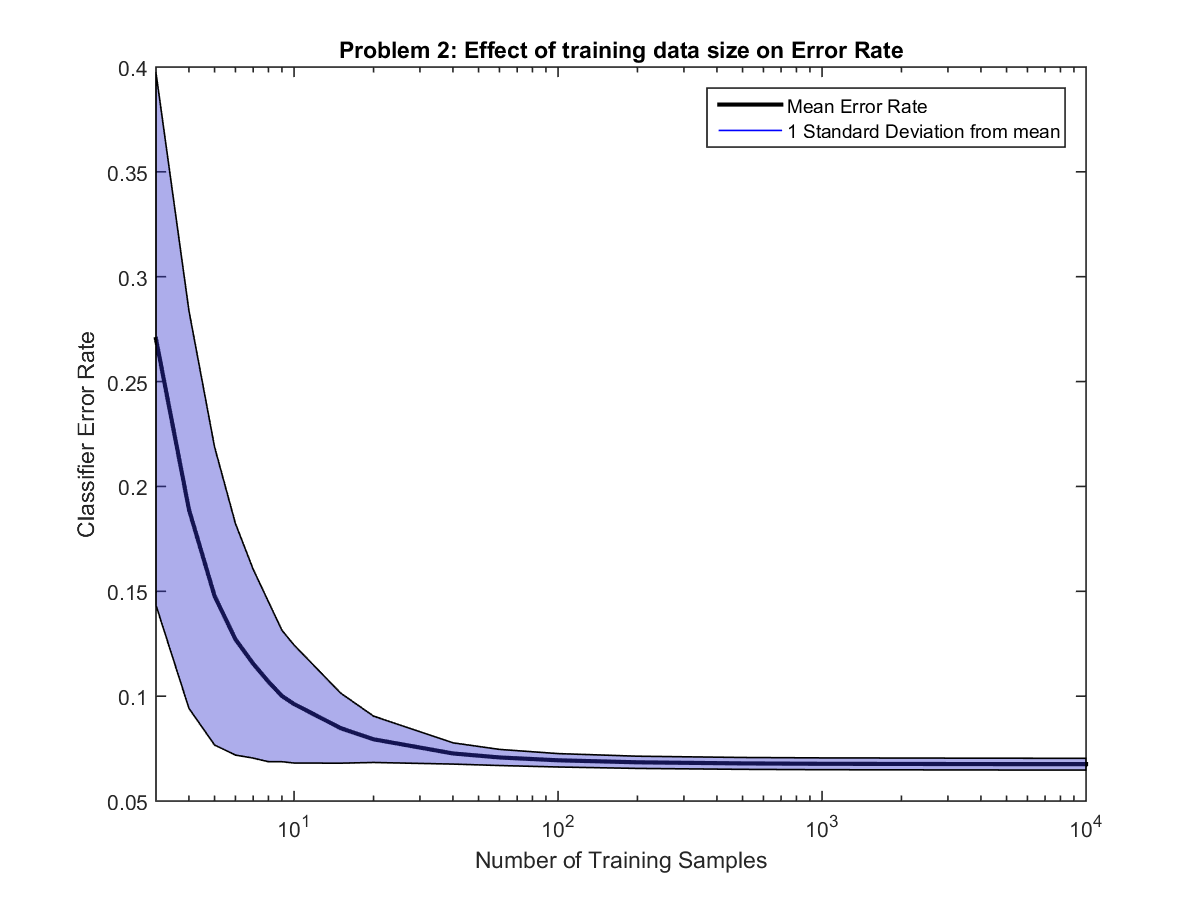


Figure 14: Effect of amount of training data on Error Rate

Figure 12 shows the results of the experiment. The black line indicates the mean error rate given some number of samples. The blue region is filled one standard deviation above and below the black line.

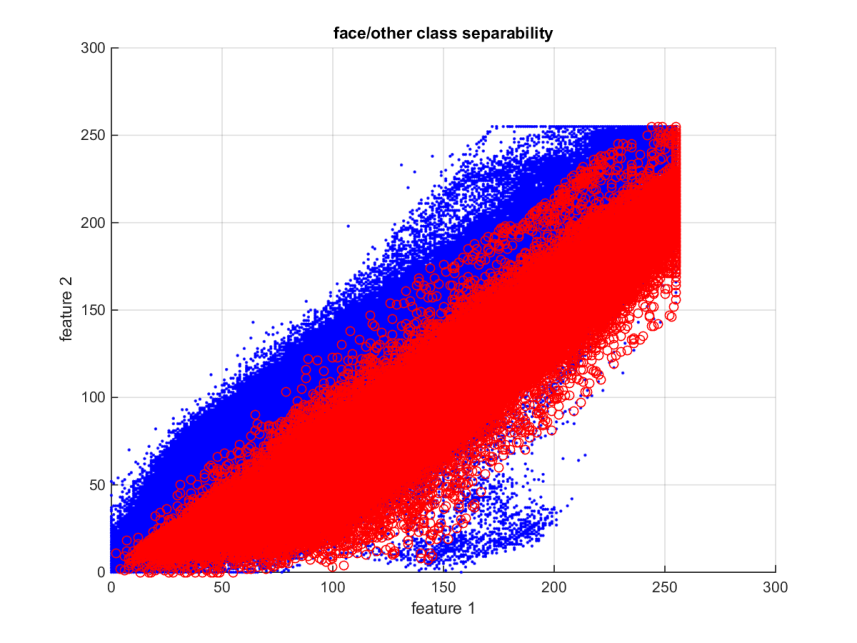
From the figure it is clear that 100, 1,000, and 10,000 training samples result in very similar classification accuracy. However, from inspection we see that less than 10 training samples results in poor performance, with 3 training samples having an average error rate of 27%. The accuracy of the classifier also seems to converge slightly more slowly than the classifier in problem 1.

## Face Detection (RP)

The face detection project is to train a classifer with one color image that has a number of human faces as class in the image along with many kinds of non-face background objects as class throughout the image, and from this one image, estimating and saving parameters for a classifier (assume Gaussian distribution for the classes) that is then tested with two other images with another set of faces. The performance of the classifiers is compared.

The features consist of RGB, chromaticity, and YCbCr color schemes, and we compare the performance of the features among all three color schemes. The assignment only requests that we compare chromaticity with YCbCr, but we use RGB features as a baseline to appreciate the impact of brightness or lightness on feature quality.

The face detector is designed using two different paradigms: the first paradigm bases the detector on a single class classifier and in the second paradigm bases the detector on a two class classifier. We chose to design both a one class classifier and a two class classifier because we found were curious how two class classifier would perform given that the background class is made of many objects, many classes, each with unique colors, reflective properties, and textures.

The single class classifier detector uses the Mahalanobis distance to perform the classification, and the two class classifier uses a Bayesian based dichotomizer as the detector assuming Gaussian features, as discussed in our paper for CS679 Project One[[7]](#footnote-9). The question we immediately had was whether or not the features were Gaussian as assumed.

We created feature files and then viewed all the features for the RBG, the chromatic, and the YCbCr color spaces, and clearly the features are not Gaussian for any of the above, see Figure 15. as an example of two of the features RGB from training image 1 (feature 1 is red and feature 2 is blue). Moreover, their class separation in Figure 13. The separation between the two classes appears to have considerable overlap poor and suggest would give poor detector performance.

Figure 15: The separation of red/blue features for the classes of faces/other.

In comparison, the CrCb features from training image 1 (feature 1 is Cr and feature 2 is Cb) are shown in Figure 14. The class separability appears better than for the two features in RGB and would suggest we should see better classifier performance.

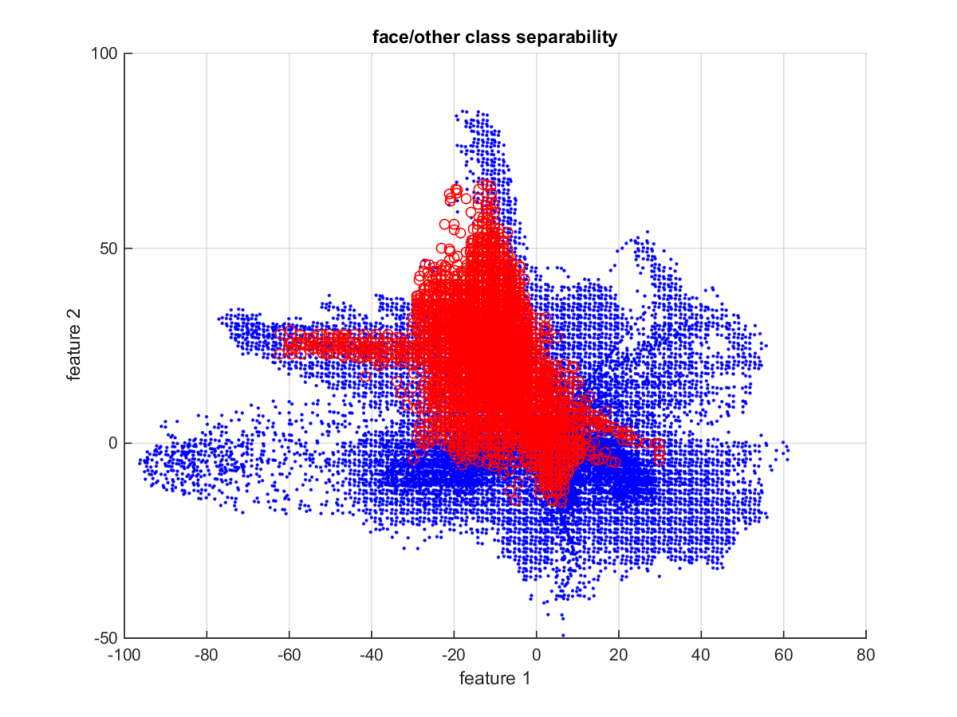


Figure 16: CrCb features plotted in scattergram

As a final consideration to apparent class separability, the class separability is better than shown. This is because in the training images there are some samples (albeit a small percentage) of skin on non-face regions, such as arms and hands, that are not highlighted as skin and are placed in the other category, and these samples of skin overlap with the skin features reducing the apparent separability of the classes.

In the output of the trained classifiers below, it is clear that the skin on the hands and arms are in the same region of the feature space as the skin on the faces of the people photographed. We note this because these non-face skin samples are detected as faces, and they increase the false positive rates which in turns misleads us as to the actual classifier performance. If the non-face skin samples were in the training for face samples, we would have better true positive rates and lower false negative rates. We discuss this topic in the following sections.

### Problem 3a (Chromatic Color Space) (RP)

#### One-class classifier results

#### Two-class classifier results

### Problem 3b (YCbCr Color Space) (RP)

#### One-class classifier results

#### Two-class classifier results

### Problem 3c (RGB Color Space) (RP)

#### One-class classifier results

#### Two-class classifier results

### Conclusion

# Conclusion

The Bayesian classifier for all test cases in this project produces results in which the error is minimized over all classifiers. This minimization occurs because the classifier was designed as a minimum risk classifier using a zero-one loss function.[[8]](#footnote-10) The classifier is known as a Bayesian classifier because the decision rule is based on estimating a-posteriori class probabilities using the Bayes rule that

|  |  |  |
| --- | --- | --- |
|  |  | (54) |

The resulting classifier is both a Bayes minimum risk classifier and minimum probability of error classifier.[[9]](#footnote-11)

# Contributors

Josh Gleason and Rod Pickens each wrote their own MATLAB software to perform the classification, the maximum likelihood estimation, the error estimation, the calculation of the Bayes error using false negative and false positive rates. The Bhatacharayya and Chernoff errors bounds were not included in this report.

Josh generated the maximum likelihood performance charts, and Rod generated the face detection performance charts.

Josh wrote the theory section on maximum likelihood along with the results section. Rod Pickens wrote the theory section on face detection along with the results section.

## Appendix

### Receiver operator characteristics

1. Duda, Richard O., Hart, Peter O. and Stork, David G, “Pattern Classification,” Wiley Interscience, Second Edition, page 25, equation 12. [↑](#footnote-ref-3)
2. Rabia Jafri and Hamid Arabnia, “A Survey of Face Recognition Techniques,” Journal of Information Processing Systems, Vol. 5, No. 2, June 2009. [↑](#footnote-ref-4)
3. Ibid, pages 46-47. [↑](#footnote-ref-5)
4. Jie Yang and Alex Waibel, “A Real-time Face Tracker,” School of Computer Science, Carnegie Mellon University, Pittsburgh, Pa., IEEE Applications of Computer Vision, 1996, WACV, 96, pages 142-147. [↑](#footnote-ref-6)
5. William K. Pratt, Digital Image Processing, Wiley-Interscience Publication, John Wiley and Sons, 1978, pages 59-61. [↑](#footnote-ref-7)
6. Jie Yang and Alex Waibel, “A Real-time Face Tracker,” School of Computer Science, Carnegie Mellon University, Pittsburgh, Pa., IEEE Applications of Computer Vision, 1996, WACV, 96, pages 142-147. [↑](#footnote-ref-8)
7. Josh Gleason and Rod Pickens, “Bayesian Classification for Classes with Features that are Normally Distributed,” Computer Science 679, February 23, 2015. [↑](#footnote-ref-9)
8. Ibid., page 26, Equation 19. [↑](#footnote-ref-10)
9. Ibid., page 27, Equation 21. [↑](#footnote-ref-11)