|  |
| --- |
| **Bayesian Classification for Classes with Features that are Normally Distributed** |
| **Josh Gleason and Rod Pickens** |
| **Programming Assignment 1** |
| **Computer Science 679 – Pattern Recognition, UNR, Dr. Bebis** |
| **Due: Monday February 23, 2015** |
| **Submitted: Monday February 23, 2015** |

Contents

[1 Abstract 3](#_Toc412293459)

[2 Project: Problems 1 and 2 3](#_Toc412293460)

[2.1 Problem 1 4](#_Toc412293461)

[2.2 Problem 2 5](#_Toc412293462)

[3 Results 7](#_Toc412293463)

[3.1 Problem 1a 7](#_Toc412293464)

[3.2 Problem 1b 8](#_Toc412293465)

[3.3 Problem 2a 10](#_Toc412293466)

[3.4 Problem 2b 12](#_Toc412293467)

[4 Conclusion 14](#_Toc412293468)

[5 Contributors 14](#_Toc412293469)

[6 Appendix 15](#_Toc412293470)

[6.1 Theory 15](#_Toc412293471)

[6.1.1 Minimum Risk Classifier 15](#_Toc412293472)

[6.1.2 Minimum Error Rate Classifier 16](#_Toc412293473)

[6.1.3 Error Bounds 18](#_Toc412293474)

[6.1.4 Discriminant Functions 19](#_Toc412293475)

[6.1.5 Decision Boundaries 20](#_Toc412293476)

[6.2 Procedure 21](#_Toc412293477)

[6.2.1 Independent Features 21](#_Toc412293478)

[6.2.2 Linear classifier 21](#_Toc412293479)

[6.3 The Gaussian Random Numbers 22](#_Toc412293480)

[6.3.1 Case 1: Want given : 22](#_Toc412293481)

[6.3.2 Case 2: Want given : 22](#_Toc412293482)

[6.3.3 Case 3: Want : 22](#_Toc412293483)

[6.3.4 Box-Mueller Method 23](#_Toc412293484)

[6.3.5 Goodness of Fit Test 24](#_Toc412293485)

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

# Abstract

This paper describes our research regarding the first 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, the classifiers used in the project, and the results of the classification. The topic discussed in the appendix are a description of the theory of a Bayesian classifier and the derivation of the two classifiers used in the project, the theoretical errors expected from a minimum risk Bayesian classifier with a zero-one loss function, as well as the generation of normal Gaussian random variables and the evaluation of the Gaussian random numbers using a Goodness of Fitness test.

# Project: Problems 1 and 2

The project is to design a minimum error Bayesian classifier to assign a collected sample from a population that contains two classes, where = 1 or 2, to one or the other classes within the population. The project states that the features to be used have a Gaussian distribution with known mean and known covariance matrix. Since the covariance matrices are defined as diagonal and the features are defined as a joint distribution, it is known that the features are independent.[[1]](#footnote-1) The a-priori probabilities of each class are likewise specified.

The project does not explicitly specify costs for correct and incorrect decisions, so we do not build costs into the classifier. However, the costs can be inferred as coming from a zero-one loss function, or symmetric loss function, because the assignment states that the classifier be minimum error.[[2]](#footnote-2) The derivation of the minimum risk and minimum error classifiers are discussed in the appendix Sections 6.1.1 and 6.1.2.

The classification is performed using two measurements, or features, from each sample. The samples are to be generated using random number generators, and the approach we used to generate the features is described in the Appendix.

The project asks that 10,000 samples be generated for each class and . The features for each class have 2D joint probability distributions.

Problem 1 is to run the following experiment twice with different a-priori class probabilities:

1. Design Bayes classifier for minimum error.
2. Plot the Bayes decision boundary together with the generated samples.
3. Classify the samples by the classifier and count the number of misclassified samples.
4. Plot the Chernoff bound as a function of β and find the optimum β for the minimum.
5. Calculate the Bhattacharyya bound.

Problem 2 is to again repeat the above steps, but with different feature density functions. These problems are discussed below.

## Problem 1

For Problem 1a, the a-priori class probabilities should be set such that , and for Problem 1b, the a-priori class probabilities should be set such that and

The normal probability distribution for the features is defined as follows:

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

And normal probability distribution for the features is defined as follows

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

Note that this description of features corresponds to designing a classifier that corresponds to Case 1[[3]](#footnote-3) in Duda, Hart, and Stork where . The cases are discussed further in Section 6.1.5.

The classifier chosen for Problem 1a is a dichotomizer because it is defined for a two-class decision problem in which the features are uncorrelated and have identical covariance matrices for both classes.[[4]](#footnote-4) The dichotomizer is defined as the following:

|  |  |  |
| --- | --- | --- |
|  | = 0 | (3) |

Because the features are independent, normally distributed and have identical covariance matrices for each class, the dichotomizer reduces to the following discriminant function

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

Where

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

And

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

## Problem 2

Problem 2 is to rerun the above experiment except that the probability distribution for the features changes. The features for are defined as follows

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

And the features for are defined as follows

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

Since the covariance matrix for each class is different, the classifier used in the Problem 2 is based on the classifier defined in Duda, Hart, and Stork in Section 2.6.3 where Given these facts, the discriminant function chosen for each class is defined as follows:

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

Where

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

and

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

and

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

For Problem 2a, the a-priori class probabilities should be set such that , and for Problem 2b, the a-priori class probabilities should be set such that and The a-priori probabilities are the same as in Problem 1.

# Results

## Problem 1a

For Problem 1a, we generate a plot of the ten-thousand samples from two classes along with the decision boundary that separates the two classes. Note that the distributions for the features of each class are defined as follows:

|  |  |  |
| --- | --- | --- |
|  |  | (13) |
|  |  | (14) |

The a-priori probabilities are as follows

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

In Figure 1, 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 clearly an affine function (a linear function with an additive intercept). The Bayes error is 1.76 percent.

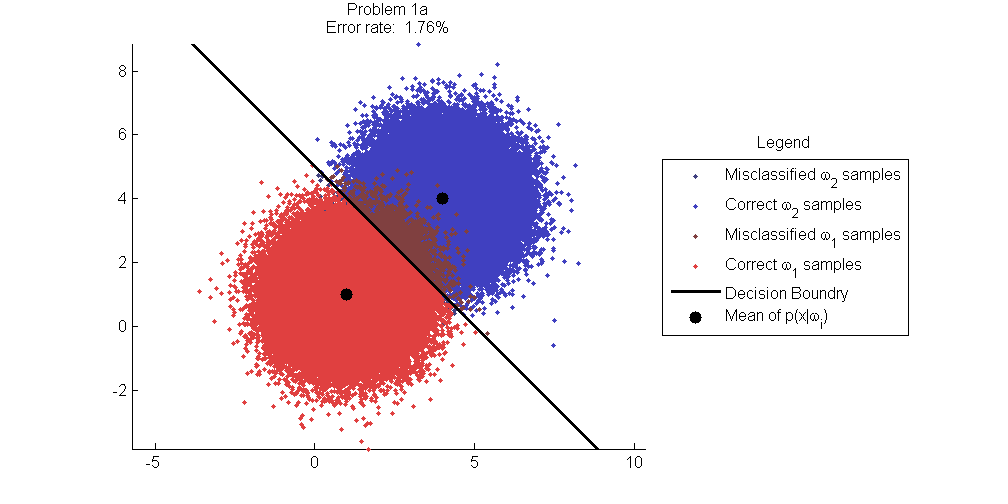


Figure : Classification results for Problem 1a.

The Bayes error (notes as true error in the Figure) and the Chernoff and Bhattacharyya error bounds for the classification are given in Figure 2. The true error at 1.69% is significantly lower than are either the Chernoff and Bhattacharyya error bounds, both at 5.27%.

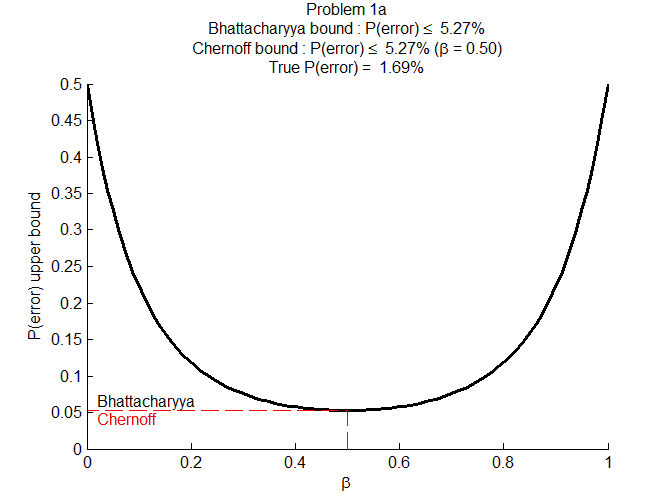


Figure : Chernoff and Bhattacharyya bounds for Problem 1a.

## Problem 1b

For Problem 1b, we again generate a plot of the ten-thousand samples from two classes along with the decision boundary that separates the two classes. Note that the distributions for the features of each class are defined as follows:

|  |  |  |
| --- | --- | --- |
|  |  | (16) |
|  |  | (17) |

The a-priori probabilities are different and are set as follows

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

In Figure 3, the two classes are colored differently: class 1 (red dots) is and class 2 (blue dots) is Note that the distribution of the samples are identical to the samples in Problem a, which they should identically resemble unless a different random seed was used which was not. The decision boundary between the two classes is again clearly an affine function (a linear function with an additive intercept). The Bayes error is 1.56%. This improvement over Problem 1a is due to the fact that class 2 is more likely than class 1 .

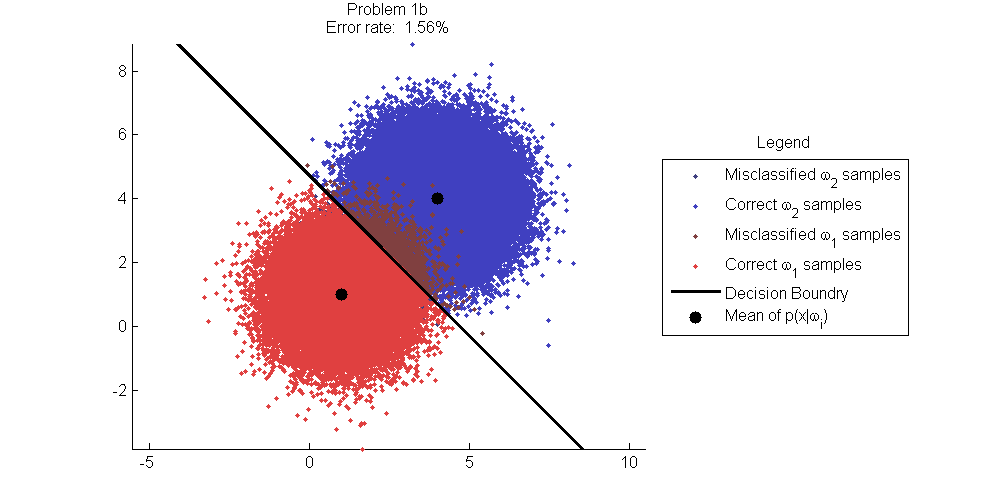


Figure : Classification results Problem 1b.

The Bayes error (labeled as true error) and the Chernoff and Bhattacharyya error bounds for the classification are given in Figure 4. The true error is 1.53% and is again significantly lower than are either the Chernoff and Bhattacharyya error bounds; the Chernoff at 4.73% and the Bhattacharyya at 4.83%. This difference in error bounds matches our expectations in that Bayes error Chernoff error Bhattacharyya.

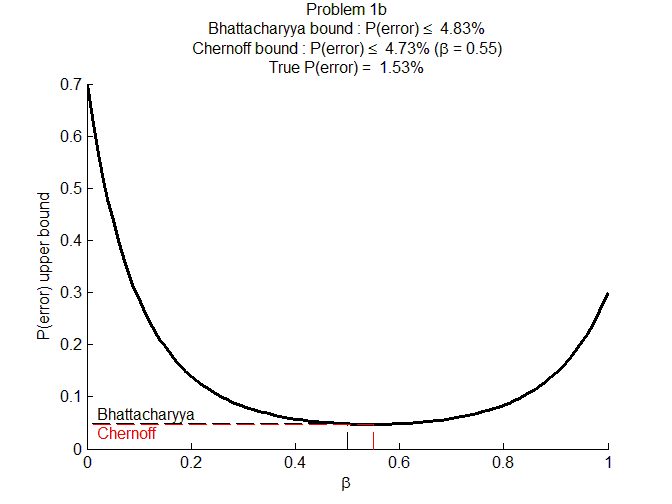


Figure : Chernoff and Bhattacharyya bounds for Problem 1b.

## Problem 2a

For Problem 2a, we again generate a plot of the ten-thousand samples from two classes along with the decision boundary that separates the two classes. Note that the distributions for the features of each class are defined as follows:

|  |  |  |
| --- | --- | --- |
|  |  | (19) |
|  |  | (20) |

The a-priori probabilities are as follows

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

In Figure 5, the two classes are colored differently: class 1 (red dots) is and class 2 (blue dots) is Note that the distribution of the samples are identical to the samples in Problem 1a, which they should identically resemble unless a different random seed was used which was not. The decision boundary between the two classes is clearly an ellipse. The Bayes error is 6.71 percent.

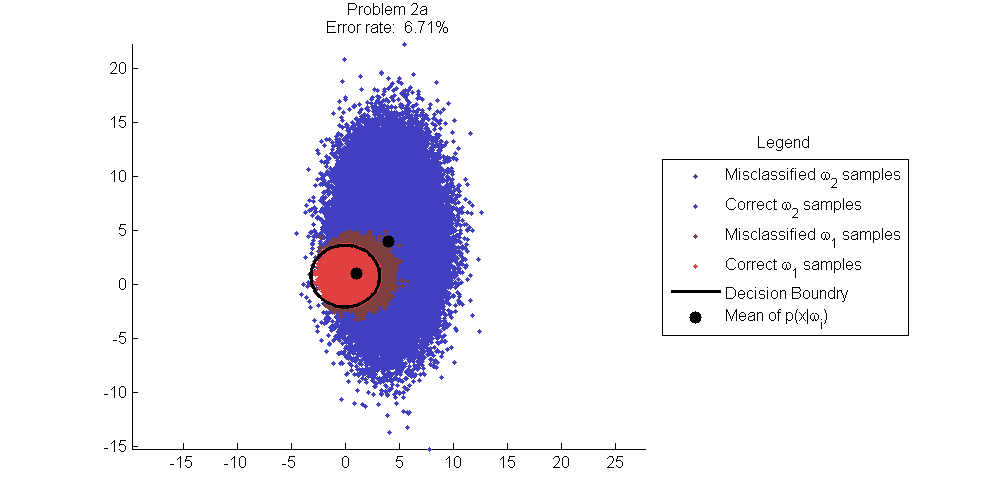


Figure : Classification results Problem 2a.

The Bayes error (labeled as true error) and the Chernoff and Bhattacharyya error bounds for the classification are given in Figure 6. The true error is 6.76% and is again significantly lower than are either the Chernoff and Bhattacharyya error bounds; the Chernoff at 14.78% and the Bhattacharyya at 17.14%. This difference in error bounds matches our expectations in that Bayes error Chernoff error Bhattacharyya. Also, there is considerable overlap in the feature spaces between the two classes, and we would expect larger errors than observed in Problems 1a and 1b.

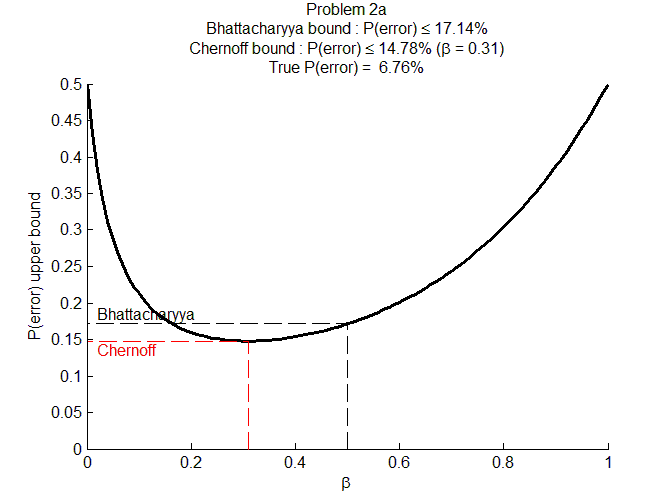


Figure : Chernoff and Bhattacharyya bounds for Problem 2a.

## Problem 2b

For Problem 2b, we again generate a plot of the ten-thousand samples from two classes along with the decision boundary that separates the two classes. Note that the distributions for the features of each class are defined as follows:

|  |  |  |
| --- | --- | --- |
|  |  | (22) |
|  |  | (23) |

The a-priori probabilities are as follows

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

In Figure 7, the two classes are colored differently: class 1 (red dots) is and class 2 (blue dots) is Note that the distribution of the samples are identical to the samples in Problem 1a, which they should identically resemble unless a different random seed was used which was not. The decision boundary between the two classes is clearly an ellipse. The Bayes error is 7.35%.

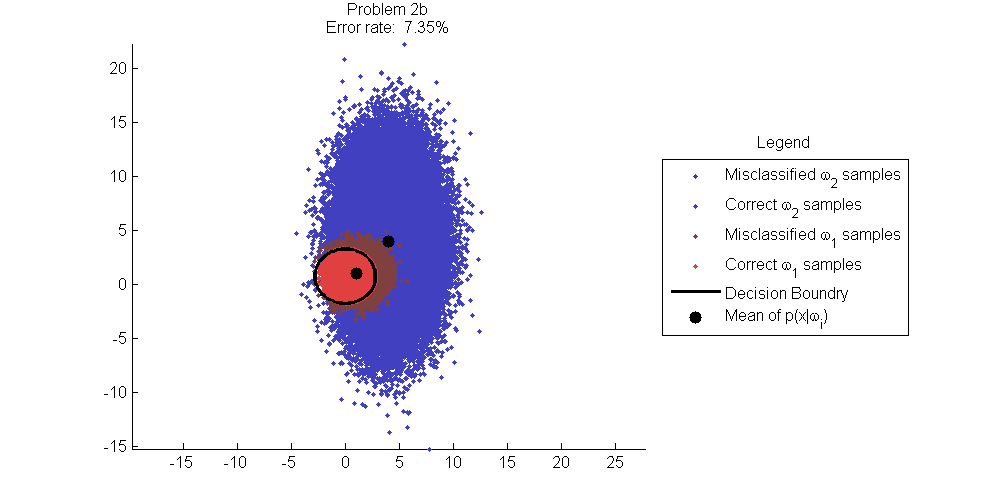


Figure : Classification results Problem 2b.

The Bayes error (plotted as the true error in the Figure) and the Chernoff and Bhattacharyya error bounds for the classification are given in Figure 8. The true error is 6.76% and is again significantly lower than are either the Chernoff and Bhattacharyya error bounds; the Chernoff at 15.31% and the Bhattacharyya at 15.71%. This difference in error bounds matches our expectations in that Bayes error Chernoff error Bhattacharyya. Also, there is considerable overlap in the feature spaces between the two classes, and we would expect larger errors than observed in problem 1a and 1b. As observed early, overall errors are smaller than in problem 2a because class 2 is favored probabilistically over class 1 (the a-priori probabilities differ).

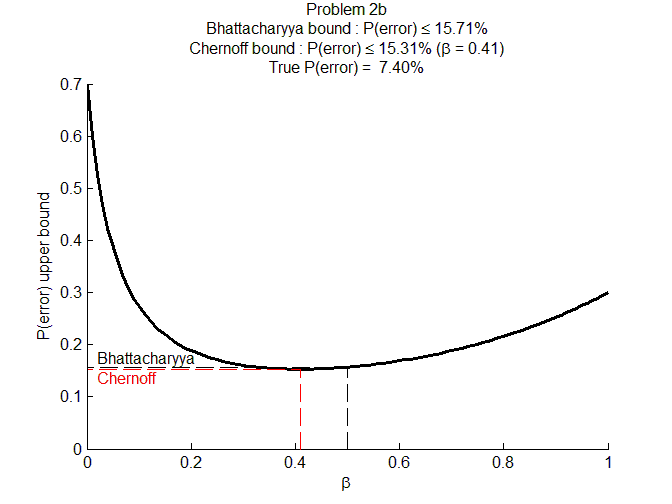


Figure : Chernoff and Bhattacharyya bounds Problem 2b.

# 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.[[5]](#footnote-6) 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

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

The resulting classifier is both a Bayes minimum risk classifier and minimum probability of error classifier.[[6]](#footnote-7)

# Contributors

Josh Gleason and Rod Pickens each wrote their own MATLAB software to perform the classification, the error estimation, the calculation of the Bayes error, the Chernoff error bound, and the Bhattacharyya error bound. The plots used in this report were generated by Josh Gleason’s software. Josh and Rod wrote different sections of the project report. Josh Gleason wrote the theory section on minimum error classification, error bounds, discriminant functions, and decision boundaries. Rod Pickens wrote the theory section on minimum risk classification, the procedure section on selecting the classifiers to use, the results section from the experiments, and the approaches used to generating the random numbers along with a short description of the Box-Muller transform, and evaluated the performance of the Box-Muller method of generating Gaussian random numbers by applying the Goodness of Fit test to the random number and documented the conclusion of this work in the Appendix.

# Appendix

## Theory

In pattern recognition a classifier is a machine that makes a decision among candidate choices given evidence conditioned on the choice under consideration, if available. When evaluating the conditioned evidence particular to a choice, the machine makes a decision on which candidate to select based on minimizing the overall risk for all actions choices or minimizing the probability of error, both based on the outcome of the choice. Without conditioned evidence, a classifier will make the choice based on the most likely choice.

Conditioned evidence is described as an n-dimensional mathematical vector

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

Where ‘ represents the transpose of the vector. Each dimension of the vector

is a measurement of one aspect of the object being evaluated prior to that object being classified, and each measurement is defined as one of n features for that object.

### Minimum Risk Classifier

The approach used in this paper to designing the classifier for the class project is to derive the classifier by minimizing the overall risk and then setting the costs associated with the risk in such a manner as to minimize the probability of error.

The overall risk is defined in Hart[[7]](#footnote-8) as

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

The risk R corresponds to the expected value of the conditional risk function ) given we know the probability density of the evidence . The R is the average because by definition the expected value of a function is given as

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

The action taken by the classifier in Equation (26) is a function of the evidence. Assuming a finite number of discrete actions , the risk R is minimized by computing the conditional risk for each action

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

Where is the loss function associated with action when the true state of nature is . The machine choses the action for which is minimum.

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

The resulting overall risk is called Bayes risk.[[8]](#footnote-9) The class selected given the machine choice depends upon the mapping from action to outcome.

For a two-category classification, the conditional risk derived from Equation (28) is

|  |  |  |
| --- | --- | --- |
|  |  | (30) |
|  |  | (31) |

With the two-category classification described in Equations (30) and (31), Equation (29) is reduced to

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

### Minimum Error Rate Classifier

A special case of classifier is the minimum error classifier. This type of classifier defines the decision rules such that the average probability of error is minimized. The average probability of error is described by Hart[[9]](#footnote-10) as

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

where is the probability of error given a particular observation . It is clear from Equation (33) that minimizing for all will minimize the average probability of error .

In general, the decision rule is then defined as follows.

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

The probability of error is simply the probability that a decision is wrong, which may be stated as

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

where is the probability of the complementary event.

Let action be the decision that the true state of nature is and Equation (34) becomes

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

The constant may be removed from the minimization and Equation (36) becomes a maximization describing the simple decision rule.

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

For the two class case the decision rule becomes

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

and, from Equation (35), the minimized probability of error becomes

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

Substituting the minimized probability of error into the average error formula leads to the following definition of minimum error for the two class case

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

It is important to point out an assumption that was made during the derivation of the minimum error classifier which was that no alternative decisions were addressed such as refusing to classify. This is because decisions such as refusing to classify an observation may be interpreted as an error, making the probability of error maximal (i.e. ) which means that alternative actions may be disregarded.

Now that the decision rule for the minimum error classifier has been derived, it needs to be related to the more general minimum risk classifier. To form this relation, the following cost function known as the symmetrical or zero-one loss function[[10]](#footnote-11) is considered

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

Using Equation (28) the application of the zero-one loss function leads to the following conditional risk

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

And the general decision rule from Equation (29) becomes

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

which is identical to the Equation (37) decision rule for the minimum error rate classifier. This leads to the conclusion that using the zero-one loss for a minimum risk classifier yields a minimum error classification.

### Error Bounds

This project also calls for the computation of error bounds which compute an upper bound for the average probability of error for the classifiers. The error bounds used and discussed here are known as the Chernoff and Bhattacharyya bounds.

The bounds are derived for the two case minimum error rate classifier using the following inequality described in Hart[[11]](#footnote-12)

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

By applying Bayes rule to Equation (40) the average probability of error may be expressed as follows

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

Applying the inequality in Equation (44), the following inequality is established

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

Minimizing the inequality in Equation (46) for the argument yields the Chernoff bound, and simply using gives the slightly looser Bhattacharyya bound.

For the special case where the likelihood probabilities are normal, the inequality becomes

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

Where[[12]](#footnote-13)

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

### Discriminant Functions

In many cases, there are ways to simplify the decision, rather than computing the posterior probability directly. A discriminant function is a where the following decision rule minimizes the overall risk

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

For example

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

is a valid discriminant function (derived from Equation (29))

An important property of discriminant functions is that if all discriminate functions are transformed by the same monotonically increasing function, then the minimization in Equation (49) is maintained. That is to say

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

The discriminate functions for the minimum error classifier may be expressed as follows

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

where

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

The discriminate functions in Equation (53) are particularly useful for the case where the features are sampled from a normal distributions.

### Decision Boundaries

Decision boundaries for a classifier are locations in feature space where two or more discriminate functions are equal. The decision boundary for a two class minimum error classifier with normal likelihoods is considered further in this section.

Case 1:

In the case where all the features are independent, have the same variance, and each class has the same covariance, the decision boundary is a hyperplane normal to the vector . If the prior probabilities for both distributions are equal then the hyperplane intersects the midpoint directly between the two means. If the priors are not equal then, the hyperplane is shifted away from the midpoint in the direction opposite of the more likely class.

Case 2:

In this case, the features for both classes have the same covariance matrix and the decision boundary is still linear, however it is not generally normal to the line connecting the means.

Case 3:

This is the general case and the decision boundary is a hyperquadric surface which is generally non-linear.

## Procedure

The class project is to design a Bayesian classifier for a two class problem in which the feature vectors for each class are 2-dimensional normal Gaussian random variables. The consequent classifier is a dichotomizer which is a machine that decides between two classes.

### Independent Features

In the class project description, Dr. Bebis defined the features to be independent, and with this, the resulting classifier is a linear machine as will be discussed in the next section.

### Linear classifier

For the first assignment, The class means for each class I = 1 or 2 are 2-dimensional random variables and , and the class covariance matrix for each class i = 1,…,2 will be . Given this, the derivation of the classifier will follow that of Hart[[13]](#footnote-14) for the case when the class covariance matrices are . Also, with this information, we decided upon using a discriminant function for each class of the functional form

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

We combine the two discriminant functions into a single discriminant function, known as a dichotomizer, defined as

|  |  |  |
| --- | --- | --- |
|  |  | (55) |

The decision boundary for the dichotomizer between the two classes is defined when equals zeros, or as follows

|  |  |  |
| --- | --- | --- |
|  |  | (56) |

For the above classification problem, the dichotomizer reduces to the discriminant function

|  |  |  |
| --- | --- | --- |
|  |  | (57) |

where

|  |  |  |
| --- | --- | --- |
|  |  | (58) |

and

|  |  |  |
| --- | --- | --- |
|  |  | (59) |

The classifier decision boundary corresponds to a line in 2D space with the boundary between the classes being perpendicular to the vector that connects the means of the two classes.

## The Gaussian Random Numbers

The features of the classes are defined to be independent normal Gaussian random variables where independency is defined as follows:

|  |  |  |
| --- | --- | --- |
|  |  | (60) |

We discuss three approaches to generating the independent normal random feature vectors given random number generators that generate numbers with and , viz. , or with and , viz. :.

### Case 1: Want given :

This case describes the random number generator that allows a user to define the mean , the standard deviation , and the number n of samples to return. This type of random number generator is called with syntax similar to the following:

|  |  |  |
| --- | --- | --- |
|  |  | (61) |

### Case 2: Want given :

This case describes the random number generator that produces only standard normal random numbers, or random numbers that have and . To produce n random numbers with mean and standard deviation , this type of random number generator is called with syntax as follows

|  |  |  |
| --- | --- | --- |
|  | i = 1 or 2 | (62) |

### Case 3: Want :

This case describes the production of random numbers using the concepts from linear algebra and the application of coloring and whitening transforms. The approach to be developed finds the linear transformation A that converts the random vector to the random vector as follows

|  |  |  |
| --- | --- | --- |
|  |  | (63) |

such that the linear operator A is constrained to leave the expectation of the outer product of equal to a semi-positive definite, symmetric matrix

|  |  |  |
| --- | --- | --- |
|  |  | (64) |

It turns out that the matrix A is the inverse of the matrix that uncorrelates, or whitens, a set of correlated random variables as follows

|  |  |  |
| --- | --- | --- |
|  |  | (65) |

Where is constrained as follows

|  |  |  |
| --- | --- | --- |
|  |  | (66) |

The resulting whitening matrix is defined as follows

|  |  |  |
| --- | --- | --- |
|  |  | (67) |

is an nxn matrix of the eigenvectors of the matrix and the matrix are the eigenvalues of the matrix . The approach to generating the random variables are to transform white normal Gaussian random variables as follows:

|  |  |  |
| --- | --- | --- |
|  |  | (68) |

Where the vector is generated using one of the above methods in Case 1 or Case 2 such as

|  |  |  |
| --- | --- | --- |
|  |  | (69) |

### Box-Mueller Method

The Box-Muller transform is a clever algorithm, in these authors opinion, for generating pseudo-random Gaussian numbers. Figure 9[[14]](#footnote-15) found on Wikipedia visually describes the algorithm developed by George Box and Mervin Muller in 1958 to generate Gaussian random numbers.

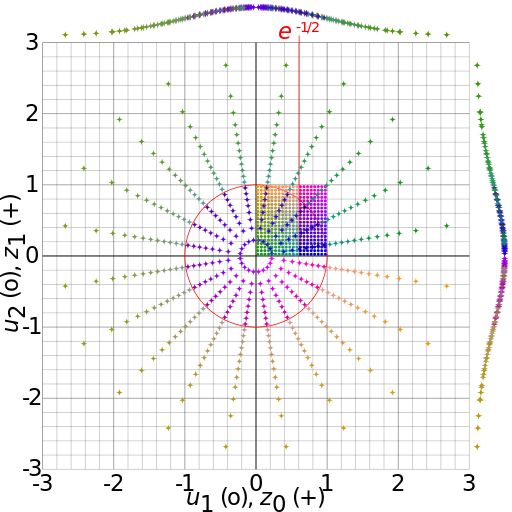


Figure : Box-Muller Diagram

As illustrated in Figure 9, the algorithm uses a uniform random number generator to produce uniform random samples and that are placed in Cartesian coordinates <x=,y=>, and these coordinates are converted to polar coordinates <R,> and then again to Cartesian coordinates < as follows[[15]](#footnote-16)

|  |  |  |
| --- | --- | --- |
|  |  | (70) |
|  |  | (71) |

### Goodness of Fit Test

The Box-Muller Gaussian Random Number generator used in the class project was tested using the Goodness of Fit Test[[16]](#footnote-17) to determine if the distribution of the random numbers appeared to be Gaussian. Visually, the distributions of the random numbers look Gaussian, which is an important first test, but not conclusive because it is an ad-hoc method. To perform a better test, we evaluated each distribution evaluating their reduced scores.

Problem 1a: = 0.997456 = 1.002917   
Problem 1b: = 0.992626 = 0.998540   
Problem 2a: = 1.002041 = 1.003625   
Problem 2b: = 0.997948 = 1.002234

From the reduced test, these results indicate that the random number generator is nearly normal Gaussian Random variables.[[17]](#footnote-18) [[18]](#footnote-19) The authors believe that the classifier design and the results of the class project represent that expected from features with Gaussian random numbers.

Rod Pickens also coded and tested the polar form[[19]](#footnote-20) of the Box Muller Gaussian number generator, and he found that it generated Guassian random numbers but nominally ten times faster than the above algorithm.

1. http://en.wikipedia.org/wiki/Normally\_distributed\_and\_uncorrelated\_does\_not\_imply\_independent [↑](#footnote-ref-1)
2. Duda, Richard O., Hart, Peter O. and Stork, David G, “Pattern Classification,” Wiley Interscience, Second Edition, page 26, equation 19. [↑](#footnote-ref-2)
3. Ibid, page 36, Section 2.6.1. [↑](#footnote-ref-3)
4. Ibid, page 41, Section 2.6.3. [↑](#footnote-ref-4)
5. Ibid., page 26, Equation 19. [↑](#footnote-ref-6)
6. Ibid., page 27, Equation 21. [↑](#footnote-ref-7)
7. Duda, Richard O., Hart, Peter O. and Stork, David G, “Pattern Classification,” Wiley Interscience, Second Edition, page 25, equation 12. [↑](#footnote-ref-8)
8. Ibid, page 25. [↑](#footnote-ref-9)
9. Ibid, page 22. equation 5. [↑](#footnote-ref-10)
10. Ibid, Page 20. [↑](#footnote-ref-11)
11. Ibid, page 46. equation 72. [↑](#footnote-ref-12)
12. Ibid, page 47. equation 75. [↑](#footnote-ref-13)
13. Ibid, page 36. [↑](#footnote-ref-14)
14. By Cmglee (Own work) [CC BY-SA 3.0 (http://creativecommons.org/licenses/by-sa/3.0) or GFDL (http://www.gnu.org/copyleft/fdl.html)], via Wikimedia Commons [↑](#footnote-ref-15)
15. http://en.wikipedia.org/wiki/Box%E2%80%93Muller\_transform [↑](#footnote-ref-16)
16. Meyer, Paul L., Introductory Probability and Statistical Applications, Addison-Wesley Publishing Company, Inc, pages 306-308. [↑](#footnote-ref-17)
17. http://neutrons.ornl.gov/workshops/sns\_hfir\_users/posters/Laub\_Chi-Square\_Data\_Fitting.pdf [↑](#footnote-ref-18)
18. http://en.wikipedia.org/wiki/Goodness\_of\_fit [↑](#footnote-ref-19)
19. http://www.design.caltech.edu/erik/Misc/Gaussian.html [↑](#footnote-ref-20)