Programming Assignment 1 Report

CS 679 Pattern Recognition

Tyler Becker, Mayamin Hamid Raha

Objective	**Completed By**
Programming:	
– Experiment 1	Tyler Becker
– Experiment 2	Tyler Becker
– Experiment 3	Mayamin Hamid Raha
– Experiment 4	Mayamin Hamid Raha
Report:	
– Theory	
– Dataset Generation	Mayamin Hamid Raha
– Bayesian Classifier	Tyler Becker
– Euclidean Distance Classifier	Mayamin Hamid Raha
– Bhattacharyya Error Bound	Mayamin Hamid Raha
– Results and Discussion	
– Experiment 1	Tyler Becker
– Experiment 2	Tyler Becker
– Experiment 3	Mayamin Hamid Raha
– Experiment 4	Mayamin Hamid Raha
– Conclusion	Tyler Becker, Mayamin Hamid Raha

Theory:

In this section the theory of the classifiers and errors calculations used in the assignment will be described. As well, the basic tasks for each experiment will be quickly explained for clarity.

We are provided with a set of mean and covariances corresponding to two datasets. Our initial task is to generate a dataset that follows gaussian distribution. In experiment 1, the task was to create a Bayesian classifier to discriminate points belonging to two different normal distributions, both of which shared a covariance matrix but had different means. For experiment 2 the task was repeated on a new dataset which had heterogeneous covariance matrices. Finally, for experiments 3 and 4 the task was to create a simple euclidean distance classifier for the dataset in experiment 1 and 2 respectively, and to then compare the results from the euclidean distance classifier to the results from the bayesian classifiers.

Dataset generation:

We have generated 60,000 random samples that have a gaussian distribution having mean and standard deviation value of 1 using Box Muller transformation . We have generated another 140,000 samples that have a normal distribution using the same procedure with a mean and standard deviation value of 4 and 1 respectively. In this way we created Dataset A and for creating Dataset B we followed the same approach with the mean and standard deviation given for Dataset B. We calculated the square root of the given covariance values, in order to find values for standard deviation which acted as an input for box muller transformation.

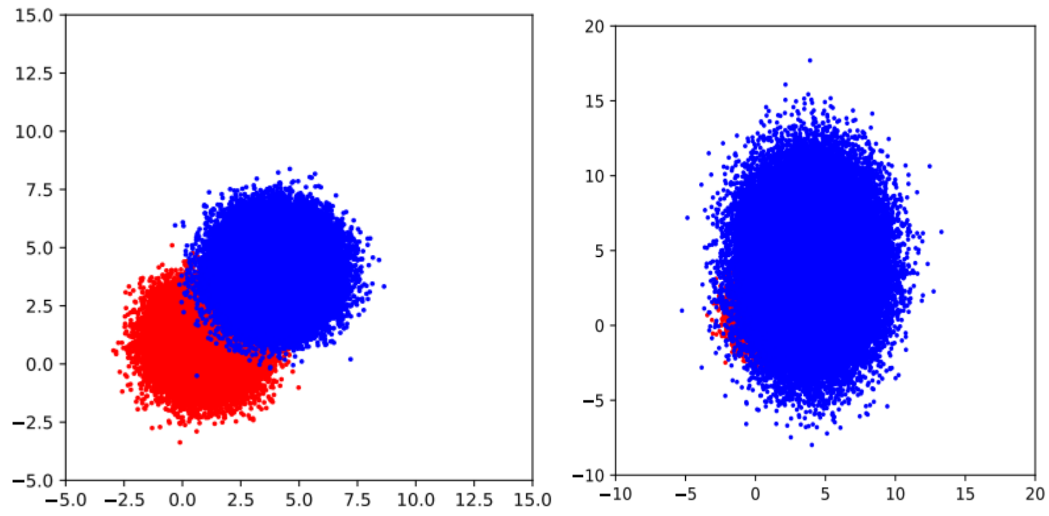

Figure 1: Visualization of Dataset A and Dataset B . Here Red color is used for class 0 features and Blue color for class 1 features. The X and Y axes represent feature values x and y respectively.

In order to find out whether our data generation was correct we used square plots to check whether the mean and variance values were reflected there. Let's have a look at the mean and covariance matrices provided for Dataset A.

$$\mu_0 = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \Sigma_0 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \mu_1 = \begin{bmatrix} 4 \\ 4 \end{bmatrix} \Sigma_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

We see that for class 0 the mean is (1,1) and the covariance has equal variance values along the x and y axis. For class 1, the mean is (4,4) and the variance values along the x and y axis are equal.

By looking at the plot of Dataset A in the left corner of Figure 1, we can see that both clusters are of the same shape, both are round since they follow Gaussian distribution and the cluster centers are also (1,1) and (4,4). Similarly the mean and covariance matrices mentioned for Dataset B are mentioned below:

$$\mu_0 = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \Sigma_0 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \mu_1 = \begin{bmatrix} 4 \\ 4 \end{bmatrix} \Sigma_1 = \begin{bmatrix} 4 & 0 \\ 0 & 8 \end{bmatrix}$$

Here we see that the cluster center for class 0 is (1,1) and it also has unit variance along the x and y axis. So we expect the plot to be round and have a center (1,1). Again for class 1 we find that the mean is (4,4) but the variance along y axis is double of variance along x axis.i.e. this time we expect the data to be more spread out along y axis than x axis. All these assumptions or expectations regarding cluster shape only hold true for the case when we are using a square frame to plot (equal length of figure along a and y axis). In the plot shown on the right side of Figure 01 we see that our assumptions hold true, class 1 cluster is more spread out along the y axis than x axis.

Bayesian Classifier:

A Bayesian classifier is a classification model which leverages Bayes rule to calculate the probability that a given sample is of any given class based on probabilities calculated from some collected data. The classification given is generally of the class for which the probability was the greatest, although there are cases in which this is not true as described below. This model utilizes assumptions made about the true probability distribution of the data in order to simplify the computations necessary to perform this classification, and the amount of assumptions taken and the correctness of the assumptions will affect the accuracy of the classification.

In order to use this classifier, or any classifier, we first need to define the features that we will be measuring. In the case of this experiment, the x and y values of each data point on the graph are the features we compute, however, we could just as easily utilize other values such as closeness to other points, average class of neighboring points etc. which could affect the accuracy of the models. The features that we are using are always assumed to be sampled values from some unknown probability distribution and for simplicity a normal distribution is used. In the case of this experiment, the parameters of the distribution are given and are used to generate the data that we are using to compute these probabilities.

The data that we use to estimate the probability distribution is generally referred to as the training set. The accuracy of the model is directly related to the degree to which the training data represents the population. This means that in general having a larger and more varied training set will give a more accurate model, however, this is only true if there is no bias in the training set. An example of this would be if a probability distribution of pictures of faces were being used to select faces in pictures, however, only pictures of white faces appear in the training set. In order to avoid this problem, a section of the data is generally taken out of the training set for testing and measuring the accuracy of the model afterwards, however, again if this data is also biased then it will report higher or lower accuracy then what will be observed in the real world. For the purposes of this experiment we sample unbiasedly directly from the true values of the distribution to generate the data and therefore it is of course representative of the true values. Similarly, because the training data is so representative of the actual distribution in this experiment we would not expect a generalizability increase from utilizing a test set of data but only because we are ensuring that we are getting representative data which is not realistic outside of this experiment.

In order to make a classification decision with the model that was trained we use the following equation:

$$\text{eq1: } g_i(x) = -\frac{1}{2}(x-\mu_i)^t\Sigma_i^{-1}(x-\mu_i) - \frac{d}{2}ln(2\pi) - \frac{1}{2}ln|\Sigma_i| + lnP(\omega_i)$$

Where x is the vector of features, μ_i is the mean of class i, Σ_i is the covariance matrix of class i, d is the number of features and $P(\omega_i)$ is the prior probability of class i. While this is the general equation if we make certain assumptions then we can simplify it and therefore simplify the model. For experiment 1 of this assignment we were given two datasets which share the identity matrix as their covariance matrix and also canceled all shared constants between the different functions and therefore we simplified each discriminant to the following:

$$\text{eq2: } g_i(x) = -\frac{||x-\mu_i||^2}{2\sigma^2} + lnP(\omega_i)$$

Which is significantly easier to compute and also yields a hyperplane as the discriminant line between the two distributions which is easier to visualize. It is important to note that even if we did not know the true distribution or this assumption that each class shares a covariance matrix was incorrect we could still model the distributions this way, however, it would decrease the accuracy of the model we are using.

For experiment 2 of this assignment we were given two datasets which have different covariance matrices, and therefore we decided not to use the same assumptions because we would get a poorer accuracy that way. Instead we primarily used the first function expect we do still eliminate the term containing d because it is constant for all classes, and we end up with the following:

$$\text{eq3: } g_i(x) = -\frac{1}{2}(x-\mu_i)^t\Sigma_i^{-1}(x-\mu_i) - \frac{1}{2}ln|\Sigma_i| + lnP(\omega_i)$$

Which requires us to estimate the covariance matrix for each class as well as the mean vector.

Euclidean Distance Classifier:

We know a classifier can also be represented by a set of discriminant functions one for each class. Let $g_i(x)$ be a discriminant function for class i to class c. We will assign a feature vector x to class ω_i when $g_i(x) > g_j(x)$. For our given classification problem for dataset A we have two sets of mean and covariance matrix meaning that it is a two class classification problem. For a two class classification problem a single discriminant function (named dichotomizer) can be used instead of two. An example of dichotomizer for two class classification problem is mentioned below:

$$\text{eq 4:}\quad g\ (x) \ = \ g_1(x) - g_2(x)$$

For the above mentioned equation we decide ω_i if $g\ (x) > 0$. Let our discriminant function be

$$\text{eq 5:}\quad g_i(x) \ = \ ln\, p(x/\omega_i) \ + \ ln\, P(\omega_i)\ \text{i} = 1, \ldots, \text{c}$$

By this time in our experiment we know that our data follows gaussian distribution and we can model likelihood for each class using multivariate gaussian density.i.e. $p(x/\omega_i) \sim N(\mu_i,\Sigma_i)$. Here $N(\mu_i,\Sigma_i)$ is the pdf function for gaussian distribution.

$$\text{eq 6:}\quad N(\mu_i,\Sigma_i) = \frac{1}{(2\pi)^{\ d/2}|\Sigma|^{1/2}} exp[-\tfrac{1}{2}(x - \mu)^t\Sigma^{-1}(x - \mu)]$$

Hence our discriminant function can be expressed as

$$\text{eq 6:}\quad g_i(x) \ = \ -\tfrac{1}{2}(x - \mu_i)^t\Sigma_i^{-1}\ (x - \mu_i) - \tfrac{d}{2}ln2\pi - \tfrac{1}{2}ln|\Sigma_i| + lnP(\omega_i)$$

The computational complexity of our discriminant function depends on the number of parameters in the covariance matrix represented by Σi. There are three cases that indicate the number of parameters in the covariance matrix. Dataset A categorizes as Case 1, which is:

$$\text{eq7:}\ \Sigma_i = \sigma^2 I$$

It means the covariance matrix for every class i, is going to be equal provided the fact that they are diagonal and a product of the identity matrix. So all the diagonal elements will be equal for every covariance matrix. For dataset A the following information was given.

$$\mu_0 = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \Sigma_0 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \mu_1 = \begin{bmatrix} 4 \\ 4 \end{bmatrix} \Sigma_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

From the mean and covariance matrices given for Dataset A we can see that the covariance matrices match with Case 1. Then our discriminant function takes the following form of eq 2.

Now for the case when all the priors are equal that is in our case if we consider we have two class namely class 0 and class 1, then $P(\omega_0) = P(\omega_1)$, our discriminant can be further optimized to:

$$\text{eq 8: } g_i(x) = -\,||x - \mu_i||^2$$

which is also known as the Euclidean classifier. Now, if we look at the prior probability for class 0 and class 1 for dataset A we get,

$$P(\omega_0) = 60000 / 200000 = 0.3 \text{ and } P(\omega_1) = 140000 / 200000 = 0.7$$

Here $P(\omega_0)$!= $P(\omega_1)$, therefore, the Euclidean classifier may not provide the optimum classification results. For our dataset A, since we have 2 classes and 2 features x and y our euclidean classifier equation looks the following:

$$\text{eq 9:} \quad g_0(x) = -\sqrt{(x - \mu 0x)^2 + (y - \mu 0y)^2} \quad \text{(for class 0)}$$

$$\text{eq 10:} \quad g_1(x) = -\sqrt{(x - \mu 1x)^2 + (y - \mu 1y)^2} \quad \text{(for class 1)}$$

Where u0x, u0y are mean coordinate values along x and y axis for class 0 and u1x, u1y are mean coordinate values along x and y axis for class 1. These two equations are measuring the distances between a given feature pair (x,y) with the mean for class 0 and class 1. Then we have designed our classifier to assign a prediction level in favor of the class that has the minimum value for discriminant function.i.e. Min [$g_0(x)$, $g_1(x)$].

For Dataset B, we see that our prior probabilities are $P(\omega_0) = 60000 / 200000 = 0.3$ and $P(\omega_1) = 140000 / 200000 = 0.7$. Therefore, $P(\omega_0)$!= $P(\omega_1)$, for this dataset also the Euclidean classifier may not provide the optimum classification results since our assumptions of prior being equal dont hold true. We use the same discriminant function formula here to predict which label belongs to which class.

Bhattacharyya Error Bound:

Calculating error bounds rather than exact error can be a computationally efficient approach when designing classifiers. The goal is to design a classifier that has error less than or equal to the value given by an error bound calculation method. We can use eq 1 for this purpose;

$$\text{eq 11:} \quad P(error) \le P^{\beta}(\omega_1)P^{1-\beta}(\omega_2)e^{-k(\beta)}$$

Here P(error) is the error we get from our classifier. Bhattacharya Bound uses a β value of 0.5 making calculation of error bound a computationally simpler task. In table 1 we can see the value of Bhattacharya error bound obtained in Experiment 1 and Experiment 2.

Experiment number	Bhattacharya Error Bound
01	4.82%
02	16.2%

Table 1: Bhattarcharya error bound values for Experiment 1 and 2

In both of our experiments our total misclassification error rate was lower than the Bhattacharya error bound. For Experiment 1 and Experiment 2 we have a total misclassification error rate of 1.55% and 7.61% respectively.

Results and Discussion:

In this section the results of each experiment will be detailed and then discussed based on how well they match the theory and the expected results. As well, for experiments 3 and 4 their results will be compared with the results for experiment one and two to determine if the classifier we chose gave results that were more accurate as they made less assumptions about the true distribution of the data.

Experiment 1:

In experiment 1 we were tasked with writing a bayesian classifier for a set of data produced by two normal distributions with the following parameters:

$$\mu_0 = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \Sigma_0 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \mu_1 = \begin{bmatrix} 4 \\ 4 \end{bmatrix} \Sigma_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

As the covariance matrices are equal, and in this case also equal to the identity matrix, we decided to choose the model described in eq2. For class 0 we generated 60000 points and for class 1 we generated 140000 points for a total of 200000.The generated data, as well as the estimated discriminant line are shown in fig. 2. As well, the numerical results are summarized in Table 2.

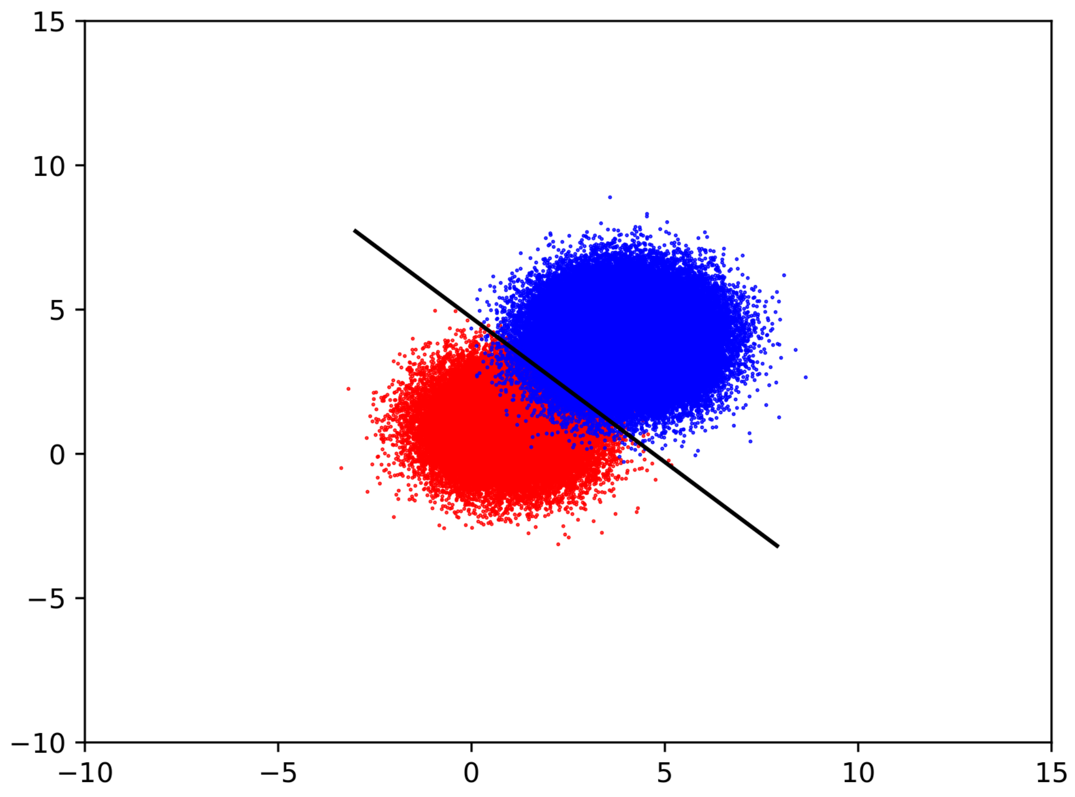

Fig 2: Shown in red is the data generated for class 0, and in blue is class 1. The line in between them is the discriminant line taken from the model defined in eq 2.

As shown visually in fig. 2 and numerically in table 2 this classifier performed very well on the dataset. The line is close to the center point between the two classes, however, because the data had more data points from class 1 rather than class 0 the prior probability for class 1 is higher and therefore the line is shifted closer to the mean of class 0. This lines up with our expectations of how the algorithm works and confirms that the theory is correct. As well, table 2 shows that the total misclassification rate is lower than 3% for all cases so we feel that the assumptions we made using class 1 were correct and represented the data correctly. We knew that the assumption was correct beforehand so this result means that we have confirmed the theory.

	Estimated Value
Misclassification Rate Class 0	2.78%
Misclassification Rate Class 1	1.02%
Total Misclassification Rate	1.55%
Bhattacharyya Error Bound	4.82%

Table 2: This table shows the results of using the model from experiment 1 on the training data set.

Experiment 2:

In experiment 2 we were again tasked with designing and implementing a bayesian classifier, however with different parameters given for the normal distributions. For class 0 we generated 60000 points and for class 1 we generated 140000 points using the following parameters:

$$\mu_0 = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \Sigma_0 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \mu_1 = \begin{bmatrix} 4 \\ 4 \end{bmatrix} \Sigma_1 = \begin{bmatrix} 4 & 0 \\ 0 & 8 \end{bmatrix}$$

As shown, the main difference is that the covariance matrix for class 1 is now different from class 0, and the variances contained are much larger. For this reason, we chose to use the discriminant functions described in eq3. The discriminant function is plotted in fig. 3 and the results of running the classifier are shown in table 3.

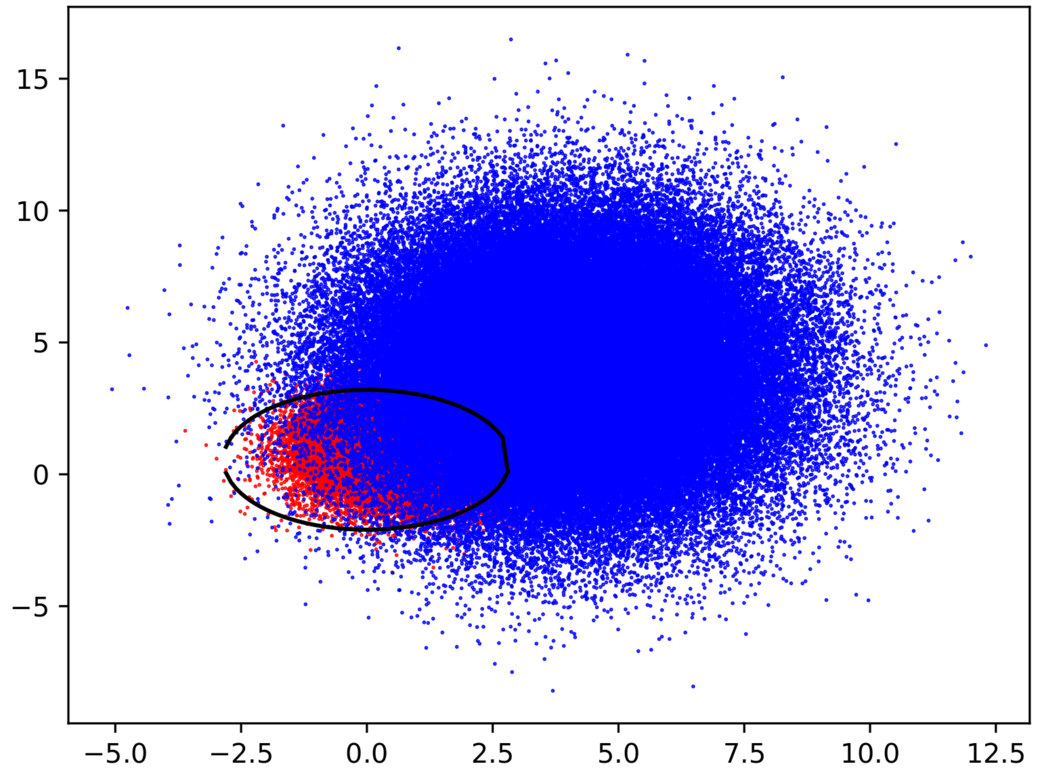

Fig.3: Plot of the training data used for experiment 2. The red points are from class 0, the blue points are from class 1, and the black ellipse represents the discriminant line drawn between them.

	Estimated Value
Misclassification Rate Class 0	8.46%
Misclassification Rate Class 1	7.24%
Total Misclassification Rate	7.61%
Bhattacharyya Error Bound	16.2%

Table 3: This table shows the results of using the model from experiment 2 on the training data set.

The error rates in table are higher than the ones from experiment 1, however, they meet the expectations that we had for this experiment. Due to the higher variance and larger number of points in class 1 the prior probability is higher and in all areas where you can find a red point in Fig. 3 you can also see a blue point. Therefore, it is not trivial to linearly separate the two classes as it was in experiment 1. Keeping this in mind, such a low error rate is somewhat surprising. In terms of the drawn ellipse it seems that it covers all but 8.46% of class 0 and only contains 7.24% of the points from class 1. Changing from calculating the prior probability of the classes to assuming a 50% prior probability for each yields the results shown in Fig. 4 and table 4.

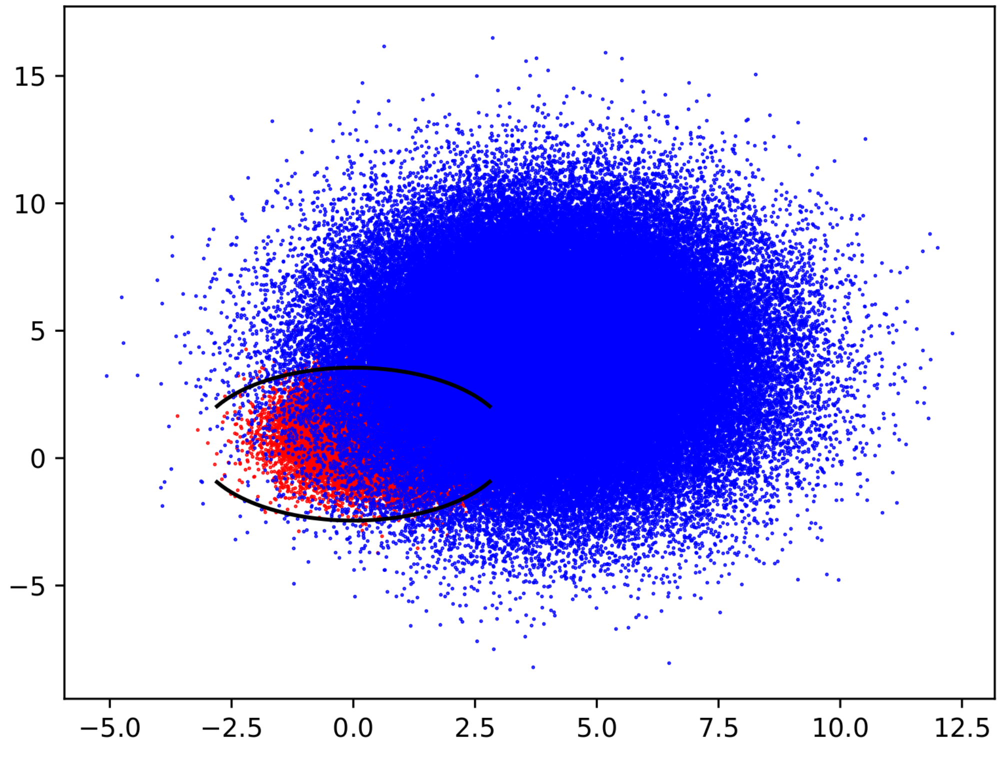

Fig. 4: Graph depicting the same data points from experiment two red representing class 0, and blue representing class 1, however, the discriminant line was calculated using equal prior probabilities and therefore is shifted to the right and up.

	Estimated Value
Misclassification Rate Class 0	3.86%
Misclassification Rate Class 1	10.21%
Total Misclassification Rate	8.39%
Bhattacharyya Error Bound	17.67%

Table 4: This table shows the results of using the model from experiment 2 on the training data set.

As shown in table 4 making the prior probabilities equal made the discriminant line better represent class 0 as we are essentially assuming that both classes should have equal rates. This then increases the overall error because although we are misclassifying less red points we are misclassifying more blue points and since there are more blue points overall the total number misclassified is higher.

Overall, the results from this experiment again coincided with our understanding of the theory and confirmed that these systems can discriminate between simple distributions. In this experiment, we noticed a higher error rate due to the fact that there are so many blue points near the center of class 0's distribution, however, the model chosen from eq3 held up surprisingly well.

Experiment 3:

After using the Euclidean classifier for Dataset A we obtained the following misclassification rates .

	Misclassification Rate Class 0	Misclassification Rate Class 1
Experiment 1	2.7 %	1.02 %
Experiment 3	1.69 %	1.71 %

Table 5: Misclassification rates from Experiment 1 and 3 on the data set A.

The features of dataset A are uncorrelated . That's why the covariance matrices are the same. We see that the Euclidean classifier classifies data of class 0 slightly better than Experiment 1 and Experiment 1 classifies data of class 1 more accurately than the Euclidean classifier. Although there are some differences in the classification results of both experiments, they don't differ much which might be due to the fact that the data clusters have similar shape with equal variance along the x and y axis (see Figure 5).

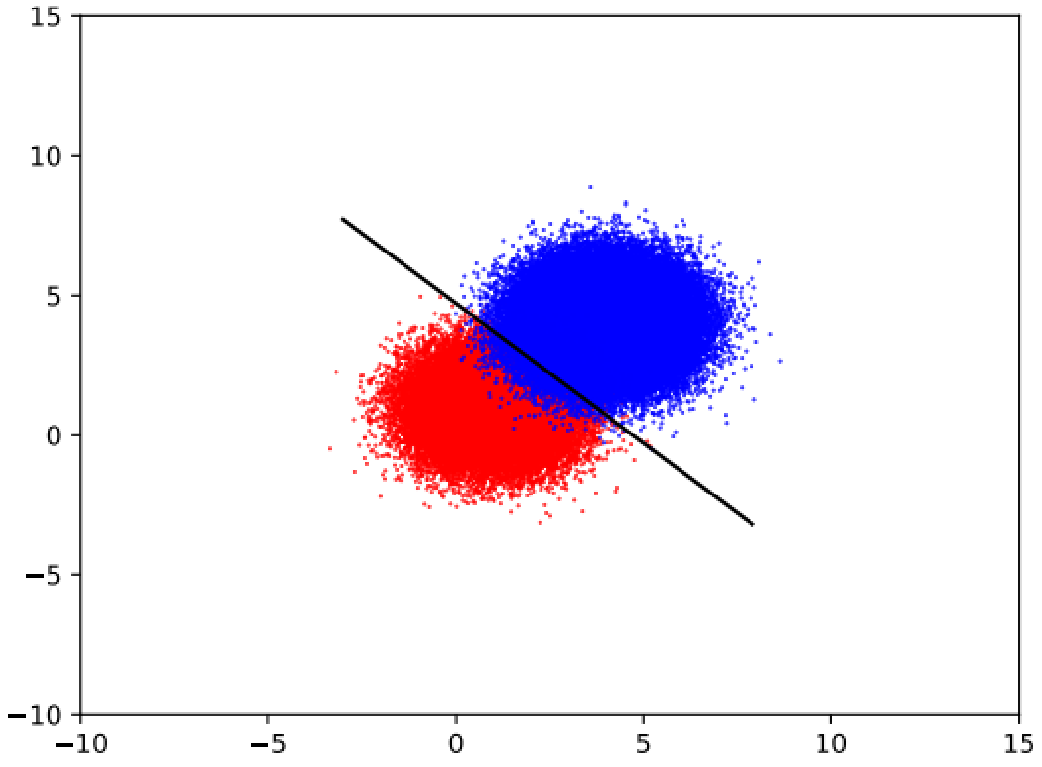

Figure 5: Decision Boundary for Dataset A using Euclidean classifier.

Experiment 4:

We see from Table 6, when the Euclidean classifier is used for Dataset B, it gives 1.72% and 19.16% misclassification rates for class 0 and class 1 respectively. On the other hand, Experiment 2 is showing a greater misclassification error for class 0 which is 8.4 % and lower misclassification error for class 1. Since the covariance matrices for dataset B were not the same, so the data is not uncorrelated. Also the prior probabilities are not the same for class 0 and 1.

	Misclassification Rate Class 0	Misclassification Rate Class 1	Dataset used
Experiment 2	8.4%	7.2%	Dataset B
Experiment 3	1.69 %	1.71 %	Dataset A
Experiment 4	1.72%	19.16%	Dataset B

Table 6: Misclassification rates from Experiment 1, 3 and 4 on various datasets

Therefore, Euclidean classifiers' high misclassification rate of 19.16% makes sense. We can get a better idea of it if we look at Figure 4. We see that there is a lot of overlap of feature points of class 1 with class 0. The decision boundary generated by the Euclidean classifier clearly misclassified a lot of features belonging to class 1 as class 0. Moreover, the classifier used in experiment 2 is designed to deal better with data where priors are not the same and the covariance matrices are arbitrary. This is the reason why from Experiment 2 we are relatively getting the lowest misclassification rates. The Euclidean distance classifier gives a linear decision boundary whereas the classifier of Experiment 2 gives non linear decision boundary resulting in separating both class 1 and 2 in a better way, we can understand this by comparing Figure 3 with Figure 6.

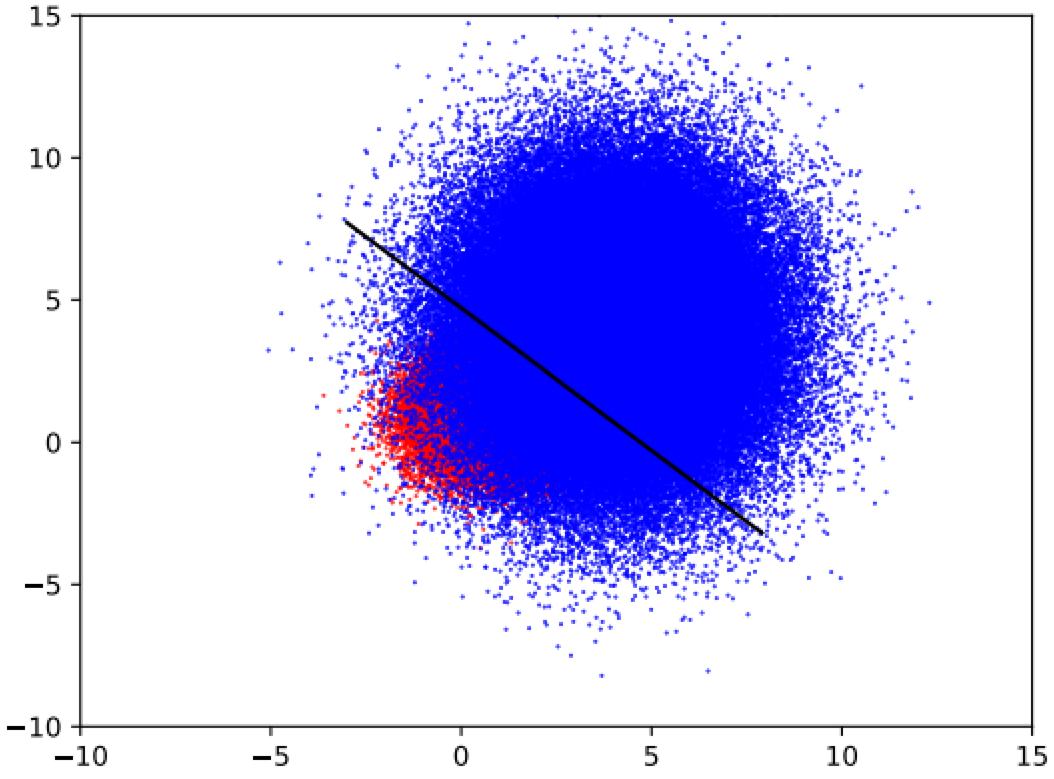

Figure 6: Decision Boundary for Dataset B using Euclidean classifier.

Again, comparing the results of Experiment 3 and 4 we see that Euclidean classifier works better for Dataset A than Dataset B. This fact is evident from Figure 3 and 4. This is because the feature clusters in Dataset A are linearly separable ,whereas, for Dataset B the clusters can be separated well using a non-linear decision boundary.

Conclusion:

Upon using various types of classifier designed for 2 datasets having different types of feature values we observed that classification performance of discriminant function based classifiers is optimum when certain assumptions hold true. For example: Euclidean classifier gives linear decision boundary so it is most suitable for linearly separable data and this classifier may yield high misclassification rate when trying to classify non-linearly separable data with unequal priors. Hence, before choosing a classifier based on discriminant functions for any application field, it is important to know about the assumptions or conditions upon which each discriminant function is dependent. In conclusion, the results here are not too surprising, but they do confirm our understanding of the theories that we are testing. While interesting, it seems obvious that bayesian classifiers not only need to estimate too many parameters, but also make too many assumptions about the data to be viable for very complex problems.