Project 1 Report

Mehar Mangat

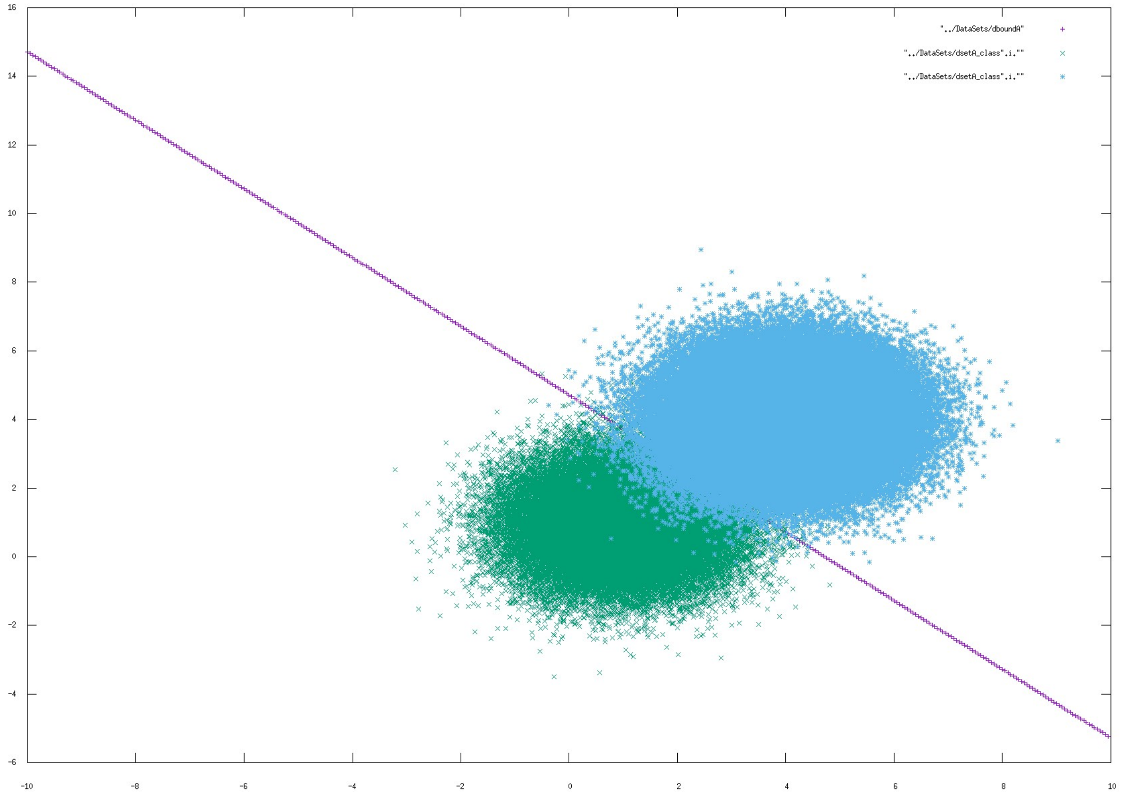
CS 479

March 28, 2021

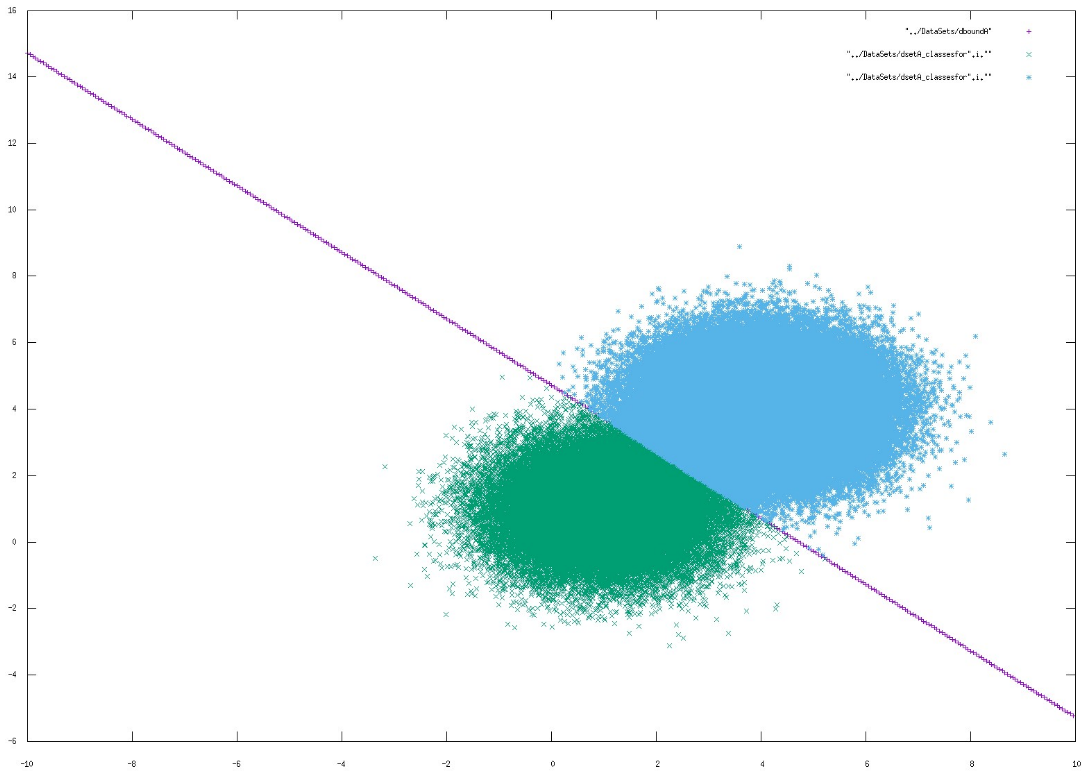
Preface: I did indeed complete the entire coding section of this project myself, without referencing my team-members' code base. Partially due to integrity, partially due to the fact that that level of C++ is well beyond my capabilities; so it would be easier/more informative if I were to start from scratch. I think looking at the difference between the two projects, this fact is fairly obvious, and I am very happy to say that I was able to take a foreign and very powerful topic and accurately understand/implement it in code.

1a) In this case, we can see that each class has the same diagonal matrix with equal diagonal elements, so we are using case one, the linear discriminant. The prior probabilities for each class would simply be the number of samples of a given class divided by the total number of samples collected.

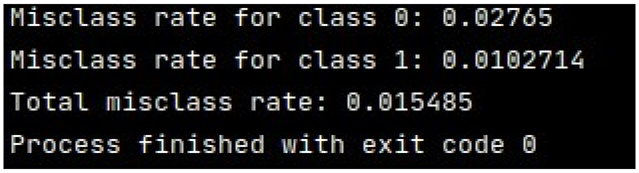
1b): Here is an image of the decision boundary (purple) dividing the classes:



1c): After classification, we can see that the data set classified looks like this:



and that our misclassification errors are the following, with class 0 corresponding to the green and class 1 corresponding to the blue:



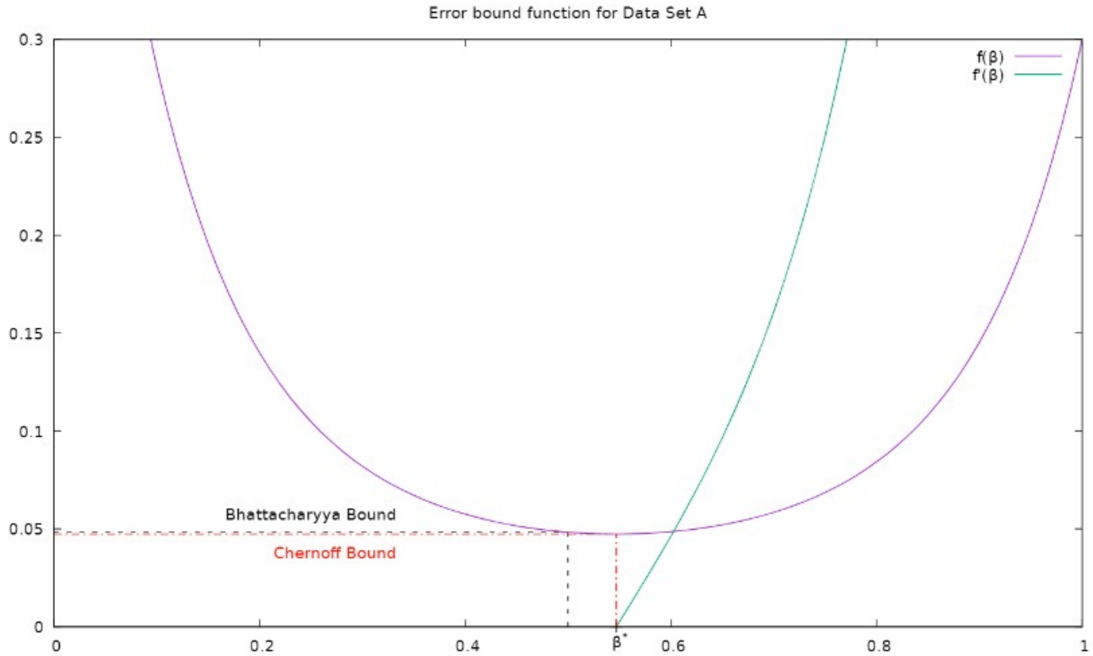
```
Misclass rate for class 0: 0.02765
Misclass rate for class 1: 0.0102714
Total misclass rate: 0.015485
Process finished with exit code 0
```

Note that although spatially it appears that the bluer area is cut off, it also has significantly higher density than the green function, so a proportionally higher amount of points are correctly classified.

1d) The Bhattacharyya bound is the upper bound for the probability of error, given by the equation:

$$\boxed{P(error) \le P^{\beta}(\omega_1)P^{1-\beta}(\omega_2)e^{-k(\beta)}}$$

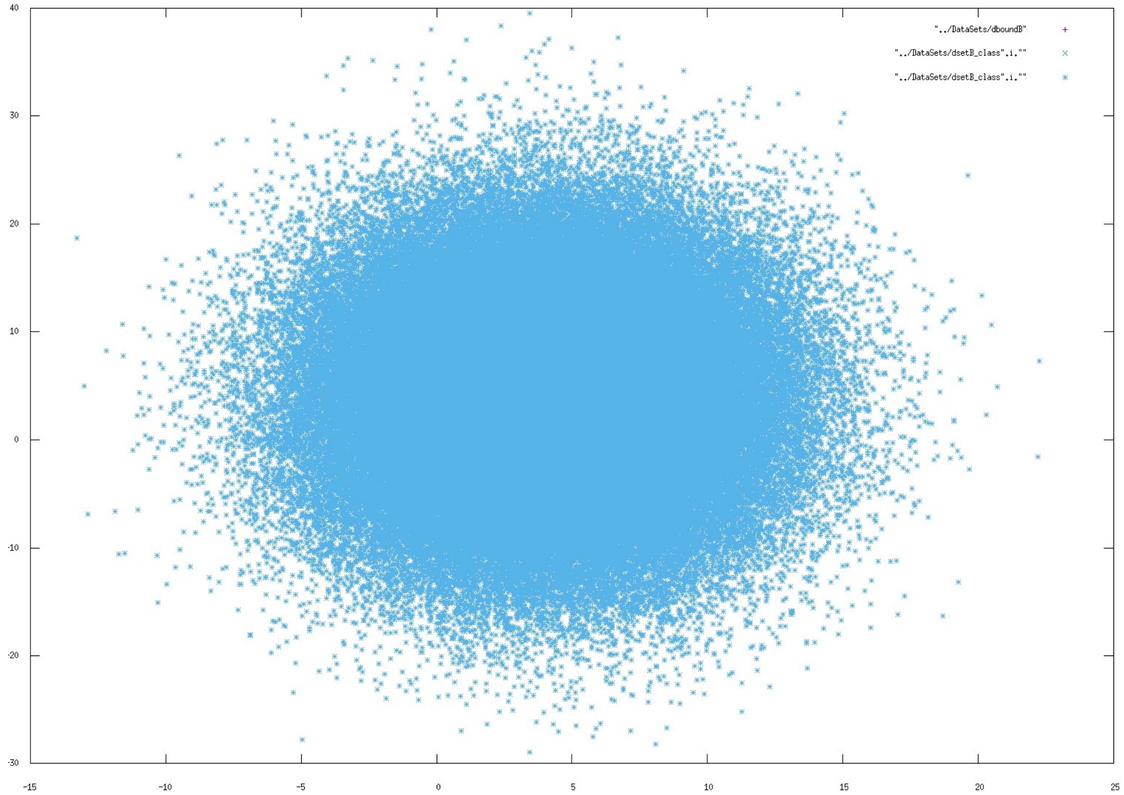
Setting β = 0.5, and priors P(w1) = 0.3 and P(w2) = 0.7, we seek to minimize the resulting function given different values of k. The calculations result in an upper error limit of 0.0483. So, we can see that our classifier has significantly outperformed this error bound. I unfortunately did not have the time to create this graph visually in c++, but the one generated by my colleagues' project can be seen below:

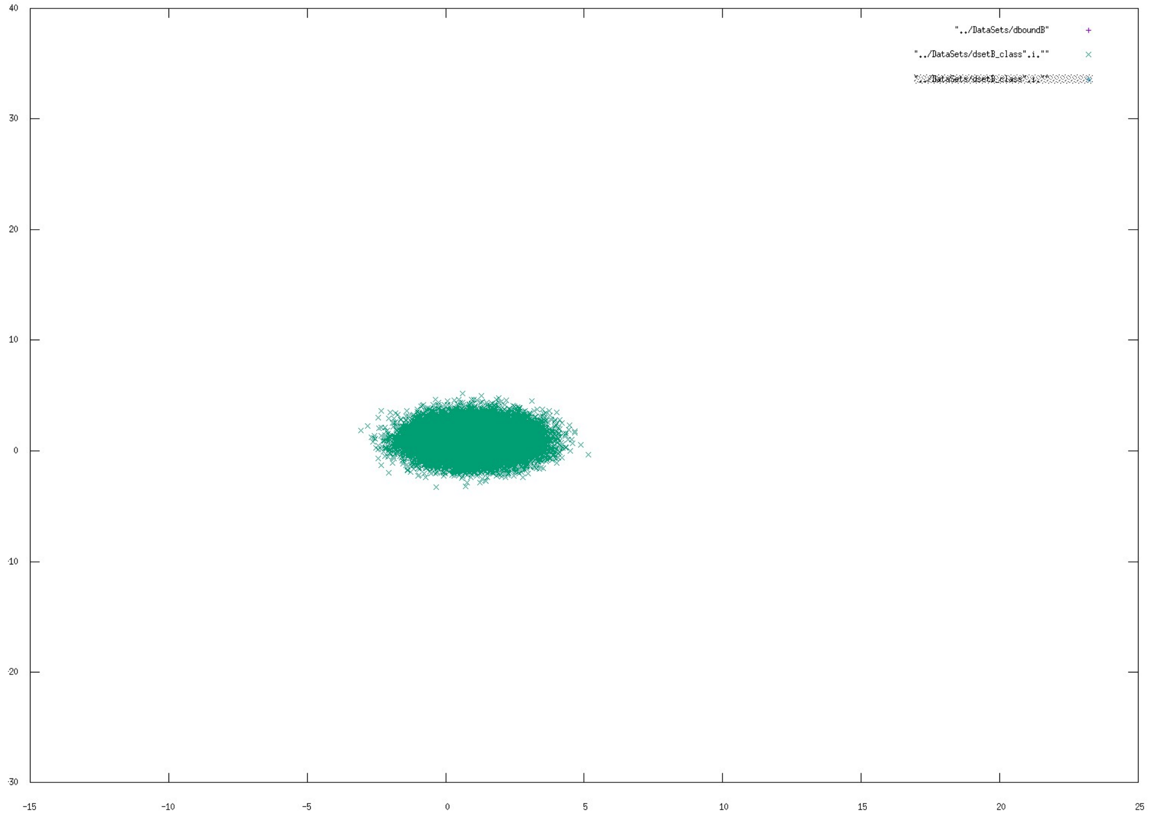


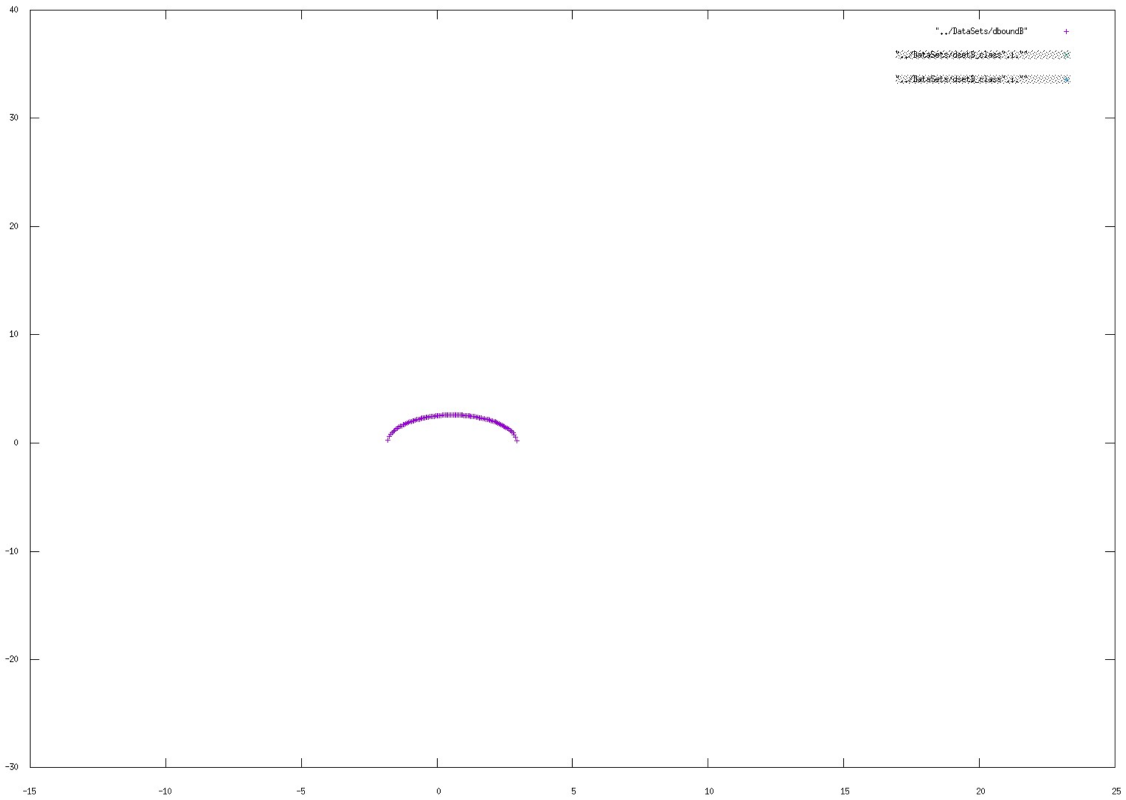
Looking closely, we see the bound being just under 0.05, agreeing with previous calculations.

2a) In this case, each class has its own arbitrary covariance matrix, and thus we are in case 3 with a hyperquadric decision boundary as opposed to a linear decision boundary. Priors are calculated in a similar manner as before, defined by the number of occurences of the class divided by the total number of samples.

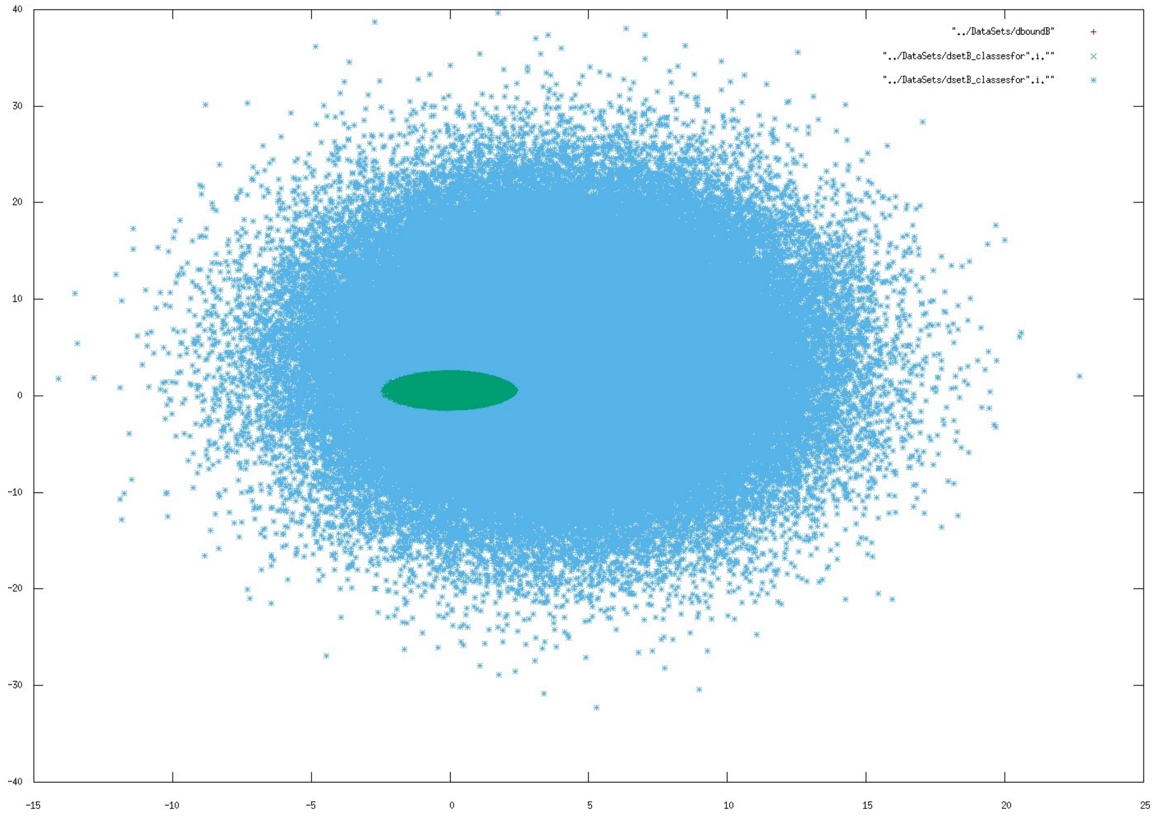
2b) Below is our decision boundary(purple) and classes(blue and green). I was unable to fully map out the boundary but have captured a projected portion of it, and we can see that it is a sort of hyperelipsoidal shape:


"../DataSets/dboundB" +
"../DataSets/dsetB_class".i."" ×
"../DataSets/dsetB_class".i."" ✳
40
30
20
10
0
-10
-20
-30
-15
-10
-5
0
5
10
15
20
25

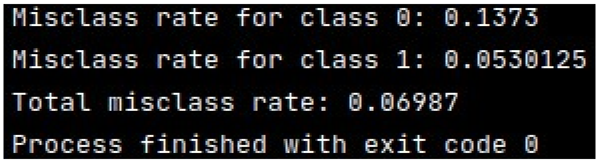

"../DataSets/dboundB"
"../DataSets/dsetB_class".i.""
"../DataSets/dsetB_class".i.""
40
30
20
10
0
-10
-20
-30
-15
-10
-5
0
5
10
15
20
25



2c) As we can see below, classification still works well:



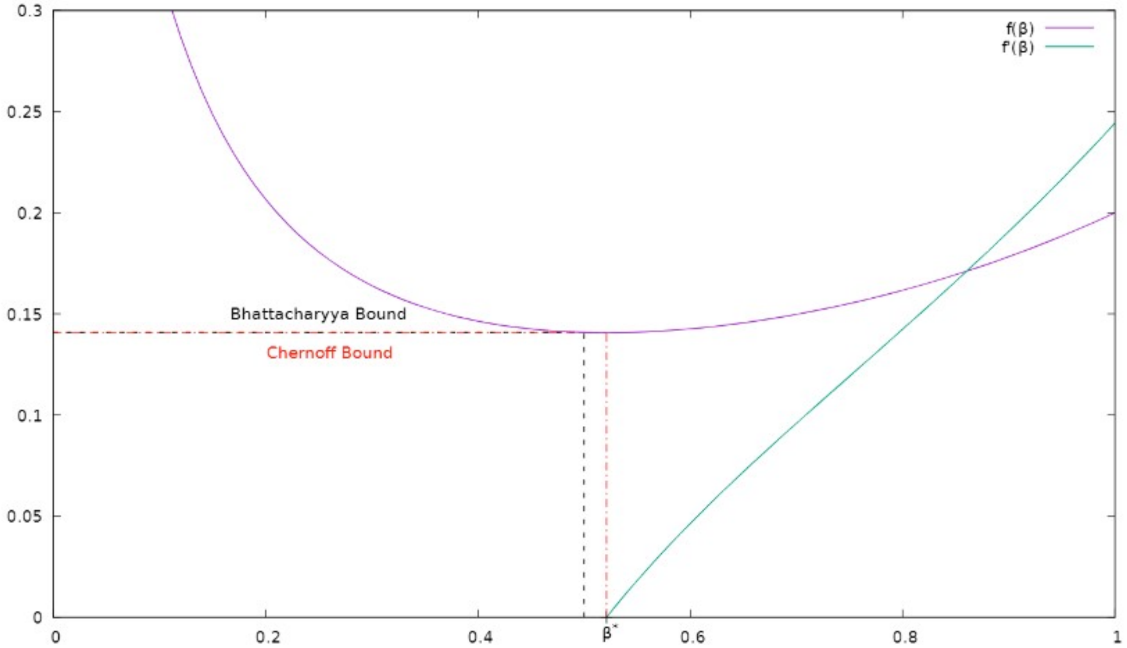
And our misclassification errors are the following, with class 0 corresponding to green and 1 to blue:



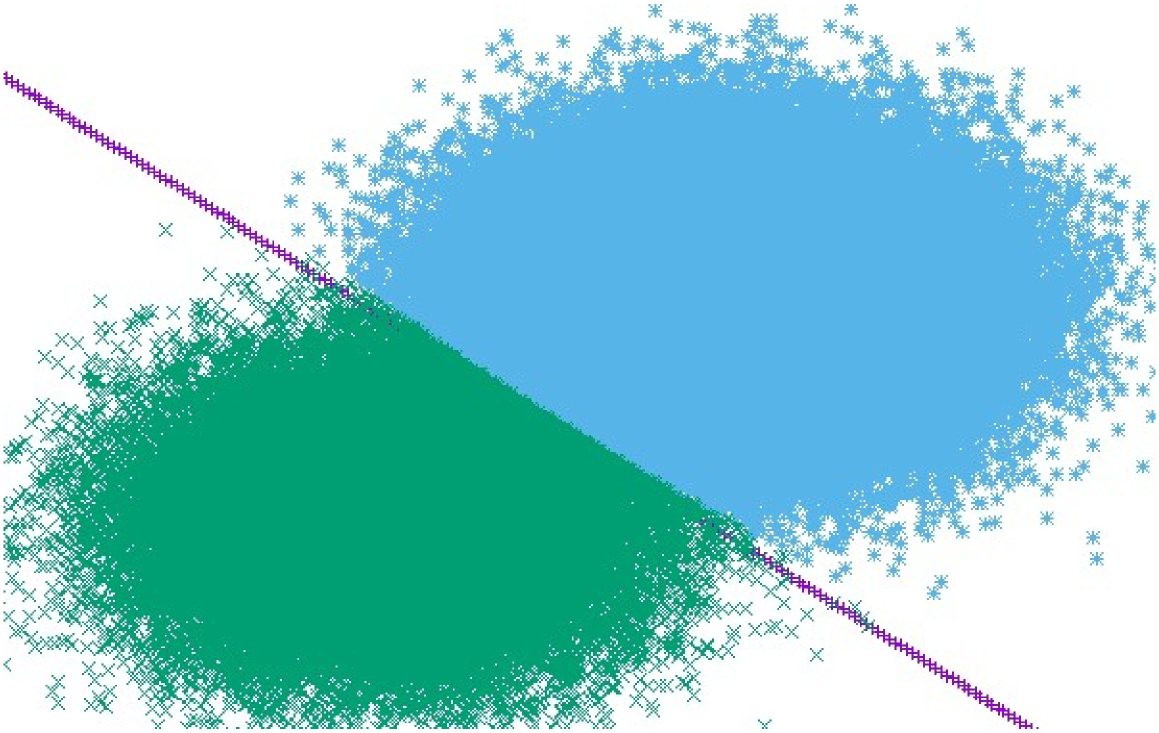
```
Misclass rate for class 0: 0.1373
Misclass rate for class 1: 0.0530125
Total misclass rate: 0.06987
Process finished with exit code 0
```

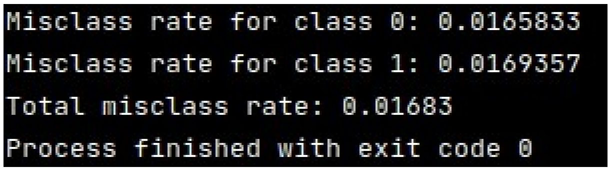
Note that our misclassifications are much higher than the previous dataset, likely due to the fact that there is a greater disparity between prior probabilities.

2d) Following the same steps as before with β = 0.5, and priors P(w1) = 0.2 and P(w2) = 0.8 , and again crediting the graph made by my colleagues, we see that the Bhattacharyya bound is roughly 0.141:



3) The Euclidean distance classifier would be an optimum only under very certain conditions - when we are dealing with a case 1 linear discriminant, AND when all priors are equal. Without these two conditions, we are not at an optimum. The results of the Euclidean distance classifier can be seen below:

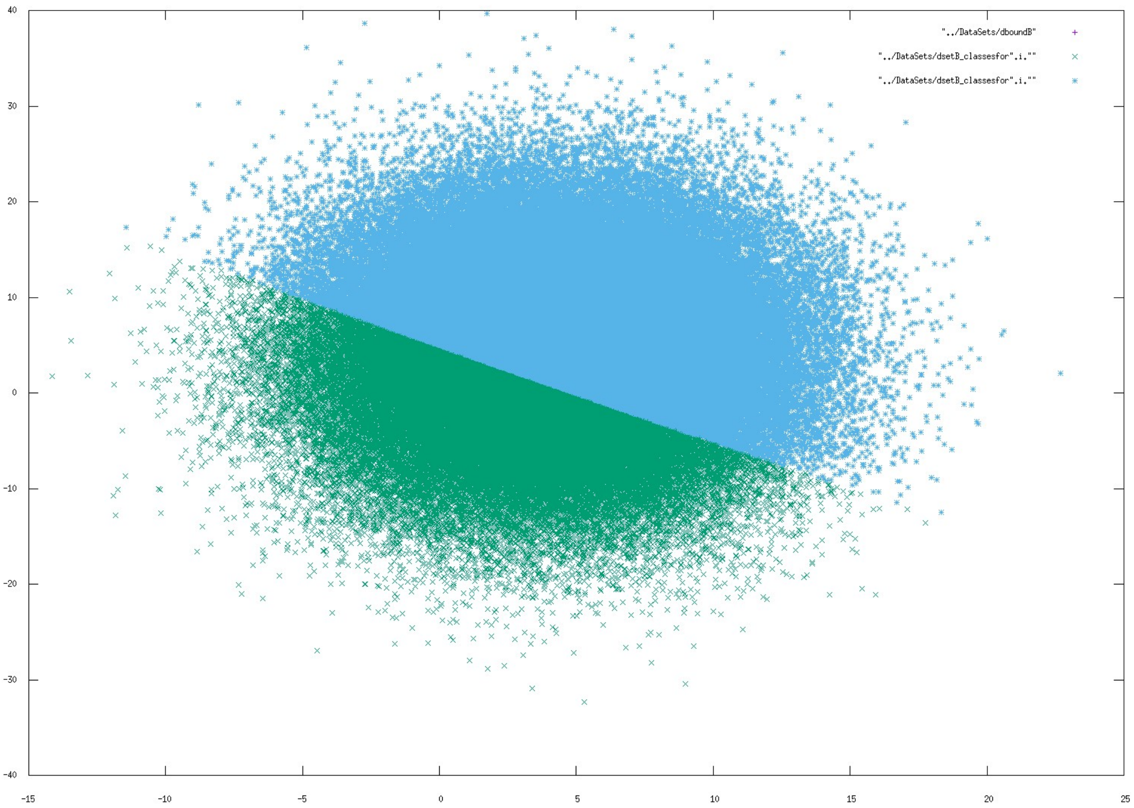


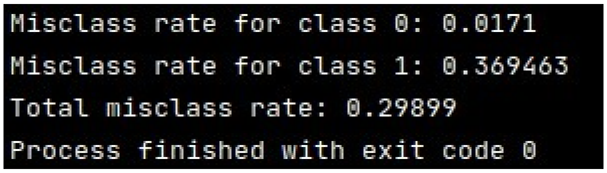


```
Misclass rate for class 0: 0.0165833
Misclass rate for class 1: 0.0169357
Total misclass rate: 0.01683
Process finished with exit code 0
```

Clearly, the Euclidean distance classifier is not an optimum for data set A. Although classification for class 0 improved, class 1 suffered, and total misclassification rate went up. This is very simply due to the fact that the priors for A are not equal, and thus the decision boundary does not account for the "pull" that classes with higher priors have.

4) Similarly, we can see that Euclidean distance classifier would not be an optimum for data set B. The classes do not have equal priors, and the arbitrary covariance matrices imply that our decision boundary must be hyperquadric:





```
Misclass rate for class 0: 0.0171
Misclass rate for class 1: 0.369463
Total misclass rate: 0.29899
Process finished with exit code 0
```

In fact, this decision boundary is radically worse than when applied to data set A due to the fact that the geometry of both boundaries are significantly different (linear vs hyperquadric) rather than being roughly identical (both linear).