UNIVERSITY OF WATERLOO

Lab 1: Clusters and Classification Boundaries

SYDE 572
University of Waterloo
Faculty of Engineering
Department of Systems Design Engineering

Mohammed Abidi, 20720554
Alex Cho, 20800781
Etido Thompson, 20765765
Océane Vandame, 20728517

March 3rd, 2023
Submitted to Dr. Alexander Wong

Introduction

This lab will explore the development of classification boundaries between classes with bivariate Gaussian distributions. Both distance-based (Minimum Euclidian Distance, Nearest Neighbour, k-Nearest Neighbour, General Euclidian Distance) and probabilistic (Maximum A Posteriori) classification strategies will be developed.

These classification strategies will be applied to a case with two classes and equal priors as well as a second case with three classes and differing priors.

Implementation and Results

Code Breakdown

This section will give a brief breakdown of the code structure. In the submitted zip file, there are two main sections: “main.m”, and “functions.” The main file is the driver file that initializes the program and calls the necessary functions to produce the plots. The structure of the main file was written such that the clusters are first initialized and then the different functions for the corresponding classification strategy is called into the main file to create the appropriate plots.

Generating Clusters

In this section, we generated clusters for each class for the given number of points, means, and covariance matrices. This was done using the “bivariatenormalfunct” function which produces normally distributed data given a mean vector and a covariance matrix. The function uses Cholesky decomposition to find the square root of the covariance matrix and then generates random samples from a normal distribution using the “randn” function resulting in bivariate normal random variables. Lastly, the mean vector is added to each row of the matrix using the “repmat” function to get the final output.

Plotting of the standard deviation contour of each class was done by finding eigenvalues of the covariance matrices to find the lengths of major and minor axes. This is done using the “$\det(S - \lambda I) = 0$” equation where the square root of the eigenvalues corresponds to the different axes. This was replicated in MATLAB using the “std_contour” script which returns the angle, and eigenvalues and eigenvectors of the covariance matrix and uses them to find the orientation and lengths of the axes.

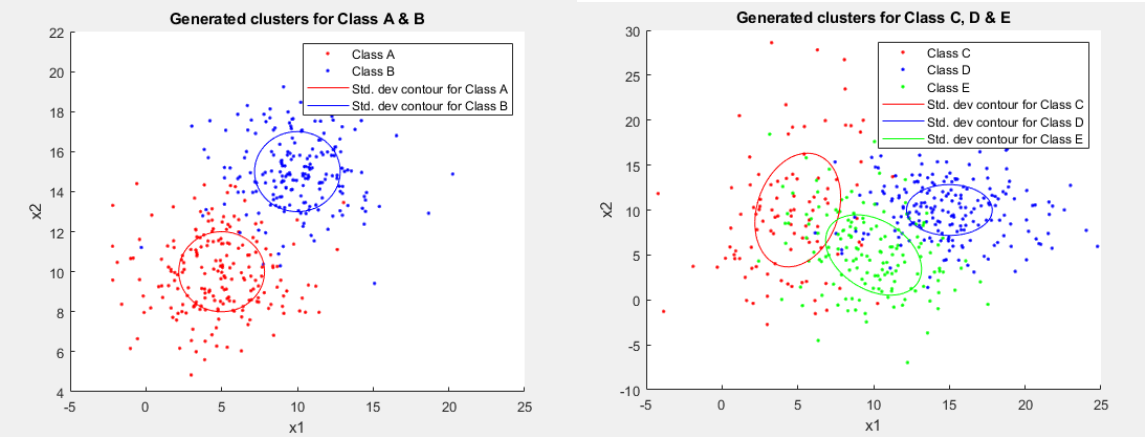

Figure 1: Sample and unit standard deviation contours for case 1 and case 2

Based on Figure 1, a large set of the data points are within a standard deviation of the mean as majority of points are encapsulated within the standard deviation contours. Additionally, Class A and Class B have similar sizes and directions signifying that they have equal variance. This isn't the case for Class C, Class D, and Class E in which Class E and Class C overlap and Class E is negatively tilted and Class C is positively tilted.

Case 1

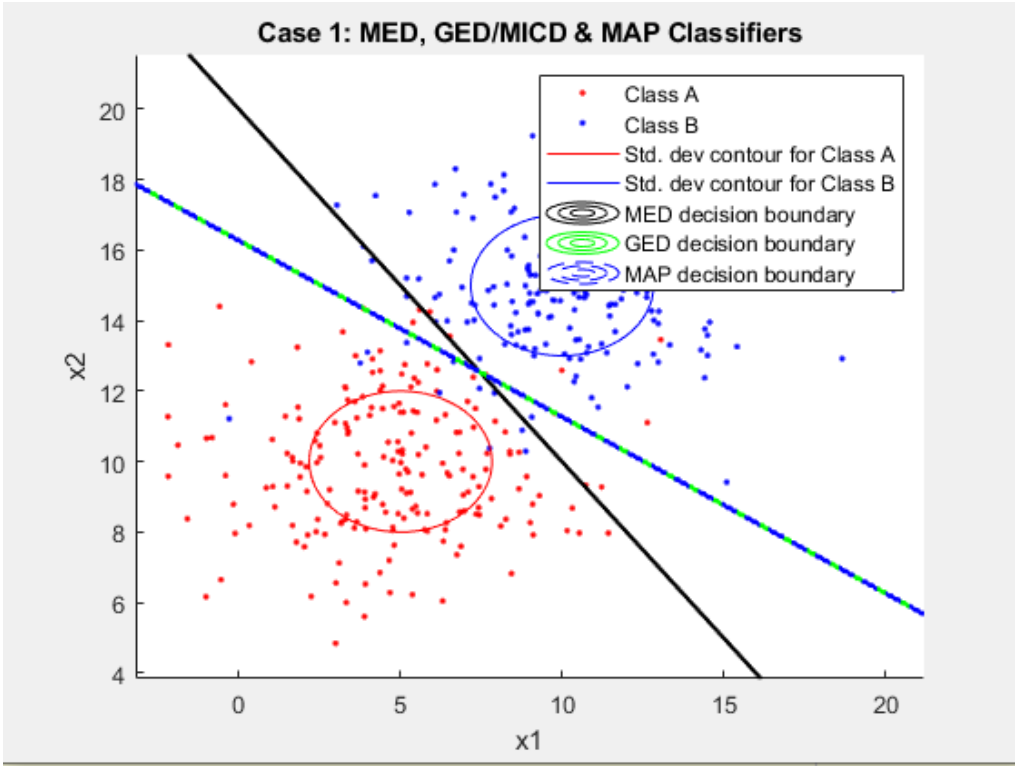

Figure 2: The MED, GED, and MAP Classifiers for case 1

Figure 2 shows the MED, GED, and MAP decision boundaries for the two-class case. Each of these classification boundaries are discussed below. For the MED classifier (shown in black), the decision boundary is a negative linear slope that intersects between the two standard deviation contours. The MED classifier works by classifying a point to a class only if the Euclidean distance between the point and the mean is less than the distance between the point and all other class means. This is done by the "MED_clf" function which calculates the discriminant value of each point and then classifies the points into either Class A or Class B. What can be seen is that the data is equally separated from the decision boundary which is because MED only accounts for means which are equidistant from the means and normal to the origin. Next, the GED decision boundary can be seen in green and is also overlapped by the MAP decision boundary. The GED decision boundary was found with the "GED_clf" function which is used to classify the points based on the distances to the mean vectors, weighted by the inverse of the covariance matrix of each class. In comparison, the MAP decision boundary was found with "MAP_clf" which calculates the discriminant function for each point using the MAP formula and then classifies each point based on the discriminant value. The MAP and GED decision boundaries overlap because of equal probabilities and covariances which causes the right-hand side of the MAP formula to be zero and thus producing the same result as the GED decision boundary.

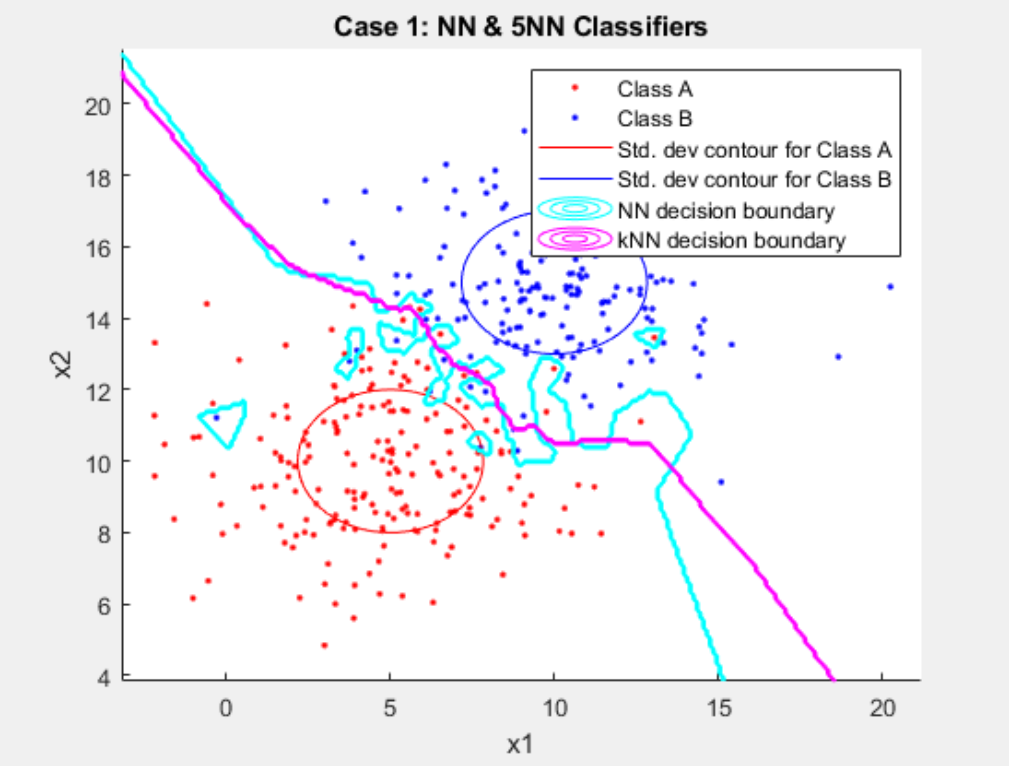

Figure 3: The NN, and KNN Classifiers for case 1

Figure 3 shows the NN and KNN classifiers for case 1. A function was made that performs NN by calculating Euclidean distances between the point and all samples in each class. It then finds the smallest distances for each class and calculates the means based on the number of neighbours

(k). The NN overfits the data and separates all the points from each other. This is because in this case, k = 1 and the prototype used is the smallest Euclidean distance between the nearest neighbour. In comparison, the KNN with a k = 5 doesn't overfit the data as much because it is using a mean of the 5 closest neighbours to run the computation.

Case 2

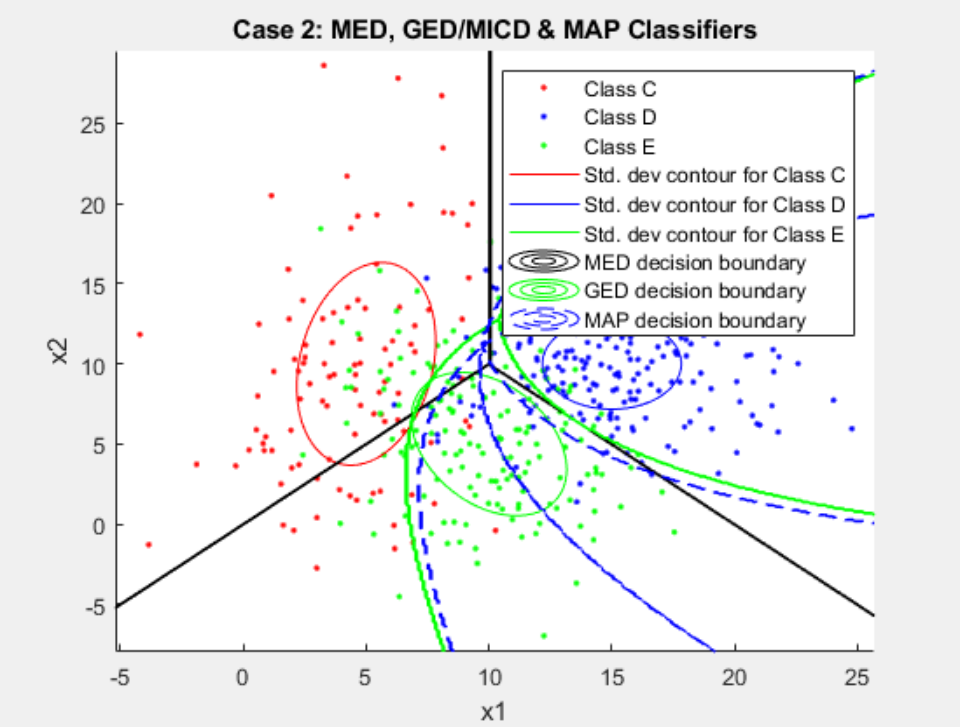

Figure 4: The MED, GED, and MAP Classifiers for case 2

Figure 4 shows the MED, GED, and MAP decision boundaries for case 2 which contains three classes that have different number of data points, means, and covariances. The functions for these classifiers were written in a very similar nature as for the two-class case thus there will be no discussion of the functions. For the MED decision boundary (black), there is a straight line that splits into two lines at ~ x2 = 10. Essentially, the graph is split up into three equal sections. This is because the MED is only considering the means of the classes and doesn't consider the covariances of each class thus the decision boundary is equidistant from the three means. In terms of the GED decision boundary (green), the decision boundary separates each contour plot but is nonlinear. This is because it considers the covariance matrices thus causing it to shift towards the right side of the graph. Lastly, the MAP decision boundary (blue) is slightly shifted to the right compared to the GED. This is because MAP also considers prior probabilities of the classes by using Bayes' theorem to calculate the posterior while GED only considers distances of data point from mean of the classes. Since Class D and E are more probable, the decision boundary shifts over towards them more.

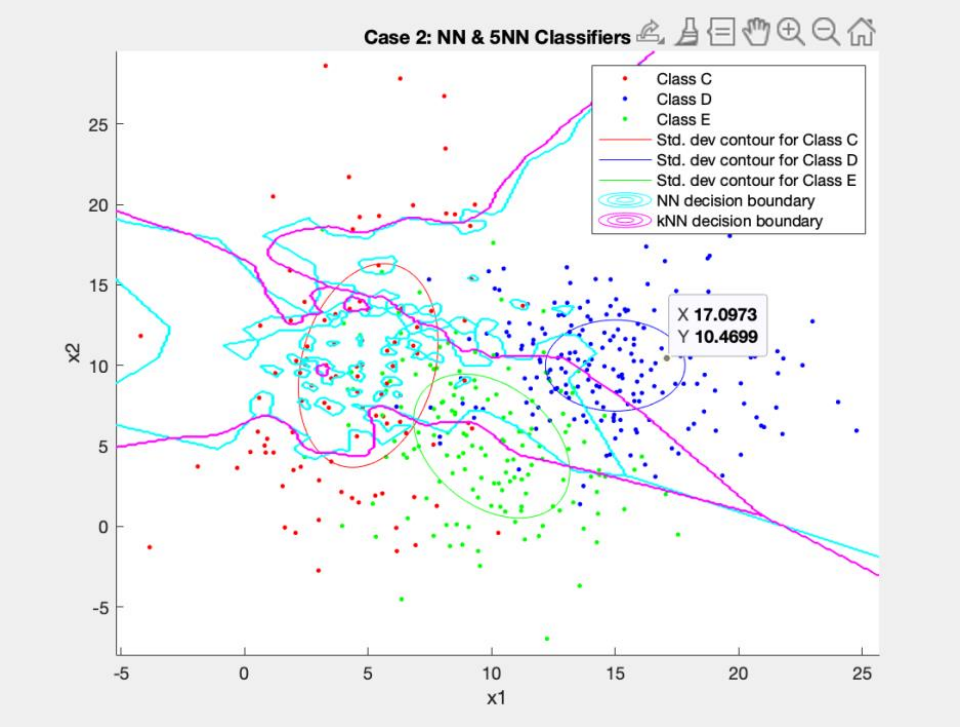

Figure 5: The NN and KNN classifiers for case 2

Figure 5 shows the NN and KNN classifiers for case 2 where there are three classes. Like case 1, there is an overfitting of data with NN having a larger overfit in comparison to KNN. This again is because NN is KNN when k = 1 thus there will be more overfitting as the minimum distance is only found between the nearest neighbour.

Error Analysis

The confusion matrix evaluates the performance of the classifiers by assigning true positives, false positives, true negatives, and false negatives. In case 1, the true positive is predicting class A as A, true negative is predicting class A as B, false positive is predicting class B as class A, and false negative is predicting class B as class B. This can be extended to case 2 where instead of 4 values, there should be 9 because of the three cases. The error analysis for each classifier will be discussed separately below.

```
MED Error analysis:
Confusion matrix for A & B:
   191    14
     9   186

P_error for A & B:
    0.0575

Confusion matrix for C, D & E:
    79     1    20
     6   176    18
    29    15   106

P(error) for C, D & E:
    0.1978
```

```
GED Error analysis:
Confusion matrix for Class A & B:
   190    12
    10   188

P_error for A & B:
    0.0550

Confusion matrix for C, D & E:
    64     1    22
     2   179    13
    24    34    82

P(error) for C, D & E:
    0.2778
```

```
KNN Error analysis:
Confusion matrix for A & B:
     0    14
   200   186

P_error for A & B:
    0.5350

Confusion matrix for C, D & E:
    92     2     6
     0   177    23
    56    28    66

P(error) for C, D & E:
    0.2556
```

```
NN Error analysis:
Confusion matrix for A & B:
     0    73
   200   127

P_error for A & B:
    0.6825

Confusion matrix for C, D & E:
    92     3     5
     8   167    25
    67    29    54

P(error) for C, D & E:
    0.3044
```

```
MAP Error analysis:
Confusion matrix for Class A & B:
   190    12
    10   188

P_error for A & B:
    0.0550

Confusion matrix for C, D & E:
    65     1    21
     3   176    16
    26    27    91

P(error) for C, D & E:
    0.2622
```

Figure 6: Error Analysis of all classifiers

Figure 6 shows the error analysis for the MED classifier in which the probability of error for case 1 was seen to be ~ 6% and 20% for case 2. These results were expected as the MED classifier takes a more simplistic approach in finding the decision boundary since it only considers the means. In comparison, GED and MAP had the same probability of error for the two-class case which was expected as they had the same decision boundary. This probability of error was also lower than MED which was expected as they take into account more than just the means. With NN/KNN the probability of error for the two-class case was very low because of the overfitting of the data but when a third class was introduced (case 2) it became very high. Generally, the probability of error in case 1 was similar as the two classes had the same covariances and similar shapes and sizes. However, in case 2 there was a greater spread in probability of error due to the differences where there were difference covariance matrices, different means, and cluster points.

Conclusions

In conclusion, this lab saw us generate contour plots and data sets for two different cases in which in case 1, there were two uncorrelated classes that equal variance and different means while case 2 had different covariances, means and sizes. We plotted contours by calculating eigenvalues and eigenvectors and several classifiers were used to classify the data. For case 1, using the MED classifier, we saw a negative linear boundary that was equidistant from the means while in the GED and MAP classifiers, they both had equal decision boundaries because the RHS of the MAP equation was zero. In comparison, the NN and kNN classifiers overfit the data (especially in the case of NN) because of the method of calculation which finds the smallest distance between nearest neighbours. For case 2, the MED classifier again produced an equidistant linear decision boundary that was equidistance to the means from the three cases, the MAP and GED classifiers were quite similar with only the difference of the MAP being shifted slightly more to the right side of the graph because of considering priors. Lastly, the NN and kNN classifers again overfit the data, with the kNN boundary being somewhat smoother because of less overfit.

Error analysis was also conducted on each classifier for both cases. In case 1, it was found that with MED there was a high probability for error, with GED and MAP having an identical and lower error. NN and kNN had very low error but are also computationally expensive. For case 2, the errors had a larger spread.