SYDE-572: Introduction to Pattern Recognition

Lab 1: Clusters and Classification Boundaries - **Report**

Marta Zheplinska - 21050969
mzheplin@uwaterloo.ca
Professor - Alexander Wong

Introduction

The purpose of the Lab 1 was to get acquainted with such classifiers as MED, GED, MAP, NN, KNN. In the course of the laboratory work, the clusters were generated based Gaussian distribution parameters, the decision boundaries of each classifier were constructed, and the errors were evaluated.

Generating Clusters

Two cases were considered in the laboratory - the first for two classes, the second for three. The first step was to generate clusters for each class according to its Gaussian distribution:

CASE 1:

$$\text{Class A:} \quad N_A = 200 \quad \mu_A = [5 \;\; 10]^T \quad \Sigma_A = \begin{bmatrix} 8 & 0 \\ 0 & 4 \end{bmatrix}$$

$$\text{Class B:} \quad N_B = 200 \quad \mu_B = [10 \;\; 15]^T \quad \Sigma_B = \begin{bmatrix} 8 & 0 \\ 0 & 4 \end{bmatrix}$$

CASE 2:

$$\text{Class C:} \quad N_C = 100 \quad \mu_C = [5 \;\; 10]^T \quad \Sigma_C = \begin{bmatrix} 8 & 4 \\ 4 & 40 \end{bmatrix}$$

$$\text{Class D:} \quad N_D = 200 \quad \mu_D = [15 \;\; 10]^T \quad \Sigma_D = \begin{bmatrix} 8 & 0 \\ 0 & 8 \end{bmatrix}$$

$$\text{Class E:} \quad N_E = 150 \quad \mu_E = [10 \;\; 5]^T \quad \Sigma_E = \begin{bmatrix} 10 & -5 \\ -5 & 20 \end{bmatrix}$$

Clusters were generated by the rand function in Matlab, after which they were modified according to the data corresponding to the classes. The covariance matrices were modified to a diagonal form using the chol function, after which the data were multiplied by the modified matrix and moved in the direction of the mean value. After that, plots were made for the two cases containing clusters, mean values (green squares), and standard deviation contours (see Figure 1 and Figure 2).

Visually contours that represent 1 standard deviation from their means, contain around more than half of all class samples. The axes of the ellipse correspond to the distribution behavior of the data, the larger axis corresponds to the largest variance of the data, and the smaller axis is plotted in the direction of the smallest variance. **Note that the standard derivation contour for class D is a circle, although it looks like an ellipse in all subsequent images due to axes rescaling.*

Figure 1: A and B classes samples

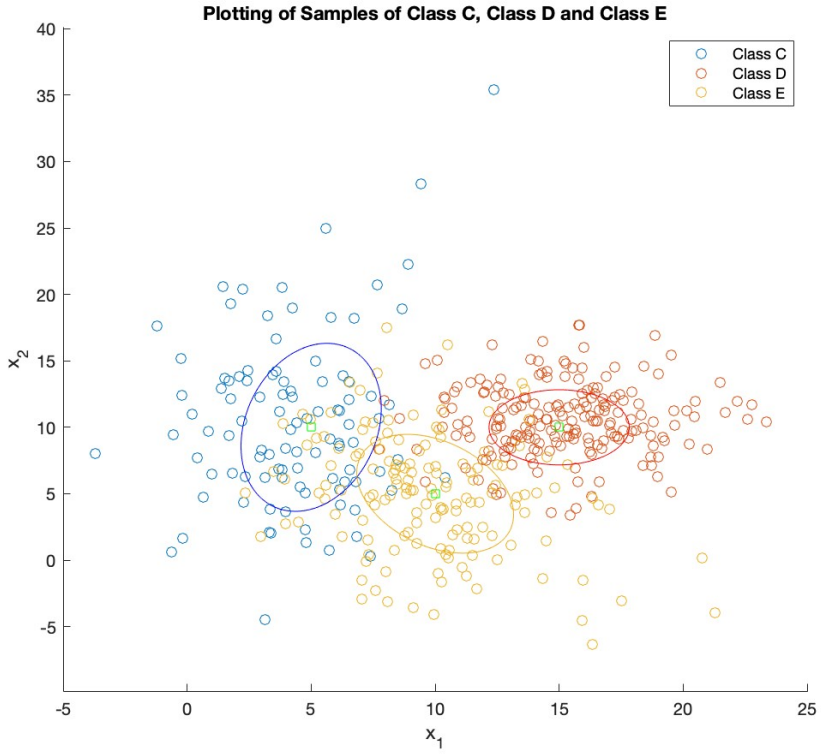

Figure 2: C, D and E classes samples

Classifiers

MED classifier

Each classifier was built numerically on a 2D grid (all points were classified and represented in specific color) using Matlab. If some points could't be classified, they belonged to the decision boundary. In the case of the MED classifier, the distances from each point to the prototype of each class (mean value) were compared. The point belongs to the class where the distance was the smallest. The equation for MED decision boundary is:

$$d_A(\bar{x}, \bar{\mu_A}) = \sqrt{(\bar{x} - \bar{\mu_A})^T(\bar{x} - \bar{\mu_A})} = \sqrt{(\bar{x} - \bar{\mu_B})^T(\bar{x} - \bar{\mu_B})} = d_B(\bar{x}, \bar{\mu_B})$$

where $\bar{x}$ is the point to classify and $\bar{\mu_k}$ is the mean value of k-th class.

The function that calculates distance between points is implemented in **MED_distance.m** file, **MED.m** file contains function that classifies points in the grid into two classes from case 1. The result of it's classification is shown in Figure 3.

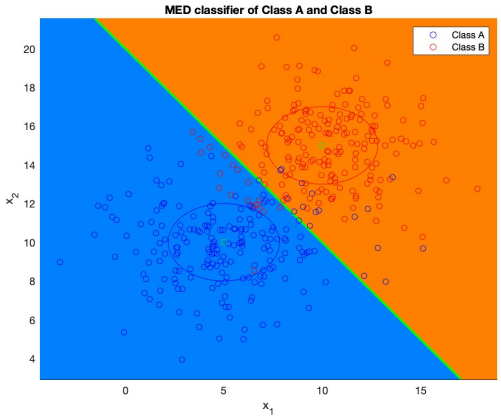

Figure 3: MED classifier for A and B classes

As expected, the decision boundary passed through the midpoint between the two maens.

The implementation of the classifier for 3 classes of case 2 can be found in the **MED_3.m** file. The result of the function is shown in Figure 4:

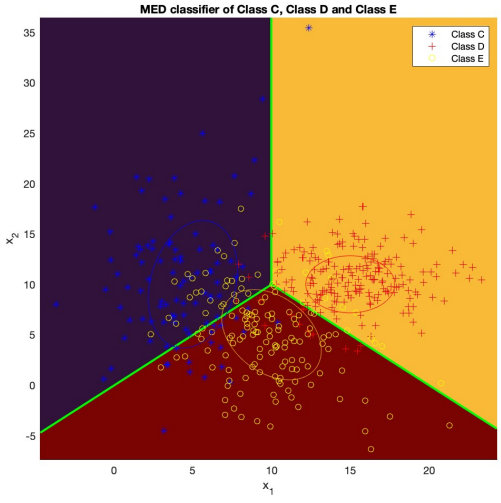

Figure 4: MED classifier for C, D and E classes

As expected, the decision boundaries passed through the midpoints between classes' means and were linear.

GED classifier

The next implemented classifier was GED. The decision boundary is described by the formula:

$$(\bar{x} - \bar{\mu_A})^T \Sigma_A^{-1} (\bar{x} - \bar{\mu_A}) = (\bar{x} - \bar{\mu_B})^T \Sigma_B^{-1} (\bar{x} - \bar{\mu_B}))$$

where $\bar{x}$ is point to classify, $\bar{\mu_k}$ is mean value of class k, and Σ_k is covariance matrix of class k.

It is also worth noting that if the left part is smaller than the right, the point belongs to class A, otherwise the point belongs to class B.

The implementation of the GED classifier function for the two-class case 1 is stored in the file **GED.m**, and for the three-class case 2 in **GED_3.m** file. The helper function that calculates the distance from the point to the prototype of the class is stored in the **GED_distance.m** file. Figure 5 below shows the action of the GED classifier in the case of two classes:

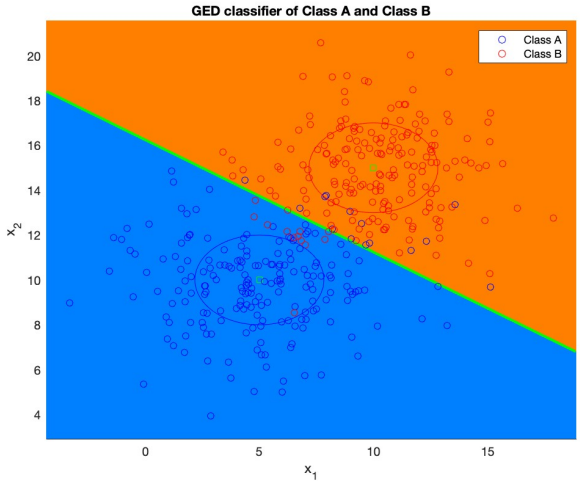

Figure 5: GED classifier for A and B classes

As expected, the decision boundary passed through the midpoint between the two standard deviation contours.

Figure 6 below shows the action of the GED classifier in the case of three classes:

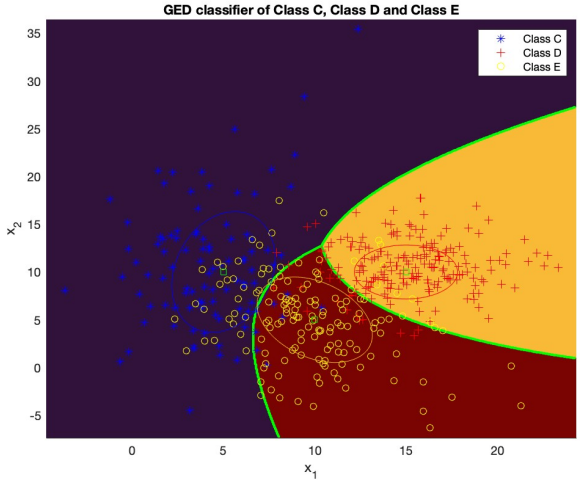

Figure 6: GED classifier for C, D and E classes

As expected, the decision boundaries passed through intersections between standard deviation contours and were non-linear.

MAP classifier

A very effective way of classifying points is a classification based on posterior probability $P(A|\bar{x})$. The point will belong to the class where this value is the greatest. The mathematical formula for decision boundary is the following:

$$P(A, \bar{x}) = P(B, \bar{x})$$

Using Bayes rule, the formula can be simplified to the form:

$$(\bar{x} - \bar{\mu_B})^T \Sigma_B^{-1} (\bar{x} - \bar{\mu_B}) - (\bar{x} - \bar{\mu_A})^T \Sigma_A^{-1} (\bar{x} - \bar{\mu_A}) = 2 \ln \frac{P(B)}{P(A)} + \ln \frac{|\Sigma_A|}{|\Sigma_B|}$$

Classifiers' code is stored in the **MAP_calc.m** file. A function that determines the class a point should belong is in the **MAP.m** file for case 1, and in the **MAP_3.m** file for case 2. The result of classifier implemented by me is shown in Figure 7 below:

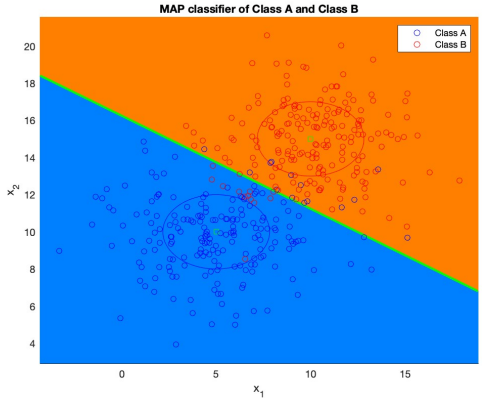

Figure 7: MAP classifier for A and B classes

Given that the class probabilities and samples numbers were the same for both classes, the right-hand side of the equation became zero. And the left part is the same as for the GED classifier. Therefore, the decision boundary is represented by the same line as for the previous classifier.

Figure 8 below shows the action of the MAP classifier in the case 2 for three classes:

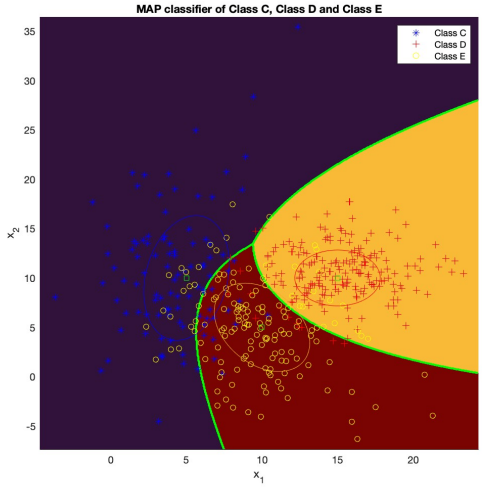

Figure 8: MAP classifier for C, D and E classes

The decision boundary in the case of three classes was not linear and fairly accurately distributed the clusters by classes. However, the experimental error still turned out to be larger than for the GED classifier.

NN classifier

Alternative ways to select a class prototype are nearest neighbor selection and k-nearest neighbors selection (next section). In Lab 1, the MED classifier was considered with the selection of the nearest neighbor as the prototype of the class, also 5 neighbors. This method differs from others, because all clusters are involved in the classification and the prototype changes depending on the point being classified. The decision boundary in this case cannot be represented by an analytical expression.

The function that calculates the distance from a point to the nearest neighbor is stored in the **NN_dist.m** file. Files **NN.m** and **NN_3.m** contain functions that classify points for 2 and 3 classes respectively. The result of performing the function for two classes with the selection of one neighbor is shown in Figure 9.

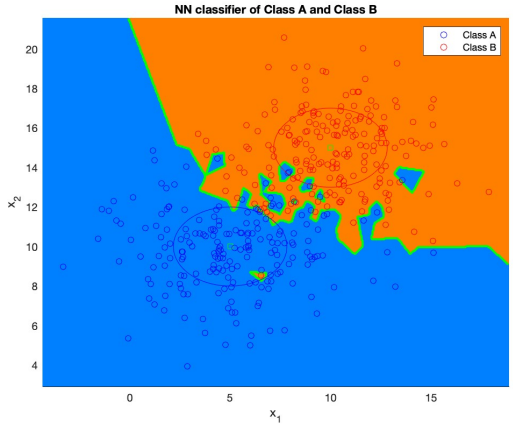

Figure 9: NN classifier for A and B classes

We can see that the decision boundary is not smooth, and passes through the middle of the distance of the two nearest clusters of classes. That is, there is a direct dependence not on the parameters of the distribution, but on the specific sample of the class. There are also small areas that highlight points that are distant from their counterparts.

All three distances of a point to its nearest neighbors were compared in NN-classifier in case of 3 classes. If one of the distances is smaller than the other two, the point belongs to the corresponding class, otherwise the point belongs to the decision boundary. Figure 10 below shows the result of NN-classifier for 3 classes:

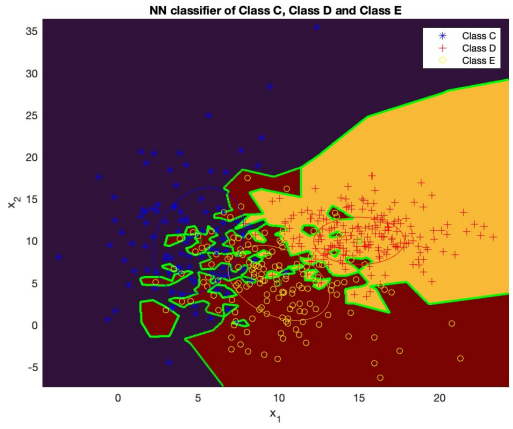

Figure 10: NN classifier for C, D and E classes

The decision boundary is not linear and contains a lot of subboundaries, as classes are close to each other.

5-NN classifier

The same classifier was tested with the selection of 5 nearest neighbors. After their selection, the middle point was taken, to which the distance was calculated.

The function that calculates the distance from a point to the 5 nearest neighbors is stored in the **KNN_dist.m** file. Files **KNN.m** and **KNN_3.m** contain functions that classify points for 2 and 3 classes respectively.

Figure 11 below shows the result of 5NN-classifier for 2 classes:

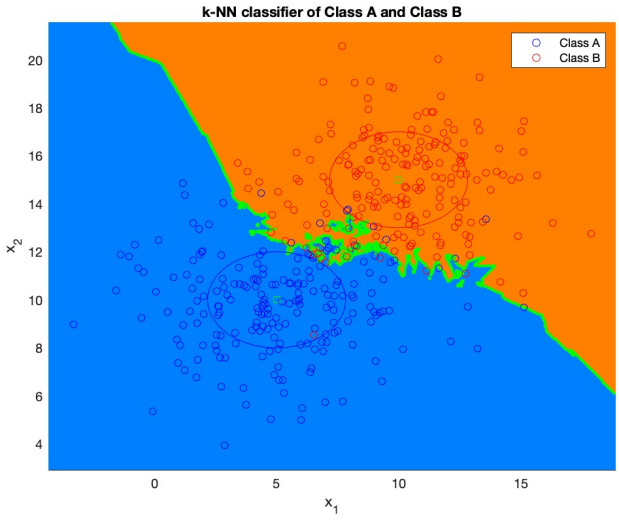

Figure 11: 5-NN classifier for A and B classes

As the number of neighbors increased, the smoothness of the curve reflecting the decision boundary increased. Also, according to the idea, since the number of clusters of classes is the same, choosing the number of neighbors - 200, as a result, as a result we will get the means of each class and the decision boundary will be identical to the decision boundary of the MED classifier. It was tested and expectations were met.

Figure 12 below shows the result of 5NN-classifier for 3 classes:

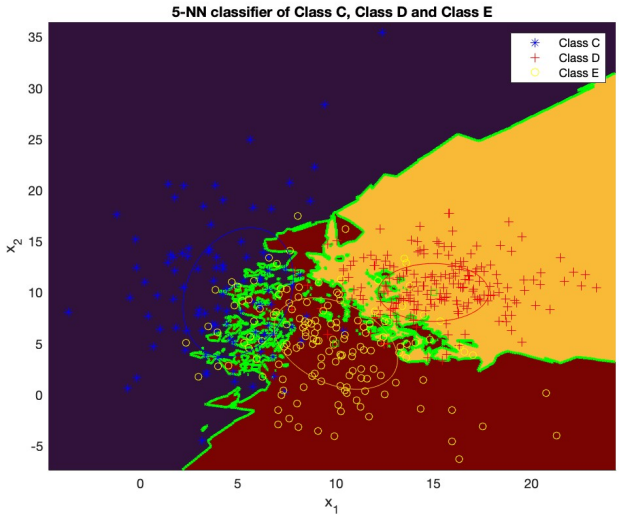

Figure 12: 5-NN classifier for C, D and E classes

In areas containing representatives of different classes, the decision boundary shows a high density in the three-class case.

Classifiers comparison

Each classifier showed a fairly high level of accuracy. The decision boundaries representation on one plot clearly shows the difference in the work of classifiers and their strengths and weaknesses in performance. To evaluate the error and confusion matrix, new test data were generated with the same mean value and covariance matrix, for which all values were calculated. That is, the training and test data are different from each other, but the same for all classifiers, in order to make a correct assessment.

Case 1

Figure 13 shows MED, GED and MAP classifiers for case 1 of two classes with the same covariance matrices and number of samples:

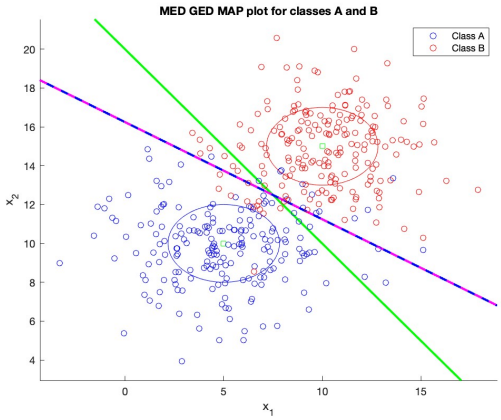

Figure 13: MED, GED, MAP decision boundaries for A and B classes

As was described before, the decision boundaries of the GED and MAP classifiers overlapped, and in the MED classifier the line runs perpendicular to the straight line connecting the mean values. Considering that the MED classifier uses only information about the mean value of class, while the GED uses information about the covariance matrix and the mean, and the MAP in general about the posterior probability, it is obvious that the performance of the MED classifier is worse.

Figure 14 shows NN and 5NN classifiers for case 1 of two classes with the same covariance matrices and number of samples:

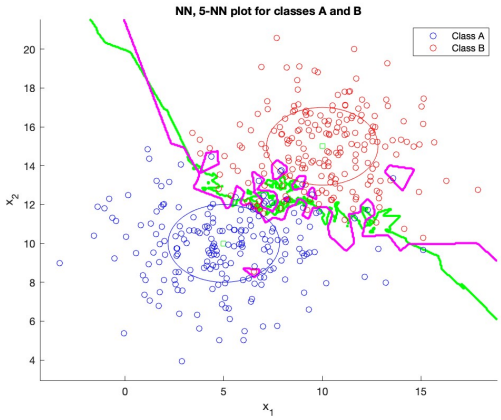

Figure 14: NN, 5-NN decision boundaries for A and B classes

Decision boundaries of KNN classifiers very accurately demarcated clusters of classes, however, with an increase in the number of neighbors from 1 to 5 as class prototypes, decision boundaries

became smoother and the number of sub-boundaries decreased.

In general, all classifiers performed quite well in such a simple case of two classes with the same covariance matrices and the same number of samples. The NN classifier best separated the class clusters, which was to be expected.

Case 2

Figure 15 shows the relative location of the decision boundaries of the MED, GED and MAP classifiers:

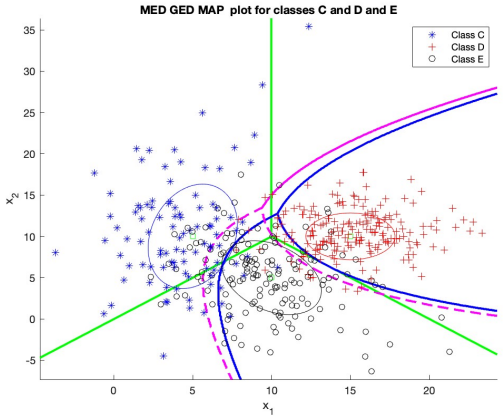

Figure 15: MED, GED, MAP decision boundaries for C, D and E classes

The three-class case with different covariance matrices, different numbers of clusters, and rather close location of class mean values was much more difficult to classify. As expected, among the MED, GED, and MAP classifiers, the first classifier, which separated the classes simply by three straight lines, did the worst. Decision boundaries are non-linear and generally delineate classes quite well.

Figure 16 demonstrates the relative location of the decision boundaries of the NN and 5NN classifiers:

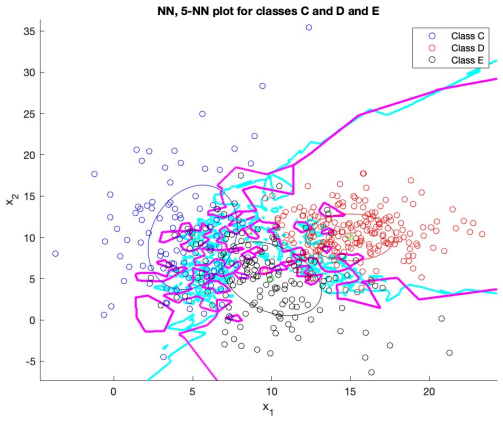

Figure 16: NN, 5-NN decision boundaries for C, D and E classes

These two classifiers coped much worse with such a difficult case, the decision boundary is very non-smooth with a lot of small areas close to each other. However, once again, as the neighborhood increased, so did the performance.

Error analysis

Experimental error is an important indicator for evaluating the performance and accuracy of the classifier. It's formula is

$$\frac{\text{number of misclassified points}}{\text{total number of points}}.$$

In the code, to calculate the experimental error, new test data were generated and each point was passed to the corresponding classifier function, if the classification result was a different class than the one it belongs to, the number of misclassified points increased.

Case 1

MED classifier

The function stored in a **MED_error_2.m** file was written to calculate the experimental error and confusion matrix for the 2 classes. The experimental error for MED classifier in case 1 is

$$P(\epsilon) = 0.0775$$

The confusion matrix is as follows:

	A	B	Total (predicted)
A	188	12	200
B	19	181	200
Total(actual)	207	193	

GED(MICD) classifier

The function stored in a **GED_error_2.m** file was implemented to calculate the experimental error and confusion matrix for the 2 classes. The experimental error for GED classifier in case 1 is:

$$P(\epsilon) = 0.0600$$

And the confusion matrix is:

	A	B	Total (predicted)
A	191	9	200
B	15	185	200
Total(actual)	206	194	

MAP classifier

The function stored in a **MAP_error_2.m** file was implemented to calculate the experimental error and confusion matrix for the 2 classes. The experimental error for MAP classifier in case 1 is:

$$P(\epsilon) = 0.0600$$

And the confusion matrix is:

	A	B	Total (predicted)
A	191	9	200
B	15	185	200
Total(actual)	206	194	

NN classifier

In order to calculate the experimental error and the confusion matrix, new samples were generated satisfying the data of the class distribution and then the accuracy of the classifier was evaluated. The error estimation implementation is stored in **KNN_error_2.m, KNN_error_3.m** files in case of taking 1 nearest neighbor as class prototype. Experimental error of NN-classifier in case of 2 classes is:

$$P(\epsilon) = 0.0900$$

And confusion matrix is:

	A	B	Total (predicted)
A	180	20	200
B	16	184	200
Total(actual)	196	204	

5-NN classifier

After generating new classes samples and testing the accuracy of the classifier, the following data were obtained: the experimental erros is:

$$P(\epsilon) = 0.0700$$

And confusion matrix is:

	A	B	Total (predicted)
A	185	15	200
B	13	187	200
Total(actual)	198	202	

When evaluating the confusion matrices, the classifiers were quite accurate, and when comparing the actual number of points belonging to the class with the number of predicted points, the 5NN classifier is the best. However, NN and 5NN classifiers trained on one data make many mistakes in the classification of test data. The MED classifier has the second highest percentage of experimental error because it uses only the class mean. As a result, MAP and GED classifiers showed the highest accuracy with experimental error 0.0675.

Case 2

MED classifier

The function stored in a **MED_error_3.m** file was implemented to calculate the experimental error and confusion matrix for the 3 classes. The experimental error for MED classifier in case 2 is:

$$P(\epsilon) = 0.2156$$

And the confusion matrix is:

	C	D	E	Total (predicted)
C	72	3	25	100
D	7	175	18	200
E	29	15	106	150
Total(actual)	108	193	149	

GED(MICD) classifier

The function stored in a **GED_error_3.m** file calculates the experimental error and confusion matrix for the 3 classes. The experimental error for GED classifier in case 2 is:

$$P(\epsilon) = 0.1822$$

And the confusion matrix is:

	C	D	E	Total (predicted)
C	85	1	14	100
D	7	175	18	200
E	26	16	108	150
Total(actual)	118	192	140	

MAP classifier

MAP_error_3.m file contains the implementation of calculation experimental error and confusion matrix in case 2 of the MAP classifier. The experimental error is:

$$P(\epsilon) = 0.1956$$

And the confusion matrix is:

	C	D	E	Total (predicted)
C	74	3	23	100
D	1	175	24	200
E	19	18	113	150
Total(actual)	94	196	160	

NN classifier

After generating test data with another seed for the same distribution parameters, classifier was tested on accuracy. Experimental error is:

$$P(\epsilon) = 0.2444$$

And confusion matrix is:

	C	D	E	Total (predicted)
C	61	5	34	100
D	0	170	30	200
E	15	26	109	150
Total(actual)	76	201	173	

5-NN classifier

The results of calculating the accuracy of the 5NN classifier for three classes are as follows: the experimental erros is:

$$P(\epsilon) = 0.2200$$

And confusion matrix is:

	C	D	E	Total (predicted)
C	66	4	30	100
D	0	170	30	200
E	14	21	115	150
Total(actual)	80	195	175	

Looking at the confusion matrices and comparing the expected amount of data with the actual one, NN classifier was the worst and the MED classifier was the closest. However, the level of experimental error is not the smallest - 0.2156. GED and MAP showed quite similar performance, but GED still showed a smaller experimental error - 0.1822 versus 0.1956. In case of NN to KNN classifiers, the value of the experimental error decreased from 0.2440 to 0.2200. However, more than one fifth of the clusters were misclassified. As a result, it can be concluded that preference should be given to MAP and GED classifiers, which showed the best performance and the smallest experimental error.

Conclusions

During the implementation of lab 1, it was determined that the GED and MAP classifiers cope best with the task in comparison with other considered classifiers. Although the NN and KNN classifiers delimit class clusters well, they show a high level of error in the classification of test data.