UNIVERSITY OF WATERLOO

Department of Mechanical and Mechatronics Engineering

LAB 1 REPORT

Prepared for:
SYDE 572

Prepared by:
Group 13

Jae Wan Cho (jw3cho)
Kapilan Satkunanathan (k3satkun)
Shraddha Shah (s222shah)
Anna Shan (yq2shan)

February 26, 2021

1. Introduction

In this report, the characteristics of various classification boundaries will be analyzed using five different classes. The Gaussian distribution parameters for each class can be found in the lab outline. Using MATLAB to generate clusters for each class, the following classification boundaries will be developed for each case: Minimum Euclidean Distance (MED), Minimum Intra-Class Distance (MICD), Maximum A Posteriori (MAP), Nearest Neighbor (NN), and k-Nearest Neighbor (k-NN). By analyzing their plots and error rates, the effectiveness and accuracy of each boundary can be analyzed.

2. Generating Clusters

2.1. Use the Matlab function randn to assist in the generation of the 2D clusters above.

To generate random sample sets for each class, Equation 1 was used:

$$\begin{bmatrix} x_{11} & x_{12} \\ \vdots & \vdots \\ x_{n1} & x_{n2} \end{bmatrix} = \begin{bmatrix} \uparrow & \uparrow \\ u_{x_1} & u_{x_2} \\ \downarrow & \downarrow \end{bmatrix} + \underbrace{\begin{bmatrix} r_{11} & r_{12} \\ \vdots & \vdots \\ r_{n1} & r_{n2} \end{bmatrix}}_{\text{generated by randn}} \begin{bmatrix} \sigma_{x_1}^2 & \sigma_{x_1 x_2}^2 \\ \emptyset & \sigma_{x_2}^2 \end{bmatrix}$$

Equation 1: Transformation equation

Since the values generated by the *randn* function follow a Gaussian distribution centered around 0 with a variance of 1, each set of (x_1, x_2) values needs to be scaled by the covariance matrix. Only the upper triangle of this matrix is used to avoid including the covariance factor twice. Then, the true mean is added to each set. This results in an N×2 matrix where each column contains the data points for the associated variable. This data matrix is saved and reused for all following computations to ensure consistency between sections.

2.2. Plot the samples and the unit standard deviation contour for each of the four classes. Visually, how does the unit contour relate to the cluster data?

To plot the unit standard deviation contours, ellipses centered around the true means are drawn. The lengths of the axes of the ellipse correspond to the square roots of eigenvalues of the covariance matrix (Σ). The angle of the ellipse is calculated using Equation 2:

$$\tan^{-1} \frac{\sqrt{c}}{\sqrt{a}}, \quad \Sigma = \begin{bmatrix} a & b \\ b & c \end{bmatrix}$$

Equation 2: Ellipse equation

The sign of *b* indicates the tilt direction of the ellipse (b > 0 is upwards). The following figures are plotted in MATLAB to group classes A & B and classes C, D, E.

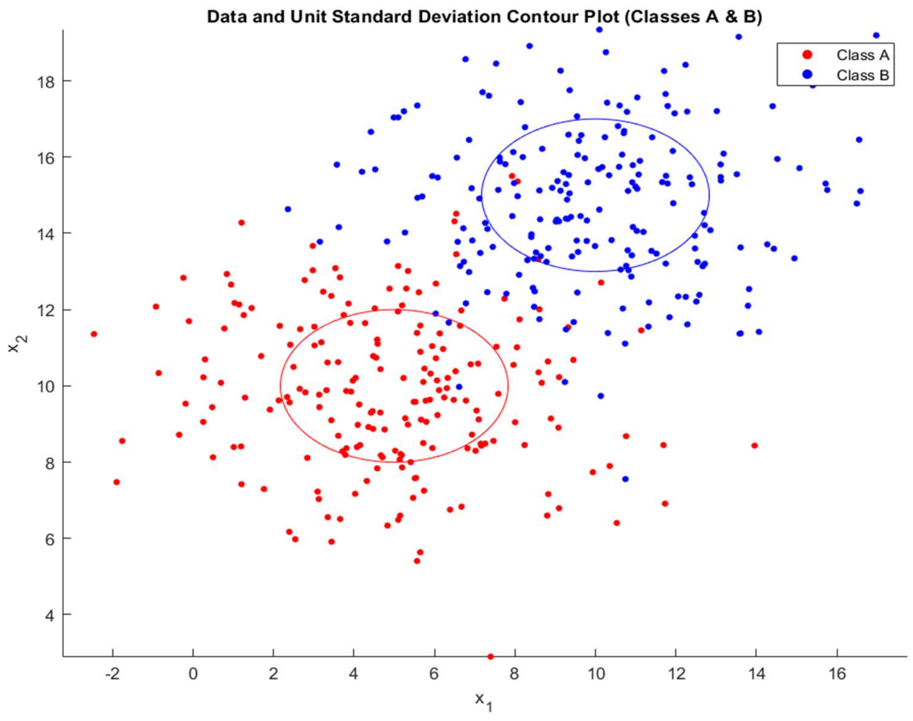


Figure 1: Data and corresponding equiprobability contour plot for Classes A & B

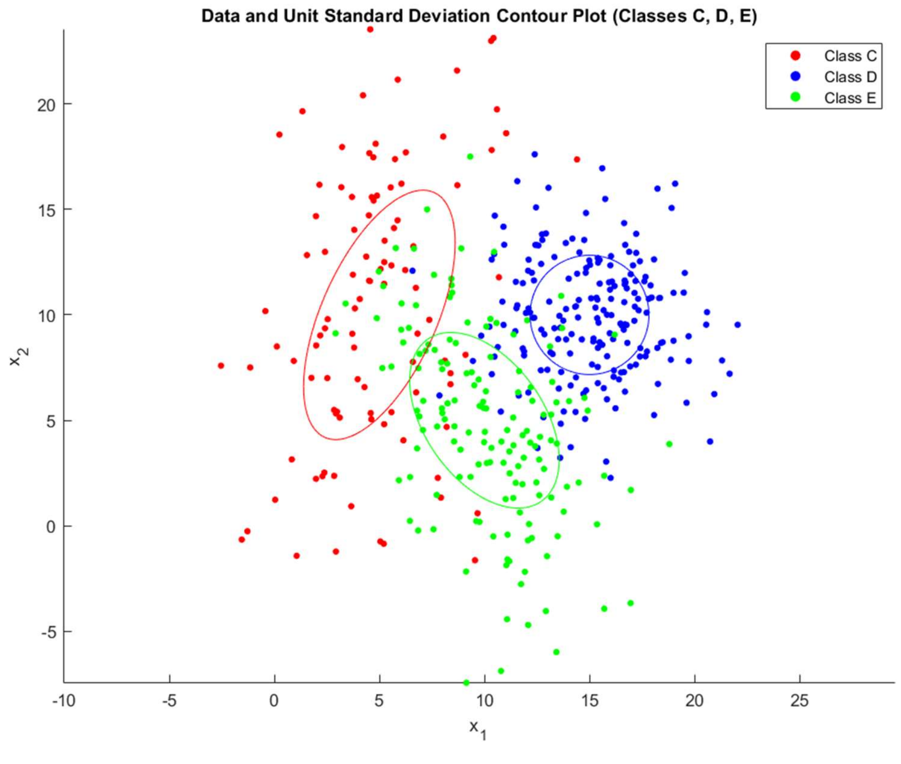


Figure 2: Data and corresponding equiprobability contour plot for Classes C, D & E

Visually, the contours are centered at the class means and closely follow the shape of the data clusters. A difference in variances in x_1 and x_2 results in an oval standard deviation contour instead of a circle (classes A, B, C, E). Non-zero covariance values result in tilted contours (classes C & E).

As well, over half of the data points seem to fall within the contours, where the clusters are more dense than the surrounding area. This observation aligns with the theory that 68.3% of sample points fall within one standard deviation of the mean for a Gaussian distribution.

3. Classifiers

Question: For the two cases, plot the classification boundaries between the classes.

For the following plots, a numerical approach is used by creating 2D grids and classifying each point on the grid using the respective classifier. The grids span the range of sample values for all classes included in the plot with a step size of 0.1. The resulting grid is a matrix composed of integers corresponding to the classes, e.g. 1's & 2's for classes A & B. Then, the contours (classification boundaries) are overlaid onto the data and unit standard deviation contour plots from the previous section. MED, MICD, and MAP boundaries are grouped into one plot while NN and kNN boundaries are displayed on a separate plot, for both cases.

3.1-3.2. Minimum Euclidean Distance (MED) and Generalized Euclidean Distance (GED), using the true means and covariances.

The same procedure is used for both methods. For each case, the MED/GED between all sample points on the 2D grid and each of the true class means is calculated by Equation 3:

$$\text{MED}: \quad d_E(\underset{\text{sample}}{x}, \underset{\text{class mean}}{\mu}) = \sqrt{(x_1-\mu_1)^2+(x_2-\mu_2)^2}$$

$$\text{GED}: \quad d_G(\underline{x}, \underline{\mu}) = \sqrt{(\underline{x}-\underline{\mu})^T S^{-1} (\underline{x}-\underline{\mu})}$$

Equation 3: MED and GED calculation

S is the true covariance matrix of the class. A sample point belongs to the class that minimizes the MED/GED from its mean to the point.

3.3. Maximum A Posterioi (MAP), using the true statistics. Set the a priori class probabilities proportional to the number of samples in each class.

For each point on the 2D grid, the posterior probability for each possible class is calculated in terms of the priors by Equation 4:

$$P(\underset{\text{class } k}{k} \mid \underset{\text{observation}}{\underline{x}}) = \frac{P(\underline{x} \mid k)P(k)}{P(\underline{x})}$$

$$= \frac{1}{\sqrt{2\pi \underset{\text{covariance matrix}}{\det(S)}}} e^{-\frac{1}{2}(\underline{x}-\overset{\text{class mean}}{\mu})^T S^{-1}(\underline{x}-\mu)} \frac{\overset{\text{\# of samples in class k}}{N_k}}{\underset{\text{\# of samples for all classes}}{N_{total}}}$$

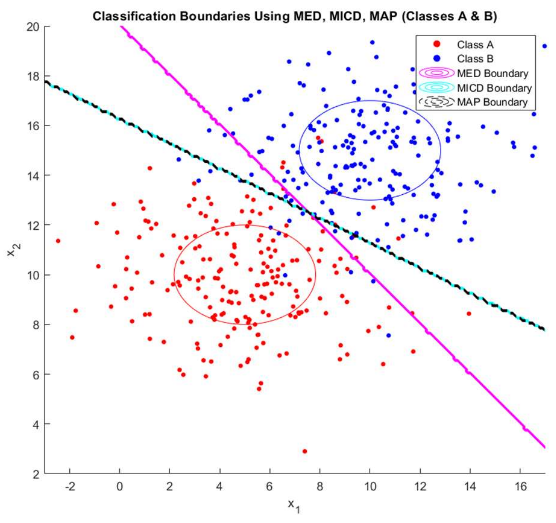
Equation 4: Prior calculation for Gaussian distribution

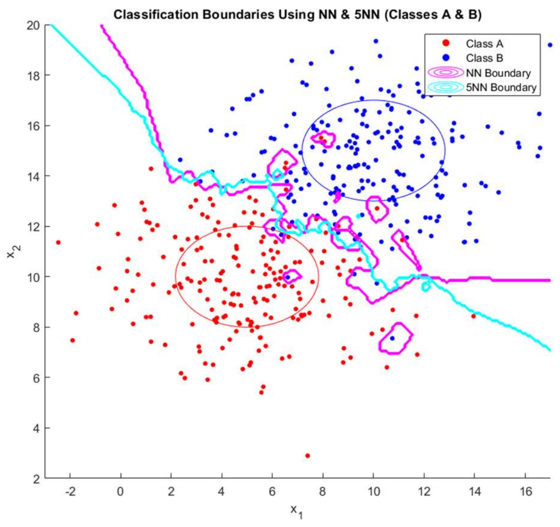
A Gaussian distribution is used. A point belongs to the class with the maximum posterior probability.

3.4-3.5. Nearest neighbor (NN) and k-Nearest neighbor (kNN) for k = 5, using Euclidean distance.

A classification model is built for each case using the *fitcknn* function in MATLAB. This function takes a set of training data points with known responses and builds a model using the Euclidean distance metric. The randomly generated Gaussian sample points for each class are set as the training input and the class is the response. Then, each point on the 2D grid is evaluated using the model through the *predict* function in MATLAB. For NN, the number of neighbours (k) is set to 1.

Question: Comment on the classification boundaries. How do the different boundaries compare?





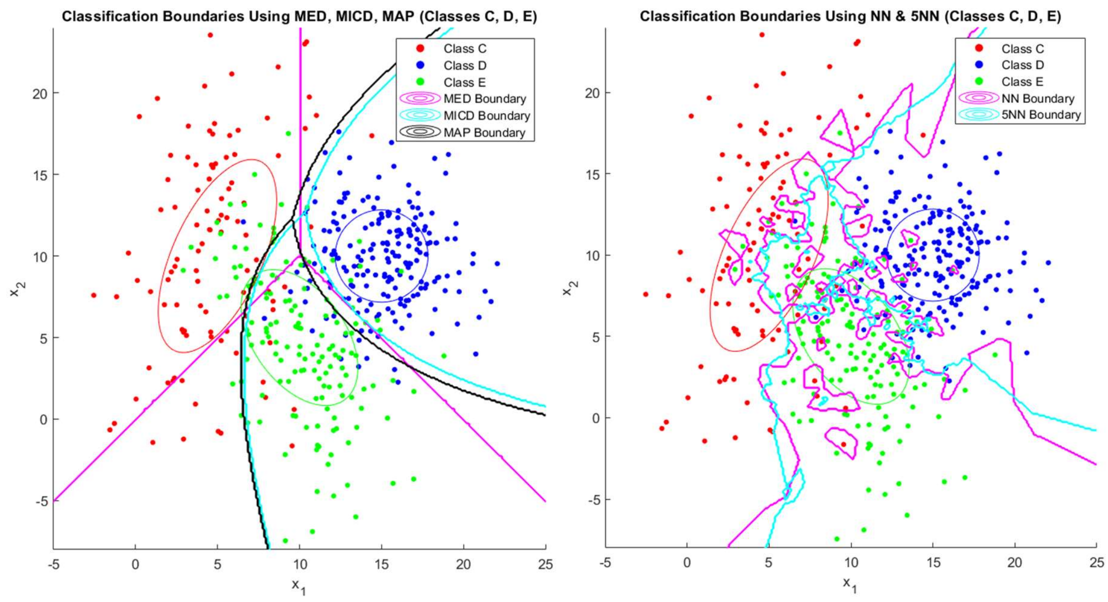


Figure 3: Data and decision boundaries for MED, MICD, MAP, NN, 5NN classifiers for Classes A, B, C, D and E

Through visual inspection of Figure 3, it can be observed that for case 1, the MICD and MAP boundaries are the same. This is due to both classes having the same priori probabilities and covariance matrix. For the second case, the boundaries are very similar, though not the exact same, since the covariance matrix and the priori probability for each class are different. Given that MICD and MED classifiers have the same form when their corresponding measurements follow a normal distribution, it makes sense that their boundaries would be very similar. The MED classifier is somewhat similar to the MICD and MAP classifiers, however it produces a much more "linear" boundary compared to the MICD and MAP boundaries. The difference between the MED classifier and the MICD/MAP classifiers can be explained by how MED classifiers do not take the covariances of any given class into account. All three classifiers produce clear boundaries.

Compared to the other three classifiers, the NN and 5NN classifiers have unclear boundaries since they are much more influenced by measurement noise and outliers. The NN classifier boundary appears to be "noisier" than the 5NN classification boundary. This intuitively makes sense given that the 5NN classifier classifies a data point based on the average distance to the 5 nearest samples in each class. Averaging the samples produces a more accurate classifier.

4. Error Analysis

To calculate the experimental error rate ($P(\epsilon)$), it possible to use the Bhattacharya formula (Equation 5) to obtain the $P(\epsilon)$.

$$P(\epsilon) \leq \sqrt{P(A)P(B)} \int \sqrt{P(x|A)P(x|B)}d\underline{x}$$

Equation 5: Bhattacharya formula

Where the individual probabilities could be calculated through Equation 6:

$$P(A) = \frac{n(A)}{n}$$

Equation 6: Prior calculation

Where n is the total amount of instances for all classes and n(A) is the number of instances for the class of A. This formula could be applied to each class yielding their individual class probability.

As for the integral the probabilities of $P(A)$ and $P(x|B)$could be calculated using Equation 7 for each respective data point in the classes and would be put through a product notation then the square root would be taken. This would then be summed to represent the integral.

$$p(x|A) = \mathcal{N}(\mu_A, \sigma_A^2) = \frac{1}{\sqrt{2\pi}\sigma_A} \exp\left[-\frac{1}{2}(\frac{x-\mu_A}{\sigma_A})^2\right]$$
$$\text{and } p(x|B) = \mathcal{N}(\mu_B, \sigma_B^2).$$

Equation 7: Likelihood calculation with Gaussian distribution

Through this process, $P(\epsilon)$ for each case is calculated and tabulated in Table 1:

	$P(\epsilon)$
Case 1	23.41%
Case 2	0.03%

Table 1: Experimental error rate for each case

Based on Table 1, the second case has lower experimental error.

An example of a confusion matrix can be seen in Figure 4, where the top axis corresponds to the true class and the left axis corresponds to the predicted class. Therefore, if a data point is passed through a classifier and is classified correctly then it will increment the value of the box in the same row and column as the true/predicted class. Note it is for that reason the green boxes are located in the diagonal. If classified incorrectly then the corresponding box is incremented shown in red.

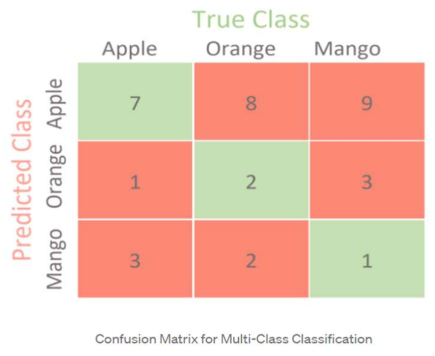


Figure 4: Confusion matrix example

We can apply this process to our generated points of passing through the various classifiers to obtain each classifier's respective confusion matrix.

The confusion matrices for each case for each type of classifier are shown in Table 2 - 11. Note for NN and KNN since they use individual points a separate testing set of data points was used.

<MED>

Case 1		True Class	
		A	B
Predicted Class	A	183	16
	B	17	184

Table 2: confusion matrix for MED case 1

Case 2		True Class		
		C	D	E
Predicted Class	C	76	2	30
	D	4	174	11
	E	20	24	109

Table 3: confusion matrix for MED case 2

<MICD>

Case 1		True Class	
		A	B
Predicted Class	A	182	14
	B	18	186

Table 4: confusion matrix for MICD case 1

Case 2		True Class		
		C	D	E
Predicted Class	C	86	3	23
	D	2	170	10
	E	12	27	117

Table 5: confusion matrix for MICD case 2

<MAP>

Case 1		True Class	
		A	B
Predicted Class	A	182	14
	B	18	186

Table 6: confusion matrix for MAP case 1

Case 2		True Class		
		C	D	E
Predicted Class	C	84	3	22
	D	3	179	16
	E	13	18	112

Table 7: confusion matrix for MAP case 2

<NN>

Case 1		True Class	
		A	B
Predicted Class	A	184	20
	B	16	180

Table 8: confusion matrix for NN case 1

Case 2		True Class		
		C	D	E
Predicted Class	C	78	5	31
	D	1	174	25
	E	21	21	94

Table 9: confusion matrix for NN case 2

<KNN>

Case 1		True Class	
		A	B
Predicted Class	A	190	17
	B	10	183

Table 10: confusion matrix for KNN case 1

Case 2		True Class		
		C	D	E
Predicted Class	C	77	2	22
	D	1	186	24
	E	22	12	104

Table 11: confusion matrix for KNN case 2

From these results, it can be noted from the confusion matrices for Case 2 that class E are getting classified as other classes at a very high rate compared to the other two classes. In addition, both Class C and Class D have a reasonable rate of getting classified properly but when wrongly classified it tends to be classified as Class E.

5. Summary of Results and Conclusions

This lab starts by transforming randomly generated white data that are uncorrelated and equally-variance into 5 classes of correlated data, namely A, B, C, D and E using the given means and covariance matrices. These data are then plotted and an equiprobability contour is drawn for each class of sample data, as shown in Figure 1 and Figure 2. This part of lab shows that if a mean and a covariance matrix of data set is known, it is possible to generate data samples with the same mean and covariance matrix using an orthonormal transformation. Also, it can be noted that equiprobability contour can be used to provide a good visualization of the statistical characteristics of data cluster, such as mean, standard deviation and covariance.

In the next part of the lab, for the two cases (A & B and C, D & E, respectively), the classification boundaries are plotted between the classes using 5 different type of classifiers, namely MED, MICD, MAP, NN and 5NN, as shown in Figure 3. It can be observed that the MED, MICD and MAP classifiers result in clean boundaries whereas the NN and 5NN classifiers result in unclear boundaries. This makes sense because the NN and 5NN classifiers tend to overfit the data and are therefore more susceptible to noises and outliers compared to the other three classifiers. Also, the NN and 5NN classifiers takes much longer time to compute because they require to calculate a new prototype for each new data point. One thing to note is that the 5NN classifier boundary appears to be less noisy than the NN classification boundary. This intuitively makes sense because the 5NN classifier uses the mean of 5 nearest samples to calculate a prototype, which averages out the noise.

Moreover, the MED classifier boundary appears to be more linear than the MICD and MAP classifier boundaries. This makes sense because the MED classifier only takes mean values into account and does not take into account covariances like the MICD classifier or probabilistic behaviours like the MAP classifier. It should be noted that in case 1, the MICD and MAP classifier boundaries are the same because their priori and their covariance matrix (distribution volume) are the same and in case 2 those boundaries are somewhat different because each class has different priori probabilities and covariance matrix (distribution volume) from others. In conclusion, a classifier for a data set should be chosen appropriately based on the statistical and probabilistic characteristics of the data set. In our cases, either the MICD or MAP classifier appears to work better than the other classifiers because each class is Gaussian, not very complex in shape, correlated and has significant noises and outliers.

In the last part of the lab, the experimental error rate and the confusion matrix for each classifier are calculated for each case and tabulated in Table 1 and Table 2 - 11, respectively. As seen in Table 1, the error rate for case 2 is much lower than case 1. Also, by looking at the confusion matrix, a trend can be observed that more data points in class E are getting classified as other classes compared to the data points in the other two classes. Also, the data points in both Class C and Class D have a reasonable rate of getting properly classified but when wrongly classified they tend to be classified as Class E. It can be concluded that the experimental error bound can be found using the Bhattacharya formula and it allows to see the theoretical limit on the expected performance. Moreover, confusion matrix is a great visual tool that quantitatively shows how many data points from each class are classified correctly/incorrectly to what class.