University of Waterloo

SYDE 372: Pattern Recognition

Lab 2: Model Estimation and Discriminant Functions

Team Members:

Adrian Chan | 20548981

Maria Cheng | 20567883

Devika Khosla | 20570142

Michael Lim | 20567193

Vyshakh Sanjeev | 20554572

Table of Contents

List of Figures

1. Introduction

In this lab, given data, we demonstrate class conditional probability density functions (PDF) estimation by parametric and non-parametric means. Parametric estimation is performed by estimating the PDFs in both one-dimensional and two-dimensional cases by assuming a functional form of the PDF. In this report, we study the effects of assuming various functional forms of the PDF (gaussian, exponential, and uniform) and its effects on the performance of the classifier. For comparison, the PDF is also estimated by non-parametric estimation, with Gaussian Parzen windows, and advantages and disadvantages of each approach are discussed. Furthermore, we also study the effect of the standard deviation of the Gaussian window used. In the latter half of the lab, a sequential classifier is defined through aggregation of multiple minimum Euclidean distance (MED) classifiers, and the performance of the learned classifiers is discussed.

2. Model Estimation 1-D Case

In this section, we attempt to learn the probability density functions (PDFs) of two different datasets using a variety of parametric estimation methods as well as non-parametric (Parzen window) estimation. The provided datasets have the following properties:

- Variable a: Gaussian samples with $\mu = 5$ and $\sigma = 1$
- Variable b: Exponential samples with $\lambda = 1$

2.1 Parametric Estimation: Gaussian

By assuming the unknown PDFs follow a Gaussian distribution, and that the 1-dimensional samples are independent, we expect the PDF to have the form,

$$P(\{x_i\} \mid \theta) = \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi}\sigma} e^{-\left(\frac{x_i-\mu}{\sigma}\right)^2}$$

where θ is the parameter set μ and σ, which are the mean and standard deviation of the PDF, respectively.

The maximum likelihood (ML) estimate attempts to maximize the likelihood of the data given a set of parameters. Thus, by setting the first derivative of the above distribution to zero, one can learn the parameters μ and σ:

$$\frac{dP(\{x_i\} \mid \theta)}{d\theta} = \frac{d}{d\theta}\left[\prod_{i=1}^{N} \frac{1}{\sqrt{2\pi}\sigma} e^{-\left(\frac{x_i-\mu}{\sigma}\right)^2}\right] = 0$$

Since working with logarithms (to the base e – natural log) are more convenient when handling exponentials, and the logarithmic function is monotonic, we can use the log form of the PDF. Furthermore, using the product property of logarithms, we can change the product above to a summation and thus, simplify this to:

$$\frac{dP(\{x_i\} \mid \theta)}{d\theta} = \frac{d}{d\theta}\left[\frac{1}{2}\sum_{i=1}^{N}\left(\frac{x_i-\mu}{\sigma}\right)^2 + \ln(\sqrt{2\pi}\,\sigma)\right] = 0$$

We can then take the derivative of the components and solve for each component to get,

$$\hat{\mu}_{ML} = \frac{1}{N}\sum_{i=1}^{N} x_i$$

$$\hat{\sigma}^2{}_{ML} = \frac{1}{N}\sum_{i=1}^{N}(x_i - \hat{\mu}_{ML})^2$$

Although the ML estimate of the mean is unbiased, the ML estimate of variance is biased. To get the unbiased variance, we simply multiply the variance by a factor of $\frac{N}{N-1}$ and thus, can define the unbiased ML estimate of the variance to be,

$$\hat{\sigma}^2{}_{ML} = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \hat{\mu}_{ML})^2$$

The two parameters can then be estimated for the two datasets to be:

- Variable a: $\hat{\mu}_{ML} = 5.0763$ and $\hat{\sigma}_{ML} = 1.0671$
- Variable b: $\hat{\mu}_{ML} = 0.9633$ and $\hat{\sigma}_{ML} = 0.9343$

The results are shown below in Figure 1. As expected, the ML approach assuming a Gaussian PDF learns dataset a, which was known to have a Gaussian distribution, well, but does not adequately learn the distribution of dataset b, which we know to have an exponential distribution.

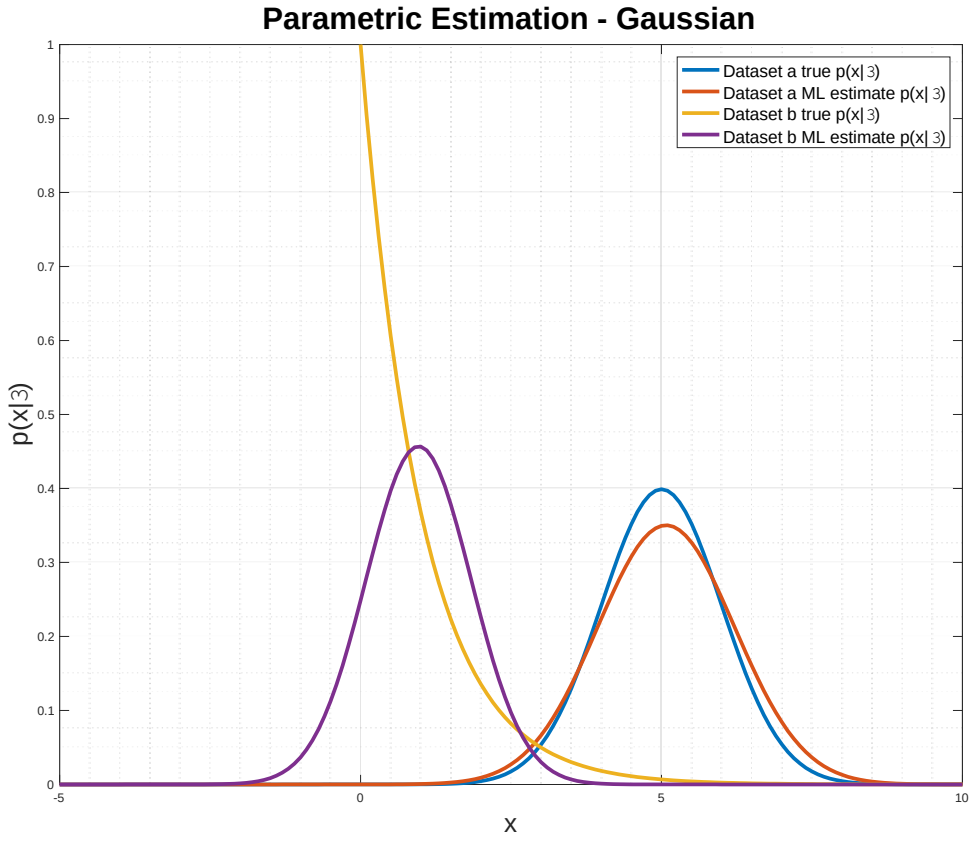

Figure 1. Parametric Estimation assuming a Gaussian distribution for datasets a and b

2.2 Parametric Estimation: Exponential

By assuming the unknown PDFs follow an exponential distribution, and that the 1-dimensional samples are independent, we expect the PDF to have the form,

$$P(\{x_i\} \mid \lambda) = \prod_{i=1}^{N} \lambda e^{-\lambda x} \; H(x)$$

where λ is the rate parameter, and $H(x)$ is the Heaviside step function. Given that in both datasets a and b, we only have samples greater than zero, we can simply write this as,

$$P(\{x_i\} \mid \lambda) = \prod_{i=1}^{N} \lambda e^{-\lambda x}$$

Once again, by setting the first derivative of the above distribution to zero, we can learn the parameter λ.

$$\frac{dP(\{x_i\} \mid \lambda)}{d\lambda} = \frac{d}{d\lambda}\left[\prod_{i=1}^{N} \lambda e^{-\lambda x}\right] = 0$$

Like before, working with logarithms is more convenient in this situation, and by taking the log form of the PDF, we get:

$$\frac{dP(\{x_i\} \mid \lambda)}{d\lambda} = \frac{d}{d\lambda}\left[\ln(\lambda) - \sum_{i=1}^{N} \lambda x\right] = 0$$

We can then take the derivative of λ and solve it to get:

$$\hat{\lambda}_{ML} = \frac{N}{\sum_{i=1}^{N} x_i}$$

Since $E\left[\hat{\lambda}_{ML}\right] = \lambda$, the ML estimate of λ is unbiased and thus, it can be estimated for the two datasets to be:

- Variable a: $\hat{\lambda}_{ML} = 0.1970$
- Variable b: $\hat{\lambda}_{ML} = 1.0381$

The results are shown below in Figure 2. As expected, the ML approach assuming a Gaussian PDF learns dataset a, which we know to have a Gaussian distribution, poorly, but learns the distribution of dataset b, which we know to have an exponential distribution, much better, resembling it very closely.

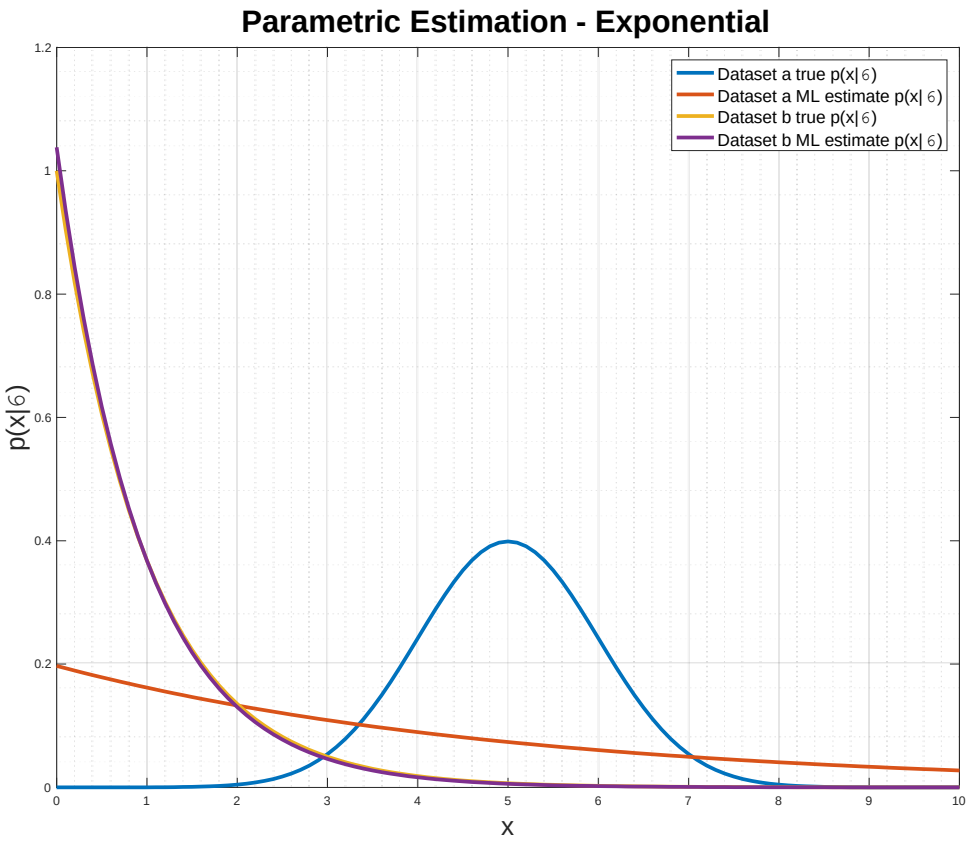

Figure 2. Parametric Estimation assuming an Exponential distribution for datasets a and b

2.3 Parametric Estimation: Uniform

By assuming the unknown PDFs follow a Uniform distribution, and that the 1-dimensional samples are independent, we expect the PDF to have the form,

$$P(\{x_i\} \mid \theta) = \begin{cases} 0, & x < \alpha \\ \prod_{i=1}^{N} \frac{1}{\beta - \alpha} = \frac{1}{(\beta - \alpha)^N}, & \alpha \leq x \leq \beta \\ 0, & x > \beta \end{cases}$$

where θ is the parameter set α and β, which are the two bounds of the uniform distribution, beyond which it is zero. Since the uniform distribution must span the dataset, and the function is defined to be a 0 outside the two bounds, it must be true that $\beta \geq x_{max}$ and $\alpha \leq x_{min}$. From the PDF shown above, we immediately we can see that this is a decaying function, and thus is maximized when $(\beta - \alpha)$ is as small as possible. Combining the two conditions together, we get that,

$$\hat{\beta} = x_{max}$$

$$\hat{\alpha} = x_{min}$$

Since both α and β are unbiased, it can be estimated for the two datasets to be:

- Variable a: $\hat{\alpha} = 2.7406$ and $\hat{\beta} = 8.3079$
- Variable b: $\hat{\alpha} = 0.0143$ and $\hat{\beta} = 4.2802$

The results are shown below in Figure 3. As expected, the ML approach assuming a uniform PDF learns both datasets a and b (which we know to be Gaussian and Exponential distributed, respectively), very poorly.

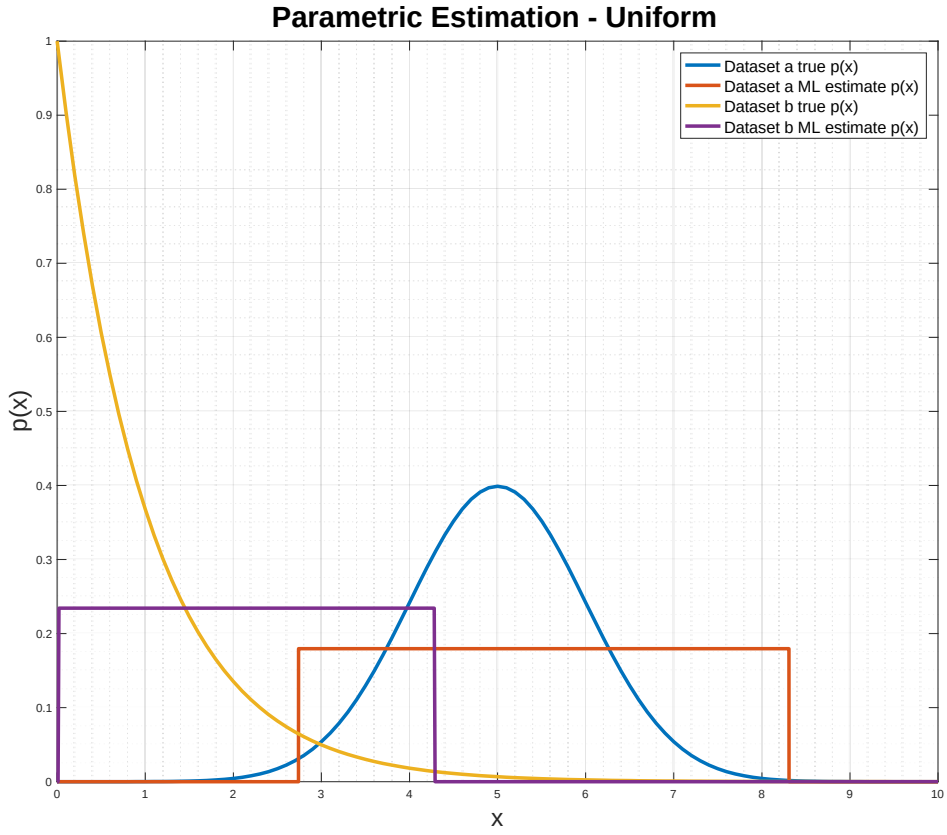

Figure 3. Parametric Estimation assuming a Uniform distribution for datasets a and b

2.4 Non-parametric Estimation: Parzen Window

Unlike in parametric estimation methods, where a class condition PDF was assumed, in non-parametric estimation, we assume that we have no information regarding the class conditional PDF. Thus, non-parametric estimation attempts to estimate the distribution directly from the samples. The Parzen window estimation is a type of non-parametric estimation, which relies on that fact that the greater the sample density, the larger the PDF in that region. In other words, the PDF at any point must be directly proportional to the local sample density at that point. Thus, by estimating the PDF to be the sum of all the samples within a local window, we can recreate the PDF.

$$\hat{P}(x) = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{h}\phi\left(\frac{x - x_i}{h}\right)$$

where ϕ is the window function and h is a scaling factor that can stretch or compress the window function to change the locality of influence of a sample. Here, we use the Gaussian function with a standard deviation of $\sigma = 0.1$ and $\sigma = 0.4$ for the window function and set the scaling factor, h, to 1. The results are shown below in Figures 4 and 5.

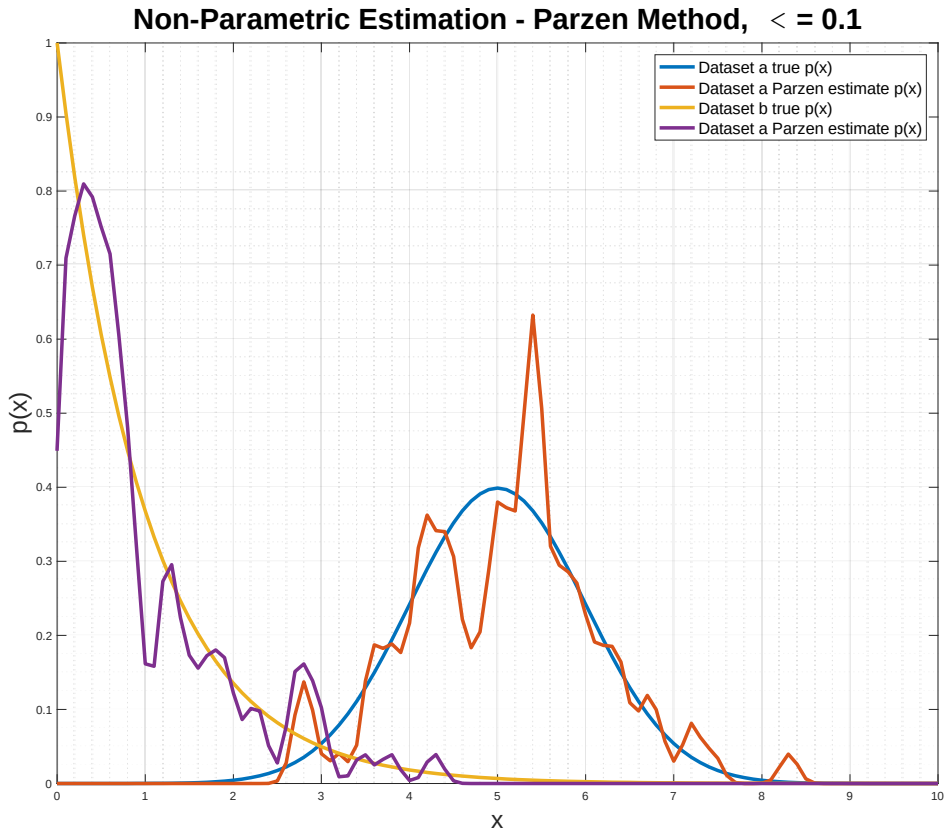

Figure 4. Non-parametric estimation using Parzen window estimation, with σ=0.1, for datasets a and b

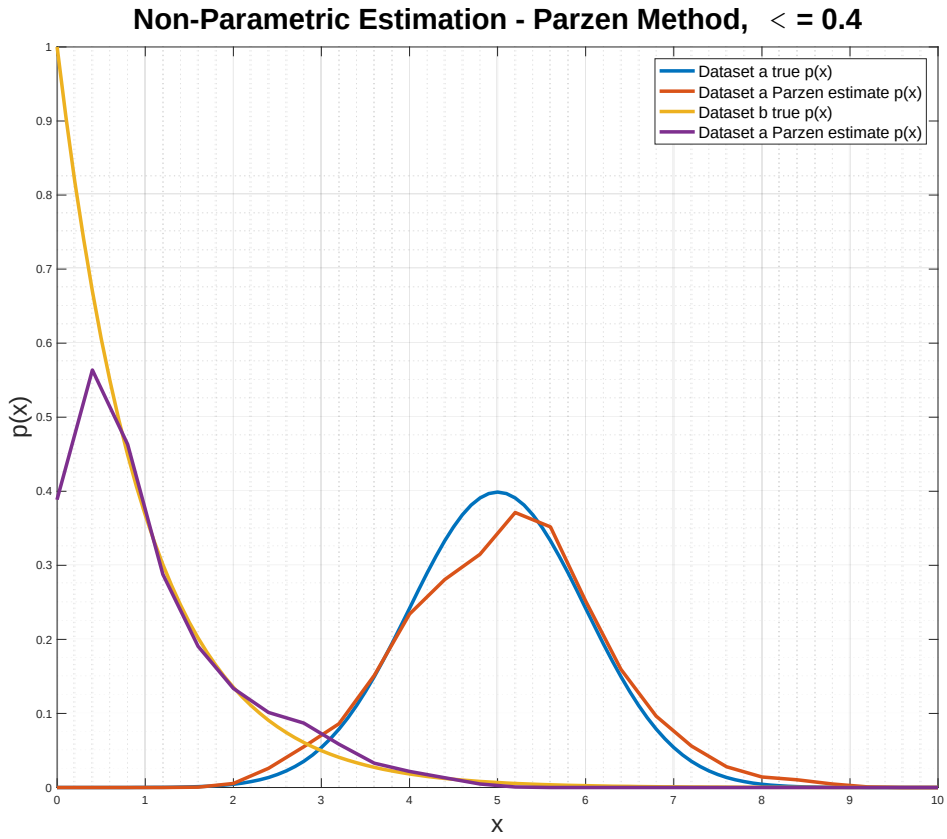

Figure 5. Non-parametric estimation using Parzen window estimation, with σ=0.4, for datasets a and b

From the plots above, we see that the Parzen window estimation learns the PDF quite well as it can be seen to closely resemble the true PDF of the class. It is however evident that the estimated PDF in Figure 4 appears to be noisier than that seen in Figure 5. This is because of the standard deviation of the Gaussian window. When the standard deviation is small, local sample densities have a very strong effect and thus you see peaks where there are more samples. This makes the PDF estimate noisy. By increasing the standard deviation, the method is less sensitive to highly localized densities and effectively looks at a wider range, thus smoothing out any noise in the sample. Interestingly, the Parzen window estimation method can be seen to perform poorly for very small values of x in the case of the exponential function. This is because we are using a Gaussian window on a rapidly decaying function. Since the exponential distribution decays very quickly, a large standard deviation will not be sensitive enough to rapid change as it will include both a high and low density of samples within the same window. Thus here, we can see the lower value of standard deviation, $\sigma = 1$, performs better.

The window function can be thought to work like a low-pass filter or a moving average and thus increasing the window size can be thought of as reducing the noise. Like in a moving average, too small a window yields in a noisy output whilst too large a window will wash out all resolution. Thus, having to optimize the window parameters to appropriately represent an unknown dataset is a drawback of the Parzen window estimation.

In the case of datasets a and b, $\sigma = 0.4$ can be deemed to better represent the true PDFs than $\sigma = 0.1$, expect for small values of x, $x < 0.1$, in dataset b, where $\sigma = 0.1$ performs slightly better than $\sigma = 0.4$.

2.5 Analysis

From the results above, it is evident that the dataset a, which we know has Gaussian distributed samples, is best represented by the parametric estimation method assuming a Gaussian PDF. Since the assumption matches the true distribution, the estimation method learns the parameters quite accurately. The estimated

parameters $\hat{\mu}_{ML} = 5.0763$ and $\hat{\sigma}_{ML} = 1.0671$ are very close to the true value of $\mu = 5$ and $\sigma = 1$. Assuming an exponential or a uniform distribution for the PDF yield in very poor estimation of the true PDF, as seen above. This is however expected as neither of these can represent a Gaussian distribution. The Parzen window estimation method was also able to closely resemble the true PDF. It was seen that with $\sigma = 0.1$, the estimate was noisy, but with $\sigma = 0.4$, the estimate seemed more accurate. This is mainly because the Gaussian distribution in dataset a has a standard deviation of 1, which is smoothly varying with respect to the standard deviations used for the Parzen window. Thus, the larger of the two standard deviations was able to wash out the noise in the sample whilst adhering to the inherent shape of the true distribution.

Similarly, for dataset b, which we know has exponential distributed samples, is best represented by the parametric estimation method assuming an exponential PDF. The estimated parameter $\hat{\lambda}_{ML} = 1.0381$ is very close to the true value $\lambda = 1$. Like in dataset a, the Parzen window estimation was able to estimate the PDF quite well, showing lower noise with $\sigma = 0.4$ than $\sigma = 0.1$, like in the case of dataset a. The reasoning is again similar to that above, where the smoothly varying distribution is better represented by $\sigma = 0.4$ that can wash out the sample noise. However, at small x values, the distribution changes too rapidly, compared to the standard deviation of the window and thus, in this region, the small standard deviation, $\sigma = 0.1$, better represents the PDF.

In general, it is not always possible to use a parametric estimation approach. In order to use the parametric estimation approach, the class conditional PDF of the samples must be known, and in these cases, the parametric estimation approach is certainly recommended over the non-parametric estimation approach. In many practical situations however, the class conditional PDF of the samples is unknown. In such cases, the non-parametric estimation approach must be used. Evidently from the results above, assuming the wrong distribution for a parametric estimation approach results in very poor estimation of the true PDF, and thus, non-parametric estimation approach would be preferred when the functional form of the PDF is unknown.

3. Model Estimation 2-D Case

In this section, we attempt to learn the probability density functions (PDFs) of three different 2-dimensional datasets using a parametric estimation method, assuming a Gaussian distribution of samples, as well as a non-parametric (Parzen window) estimation. Unlike the previous case, here the true distributions of samples are unknown.

3.1 Parametric Estimation: Gaussian

We first use a parametric estimation method, assuming the samples are Gaussian distributed. The sample mean and sample covariance can then be computed to estimate the class conditional PDF. Since 3-dimensional plots are often messy and difficult to wrap our heads around, we generate a ML decision boundary instead.

By assuming the 2-dimensional samples are independent, the PDF will have the form,

$$P(\{x_i\} \mid \theta) = \prod_{i=1}^{N} \frac{\Sigma^{\frac{1}{2}}}{2\pi} \exp\left[-\frac{1}{2}(x_i - \mu)^T \Sigma^{-1}(x_i - \mu)\right]$$

where θ is the parameter set, and μ and Σ, are the sample mean and sample covariance of the PDF, respectively. Here, x_i are the 2-dimensional samples, and thus it is a vector.

Similar to the 1-dimensional case, we first take the logarithmic form of the PDF and then set the derivative with respect to each component to zero to get,

$$\hat{\mu}_{ML} = \frac{1}{N}\sum_{i=1}^{N} x_i$$

$$\hat{\Sigma}_{ML} = \frac{1}{N}\sum_{i=1}^{N} (x_i - \hat{\mu}_{ML})(x_i - \hat{\mu}_{ML})^T$$

The two parameters can then be estimated for the three datasets to be:

- Cluster al:

$$\hat{\mu}_{ML} = \begin{pmatrix} 347.16 \\ 131.20 \end{pmatrix}$$

$$\hat{\Sigma}_{ML} = \begin{pmatrix} 1749 & -1594.5 \\ -1.594.5 & 3310.1 \end{pmatrix}$$

- Cluster bl:

$$\hat{\mu}_{ML} = \begin{pmatrix} 291.84 \\ 224.02 \end{pmatrix}$$

$$\hat{\Sigma}_{ML} = \begin{pmatrix} 3282.6 & 1164.3 \\ 1164.3 & 3379.8 \end{pmatrix}$$

- Cluster cl:

$$\hat{\mu}_{ML} = \begin{pmatrix} 119.55 \\ 346.67 \end{pmatrix}$$

$$\hat{\Sigma}_{ML} = \begin{pmatrix} 2711.1 & -1313.9 \\ -1313.9 & 1682.3 \end{pmatrix}$$

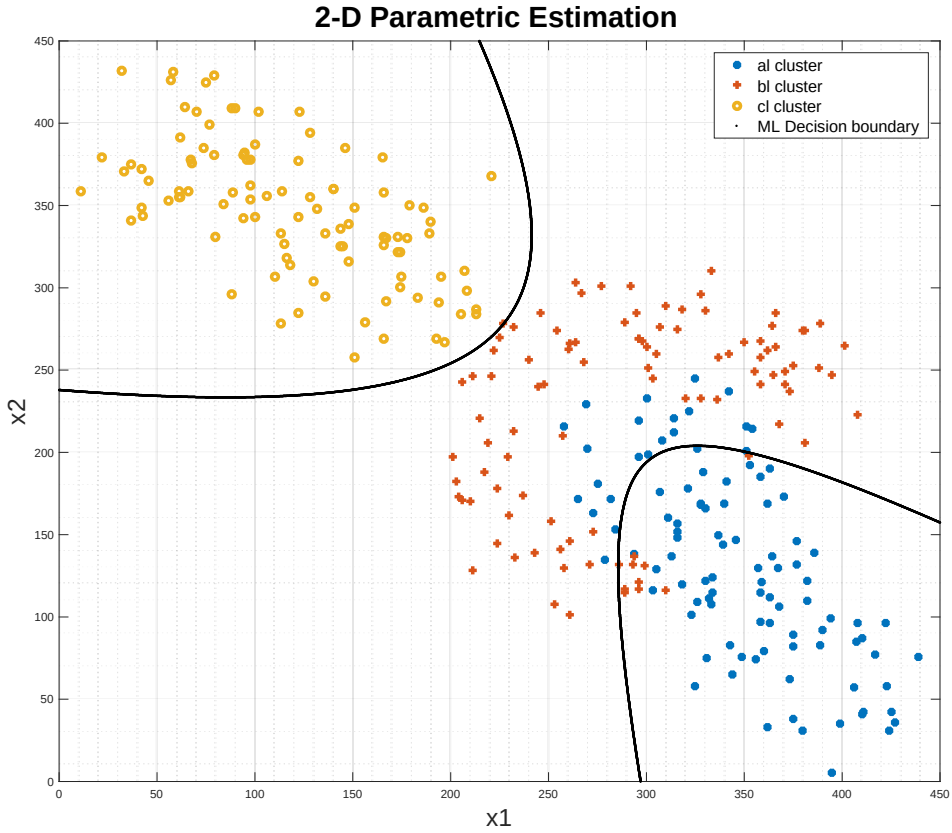

Figure 6. Parametric Estimation assuming a Gaussian distribution for datasets a and b

The parameters above then define the Gaussian PDF estimate and can be used to calculate the likelihood of each class. In order to generate the decision boundaries numerically, we first discretize the x_1-x_2 plane and calculate the likelihood, $P(x \mid \theta)$, at each of the points. By then identifying which of the three PDFs is the greatest at every point, we can assign each to one of the three classes. This can be done using a *max* function. We then use an edge detection filter to identify the decision boundaries. Here, we use the *Canny* built-in function on MATLAB to generate the decision boundary seen above in Figure 6.

3.2 Non-parametric Estimation: Parzen Window

The decision boundaries in Figure 6 appear to be fairly accurate, but evidently misclassify several points. It is however important to note that although clusters *al* and *cl* can be visually seen to resemble a correlated Gaussian distribution, cluster *bl* does not appear to be Gaussian distributed. Since the true distribution of the clusters are unknown, naively assuming them to be Gaussian distributed, particularly when one of the clusters don't appear to be, leads to incorrect boundaries resulting in wrong classification.

Thus, here we attempt to use a non-parametric approach using the Parzen window estimate. We use a Gaussian window function with a standard deviation of 20 to estimate the PDF of each of the three clusters. The function to implement the 2-D Parzen window is provided on the course website. This function takes in the cluster, the resolution, which was set to 1, and the window, which was input as a 1-dimensional vector that was generated using the *gausswin* built-in function on MATLAB. The decision boundaries were then identified the same way it was for the parametric estimation approach described above, where the x_1-x_2 plane was first discretized and the likelihood, $P(x \mid \theta)$, at each of the points for all three clusters were calculated. Each point was then assigned one of the three classes using a *max* function and the boundaries were identified using the *Canny* edge detection filter. The result can be seen below in Figure 7.

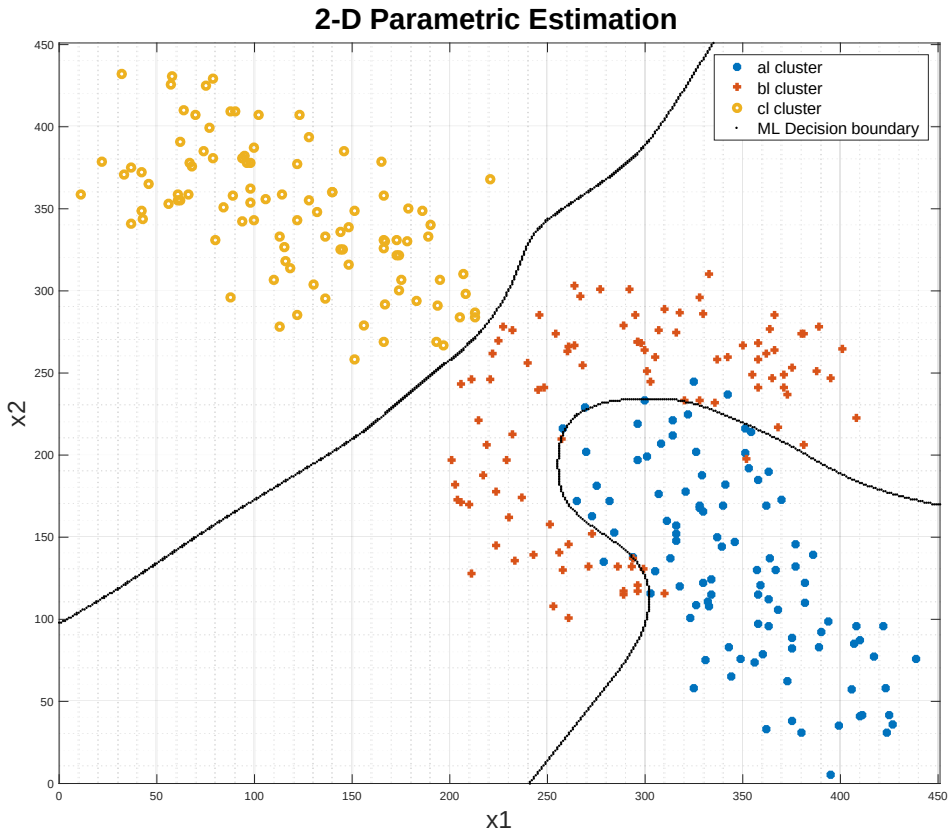

Figure 7. Non-parametric Estimation using a 2-D Parzen window estimation method, with σ=20

From Figure 7, we can see that the non-parametric approach misclassifies much fewer sample points compared to the parametric approach. By not assuming a functional form of the PDF and solely relying on the local sample density to represent the PDF, we have created a more robust classifier. This is particularly advantageous as the true distribution of the classes is unknown, making the non-parametric approach the optimal choice.

3.3 Analysis

Here, we have 3 clusters where the PDF of the three are unknown. First, we try to estimate the PDFs using a parametric estimation method, assuming a Gaussian PDF. Clusters *al* and *cl* appear to have a correlated Gaussian distribution and thus the samples are well classified using this approach. Cluster *bl* however can be seen not to have a Gaussian distribution. Thus, this approach is not adequate for it and we clearly see that the classifier misclassifies several samples. The greater the deviation of the true PDF from the Gaussian assumption, the worse is the performance of the classifier.

Comparing this to the non-parametric approach using Parzen window estimation, we see that the performance of the classifier is much better. This is because the Parzen window estimation approach only depends on the local sample densities in determining the PDF at that point. Unlike in a parametric estimation approach, no assumptions on the true PDF are being made here.

Thus, it can be concluded that unless the distributions of the samples are already known or can be assumed to be of a certain functional form, based on the samples, with high confidence, a non-parametric approach is recommended. Faulty assumptions of the PDF would lead to poor performance when using a parametric approach, as seen here, where numerous samples are misclassified. The non-parametric approach using Parzen window estimation does however have the drawback of having to choose an appropriate window

function and scaling factor. In cases where the functional form of the PDF is known or assumed given the data, a parametric approach is recommended.

4. Sequential Discriminants

Data points belonging to two different classes were used to build a sequential classifier consisting of many discriminants, G, such that $P(true\ class\ is\ C_j \mid G\ says\ C_i) = 1$ for at least one class, C_i. This was done by randomly selecting one point from class A, one point from class B and creating an MED discriminant using the two points as prototypes. Using this decision boundary, every data point in the two classes were classified, and the error was calculated, i.e. $P(true\ class\ is\ C_A \mid G\ says\ C_B)$. If no points from either class were correctly classified in its entirety, two new points were randomly selected as the prototype, one from class A and one from class B, another discriminant was generated, and the classification error was calculated again. Otherwise, the discriminant and the number of points misclassified are stored. Whichever class was correctly fully classified, those points are removed from the pool of available points, and the process is restarted by selecting two new random points, one from both class A and B. In the end, there will be a list of discriminant functions with which new data sets can be classified.

New data sets are classified by applying the discriminant functions in the order they were learned. For example, if there are N discriminant functions, the new data set is classified using $G_1 \rightarrow G_N$. G_1 is applied to the new data set and points are classified depending on whether this discriminant misclassified points from class A or class B during the learning stage (i.e. if $n_{a,B} = 0$, classify as B, if $n_{b,A} = 0$, classify as A). The points that are classified correctly are removed from the process, and the remaining points are classified using the remaining discriminant functions.

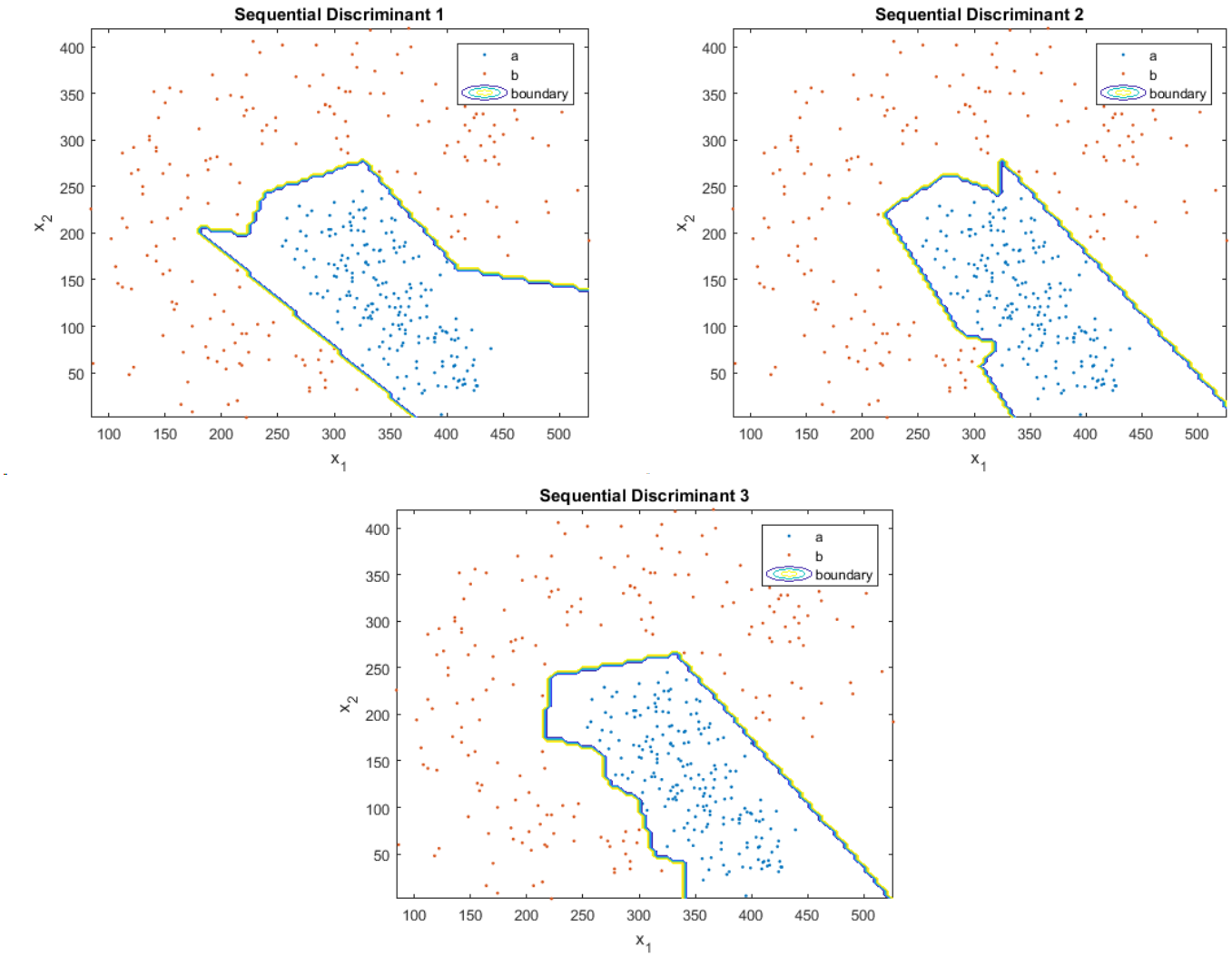

Figure 8. Learned Sequential Classifiers

The plots above show three sequential classifiers that were learnt. All of them completely separate class A from class B, providing a perfectly fitted boundary between the two classes.

If the sequential classifiers are tested with training data, the probability of error will be 0 because the classification boundaries are perfectly fitted to the data.

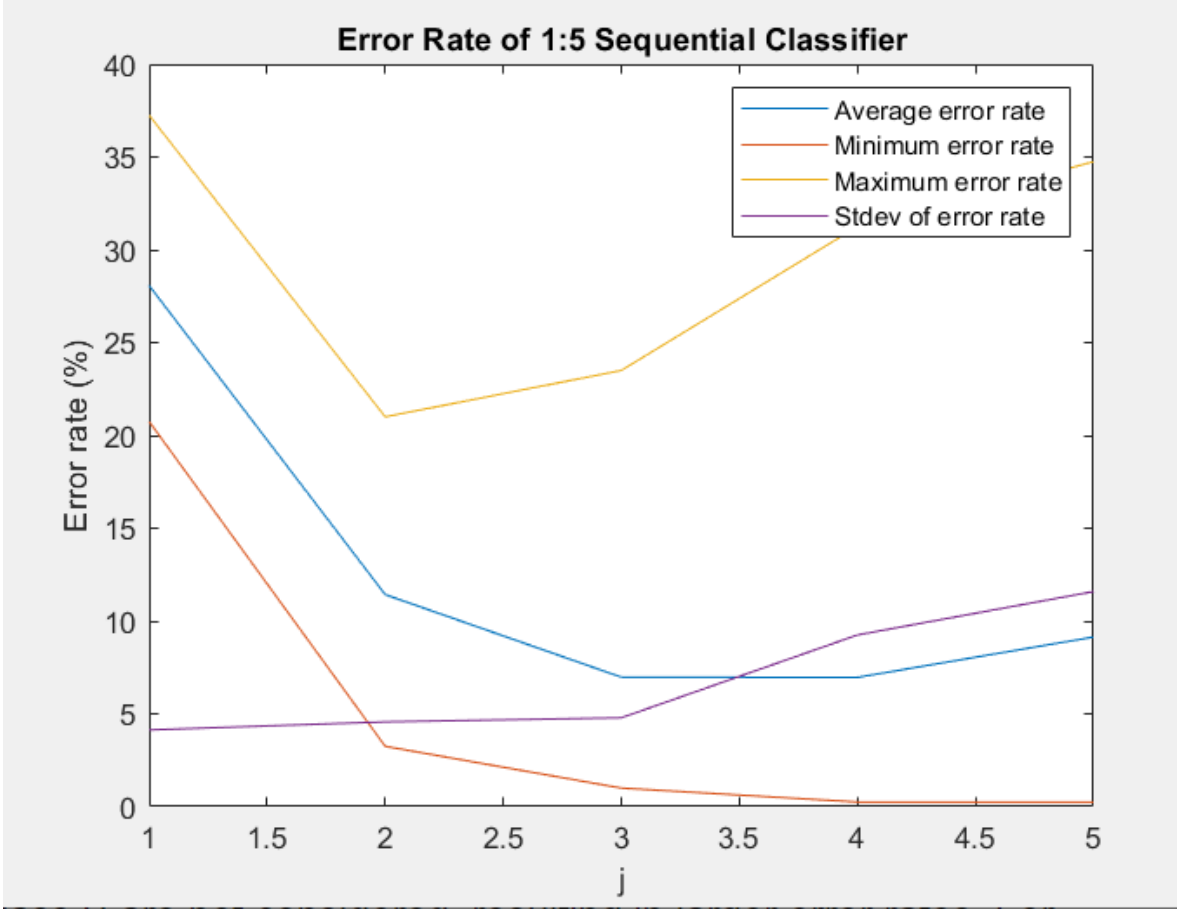

Figure 9. Statistical behaviour of error rates

The figure above shows the statistical behaviour of error rates when limiting the number of classifiers, 'J', when learning sequential classifiers. The mean error rate shows an initial decrease, which was expected since if there are too few discriminants, the shape and span of the class distributions aren't entirely captured, resulting in poor classification performance. The average error rate increased slightly from J = 4 to J = 5, and this can be attributed to overfitting the classifier to the training data which occurs with a larger number of sequential classifiers. Briefly, given the shape of the data, one can see that at minimum 3 discriminants are needed to create an accurate decision boundary. At J=4, and J=5, it is possible that most of the points are already classified with the first 3 discriminants, so additional discriminants can land in less useful places, creating more misclassified points in one set (but correctly classifying a few points from the other set). Note that the error was simply calculated as the number of misclassified points at the last iteration (error = $(n_{aB}+n_{bA})$/(total number of points)).

For each J, 20 sequential classifiers were learnt, and the minimum and maximum error rates resulting from testing all 20 classifiers were extracted. The minimum error rate was observed to decrease while the maximum error rate decreased from J = 1 to J = 2, then continued to rise through to J = 5. The reason for this is because with more sequential classifiers, there is more potential for a larger variation in error. To elaborate, for J = 1, it's unlikely that the one discriminant generated performs well as a classifier, however J = 1 discriminants will consistently perform badly, resulting in a small variance in error rates. Contrarily, for J = 5, there is ample opportunity to generate both effective and ineffective discriminant classifiers, resulting in a larger variation of error rates. This variation in error rates is reflected through the total difference between the maximum and minimum error rates - the total range of error rates. For J = 1, the difference between the max and min error rates is at a minimum while for J = 5, this difference is at a maximum. This variation in error rates is better represented through the standard deviation which is seen to be relatively constant from J = 1 to J = 3, then increases through the J = 5. Again, this is because with a larger number of sequential classifiers, there is more variation in the classification performance- there is more opportunity to learn both good and bad discriminant functions.

If the number of points pairs are limited when building the sequential classifier, it's possible to prevent overfitting the classifier to the training data. This could result in better performance and lower error rates when applying the classifier to other data sets, since the classifier wouldn't be constrained and overfitted to the training set. However, if there were too few points, the classifier may never converge.

5. Conclusion

In conclusion, from the first two questions, we learn that parametric estimation is an appropriate method when the functional form of the PDF is known or can be assumed with high confidence based on the sample points. However, if the assumed PDF is wrong, it leads to a poor classifier that misclassifies the data. Thus, in cases where the form of the PDF is unknown, the non-parametric estimation approach is certainly more appropriate. Here, for all non-parametric estimation approaches, we use the Parzen window estimation method. For the 1-dimensional case, we studied the effect of the standard deviation of the Gaussian function used for the window function – too small a standard deviation results in a noisy estimation of the PDF, with spikes at the samples. A sufficiently large standard deviation can better encapsulate the inherent distribution of the samples. However, when the true distribution changes rapidly, like at very small values of the exponential distribution, a small standard deviation better approximates the true PDF. This also highlights a drawback in using the Parzen window estimation, which is the choice of the window function.

When learning sequential classifiers, if the number of discriminants is not limited, the classification boundaries perfectly fit the training data. When the number of discriminants is limited, the classification error rates increase, decreasing the performance of the classifier. However, after a certain threshold, a greater number of sequential classifiers increases the error rate slightly due to overfitting of the training data. In order to generate a classifier that does not overfit the training data, the number of pair points can be limited when learning the classifier, to improve its performance.