1 Mixture of Gaussians and Bayes Classification

A) Assume that you have two classes of datapoints, each of which has been fitted with a single Gauss function. The two classes have the same number of points and the Gaussians are not normalized. One can determine to which of the two classes each group of datapoints belongs by comparing the likelihoods of the data under each Gauss function. The Gauss function corresponding to the highest likelihood wins. Draw the classification boundary for each of the three examples illustrated in fig. 1.

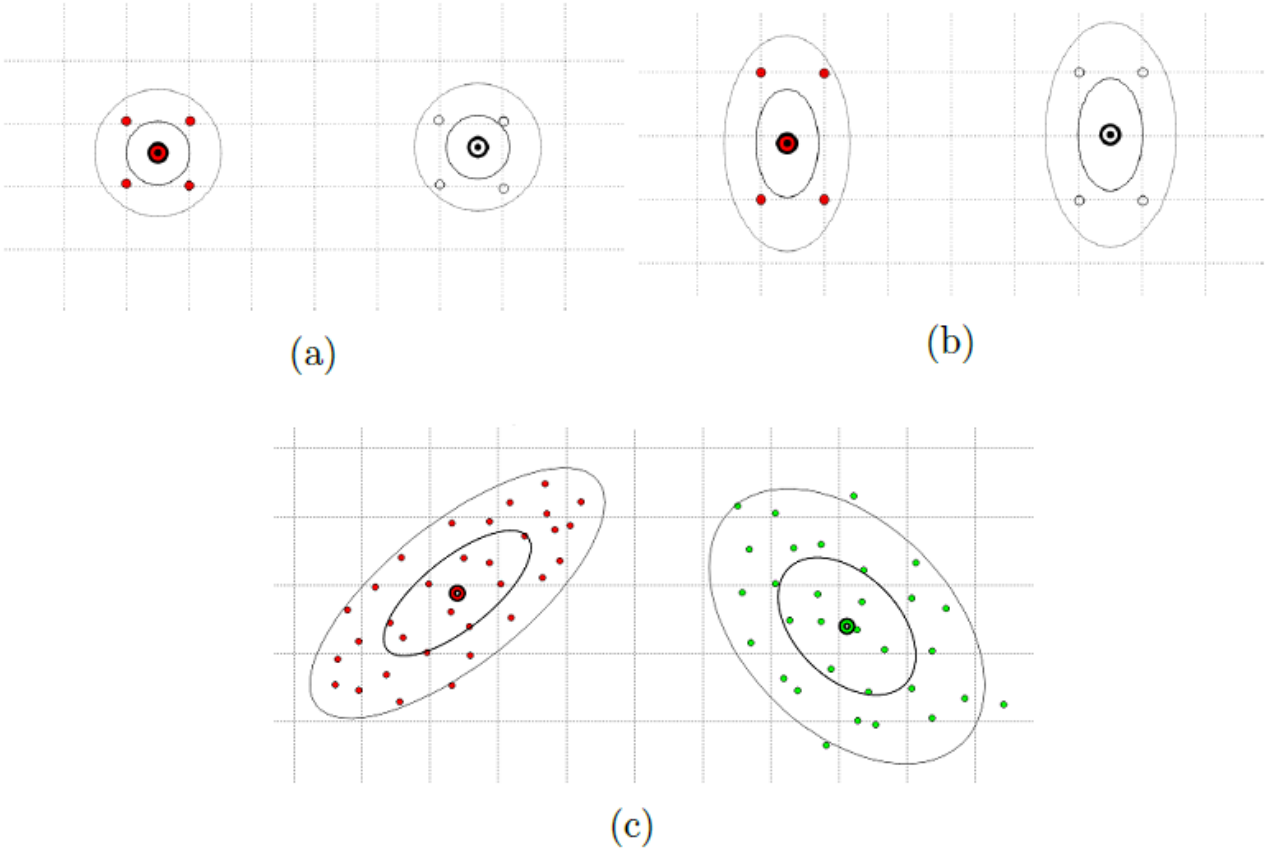


Figure 1: Three datasets comprised of two classes of datapoints. Each class is fitted with a single Gauss function. The isolines correspond to one and two standard deviations. In this question, we assume that the isolines have equal values for all Gauss functions.

Note: In MLDemos and in this question, the Gaussians are not normalized; meaning that the isolines corresponding to one and two standard deviations give the same probabilities for each class. Since the number of points is similar for both classes, the boundary can be obtained through the intersection of isolines of the same probabilities.

In a Bayesian framework, Gaussians are usually normalized, i.e., the value of a Gaussian at point x is computed as

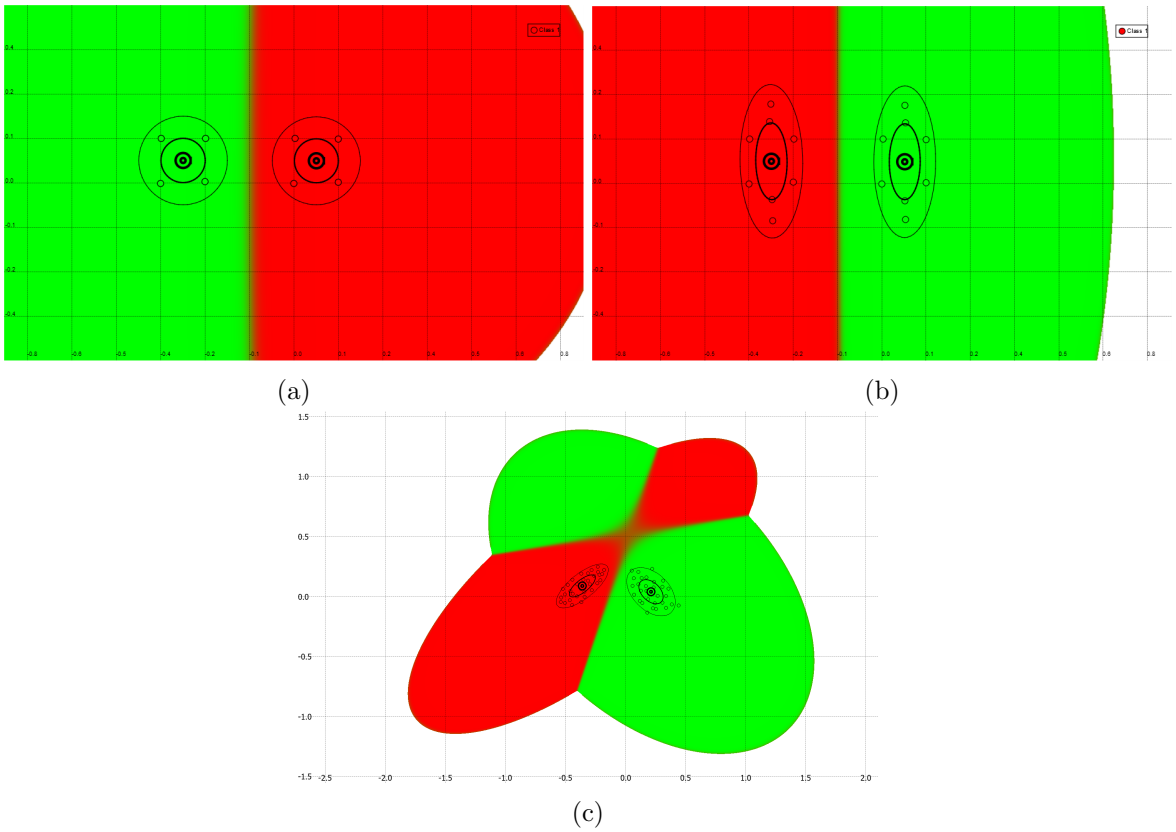
$$\frac{1}{(2\pi)^{N/2}|\Sigma|^{1/2}} \exp\left[-\frac{1}{2}(x-\mu)^\top \Sigma^{-1}(x-\mu)\right],$$

where the normalization factor $\frac{1}{(2\pi)^{N/2}|\Sigma|^{1/2}}$ ensures that the function integrates to 1, defining a probability density function. Consequently, the values corresponding to each isoline may differ across Gaussian functions. In this question, we assume that the isolines have identical values for all Gaussian functions.

Solution

In cases (a) and (b), the covariance matrices are identical. Moreover, the two centers are at the same height on the second axis and each Gaussian has the same number of datapoints. Hence the boundary is equidistant to the two centroids (see figs. 2a and 2b).

In case (c), the green class overcomes the red class on the right-hand side because its variance in this direction is larger (see fig. 2c). Note that the white-colored areas correspond to regions where the likelihood for both classes becomes zero (numerically).



(a) (b)

(c)

Figure 2: Classification boundaries for each of the three examples.

■

B) Revisit the (a) and (b) classification problems above where each class is modeled with a single Gauss function aligned with the X-Y axes. Assume that the two covariance matrices for the two classes are the same. Show how the classification boundary moves as a function of the number of datapoints in each class. Consider a ratio of 1:1, 2:1, and 4:1, respectively, for the number of datapoints in class 1 versus the number of datapoints in class 2. Assume that the datapoints in each case have been generated following a Gauss distribution in each class.

Hint: Recall that the classification boundary is obtained by setting the likelihood ratio of the two classes to 1, i.e.,

$$\frac{p(y=1|x)}{p(y=2|x)} = 1.$$

Using Bayes rule, this is equivalent to

$$\frac{p(x|y=1)}{p(x|y=2)} \times \frac{p(y=1)}{p(y=2)} = 1,$$

which can be transformed using the logarithm into

$$-\frac{1}{2}(x-\mu^1)^\top\Sigma^{-1}(x-\mu^1)+\ln p(y=1)=-\frac{1}{2}(x-\mu^2)^\top\Sigma^{-1}(x-\mu^2)+\ln p(y=2).$$

Note: In MLDemos, the class ratio is not taken into consideration and you will not be able to observe the behavior in this question.

Solution

From Bayes rule, we have

$$p(y=i|x)=\frac{p(x|y=i)p(y=i)}{p(x)} \quad i=1,2.$$

In the case that the number of datapoints is not equal in both classes, we can no longer make the assumption of equal class distribution $p(y=1)=p(y=2)$. Thus, the classification boundary is obtained by setting the likelihood ratio of the two classes to 1 such that

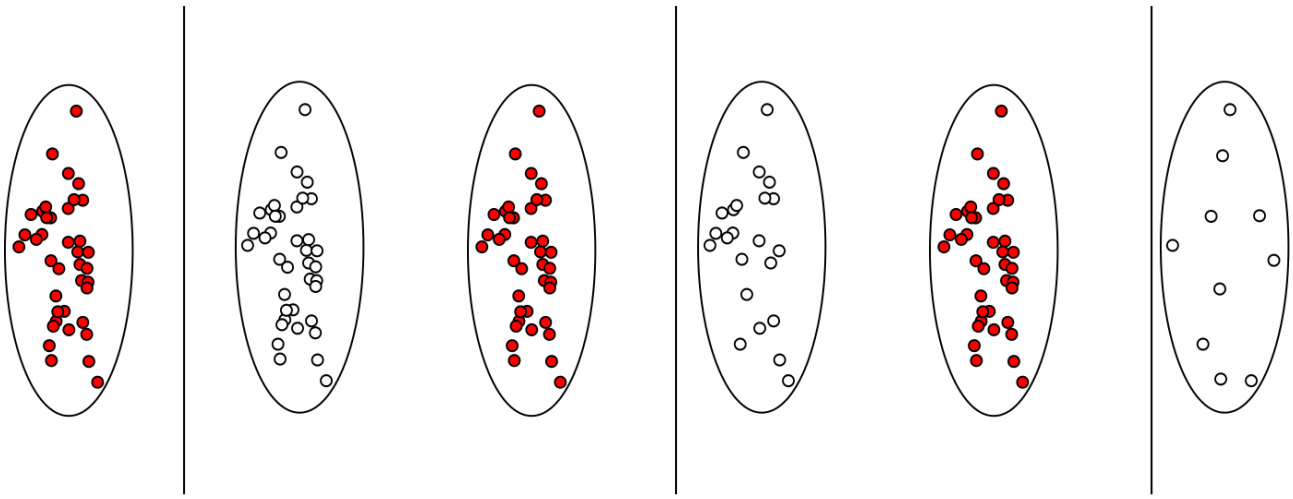
$$\frac{p(y=1|x)}{p(y=2|x)}=\frac{p(x|y=1)p(y=1)}{p(x|y=2)p(y=2)}=\frac{p(x|y=1)}{p(x|y=2)}\times\frac{p(y=1)}{p(y=2)}=1.$$

Taking the logarithm of both sides of this equation leads to

$$-\frac{1}{2}(x-\mu^1)^\top\Sigma^{-1}(x-\mu^1)+\ln p(y=1)=-\frac{1}{2}(x-\mu^2)^\top\Sigma^{-1}(x-\mu^2)+\ln p(y=2).$$

Note that the appearance of the term $\ln p(y=1)$ is due to the fact that we do not assume equal class distribution. When we have equal number of data points in the two classes (1:1 ratio) only the likelihood terms define the classification boundary because the prior terms $p(y=1)$ and $p(y=2)$ are equal. In this case, since the covariance matrices are equal, the classification boundary is the perpendicular bisector of the line joining the two Gaussian centers.

As the number of datapoints in one class (say 1) is increased, the term p(y=1) will increase and p(y=2) will decrease. To make the two sides of the above equation equal, the likelihood term of class 1 must decrease and that of class 2 must increase. The boundary then shifts by a distance that is proportional to the difference between the two priors, i.e., $p(y=1)-p(y=2)$ (see fig. 3).



(a) 1:1 ratio (b) 2:1 ratio (c) 4:1 ratio

Figure 3: Boundaries for the binary classification with varying ratio of datapoins in two classes.

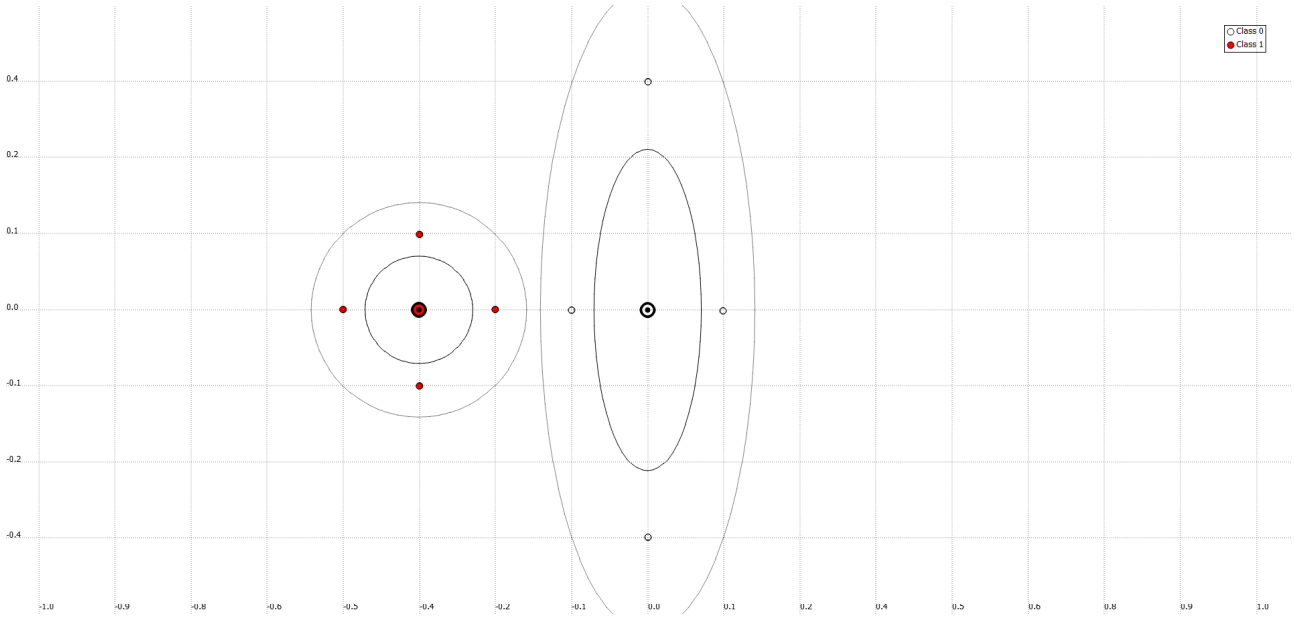


Figure 4: Binary classification task. Each class is fitted with a single Gauss function. The isolines correspond to one and two standard deviations. In this question, we assume the isolines do not have equal values for all Gauss functions. The variance along the X-axis is the same for both Gaussian distributions.

C) Now, we consider the case where the Gaussians are normalized and their covariance matrix is different. What would be the boundary for the example shown in fig. 4?

Solution

Although both Gaussian distributions have the same variance along the X-axis, the intersection of the boundary and the X-axis is not equidistant from both centers. This is a consequence of the normalization of the Gauss functions. Since the Gaussian distribution of the white class is more elongated along the Y-axis, the normalization significantly decreases the value of the Gaussian distribution for this class compared to that of the red class over the whole space. In other words, the isolines of the white class have a lower probability than those of the red class. Hence, the boundary is shifted towards the white class to compensate for the decrease in values. The shape of the boundary is also changed taking into consideration the elongation away from the X-axis. More specifically, the more we get away from the X-axis, the more the white class dominates (see fig. 5).

If we now consider identical spherical covariance matrices (like question A), the boundary becomes a straight line[1] and its intersection with the X-axis is equidistant from the two centers as shown in fig. 6.

[1] The small discrepancy is simply due to the approximate positions of the points (hand-drawn) and the estimation of the spherical covariance matrices, which are not exactly the same.

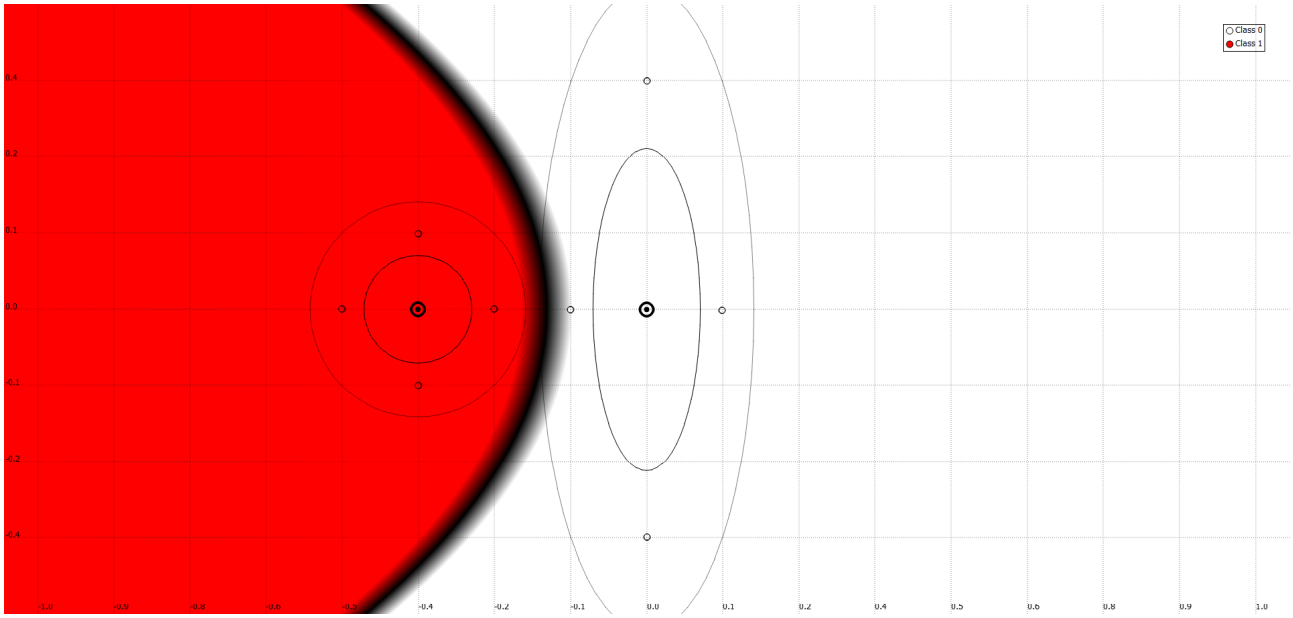


Figure 5: Boundaries for the binary classification task given in fig. 4. Each class is fitted with a single Gauss function. The isolines correspond to one and two standard deviations and do not have equal values for all Gauss functions since they do not have the same covariance matrix.

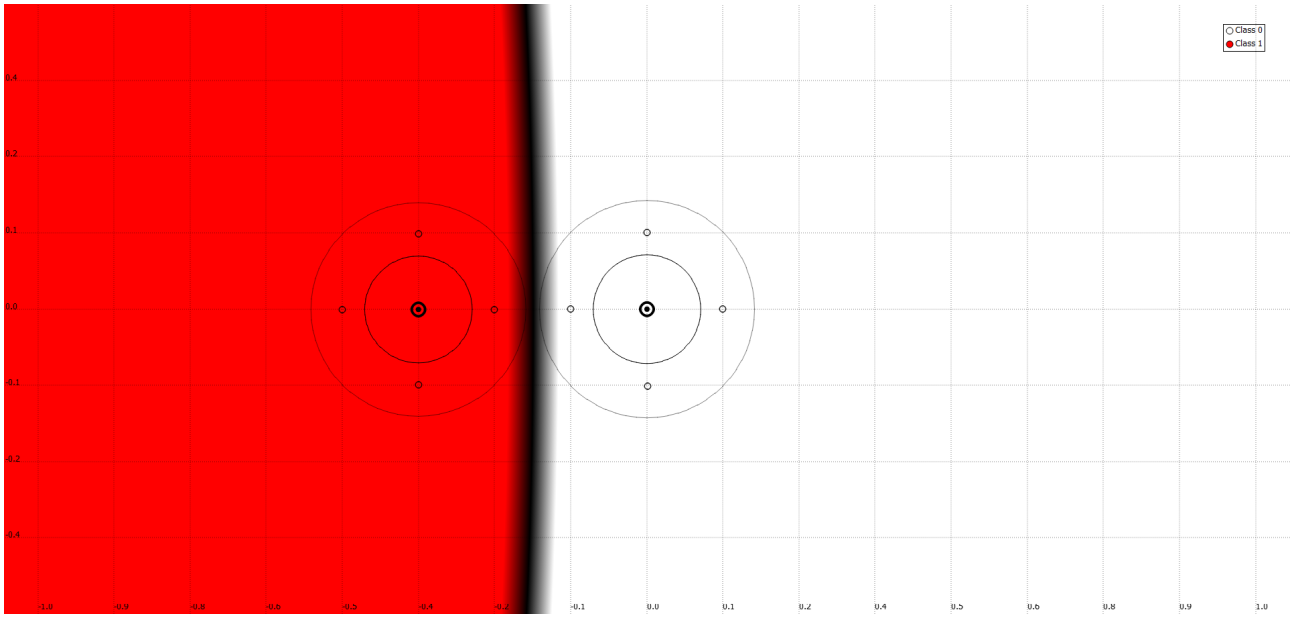
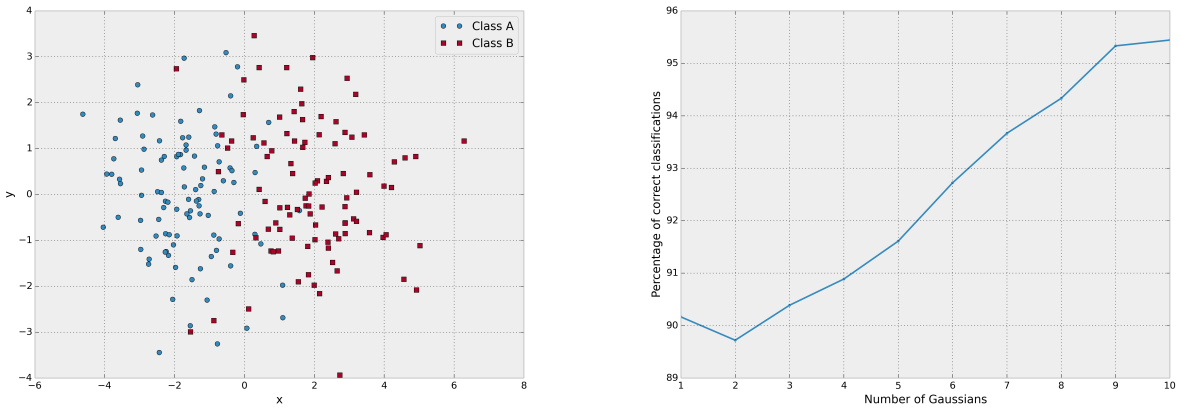


Figure 6: Boundaries for the binary classification task given in fig. 4 with (almost) identical covariance for both distributions. Each class is fitted with a single Gauss function. The isolines correspond to one and two standard deviations and approximately have equal values.

2 Overfitting, Generalization, and Computational Costs

A) Joe was given the labeled dataset represented on fig. 7a. With the goal of classifying the future unlabeled data, Joe trains separately a GMM for each class. He does not know the number of Gaussians to use in his GMM; thus, he tries different values and tests the performance of the classifier on the training dataset, obtaining the plot presented in fig. 7b. He decides to use mixtures of 10 Gaussians. Do you agree with Joe's decision? What would you do in his place?



(a) (b)

Figure 7: (a) Labeled dataset. (b) Percentage of correct classifications for GMM with different number of Gaussians.

Solution

By increasing the number of parameters, the model will adapt better to the dataset on which it is being trained. However, if the number of parameters increase too much, instead of capturing only the general distribution of the data, the model will be able to adapt to the specificities of the training dataset and, in the limit, it will be able to adapt to each datapoint. Sometimes these datapoints are representatives of rare events or even occur due to noise; therefore, this extreme adaptation causes overfitting and the model will not be able to generalize well to new datapoints (see fig. 8).

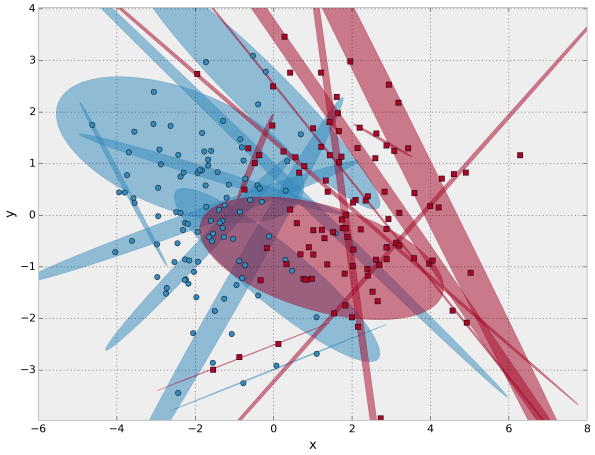


Figure 8: This figure shows the case where 10 Gaussians used to represent each class. Evidently, the model is very specific to this dataset and will not generalize well. In other words, the model is overfitted.

To avoid this and to test the model generalization power, Joe should not evaluate his GMM with the same data that he used to train the model. Instead, he could randomly divide his data into a training dataset and a validation dataset. He could then train his GMM in the training dataset and evaluate its generalization power in the validation dataset obtaining the result illustrated in fig. 9. To divide the data, he could do a K-fold cross-validation, where he randomly divides the samples in K equally-sized subsets. He could then perform the cross-validation K times, using a different subset as the validation dataset each time. The results are then averaged across the K tests. To avoid overfitting, Joe could have also used an information criterion, such as BIC, to evaluate his model.

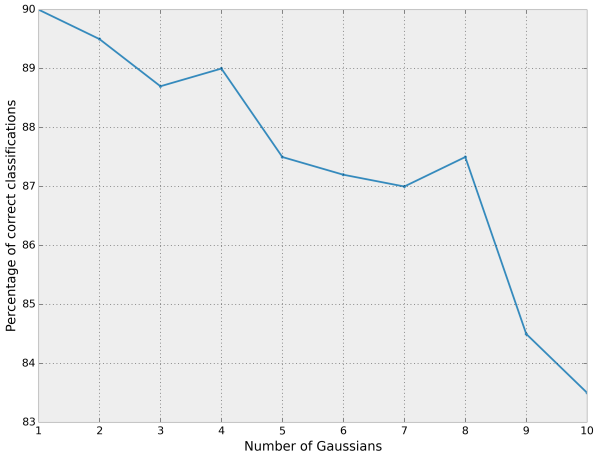
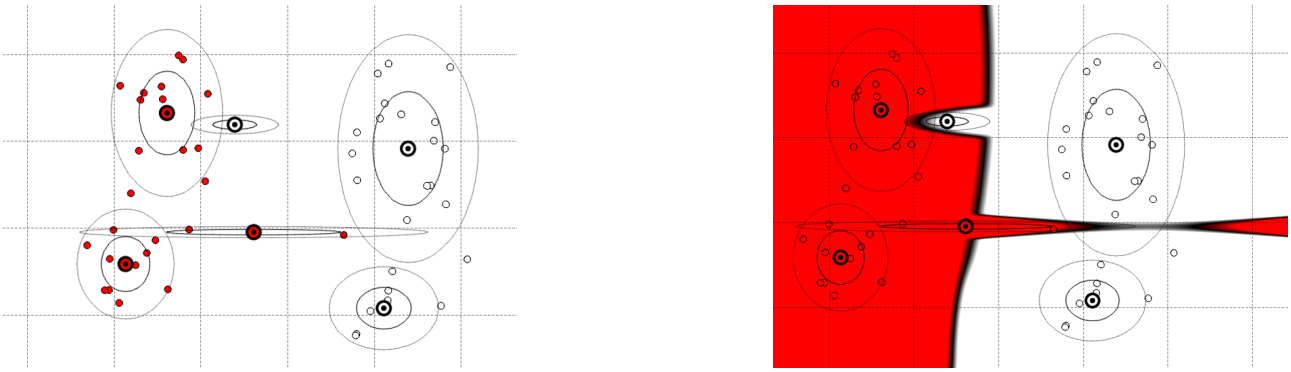


Figure 9: This plot was achieved by doing a 10-fold cross-validation. The best score is obtained when only one Gaussian is employed.

B) Draw one example dataset and choice of model for GMM that would lead to overfitting.

Solution

An example dataset and choice of GMM leading to overfitting is provided in fig. 10a. The dataset is composed of two classes that are well-separated. One noisy point belonging to each class has been added close to the other one. By using a GMM composed of three Gaussians with diagonal covariances matrices, we can observe for both classes that one Gaussian will be used to capture the noisy point. This affects the boundary (see fig. 10b).



(a) (b)

Figure 10: Example of a dataset and GMM model leading to overfitting.

C) What is the number of parameters to be estimated for GMM? What is the computational cost per iteration of the update step for GMM? Discuss the effect of choosing spherical, diagonal, or full covariance matrices.

Solution

The number of parameters to be estimated depends on the type of covariance matrix used for GMM. The possible types of covariance matrices for an N-dimensional dataset are

- **Full:** $\Sigma_{\text{full}} = \begin{bmatrix} \sigma_1^2 & \dots & \sigma_{1N}^2 \\ \vdots & \ddots & \vdots \\ \sigma_{N1}^2 & \dots & \sigma_N^2 \end{bmatrix}$.
- **Diagonal:** $\Sigma_{\text{dia}} = \text{diag}(\sigma_1^2, \dots, \sigma_N^2)$.
- **Spherical:** $\Sigma_{\text{iso}} = \text{diag}(\sigma^2, \dots, \sigma^2)$.

Considering that one must store the weight of each Gaussian α_k, the means μ^k, and the covariance matrices Σ^k, the total number of parameters to store is $(KN^2 + KN + K)$, $(KN + KN + K)$, and $(K + KN + K)$ for full, diagonal and spherical covariance matrices, respectively[2].

The update Step (M-Step) for GMM accounts for recomputing the means, covariance matrices, and priors, given the responsibilities estimated in the estimation Step (E-Step) such that

$$p(k|x^j, \Theta^{(t)}) \quad \forall k = 1, \dots, K, \quad \forall j = 1, \dots, M,$$

where K is the number of Gaussians per class and M is the number of datapoints.

More formally, one sequentially computes the

1. **Weights**

$$\alpha_k^{(t+1)} = \frac{1}{M} \sum_j p(k|x^j, \Theta^{(t)}).$$

2. **Means**

$$\mu^{k\,(t+1)} = \frac{\sum_j p(k|x^j, \Theta^{(t)}) x^j}{\sum_j p(k|x^j, \Theta^{(t)})}$$

3. **Covariance matrices**

$$\Sigma_{\text{full}}^{k\ (t+1)} = \frac{\sum_j p(k|x^j, \Theta^{(t)})(x^j - \mu^{k\,(t+1)})(x^j - \mu^{k\,(t+1)})^\top}{\sum_j p(k|x^j, \Theta^{(t)})}$$

$$\Sigma_{\text{dia}}^{k\ (t+1)} = \text{diag}((\sigma_1^{k\,(t+1)})^2, \dots, (\sigma_N^{k\,(t+1)})^2)$$

$$\Sigma_{\text{iso}}^{k\ (t+1)} = \text{diag}((\sigma^{k\,(t+1)})^2)$$

where

$$(\sigma_i^{k\,(t+1)})^2 = \frac{\sum_j p(k|x^j, \Theta^{(t)})(x_i^j - \mu_i^{k\,(t+1)})^2}{\sum_j p(k|x^j, \Theta^{(t)})} \quad \forall i = 1, \dots, N$$

[2]The number of parameters to estimate is actually lower, i.e., $(KN(N+1)/2 + KN + K - 1)$, $(KN + KN + K - 1)$, and $(K + KN + K - 1)$, respectively, due to symmetry and the constraint $\sum_{k=1}^K \alpha_k = 1$

are the variances along each dimension i and

$$(\sigma^{k^{(t+1)}})^2 = \frac{\sum_j p(k|x^j, \Theta^{(t)}) \|x^j - \mu^{k^{(t+1)}}\|^2}{N \sum_j p(k|x^j, \Theta^{(t)})}$$

is the variance averaged over all samples.

From the above equations, the computational complexity of the M-Step for GMM appears to lie in the update of the covariance matrices. Thus, the complexity of the M-Step with full covariance matrices grows with $\mathcal{O}(KMN^2)$ and drops to $\mathcal{O}(KMN)$ for diagonal and $\mathcal{O}(KM)$ for spherical covariance matrices, respectively. In comparison, the computational complexity of the update step of K-means and Soft K-means respectively grow with $\mathcal{O}(KMN)$ and $\mathcal{O}(K^2MN)$, where K is the number of clusters/Gaussians.

K-means is the cheapest method. It, however, can solely generate clusters with isotropic distributions. It is also very sensitive to the choice of K. Soft K-means and GMM relaxes this constraint. Hence, GMM with full covariance matrices will be preferred only when computational costs are of no real concern considering that it offers maximal flexibility. Variants of GMM with either diagonal or isotropic covariance matrices may be considered to reduce the computational costs (and the required amount of memory) while relaxing the constraint of choosing the number of clusters since the optimal K value can be estimated using BIC and AIC.

■

D) On another dataset, Joe decides to use the KNN classification method. In order to find the best hyperparameter K, he decides to perform cross-validation and plot the mean train and test F-measure for different K (see fig. 11). What value of K should he select for his classification problem?

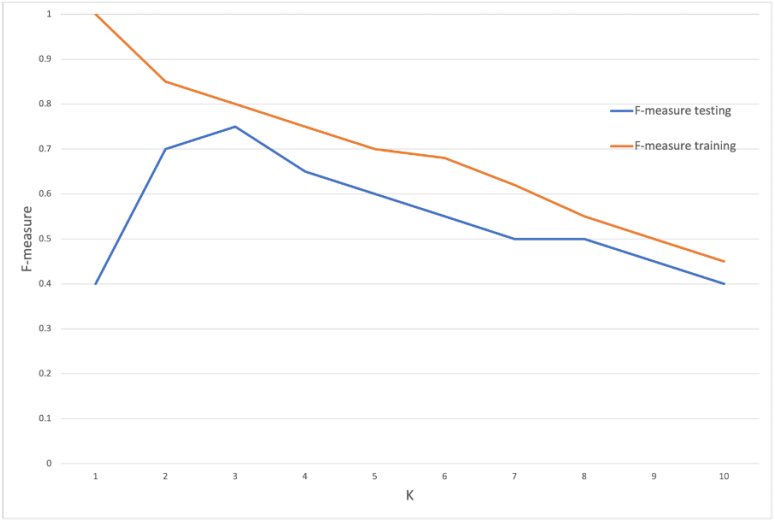


Figure 11: Mean train and test F-measure for different K.

Solution

Joe should select $K = 3$ because it is where he gets a high F-measure on both the training and testing sets. For $K = 1$ we get an over-fitted model which fits very well the training set but is unable to generalize well. For $K \geq 3$, we get more and more under-fitted models which cannot learn the correct classification function, resulting in poor performances on both the training and testing dataset.

■