CSE176 Introduction to Machine Learning
Fall semester 2023

Lab: Gaussian classifiers
Miguel Á. Carreira-Perpiñán

The objective of this lab is for you to explore the behavior of Gaussian classifiers in Matlab by applying them to some datasets. The TA will first demonstrate the results that Gaussian classifiers (of various covariance matrix types) give on toy datasets and the MNIST dataset, including the corresponding ROC curve or confusion matrix. Then you will replicate those results and further explore the datasets and classifiers.

We provide you with the following:

- The script `lab02.m` sets up the problem (toy dataset or MNIST) and plots various figures. The actual algorithms are implemented in the functions below.
- `GMclass.m` and other functions from the Gaussian mixture tools train a Gaussian classifier and do other things.

Look at those functions and try to understand how they work and how they produce plots. It will help you to see how the machine learning algorithms were implemented, become proficient in Matlab and produce useful plots.

I Datasets

Construct your own toy datasets in 1D and 2D, such as Gaussian classes with more or less overlap, or classes with curved shapes as in the 2moons dataset. You will also use the MNIST dataset of handwritten digits  .

II Using a Gaussian classifier

In a Gaussian classifier, for each class $k = 1, \dots, K$, the class-conditional probability distribution $p(C_k|\mathbf{x})$ for a point $\mathbf{x} \in \mathbb{R}^D$ is a Gaussian distribution with parameters $(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ (mean vector of $D \times 1$ and covariance matrix of $D \times D$). Also, each class has another parameter, its proportion $\pi_k = p(C_k) \in [0, 1]$ (its prior distribution). We will consider 6 types of covariance matrix $\boldsymbol{\Sigma}_k$:

	Non-shared (separately for each class)	Shared (equal for all classes)
full	`'F'`: $\boldsymbol{\Sigma}_k$	`'f'`: $\boldsymbol{\Sigma}_k = \boldsymbol{\Sigma}\ \forall k$
diagonal	`'D'`: $\boldsymbol{\Sigma}_k = \mathbf{D}_k$	`'d'`: $\boldsymbol{\Sigma}_k = \mathbf{D}\ \forall k$
isotropic	`'I'`: $\boldsymbol{\Sigma}_k = \sigma_k^2\mathbf{I}$	`'i'`: $\boldsymbol{\Sigma}_k = \sigma^2\mathbf{I}\ \forall k$

To use a Gaussian classifier, we need to solve two problems:

1. **Training**: to learn the parameters $\{(\pi_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)\}_{k=1}^K$ from a training set. This is given by the maximum likelihood estimate (MLE). For the prior distribution and mean vectors, this is:

$$\text{prior distribution: } p(C_k) = \pi_k = \frac{N_k}{N} \qquad \text{mean vector: } \boldsymbol{\mu}_k = \frac{1}{N_k}\sum_{\text{class } k} \mathbf{x}_n$$

   where the sum above is over the data points in class k. For the covariance matrix, the MLE depends on the type of covariance:

`'F'`: $\boldsymbol{\Sigma}_k = \frac{1}{N}\sum_{\text{class } k}(\mathbf{x}_n - \boldsymbol{\mu}_k)(\mathbf{x}_n - \boldsymbol{\mu}_k)^T$ (the covariance of points in class k)	`'f'`: $\boldsymbol{\Sigma} = \sum_{k=1}^K \pi_k \boldsymbol{\Sigma}_k$
`'D'`: $\mathbf{D}_k = \text{diag}\,(\boldsymbol{\Sigma}_k)$ (the diagonal elements of $\boldsymbol{\Sigma}_k$)	`'d'`: $\mathbf{D} = \sum_{k=1}^K \pi_k \,\text{diag}\,(\boldsymbol{\Sigma}_k)$.
`'I'`: $\sigma_k^2 = \frac{1}{D}\sum_{d=1}^D \sigma_{kd}^2$, where $\sigma_{kd}^2 = d$th diagonal element of $\boldsymbol{\Sigma}_k$	`'i'`: $\sigma^2 = \frac{1}{D}\sum_{d=1}^D \sum_{k=1}^K \pi_k \sigma_{kd}^2$

   So the shared-case MLE is the weighted average of the non-shared MLE over the classes using π_k as weights. Also, we ensure each covariance matrix is full-rank by adding to its diagonal a small positive number, e.g. 10^{-10}.

2. **Testing**: to compute the posterior probabilities $\{p(C = k|\mathbf{x})\}_{k=1}^K$ for a test point $\mathbf{x}$ (typically not in the training set, although it can be any point in $\mathbb{R}^D$). This is done by `GMpdf.m`, see `lab02.m`.
   Given these posterior probabilities, we can classify $\mathbf{x}$ as $\arg\max_{k\in\{1,\dots,K\}} p(C = k|\mathbf{x})$, or construct an ROC curve (for $K = 2$) or confusion matrix (for any K) over a test set.

For each classifier, we plot the following figures:

- For 1D datasets: $p(x|C)$, $p(x|C)p(C)$, $p(C|x)$, for each class $C = 1, \ldots, k$, and $\max_C p(C|x)$ for each x value.
- For 2D datasets: contour plot of $p(x|C)$ for each class and class boundaries.
- For any dataset: we plot either the confusion matrix (for $K > 2$) or the ROC curve (for $K = 2$) and give the area-under-the-curve (AUC) value.

Explore the classifiers and plots with different datasets, number of classes, classes with more or less overlap, etc. See the end of file `lab02.m` for suggestions of things to explore.