Probabilistic Clustering

EM, Mixtures of Gaussians, RBFs, etc
Multi-variate density estimation

● A mixture of Gaussians model

\[ p(x|\theta) = \sum_{i=1}^{k} p_j p(x|\mu_j, \Sigma_j) \]

where \( \theta = \{p_1, \ldots, p_k, \mu_1, \ldots, \mu_k, \Sigma_1, \ldots, \Sigma_k\} \) contains all the parameters of the mixture model. \( \{p_j\} \) are known as mixing proportions or coefficients.
Mixture density

- Data generation process:

\[
p(x|\theta) = \sum_{j=1,2} P(y = j) \cdot p(x|y = j)
\]

(generic mixture)

\[
= \sum_{j=1,2} p_j \cdot p(x|\mu_j, \Sigma_j)
\]

(mixture of Gaussians)

- Any data point \( x \) could have been generated in two ways
Mixture density

- If we are given just $x$ we don’t know which mixture component this example came from

$$p(x|\theta) = \sum_{j=1,2} p_j p(x|\mu_j, \Sigma_j)$$

- We can evaluate the posterior probability that an observed $x$ was generated from the first mixture component

$$P(y = 1|x, \theta) = \frac{P(y = 1) \cdot p(x|y = 1)}{\sum_{j=1,2} P(y = j) \cdot p(x|y = j)} = \frac{p_1 p(x|\mu_1, \Sigma_1)}{\sum_{j=1,2} p_j p(x|\mu_j, \Sigma_j)}$$

*But only if we are given the distributions and prior*

- This solves a credit assignment problem
Mixture density estimation

- Suppose we want to estimate a two component mixture of Gaussians model.

\[ p(x|\theta) = p_1 p(x|\mu_1, \Sigma_1) + p_2 p(x|\mu_2, \Sigma_2) \]

- If each example \( x_i \) in the training set were labeled \( y_i = 1, 2 \) according to which mixture component (1 or 2) had generated it, then the estimation would be easy.

- Labeled examples \( \Rightarrow \) no credit assignment problem
Mixture density estimation

When examples are already assigned to mixture components (labeled), we can estimate each Gaussian independently

- If $\hat{n}_j$ is the number of examples labeled $j$, then for each $j = 1, 2$ we set

\[
\begin{align*}
\hat{p}_j & \leftarrow \frac{\hat{n}_j}{n} \\
\hat{\mu}_j & \leftarrow \frac{1}{\hat{n}_j} \sum_{i:y_i=j} x_i \\
\hat{\Sigma}_j & \leftarrow \frac{1}{\hat{n}_j} \sum_{i:y_i=j} (x_i - \hat{\mu}_j)(x_i - \hat{\mu}_j)^T
\end{align*}
\]
Mixture density estimation: credit assignment

- Of course we don’t have such labels ... but we can guess what the labels might be based on our current mixture distribution

- We get soft labels or posterior probabilities of which Gaussian generated which example:

\[ \hat{p}(j|i) \leftarrow P(y_i = j|x_i, \theta) \]

where \( \sum_{j=1,2} \hat{p}(j|i) = 1 \) for all \( i = 1, \ldots, n \).

- When the Gaussians are almost identical (as in the figure), \( \hat{p}(1|i) \approx \hat{p}(2|i) \) for almost any available point \( x_i \).

Even slight differences can help us determine how we should modify the Gaussians.
The EM algorithm

**E-step:** softly assign examples to mixture components

\[ \hat{p}(j|i) \leftarrow P(y_i = j|x_i, \theta), \quad \text{for all } j = 1, 2 \text{ and } i = 1, \ldots, n \]

**M-step:** re-estimate the parameters (separately for the two Gaussians) based on the soft assignments.

\[ \hat{n}_j \leftarrow \sum_{i=1}^{n} \hat{p}(j|i) = \text{Soft \# of examples labeled } j \]

\[ \hat{p}_j \leftarrow \frac{\hat{n}_j}{n} \]

\[ \hat{\mu}_j \leftarrow \frac{1}{\hat{n}_j} \sum_{i=1}^{n} \hat{p}(j|i) x_i \]

\[ \hat{\Sigma}_j \leftarrow \frac{1}{\hat{n}_j} \sum_{i=1}^{n} \hat{p}(j|i) (x_i - \hat{\mu}_j)(x_i - \hat{\mu}_j)^T \]
Mixture density estimation: example
Mixture density estimation
Mixture density estimation
The EM-algorithm

- Each iteration of the EM-algorithm *monotonically* increases the (log-)likelihood of the $n$ training examples $x_1, \ldots, x_n$:

$$\log p(\text{data} | \theta) = \sum_{i=1}^{n} \log \left( \frac{p(x_i | \theta)}{p_1 p(x_i | \mu_1, \Sigma_1) + p_2 p(x_i | \mu_2, \Sigma_2)} \right)$$

where $\theta = \{p_1, p_2, \mu_1, \mu_2, \Sigma_1, \Sigma_2\}$ contains all the parameters of the mixture model.
http://www.ncrg.aston.ac.uk/netlab/

- PCA
- Mixtures of probabilistic PCA
- Gaussian mixture model with EM training
- Linear and logistic regression with IRLS
- Multi-layer perceptron with linear, logistic and softmax outputs and error functions
- Radial basis function (RBF) networks with both Gaussian and non-local basis functions
- Optimisers, including quasi-Newton methods, conjugate gradients and scaled conj grad.
- Multi-layer perceptron with Gaussian mixture outputs (mixture density networks)
- Gaussian prior distributions over parameters for the MLP, RBF and GLM including multiple hyper-parameters
- Laplace approximation framework for Bayesian inference (evidence procedure)
- Automatic Relevance Determination for input selection
- Markov chain Monte-Carlo including simple Metropolis and hybrid Monte-Carlo
- K-nearest neighbour classifier
- K-means clustering
- Generative Topographic Map
- Neuroscale topographic projection
- Gaussian Processes
- Hinton diagrams for network weights
- Self-organising map
Data sampled from Mixture of 3 Gaussians

Spectral Clustering
Original Data

Gaussian Mixture Model Classification