Unsupervised Learning

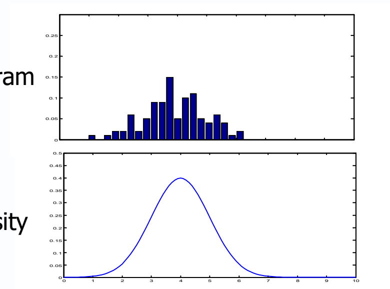
Density Estimation

Ioannis Patras

Density Estimation

Data histogram

Learned density

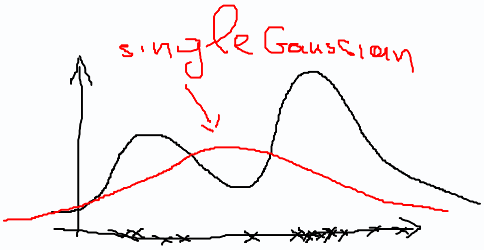


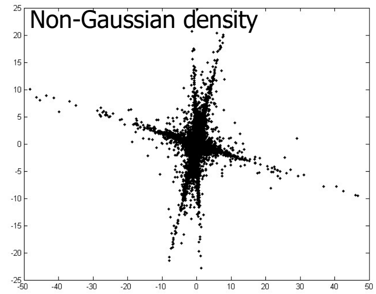
Model the density of the probability from which x is drawn. $\boldsymbol{x} \sim p(\boldsymbol{x};\boldsymbol{w})$, where $\boldsymbol{w}$ represents an unknown parameter vector that needs to be learned.

Density estimation

Not all data is well represented by Gaussian models…

Therefore need a possible way of representing more general densities



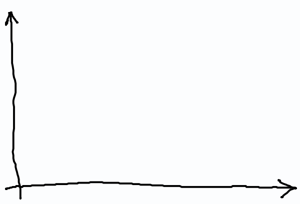


Mixture Models

Idea of mixture model: Assume there are some unobservable discrete variables (also called latent variables) underlying the observed data (c.f. classification)

Fish example. We observe 'fish lightness', x, and have two models, one for Salmon and on for Sea Bass. If we are not interested in classifying the fish (but only in the lightness variability) → Total probability model:

$$P(\mathrm{x}) = P(\mathrm{Salmon})P(x|\mathrm{Salmon}) + P(\mathrm{SeaBass})P(x|\mathrm{SeaBass})$$



Mixture Models

Idea of mixture model: Assume there are some unobservable discrete variables (also called latent variables) underlying the observed data (c.f. classification)

The general *mixture model* with K different components has the following form

$$P(x) = \sum_k P(c_k)P(x \mid c_k) = \sum_k P(x, c_k)$$

Remember the classes c_k are not observed (unsupervised learning).

ELEM041 Machine Learning

Mixture model example:

Mixtures of Gaussians (MoGs)

One mixture model is the mixture of Gaussians (MoGs) which has the general form:

$$\begin{aligned} P(x) &= \sum_k P(c_k)P(x \mid c_k) \\ &= \sum_k P(c_k)\mathrm{N}\left(\mu_k, \Sigma_k\right) \\ &= \sum_{k=1}^{K} P(c_k)\frac{1}{\left|\Sigma_k\right|^{d/2}} \exp\left(-\frac{1}{2}(x-\mu_k)^T \Sigma_k^{-1}(x-\mu_k)\right) \end{aligned}$$

Learning mixture models I

How do we estimate mixture models? Here we will look at ML. Let θ represent the parameters of interest.

Let $p(\mathrm{x} \mid \boldsymbol{\theta}) = \sum_{\mathrm{k}=1}^{\mathrm{K}} \mathrm{P}(\mathrm{c_k}) \mathrm{P}(\mathrm{x} \mid \mathrm{c_k}, \theta_k)$ so that:

$$E = -\ln p(\chi \mid \boldsymbol{\theta}) =$$

$$= -\sum_{n=1}^{N} \ln p(x_n \mid \boldsymbol{\theta}) \quad \text{(assuming independent observations)}$$

Solve by setting the partial derivatives to zero

$$\frac{\partial E}{\partial \theta} = -\sum_{n=1}^{N} \frac{\partial \ln p(x_n \mid \boldsymbol{\theta})}{\partial \theta} = -\sum_{n=1}^{N} \frac{\partial p(x_n \mid \boldsymbol{\theta}) / \partial \theta}{p(x_n \mid \boldsymbol{\theta})} = 0 \qquad \left(\text{since } \frac{\partial}{\partial \theta} \ln f(\theta) = \frac{\partial f(\theta) / \partial \theta}{f(\theta)}\right)$$

differentiating w.r.t parameters in the kth mixture:

$$\frac{\partial E}{\partial \theta_k} = -\sum_{n=1}^{N} \frac{p(c_k)}{p(x_n \mid \boldsymbol{\theta})} \frac{\partial}{\partial \theta_k} p(x_n \mid c_k, \theta_k) = 0$$

(got by noting that $\frac{\partial}{\partial \theta_k} p(x_n \mid \boldsymbol{\theta}) = \frac{\partial}{\partial \theta_k} \sum_j p(c_j) p(x_n \mid c_j, \theta_j) = \frac{\partial}{\partial \theta_k} p(c_k) p(x_n \mid c_k, \theta_k)$)

Learning mixture models II

Applying $\partial(\ln f(\theta)) / \partial \theta = \frac{\partial f(\theta) / \partial \theta}{f(\theta)}$ again gives:

$$\frac{\partial E}{\partial \theta_k} = -\sum_{n=1}^{N} \frac{p(c_k) p(x_n \mid c_k, \theta)}{p(x_n \mid \theta)} \frac{\partial}{\partial \theta_k} \ln p(x_n \mid c_k, \theta)$$

$$= -\sum_{n=1}^{N} p(c_k \mid x_n, \theta) \frac{\partial}{\partial \theta_k} \ln p(x_n \mid c_k, \theta)$$

Applying Bayes rule $\frac{p(c_k \mid \theta) p(x_n \mid c_k, \theta)}{p(x_n \mid \theta)} = p(c_k \mid x_n, \theta)$

and noting that $p(c_k) = p(c_k \mid \theta)$)

This is a weighted version of what we would get if we knew the classes

That is: we weight the contribution of x_n with the probability that it belongs to class c_k:

Parameter estimation (cont.)

$$\sum_{n=1}^{N} p(c_k \mid x_n, \theta) \frac{\partial}{\partial \theta_k} \ln p(x_n \mid c_k, \theta_k) = 0$$

Ignoring the $p(c_k \mid x_n, \theta)$ term we have the standard ML solution *given* component c_k.

The $p(c_k \mid x_n, \theta)$ term is the probability that the point x_n belongs to the c_k component model.

Can we use this to compute the ML solution for the mixture?

EM iterative update

Although $p(c_k \mid x_n, \theta)$ is not independent of θ_k we can treat it as such and generate the following iterative algorithm:

Expectation - Maximization (beginning with initial values $\hat{\theta}^{(0)}$)

E – step

calculate: $q_{n,k} = p(c_k \mid x_n, \hat{\theta}^{(i)})$ for all k and n

M-step

maximise the weighted log likelihood:

$$\hat{\theta}_k^{(i+1)} = \underset{\theta}{\operatorname{argmax}} \sum_n q_{n,k} \ln p(x_n \mid c_k, \theta_k)$$

for all k

A converged solution is a stationary point (e.g. local maximum) of the likelihood

EM algorithm for MoGs I

We can now easily apply the EM algorithm to estimate MoG parameters. Let $w_k^{(i)}$ represent the ith estimate for $p(c_k)$:

- E-step : Here we calculate $q_{n,k} = p(c_k \mid x_n, \theta^{(i)})$ by using

$$p(c_k \mid x_n, \theta^{(i)}) = p(x_n \mid c_k, \theta^{(i)})\, p(c_k) \,/\, p(x_n \mid \theta^{(i)})$$

$$= p(x_n \mid c_k, \theta^{(i)})\, p(c_k) \,/ \sum_j p(x_n \mid c_j, \theta^{(i)})\, p(c_j)$$

Where $p(x_n \mid c_k, \theta^{(i)}) = \frac{1}{\Sigma_k^{d/2}} \exp(-\frac{1}{2}(x_n - \mu_k)^T \Sigma_k^{-1} (x_n - \mu_k))$

EM algorithm for MoGs II

- M-step :

1. $\mu_{\mathrm{k}}^{(i+1)} = \sum_n q_{n,k} x_n \Big/ \sum_n q_{n,k}$ (weighted mean)

2. $\Sigma_{\mathrm{k}}^{(i+1)} = \dfrac{\sum_n q_{n,k} (x_n - \mu_{\mathrm{k}}^{(i+1)})(x_n - \mu_{\mathrm{k}}^{(i+1)})^T)}{\sum_n q_{n,k}}$ (weighted covariance)

It is also possible to show that we can estimate the priors $p(c_k)$ for each class c_k using the following update:

3. $w_k^{(i)} = \frac{1}{N} \sum_n q_{n,k}$

All the variables can be updated as weighted averages!

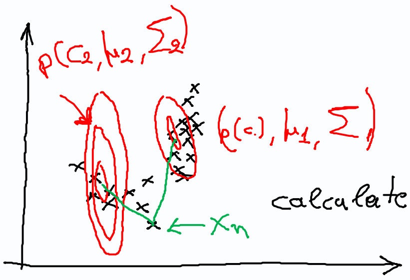
EM iterative update (E-step)

Expectation - Maximization (beginning with initial values $\hat{\theta}^{(0)}$)

E – step calculate: $q_{n,k} = p(c_k \mid x_n, \hat{\theta}^{(i)})$ for all k and n

M-step maximise the weighted log likelihood:

$$\hat{\theta}_k^{(i+1)} = \underset{\theta}{\operatorname{argmax}} \sum_n q_{n,k} \ln p(x_n \mid c_k, \theta_k) \quad \text{for all } k$$



calculate $q_{n1} = p(c_1 \mid x_n, \theta)$

$$= \frac{p(x_n \mid c_1, \theta)\, p(c_1)}{P(x_n \mid \theta)}$$

$$= \frac{p(x_n \mid c_1, \theta) \cdot p(c_1)}{\sum_k p(x_n \mid c_k, \theta)\, p(c_k)}$$

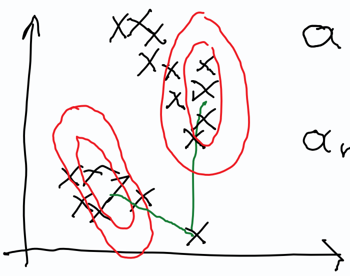
EM iterative update (E-step)

Expectation - Maximization (beginning with initial values $\hat{\theta}^{(0)}$)

E – step calculate: $q_{n,k} = p(c_k \mid x_n, \hat{\theta}^{(i)})$ for all k and n

M-step maximise the weighted log likelihood:

$$\hat{\theta}_k^{(i+1)} = \underset{\theta}{\operatorname{argmax}} \sum_n q_{n,k} \ln p(x_n \mid c_k, \theta_k) \quad \text{for all } k$$



$$a_{n1} = \frac{1}{(\pi/2)^{d/2} |\Sigma|^{1/2}} e^{-(x-\mu_1)^T \Sigma^{-1} (x-\mu_1)}$$

$$a_{n2} = \dots$$

$$q_{n1} = \frac{P(c_1) \cdot a_{n1}}{\sum_K P(c_K)\, a_{nK}}$$

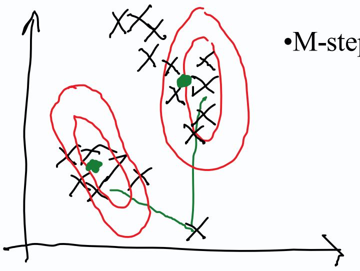
EM iterative update (M-step)

Expectation - Maximization (beginning with initial values $\hat{\theta}^{(0)}$)

E – step calculate: $q_{n,k} = p(c_k \mid x_n, \hat{\theta}^{(i)})$ for all k and n

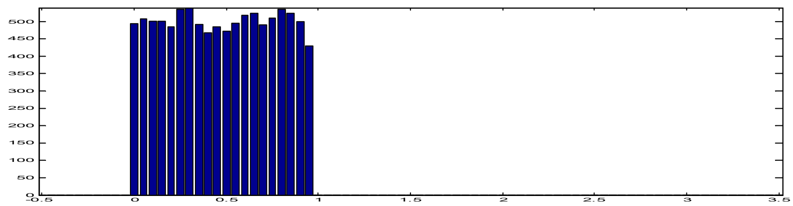
M-step maximise the weighted log likelihood:

$$\hat{\theta}_k^{(i+1)} = \underset{\theta}{\operatorname{argmax}} \sum_n q_{n,k} \ln p(x_n \mid c_k, \theta_k) \qquad \text{for all } k$$

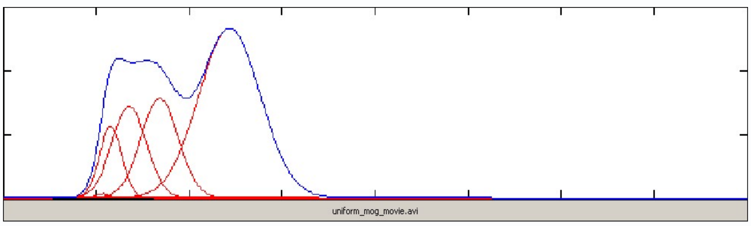


•M-step : $\mu_k^{(i+1)} = \sum_n q_{n,k} x_n \Big/ \sum_n q_{n,k}$ (weighted mean)

EM algorithm for MoG example



This demonstrates 100 EM iterations of a 5 Gaussian MoG



Mixtures of Gaussians (MoGs)

MoGs are very flexible and can be used to approximate any density:

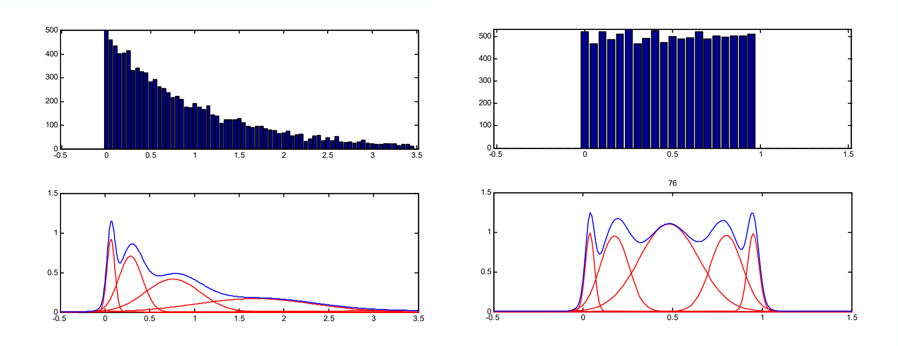
Theorem: *Any p(x) can be approximated as closely as desired by a Gaussian mixture: i.e. there is an order K and priors $p(c_k)$ for which*

For any ε>0. (Anderson and Moore, *Optimal Filtering* 1979)

$$\int_x \left|P(x) - P_{\text{mog}}(x)\right| dx < \varepsilon$$

Examples of MoGs

Here are a couple of MoG approximations using 5 Gaussians:



Exponential distribution

Uniform distribution

Notice in these two cases the individual Gaussians have no special meaning.

Convergence issues

- Number of Gaussians is assumed to be known. Determining the correct number in an unsupervised way is an ill-posed problem.

- Converges to a LOCAL minimum. Run with several random initialiasations. Select the one that maximizes the data likelihood.

$$P(x_n \mid \mu_1,\dots,\mu_K,\Sigma_1,\dots\Sigma_K) \;\propto\; \sum_{k=1}^{K} \frac{P(c_k)}{\det(\Sigma_k)^{1/2}} \exp(-\frac{1}{2}(x_n-\mu_k)^T \Sigma_k^{-1}(x_n-\mu_k))$$

Convergence issues

Overfitting. MoGs can potentially exhibit a very interesting form of overfitting – **Singular points**

Recall the likelihood for x_n is:

$$P(x_n \mid \mu_1,\dots,\mu_K,\Sigma_1,\dots\Sigma_K) \;\propto\; \sum_{k=1}^{K} \frac{P(c_k)}{\det(\Sigma_k)^{1/2}} \exp(-\frac{1}{2}(x_n-\mu_k)^T \Sigma_k^{-1}(x_n-\mu_k))$$

Now suppose $x_n = \mu_k$ for some *k* and $\Sigma_k \to 0$. Then

$$P(x_n \mid \mu_1,\dots,\mu_K,\Sigma_1,\dots\Sigma_K) \;\approx\; \frac{P(c_k)}{\det(\Sigma_k)^{1/2}} \exp(0) \to \infty$$

So the likelihood → ∞. These singular points ALWAYS exist. They can be avoided by constraining the covariances always above some minimum value.

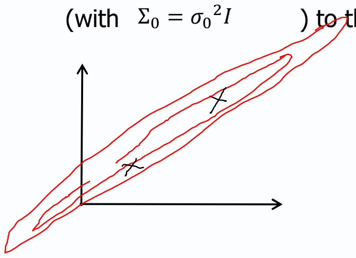
Overfitting in ML density estimation

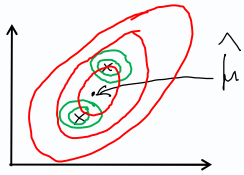
$$p(x|\mu,\Sigma) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} e^{-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)}$$

2. Regularise the covariance matrix by adding a diagonal matrix $\Sigma_0 = \sigma_0{}^2 I$

$$\hat{\Sigma} = \sum_n (x_n - \hat{\mu})(x_n - \hat{\mu})^T + \sigma_0{}^2 I = \begin{bmatrix} \sum_n (x_n(1) - \hat{\mu}(1))^2 + \sigma_0 & & \\ & \ddots & \\ & & \sum_n (x_n(d) - \hat{\mu}_n(d))^2 + \sigma_0 \end{bmatrix}$$

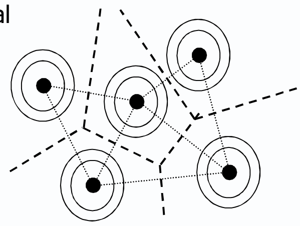
Leads to "thicker" Gaussian. Equivalent to adding zero mean Gaussian noise (with $\Sigma_0 = \sigma_0{}^2 I$) to the original dataset.





K - means MoG link

We can make the K-means/MoG link more explicit. Recall that Isotropic Gaussians (covariance = $\boldsymbol{\sigma^2 I}$) with equal weights (class priors) → templates model



E-step:

Step 1 = full classification (as opposed to weighting with probabilities) → $\boldsymbol{P_{n,k}}$ **= 0** or **1**.

This is only true if $\boldsymbol{\sigma^2} \rightarrow \mathbf{0}$

M-step:

The only parameters to be updated are the means $\boldsymbol{\mu_k}$. So this is identical to the MoG M-step.

So K-means ~ MoG with isotropic covariances $\boldsymbol{\Sigma_k} \rightarrow \boldsymbol{0}$ and equal class priors.

Summary

- Limitations of Gaussian models for unsupervised learning
- Introduced mixture modelling
  - Latent variables
  - Looked at the special case of the MoGs
  - Parameter estimation via EM
  - Convergence issues