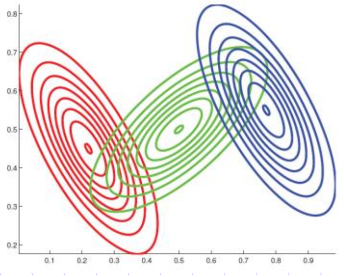
INM431 Machine Learning

Artur S. d'Avila Garcez
a.garcez@city.ac.uk
http://www.staff.city.ac.uk/~aag/

Based on Bishop's book

Content

- Gaussian Mixture Models (GMM)
- K-means
- Expectation-Maximization (EM)



Recall: Directed Graphical Models (aka Bayesian Nets)

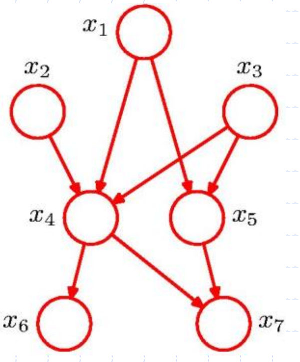
$$\begin{aligned} p(x_1, \ldots, x_7) &= p(x_1)p(x_2)p(x_3)p(x_4|x_1, x_2, x_3) \\ &\quad p(x_5|x_1, x_3)p(x_6|x_4)p(x_7|x_4, x_5) \end{aligned}$$

The joint distribution of a graph with K nodes is given by:

$$p(\mathbf{x}) = \prod_{k=1}^{K} p(x_k|\mathrm{pa}_k)$$

where pa_k denotes the set of parents of x_k

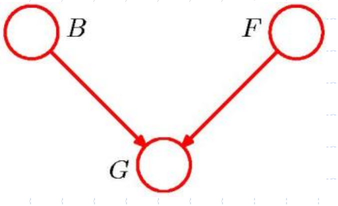
This is the **factorization** of a directed graphical model



DGMs - example (1)

When I turn on the car:

- $p(B)$: battery is charged (B={0,1})
- $p(F)$: there is fuel in the tank (F={0,1})
- $p(G)$: fuel gauge moves (G={0,1})



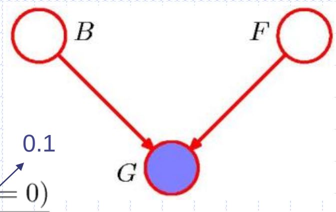
$$\begin{aligned} p(G=1|B=1,F=1) &= 0.8 \\ p(G=1|B=1,F=0) &= 0.2 \\ p(G=1|B=0,F=1) &= 0.2 \\ p(G=1|B=0,F=0) &= 0.1 \\ p(B=1) &= 0.9 \\ p(F=1) &= 0.9 \end{aligned}$$

and hence

$$p(F=0) = 0.1$$

If the gauge does not move, what is the probability that the fuel tank is empty?

DGMs - example (2)



Car out of fuel?
Recall that p(G=0|B=0,F=0) = 0.9

$$
\begin{aligned}
p(F=0|G=0) &= \frac{\overset{0.81}{\cancel{p(G=0|F=0)}}\,\overset{0.1}{\cancel{p(F=0)}}}{\underset{0.315}{\cancel{p(G=0)}}} \\
&\simeq 0.257
\end{aligned}
$$

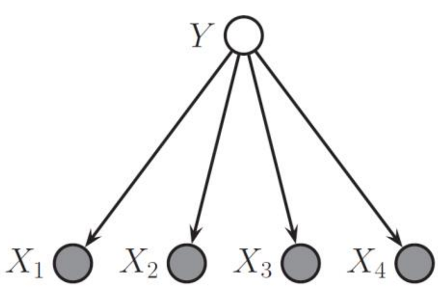
Probability of an empty tank is increased by observing G = 0.

$$
\begin{aligned}
p(F=0|G=0,B=0) &= \frac{p(G=0|B=0,F=0)p(F=0)}{\sum_{F\in\{0,1\}} p(G=0|B=0,F)p(F)} \\
&\simeq 0.111
\end{aligned}
$$

By observing also B = 0, now the probability of empty tank gets reduced.
This is known as explaining away: *battery* explains away *fuel* as a cause!

DGMs – the naïve case

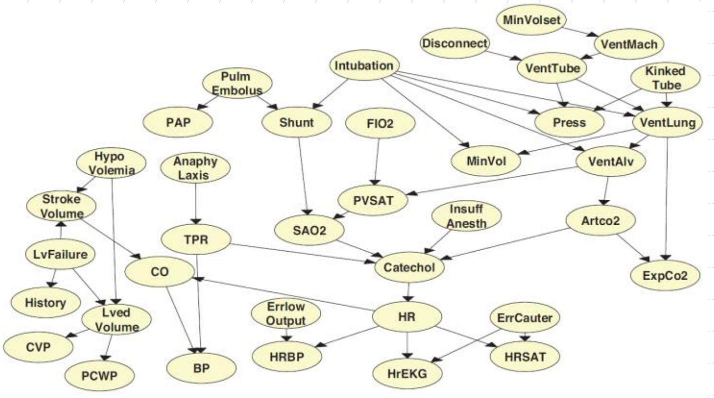
Naive Bayes Classifier (as a DGM)



$$
p(y, \mathbf{x}) = p(y) \prod_{j=1}^{D} p(x_j|y)
$$

DGMs – complex nets

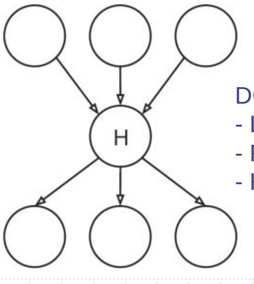
Alarm network for intensive care unit (measures features such as the breathing rate and blood pressure of a patient): 37 variables and 504 parameters



Latent Variable Models

Probabilistic models with hidden (i.e. non-observed) variables are also known as **latent variable models (LVMs).**

These latent variables can also serve as a **bottleneck,** computing a compressed representation of the data**.**



DGM example:
- Leaves: medical symptoms
- Roots: primary causes (e.g. smoking, diet)
- Hidden variable: mediating factors (e.g. heart disease)

Mixture Models

Simplest form of LVM has discrete latent states z_i

Define $p(\mathbf{x}_i|z_i = k) = p_k(\mathbf{x}_i)$

Mixture model: $p(\mathbf{x}_i|\boldsymbol{\theta}) = \sum_{k=1}^{K} \pi_k p_k(\mathbf{x}_i|\boldsymbol{\theta})$

where $\boldsymbol{\theta}$ are model parameters and π_k stands for $p(z=k)$

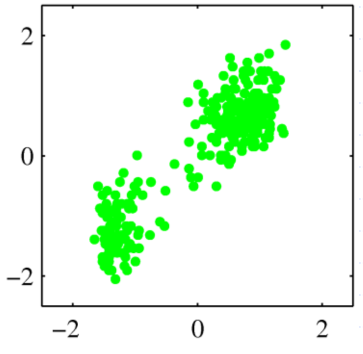
π_k are also called **mixing weights**

Such models are widely used in pattern recognition

K-means clustering (1)

We begin the discussion on mixtures by considering the problem of finding clusters in a set of data points

Approach: **K-means algorithm** (non-probabilistic technique)



unsupervised learning

K-means clustering (2)

Suppose we have a data set $\{x_1,x_2,\ldots,x_N\}$ consisting of N observations of a random D-dimensional variable

Goal: partition the data into K clusters (K is given)

Define $\boldsymbol{\mu}_k$ as a prototype associated with the k-th cluster

Define $r_{nk}=\{0,1\}$ a binary indicator variable (describes which of the clusters the data point x_n is assigned to)

Objective function: (to be minimised)

$$J = \sum_{n=1}^{N}\sum_{k=1}^{K} r_{nk}\|\mathbf{x}_n - \boldsymbol{\mu}_k\|^2$$

K-means clustering (3)

Iterative algorithm:

1. Choose initial values for $\boldsymbol{\mu}_k$
2. Minimise J wrt $\boldsymbol{r_{nk}}$
3. Minimise J wrt $\boldsymbol{\mu}_k$
4. Repeat 2-3 until convergence

K-means clustering (4)

Algorithm details:

- Updating r_{nk} :

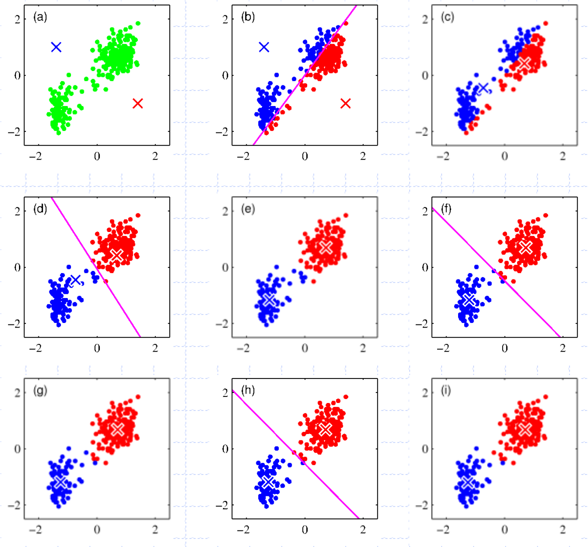
$$r_{nk} = \begin{cases} 1 & \text{if } k = \arg\min_j \|\mathbf{x}_n - \boldsymbol{\mu}_j\|^2 \\ 0 & \text{otherwise.} \end{cases}$$

- Updating $\boldsymbol{\mu}_k$:

$$\boldsymbol{\mu}_k = \frac{\sum_n r_{nk}\mathbf{x}_n}{\sum_n r_{nk}}$$

So $\boldsymbol{\mu}_k$ is the mean of the k-th cluster, thus the name k-means

K-means clustering (5)



Example

You have the following set of 2-dimensional data points: {1.9, 1.9}, {0.9, 1.1}, {1.8, 2.0}, {0.8, 1.0}, {1.1, 0.9}, {2.0, 1.9}, {1.0, 0.9}, {1.9, 1.8}.

Apply K-means with K=2 clusters using the following initial values for cluster prototypes: μ_1 = {1.0, 1.0} and μ_2 = {2.0, 2.0}.

Which are the values of the final prototypes for each cluster?

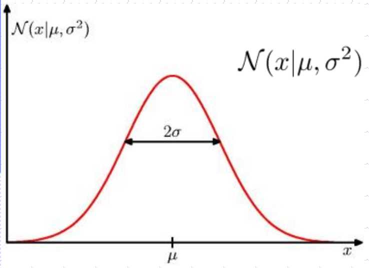
To which cluster does each data point belong to?

Model Answer

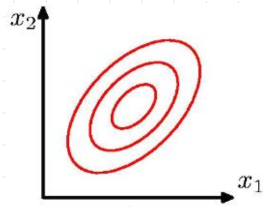
- Step 1: Initial values for cluster prototypes (given)
- Step 2: Estimating binary indicator variable
  - r1 = {0,1,0,1,1,0,1,0}, r2 = {1,0,1,0,0,1,0,1}
- Step 3: Updating mu
  - mu1 = (x2+x4+x5+x7)/4 = {0.95,0.975},
  - mu2 = (x1+x3+x6+x8)/4 = {1.9,1.9}
- Step 2 again: Estimating binary indicator variable
  - r1 = {0,1,0,1,1,0,1,0}, r2 = {1,0,1,0,0,1,0,1}
- Step 3 again: Updating mu
  - mu1 = (x2+x4+x5+x7)/4 = {0.95,0.975},
  - mu2 = (x1+x3+x6+x8)/4 = {1.9,1.9}
- The output of Step 3 is the same as in the previous iteration: convergence achieved
- Solution:
  - mu1 = {0.95,0.975}, mu2 = {1.9,1.9},
  - r1 = {0,1,0,1,1,0,1,0}, r2 = {1,0,1,0,0,1,0,1}

Mixtures of Gaussians (1)

The Gaussian Distribution:



$$\mathcal{N}(x|\mu,\sigma^2) = \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left\{-\frac{1}{2\sigma^2}(x-\mu)^2\right\}$$



$$\mathcal{N}(\mathbf{x}|\boldsymbol{\mu},\boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{D/2}} \frac{1}{|\boldsymbol{\Sigma}|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right\}$$

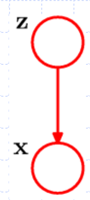
17

Mixtures of Gaussians (2)

A Gaussian mixture distribution (also called GMM):

$$p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

Define an indicator variable z_k that is characterized by:



- $z_k \in \{0,1\}$
- $p(z_k = 1) = \pi_k$
- $\sum_k z_k = 1$

If we define the joint distribution $p(\mathbf{x},\mathbf{z})$:

$$p(\mathbf{x}) = \sum_{\mathbf{z}} p(\mathbf{z})p(\mathbf{x}|\mathbf{z}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

18

Mixtures of Gaussians (3)

It might seem that we have not gained much by expressing a Gaussian mixture using a latent variable...

But: now we are able to work with $p(\mathbf{x},\mathbf{z})$ instead of $p(\mathbf{x})$, which will lead to significant simplifications

Another important quantity: the conditional probability of z given x

$$\gamma(z_k) \equiv p(z_k = 1|\mathbf{x}) = \frac{p(z_k = 1)p(\mathbf{x}|z_k = 1)}{\sum_{j=1}^{K} p(z_j = 1)p(\mathbf{x}|z_j = 1)} = \frac{\pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}.$$

Also called:
responsibility

Mixtures of Gaussians (4)

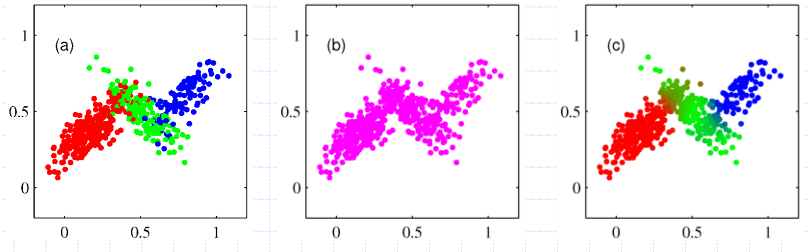


Figure: samples from a distribution of three 2-D Gaussians:

(a) True distribution
(b) Data
(c) Responsibilities

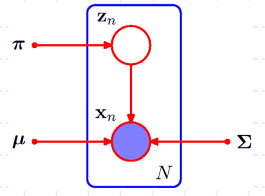
Mixtures of Gaussians (5)

Maximum likelihood for GMMs

Suppose we have data $\{x_1, x_2, \ldots, x_N\}$, represented as matrix $\mathbf{X}$

Expressing the log-likelihood of the data using a GMM:

$$\ln p(\mathbf{X}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{n=1}^{N} \ln \left\{ \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right\}$$



Maximising the above function is problematic:

- Singularities, i.e. discontinuous function
- Given a MLE, a K-component mixture will have K! solutions (identifiability problem)

EM for Gaussian Mixtures (1)

A powerful method for finding a maximum likelihood estimation MLE solution for latent variable models is the **Expectation-Maximisation algorithm (EM)**

Setting the derivatives of the log-likelihood to 0 wrt $\boldsymbol{\mu}_k$:

$$0 = -\sum_{n=1}^{N} \underbrace{\frac{\pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_j \pi_j \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}}_{\gamma(z_{nk})} \boldsymbol{\Sigma}_k (\mathbf{x}_n - \boldsymbol{\mu}_k)$$

After rearranging:

$$\boldsymbol{\mu}_k = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) \mathbf{x}_n \qquad N_k = \sum_{n=1}^{N} \gamma(z_{nk})$$

(N_k: number of points assigned to cluster k)

EM for Gaussian Mixtures (2)

Setting the derivative of the log-likelihood to 0 wrt $\boldsymbol{\Sigma}_k$:

$$\boldsymbol{\Sigma}_k = \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk})(\mathbf{x}_n - \boldsymbol{\mu}_k)(\mathbf{x}_n - \boldsymbol{\mu}_k)^{\mathrm{T}}$$

Setting the derivative of the log-likelihood to 0 wrt π_k :

$$\pi_k = \frac{N_k}{N}$$

These results do not constitute a closed-form solution, since the responsibilities depend on these parameters – but they suggest a simple **iterative scheme** for finding a solution...

23

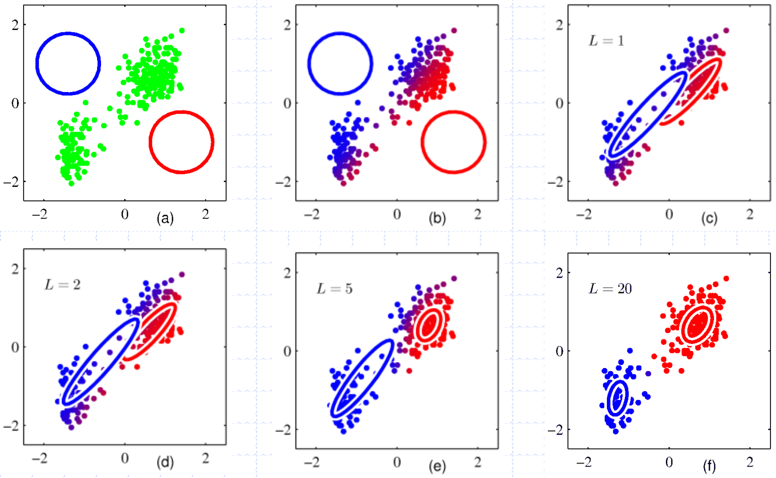
EM for Gaussian Mixtures (3)

Goal: given a GMM, maximize the likelihood function wrt the means, covariances, and mixing coefficients.

1. Initialize $\boldsymbol{\mu}_k$, $\boldsymbol{\Sigma}_k$, π_k
2. Expectation step (E-step): $\gamma(z_{nk}) = \dfrac{\pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}$
3. Maximization step (M-step):
$$\begin{aligned} \boldsymbol{\mu}_k^{\text{new}} &= \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) \mathbf{x}_n \\ \boldsymbol{\Sigma}_k^{\text{new}} &= \frac{1}{N_k} \sum_{n=1}^{N} \gamma(z_{nk}) \left(\mathbf{x}_n - \boldsymbol{\mu}_k^{\text{new}}\right) \left(\mathbf{x}_n - \boldsymbol{\mu}_k^{\text{new}}\right)^{\mathrm{T}} \\ \pi_k^{\text{new}} &= \frac{N_k}{N} \end{aligned}$$
4. Evaluate the log-likelihood and check for convergence. If criterion is not satisfied, go to step 2.

24

EM for Gaussian Mixtures (4)



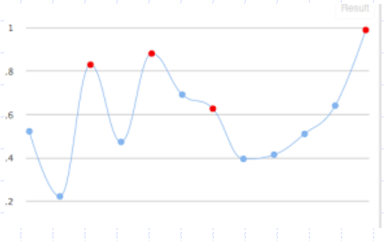
EM for Gaussian Mixtures (5)

Note that EM takes many more iterations to reach convergence compared with k-means

Common approach: initialize a GMM using k-means

There might be multiple local maxima – EM is not guaranteed to find a global maximum

But **convergence is guaranteed:**



GMM Classifier

GMM classifier: simple but useful supervised learning classification algorithm; good for the classification of faces and non-temporal pattern recognition

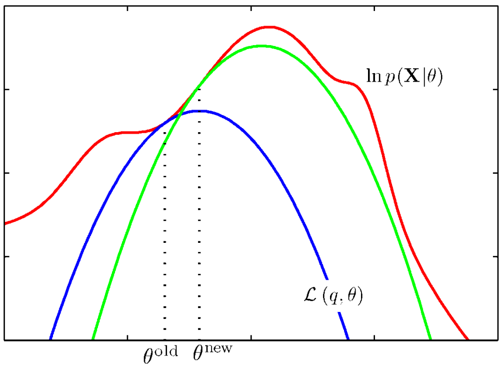
1. Train a GMM for each class (using EM)

2. Testing: compute the likelihood of the test sample for each GMM. Select as class the one that produces the largest likelihood

Applications of GMMs

- Speaker identification
- Image retrieval
- Biometric verification
- Speech/sound recognition
- Traffic flow control
- Emotion recognition
- Weather prediction

The General EM Algorithm (1)



The EM algorithm involves computing alternately a lower bound on the log likelihood for the current parameter values and then maximizing this bound to obtain the new parameter values.

EM converges to local maximum of likelihood.

The General EM Algorithm (2)

Given a joint distribution $p(\mathbf{X},\mathbf{Z}|\boldsymbol{\theta})$ over observed variables $\mathbf{X}$ and latent variables $\mathbf{Z}$, governed by parameters $\boldsymbol{\theta}$, the goal is to maximize $p(\mathbf{X}|\boldsymbol{\theta})$ wrt $\boldsymbol{\theta}$.

1. Initialize $\boldsymbol{\theta}^{\text{old}}$
2. E-step: evaluate $p(\mathbf{Z}|\mathbf{X},\boldsymbol{\theta}^{\text{old}})$
3. M-step: evaluate $\boldsymbol{\theta}^{\text{new}}$:

$$\boldsymbol{\theta}^{\text{new}} = \arg\max_{\boldsymbol{\theta}} \mathcal{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}})$$

$$\mathcal{Q}(\boldsymbol{\theta}, \boldsymbol{\theta}^{\text{old}}) = \sum_{\mathbf{Z}} p(\mathbf{Z}|\mathbf{X}, \boldsymbol{\theta}^{\text{old}}) \ln p(\mathbf{X}, \mathbf{Z}|\boldsymbol{\theta})$$

4. Check for convergence. If the convergence criterion is not satisfied: $\boldsymbol{\theta}^{\text{old}} \leftarrow \boldsymbol{\theta}^{\text{new}}$ and go to Step 2.

Comparing EM with k-means

Whereas the k-means algorithm performs a **hard assignment** from data points to clusters, EM makes a **soft assignment**.

We can derive k-means as a particular case of GMM without the need to estimate a covariance matrix

Original paper:

Maximum Likelihood from Incomplete Data via the EM Algorithm
A. P. Dempster; N. M. Laird; D. B. Rubin
Journal of the Royal Statistical Society B 39(1):1-38, 1977
http://web.mit.edu/6.435/www/Dempster77.pdf