Sapienza University of Rome

Master in Artificial Intelligence and Robotics
Master in Engineering in Computer Science

Machine Learning

A.Y. 2020/2021

Prof. L. Iocchi, F. Patrizi

14. Unsupervised Learning

L. Iocchi, F. Patrizi

with contributions from Valsamis Ntouskos

Overview

- Learning without a teacher
- K-means algorithm
- Gaussian Mixture Model
- Expectation Maximization algorithm
- General EM problem

A general technique for finding maximum likelihood estimators in latent variable models is the expectation-maximization (EM) algorithm

Reference

C. Bishop. Pattern Recognition and Machine Learning. Chapter 9.

Unsupervised Learning

we have only the input samples without labels

(labels)

Input data available $D = \{\mathbf{x}_n\}$, but target values not available.

Unsupervised data clustering: finding multiple classes from data.

Modelling input data useful when combined with supervised learning.

Gaussian Mixture Model

we motivated the Gaussian mixture model as a simple linear superposition of Gaussian components, aimed at providing a richer class of density models than the single Gaussian. We now turn to a formulation of Gaussian mixtures in terms of discrete latent variables.

we assume that the dataset contains only samples from the input space is generated from a probability distribution (this distribution is a weighted sum of gaussians)

Gaussian Mixture Model (GMM)

Mixed probability distribution P formed by k different Gaussian distributions

$$\boxed{P(\mathbf{x}) = \sum_{k=1}^{K} \overset{\text{prior}}{\pi_k} \mathcal{N}(\mathbf{x}; \overset{\text{mean}}{\boldsymbol{\mu}_k}, \overset{\text{covariance}}{\boldsymbol{\Sigma}_k})}$$

- π_k, prior probability
- $\boldsymbol{\mu}_k$, mean
- $\boldsymbol{\Sigma}_k$, covariance matrix

Unsupervised learning algorithms determine mixed probability distributions from data.

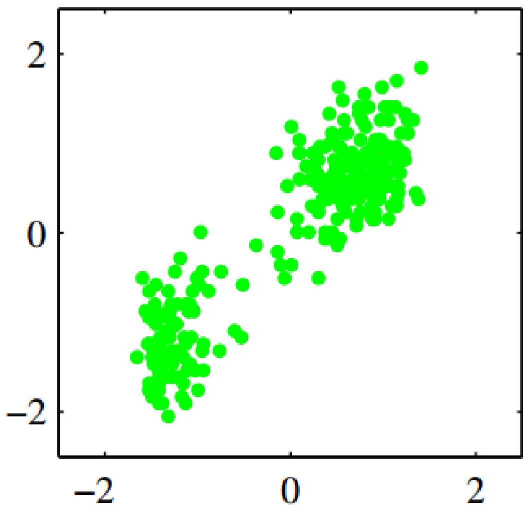
Generating data from mixture of Gaussians

Each instance $\mathbf{x}_n$ generated by

1. Choosing Gaussian k according to prior probabilities $[\pi_1, \ldots, \pi_K]$
2. Generating an instance at random according to that Gaussian, thus using $\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k$

for each gaussian we have a mean and a covariance matrix

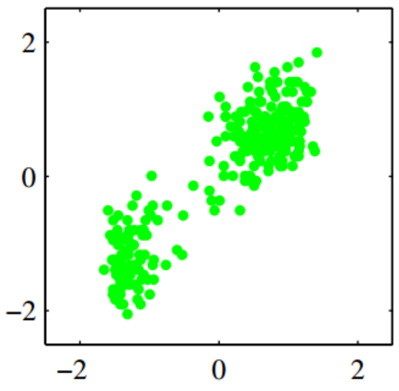
prior is the probability with which the gaussian generates the samples



K-means

simple case in which we consider only the mean

We begin by considering the problem of identifying groups, or clusters, of data points in a multidimensional space. Suppose we have a data set {x1, . . . , xN} consisting of N observations of a random D-dimensional Euclidean variable x. Our goal is to partition the data set into some number K of clusters, where we shall suppose for the moment that the value of K is given.



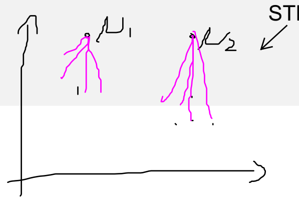
Computing K means of data generated from K Gaussian distributions.

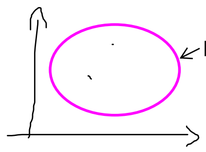
Input: $D = \{\mathbf{x}_n\}$, value K Output: $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K$

STEP 1: we want to assign each point in the dataset with the means

K-means

iterative algo





I have only point without + or - because we have no labels

the input is a set of unlabelled data

Step 1. Begin with a decision on the value of k = number of clusters

Step 2. Put any initial partition that classifies the data into k clusters. You may assign the training samples randomly, or systematically as follows

1. Take the first k training samples as single-element clusters
2. Assign each of the remaining (N-k) training samples to the cluster with the nearest centroid. After each assignment, recompute the centroid of the new cluster.

K-means

Step 3. Take each sample in sequence and compute its distance from the centroid of each of the clusters. If a sample is not currently in the cluster with the closest centroid, switch this sample to that cluster and update the centroid of the two clusters involved in the switch.

Step 4. Repeat step 3 until convergence is achieved, that is until a pass through the training sample causes no new assignments.

the algo will teminate when there are no changing of assignments from one iteration to the next one

K-means convergence

The convergence will always occur if the following conditions are satisfied:

1. For each switch in step 2, the sum of distances from each training sample to that training sample's group centroid is decreased.

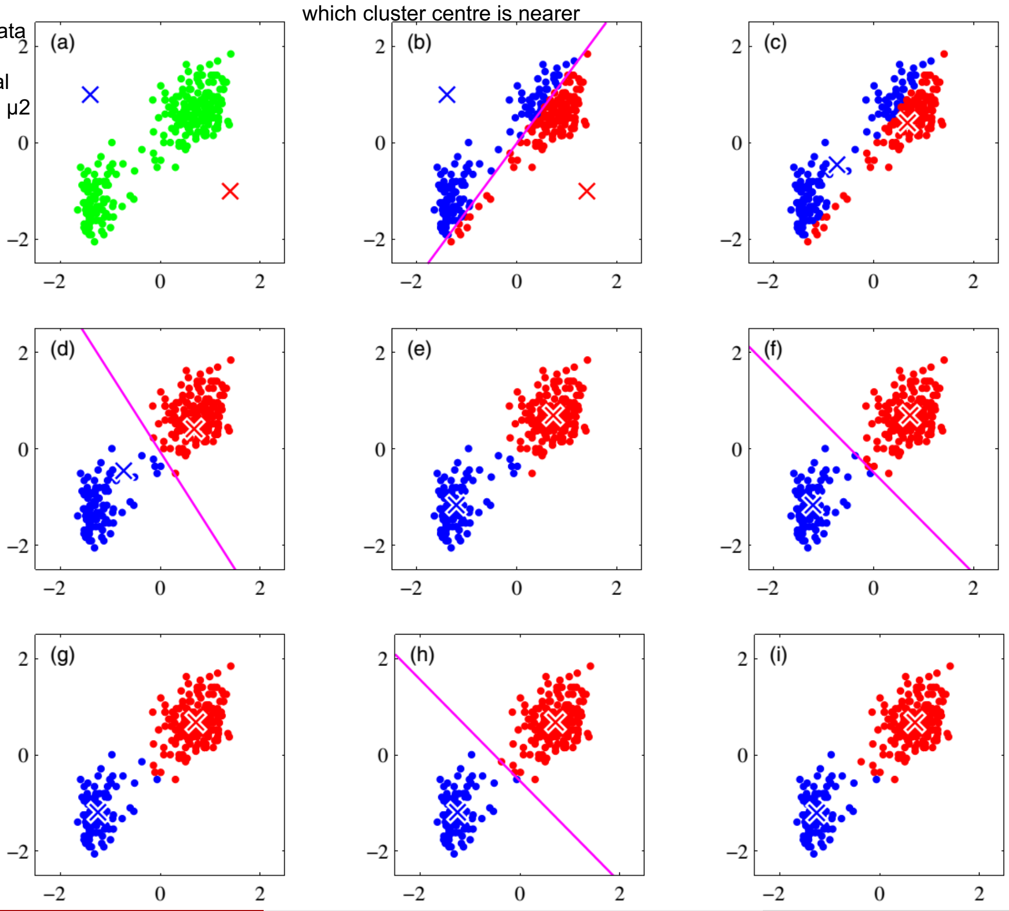
2. There are only finitely many partitions of the training examples into k clusters.

K-means example

each data point is assigned either to the red cluster or to the blue cluster, according to which cluster centre is nearer

Green points denote the data set in a two-dimensional Euclidean space. The initial choices for centres μ1 and μ2 are shown by the red and blue crosses, respectively

In the subsequent M step, each cluster centre is re-computed to be the mean of the points assigned to the corresponding cluster.



no change from (g) and so the algo stops

Remarks on K-means

K-means is done by considering tha distance but only the mean and not the covariance matrix

- The number of clusters K must be determined before hand.
- Sensitive to initial condition (local optimum) when a few data available.
- Not robust to outliers. Very far data from the centroid may pull the centroid away from the real one.
- The result is a circular cluster shape because it is based on distance.

limitation depending mostly on the distance function

Remarks on K-means

consider only some parameters

Some solutions:

- use K-means clustering only if there are many data available
- use median instead of mean
- define better *distance* functions

Gaussian Mixture Model

it considers all the parameters

LATENT VARIABLES: var that are useful for describing the problem but they are not part of the dataset

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

latent variable

Introduce new variables $z_k \in \{0, 1\}$, with $\mathbf{z} = (z_1, \dots, z_K)^T$ using a 1-out-of-K encoding (only one component is 1, all the others are 0).

for each sample in the dataset we will have that this vector is represented as 1-out-K encoding (it means that only one component of this vector will be 1 and the others will be 0). The component that is 1 denotes which is the gaussian that wll generate the sample

Let's define

$$\boxed{P(z_k = 1) = \pi_k}$$

thus

$$P(\mathbf{z}) = \prod_{k=1}^{K} \pi_k^{z_k}$$

Gaussian Mixture Model

For a given value of $\mathbf{z}$:

given zk, the probability of a point in a guassian depends only on the gaussian K

$$P(\mathbf{x}|z_k = 1) = \boxed{\mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}$$

Thus

$$P(\mathbf{x}|\mathbf{z}) = \prod_{k=1}^{K} \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)^{z_k}$$

Joint distribution: $P(\mathbf{x}, \mathbf{z}) = P(\mathbf{x}|\mathbf{z})P(\mathbf{z})$ (chain rule).

Gaussian Mixture Model

When $\mathbf{z}$ are variables with 1-out-of-K encoding and $P(z_k = 1) = \pi_k$

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

GMM distribution $P(\mathbf{x})$ can be seen as the marginalization of a distribution $P(\mathbf{x}, \mathbf{z})$ over variables $\mathbf{z}$.

Gaussian Mixture Model

Given observations $D = \{(\mathbf{x}_n)_{n=1}^N\}$, each data point $\mathbf{x}_n$ is associated to the corresponding variable $\mathbf{z}_n$ which is unknown.

Note: $z_{nk} = 1$ denotes $\mathbf{x}_n$ sampled from Gaussian k

$\mathbf{z}_n$ are called **latent variables**.

Analysis of latent variables allows for a better understanding of input data (e.g., dimensionality reduction).

Gaussian Mixture Model

Let's define the posterior

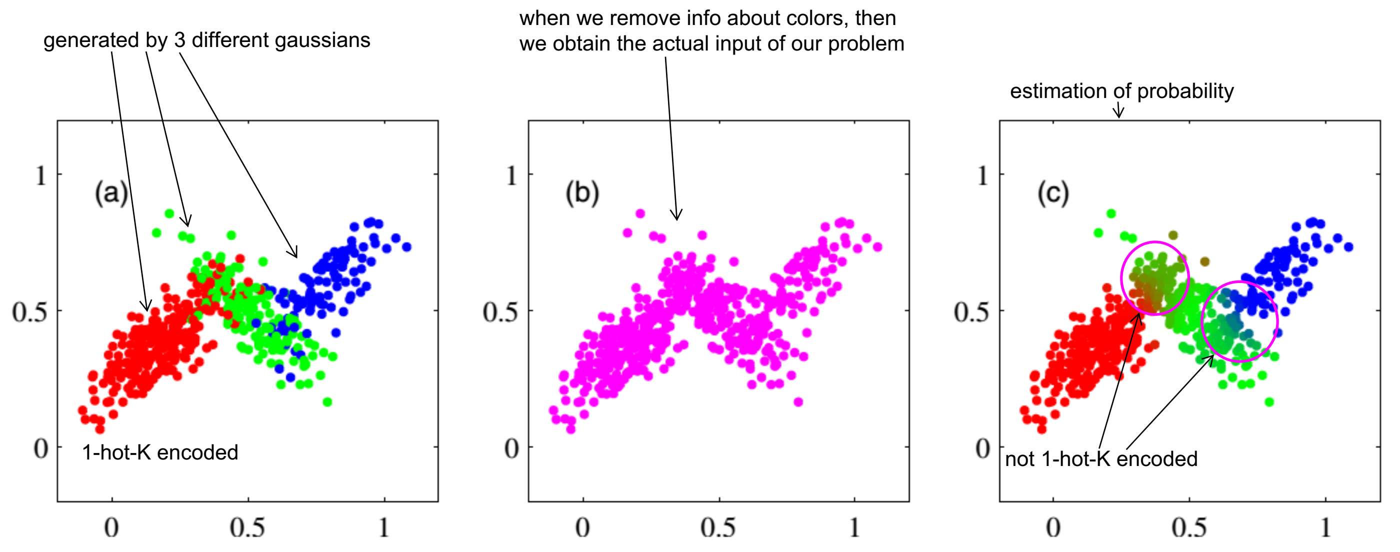
it is posterior because we compute the probability after looking at the data in the dataset

$$\gamma(z_k) \equiv P(z_k = 1|\mathbf{x}) = \frac{P(z_k = 1)\, P(\mathbf{x}|z_k = 1)}{P(\mathbf{x})}$$

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

Note:
π_k: prior probability of z_k
$\gamma(z_k)$: posterior probability after observation of $\mathbf{x}$.

Gaussian Mixture Model example



a) $P(\mathbf{x}, \mathbf{z})$ *with 3 latent variables* $\mathbf{z}$ *(red, green, blue)*
b) $P(\mathbf{x})$ *marginalized distribution*
c) $\gamma(z_{n,k})$ *posterior distribution*

Expectation Maximization (EM)

An elegant and powerful method for finding maximum likelihood solutions for models with latent variables is called the expectation-maximization algorithm, or EM algorithm

Gaussian Mixture Model

Given data set $D = \{(x_n)_{n=1}^N\}$ and GMM

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

determine $\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k, \pi_k$

Note: generalization of K-means algorithm

Expectation Maximization (EM)

Maximum likelihood

solution that maximize the likelihood

$$\underset{\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\Sigma}}{\operatorname{argmax}} \ln P(\mathbf{X}|\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\Sigma})$$

At maximum:

posterior probability

dataset

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

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

$$\boxed{\pi_k = \frac{N_k}{N}}, \quad \boxed{with \;\; N_k = \sum_{n=1}^{N} \gamma(z_{nk})}$$

Expectation Maximization (EM)

2 PARTS OF THE PROBLEM:

- **E step**
  Given $\pi_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k$, compute $\gamma(z_{nk})$
- **M step**
  Given $\gamma(z_{nk})$, compute $\pi_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k$

repeated in iterative way

Expectation Maximization (EM)

- Initialize $\pi_k^{(0)}, \boldsymbol{\mu}_k^{(0)}, \boldsymbol{\Sigma}_k^{(0)}$ random initialization of the parameters and evaluate the initial value of the log likelihood.
- Repeat until termination condition $t = 0, \ldots, T$
  - **E step** Evaluate the responsibilities using the current parameter values

$$\boxed{\gamma(z_{nk})^{(t+1)}} = \frac{\pi_k^{(t)} \mathcal{N}(\mathbf{x}_n; \boldsymbol{\mu}_k^{(t)}, \boldsymbol{\Sigma}_k^{(t)})}{\sum_{j=1}^K \pi_j^{(t)} \mathcal{N}(\mathbf{x}_k; \boldsymbol{\mu}_j^{(t)}, \boldsymbol{\Sigma}_j^{(t)})}$$

  - **M step** Re-estimate the parameters using the current responsibilities

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

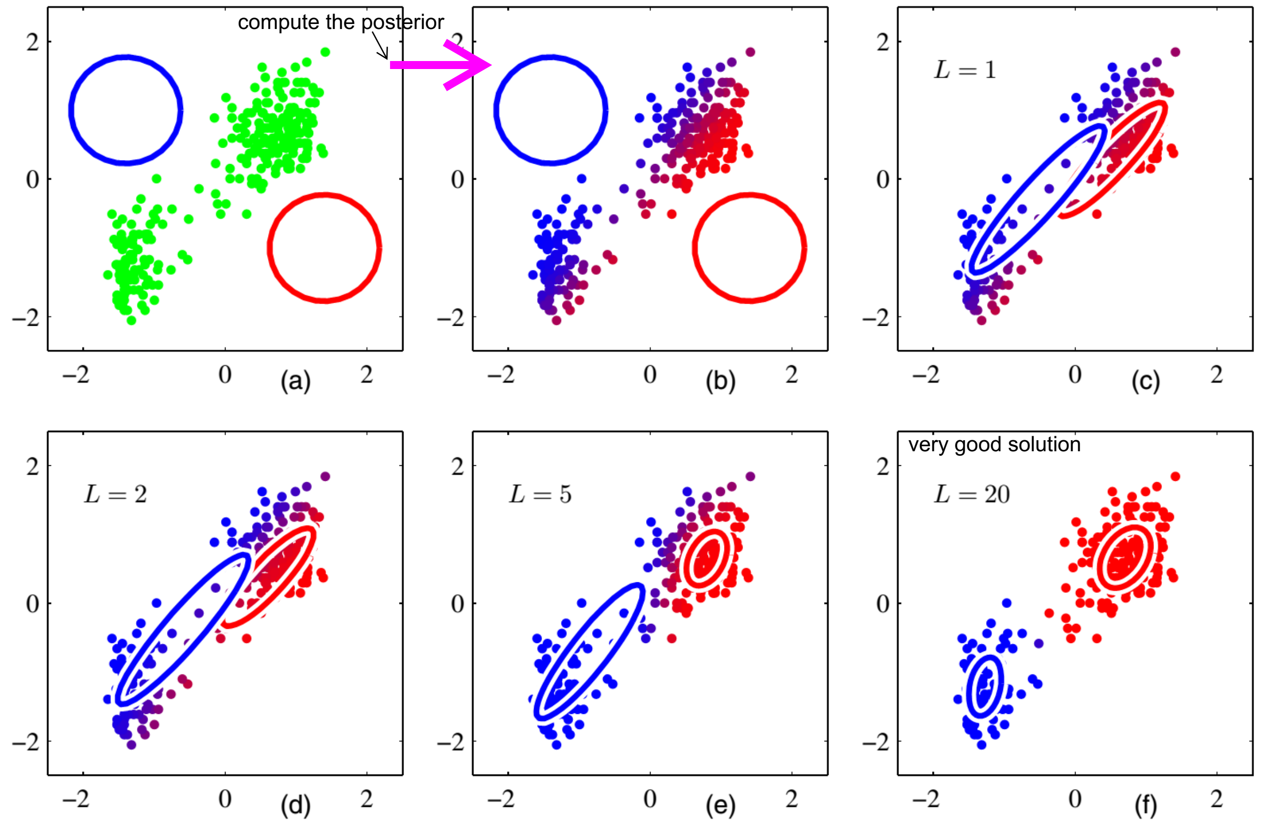
$$\boxed{\boldsymbol{\Sigma}_k^{(t+1)}} = \frac{1}{N_k} \sum_{n=1}^N \gamma(z_{nk})^{(t+1)} (\mathbf{x}_n - \boldsymbol{\mu}_k^{(t+1)})(\mathbf{x}_n - \boldsymbol{\mu}_k^{(t+1)})^T$$

$$\boxed{\pi_k^{(t+1)}} = \frac{N_k}{N}, \quad with \;\; N_k = \sum_{n=1}^N \gamma(z_{nk})^{(t+1)}$$

EM example

we have also the covariance, not only the mean

maximization step. we compute the params give the posterior



Remarks on EM Algorithm

- Converges to local maximum likelihood
- Provides estimates of the latent variables variables z_{nk}
- Extended version of K-means (probabilistic assignment to a cluster z_{nk})
- Can be generalized to other distributions (not only Gaussians)

General EM Problem

is a general technique for finding maximum likelihood solutions for probabilistic models having latent variables

Given:

- Observed data $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$
- Unobserved latent values $\mathbf{Z} = \{\mathbf{z}_1, \ldots, \mathbf{z}_N\}$
- Parametrized probability distribution $P(\mathbf{Y}|\boldsymbol{\theta})$, where
  - $\mathbf{Y} = \{\mathbf{y}_1, \ldots, \mathbf{y}_N\}$ is the full data $\mathbf{y}_n = \langle \mathbf{x}_n, \mathbf{z}_n \rangle$
  - $\boldsymbol{\theta}$ are the parameters

Determine:

- $\boldsymbol{\theta}^*$ that (locally) maximizes $E[\ln P(\mathbf{Y}|\boldsymbol{\theta})]$

Many uses:

- Unsupervised clustering
- Bayesian Networks
- Hidden Markov Models

General EM Method

set of all observed data

Define likelihood function $Q(\boldsymbol{\theta}'|\boldsymbol{\theta})$ defined on variables $\mathbf{Y} = \mathbf{X} \cup \mathbf{Z}$, using observed $\mathbf{X}$ and current parameters $\boldsymbol{\theta}$ to estimate $\mathbf{Z}$

set of all latent variables

EM Algorithm:

Estimation (E) step: Calculate $Q(\boldsymbol{\theta}'|\boldsymbol{\theta})$ using current hypothesis $\boldsymbol{\theta}$ and observed data $\mathbf{X}$ to estimate probability distribution over $\mathbf{Y}$

$$Q(\boldsymbol{\theta}'|\boldsymbol{\theta}) \leftarrow E[\ln P(\mathbf{Y}|\boldsymbol{\theta}')|\boldsymbol{\theta}, \mathbf{X}]$$

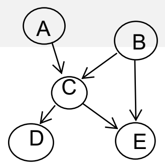
Maximization (M) step: Replace hypothesis $\boldsymbol{\theta}$ by the hypothesis $\boldsymbol{\theta}'$ that maximizes this Q function

$$\boldsymbol{\theta} \leftarrow \underset{\boldsymbol{\theta}'}{\operatorname{argmax}}\, Q(\boldsymbol{\theta}'|\boldsymbol{\theta})$$

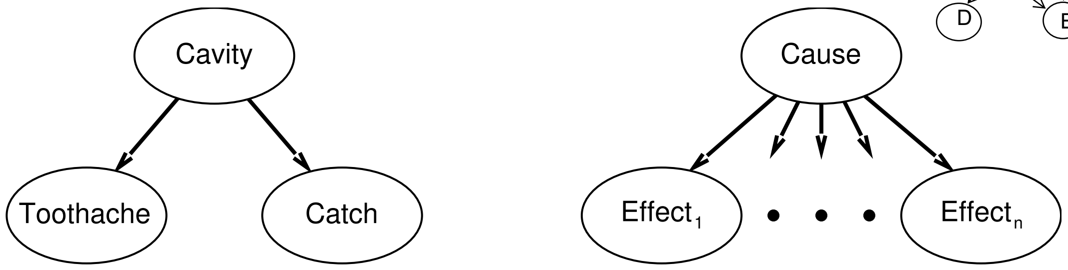
As a third example of the application of EM, we return to the evidence approximation for Bayesian linear regression

Bayesian networks

we have probability table on each edge



PGM (Probabilistic Graphical Model): graph representing conditional probability distribution among random variables



Syntax:

- a set of nodes, one per variable
- a directed, acyclic graph (link ≈ "directly influences")
- a conditional distribution for each node given its parents: $P(X_i|\mathrm{Parents}(X_i))$

In the simplest case, conditional distribution represented as a *conditional probability table* (CPT) giving the distribution over X_i for each combination of parent values.

Burglar BN Example

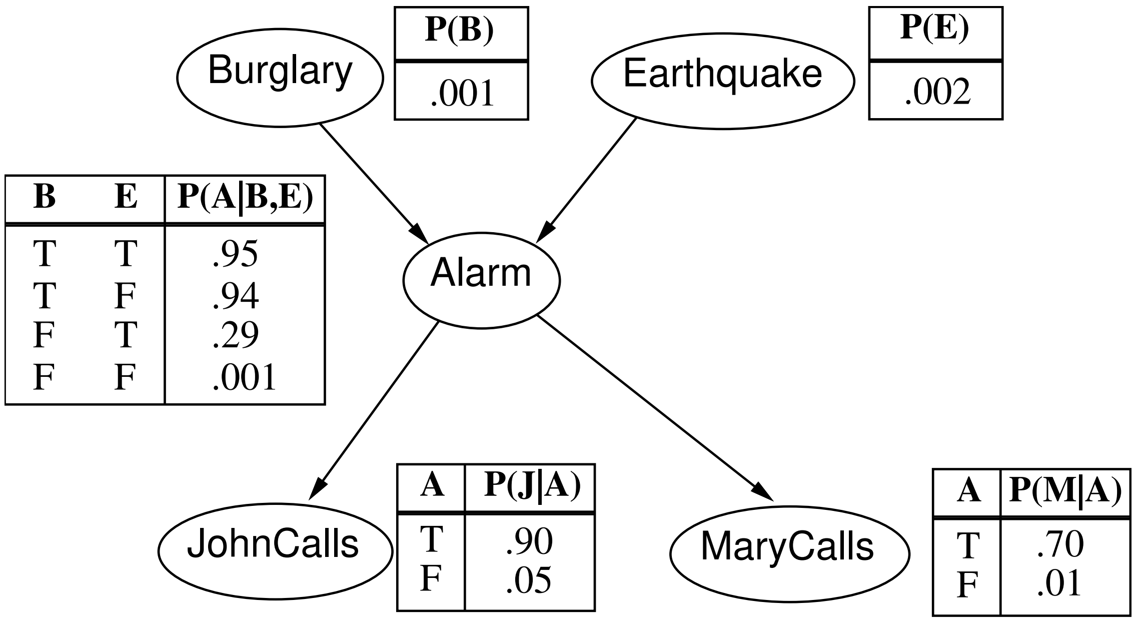
I'm at work, neighbor John calls to say my alarm is ringing, but neighbor Mary doesn't call. Sometimes the alarm is set off by minor earthquakes. Is there a burglar?

Variables: *Burglar*, *Earthquake*, *Alarm*, *JohnCalls*, *MaryCalls*
Network topology reflects "causal" knowledge:

- A burglar can set the alarm
- An earthquake can set the alarm
- The alarm can cause Mary to call
- The alarm can cause John to call

Burglar BN Example

Representation of BN



P(B)
.001

P(E)
.002

B	E	P(A|B,E)
T	T	.95
T	F	.94
F	T	.29
F	F	.001

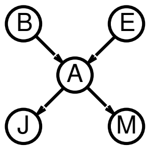
A	P(J|A)
T	.90
F	.05

A	P(M|A)
T	.70
F	.01

Computing joint probabilities

All joint probabilities computed with the chain rule:

$$P(x_1, \ldots, x_n) = \prod_{i=1}^{n} P(x_i|\mathrm{Parents}(X_i))$$

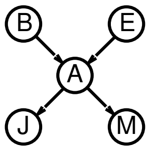


e.g., $P(j \wedge m \wedge a \wedge \neg b \wedge \neg e)$

$$\begin{aligned} &= P(j|a)P(m|a)P(a|\neg b, \neg e)P(\neg b)P(\neg e) \\ &= 0.9 \times 0.7 \times 0.001 \times 0.999 \times 0.998 \\ &\approx 0.00063 \end{aligned}$$

Compactness

A CPT for Boolean variable X_i with k Boolean parents has 2^k rows for the combinations of parent values



Each row requires one number p for $X_i = true$ (the number for $X_i = false$ is just $1 - p$)
If each variable has no more than k parents, the complete network requires $O(n \cdot 2^k)$ numbers
I.e., grows linearly with n, vs. $O(2^n)$ for the full joint distribution
For burglary net, $1 + 1 + 4 + 2 + 2 = 10$ numbers (vs. $2^5 - 1 = 31$)

Learning BN with observable variables

Bayesian Networks model dependencies among random variables.

When structure **known** and all variables **observable**

conditional probabilities can be estimated with maximum likelihood.

Example

$$P(j|a) \approx \frac{|\{\langle a, j, \ldots \rangle\}|}{|\{\langle a, \ldots \rangle\}|}$$

Learning BN with hidden variables

Unsupervised learning can be seen as learning with a BN with one hidden variable (the *class* of the instances).

This can be generalized to general BN with multiple hidden variables.

Learning BN with hidden variables: example

Consider three random variables $X \in \{0,1\}$, $A \in \{a_1, a_2\}$, $B \in \{b_1, b_2\}$, with X unobservable.

How can we learn BN parameters for $P(X)$, $P(A|X)$, and $P(B|X)$ from instances $D = \{d_1, ..., d_n\}$, with $d_k = \langle a_k, b_k \rangle$?

Define:
$P(X=0) = \theta_0$, $P(A=a_1|X=0) = \theta_1$, $P(A=a_1|X=1) = \theta_2$,
$P(B=b_1|X=0) = \theta_3$, $P(B=b_1|X=1) = \theta_4$

$\boldsymbol{\theta} = \langle \theta_0, \theta_1, \theta_2, \theta_3, \theta_4 \rangle$

Apply EM method to find maximum likelihood wrt $\boldsymbol{\theta}$ from D.

Learning BN with hidden variables: example

Estimation of BN parameters:

$$\boxed{\begin{aligned} P(X=x_j) &= \frac{1}{n} E[\hat{N}(X=x_j)] \\ P(A=a_i|X=x_j) &= \frac{E[\hat{N}(A=a_i, X=x_j)]}{E[\hat{N}(X=x_j)]} \end{aligned}}$$

Note that

$$E[\hat{N}(\cdot)] = E[\sum_k I(\cdot|d_k)] = \sum_k P(\cdot|d_k)$$

Learning BN with hidden variables: example

Estimation of BN parameters:

$$\begin{aligned} P(X = x_j) &= \frac{1}{n}\sum_{k=1}^{n} P(X = x_j | d_k) \\ P(A = a_i | X = x_j) &= \frac{\sum_{k=1}^{n} P(A = a_i, X = x_j | d_k)}{\sum_{k=1}^{n} P(X = x_j | d_k)} \\ P(B = b_l | X = x_j) &= \ldots \end{aligned}$$

Apply Bayes rule

$$\begin{aligned} P(x_j | d_k) &= P(x_j | \langle a_k, b_k \rangle) = \frac{P(a_k|x_j)P(b_k|x_j)}{\sum_i P(a_k|x_i)P(b_k|x_i)P(x_i)} = \phi_1(\boldsymbol{\theta}) \\ P(a_i, x_j | d_k) &= P(a_i | x_j, d_k) P(x_j | d_k) = \phi_2(\boldsymbol{\theta}) \\ P(b_l, x_j | d_k) &= P(b_l | x_j, d_k) P(x_j | d_k) = \phi_3(\boldsymbol{\theta}) \end{aligned}$$

... to define $Q(\boldsymbol{\theta}' | \boldsymbol{\theta})$

Summary

it's good for limited spaces

- Unsupervised learning useful to deal with unknown variables
- Clustering when labeled data are not available
- EM algorithm is a general method to estimate likelihood for mixed distributions including observed and latent variables
- Concepts to be extended to continuous latent variables