Unsupervised learning

Václav Hlaváč

Czech Technical University in Prague

Czech Institute of Informatics, Robotics and Cybernetics

160 00 Prague 6, Jugoslávských partyzánů 1580/3, Czech Republic

http://people.ciirc.cvut.cz/hlavac, vaclav.hlavac@cvut.cz

also Center for Machine Perception, http://cmp.felk.cvut.cz

Courtesy: M.I. Schlesinger, B. Flach.

Outline of the talk:

- Scope, motivation, cluster analysis.
- K-means, deterministic variant.
- Unsupervised learning, statistical approach.
- Use of unsupervised learning.
- K-means, statistical view.
- EM algorithm.

The scope of unsupervised learning informally

- Unsupervised learning is used to analyze observations (or data) when the information from a teacher (labels in a training multi-set) is not available. The alternative name: cluster analysis.
- The principle informally: data belonging to one class are more similar each to other than data belonging to other classes.
- Observed data should be explained by a mathematical model (to be found).

Approaches:

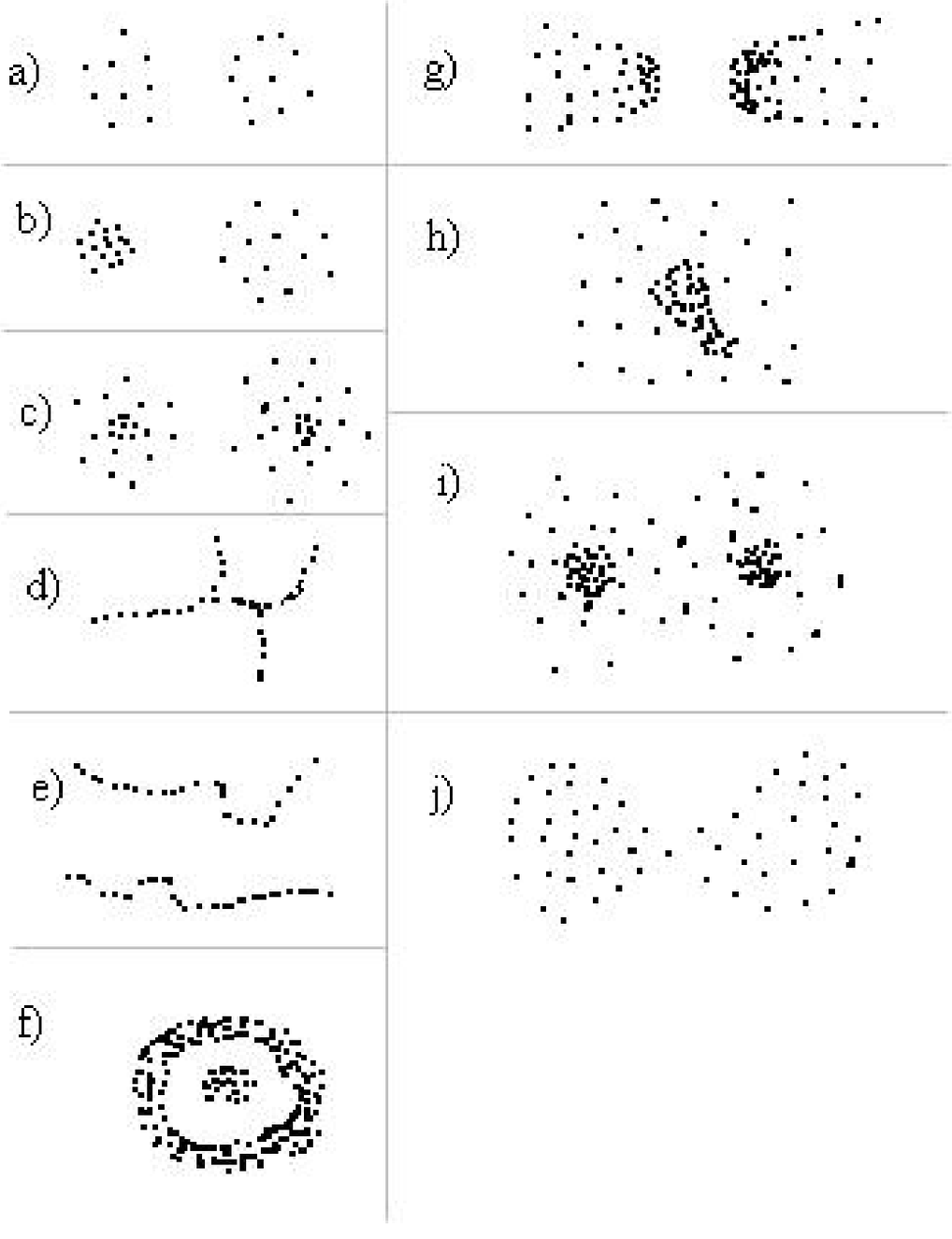
Parametric × non-parametric

- The underlying class-conditional probability distributions (likelihoods) are modeled with a mixture of parametric distributions.
- The objective is to find the model parameters.

Deterministic × statistical

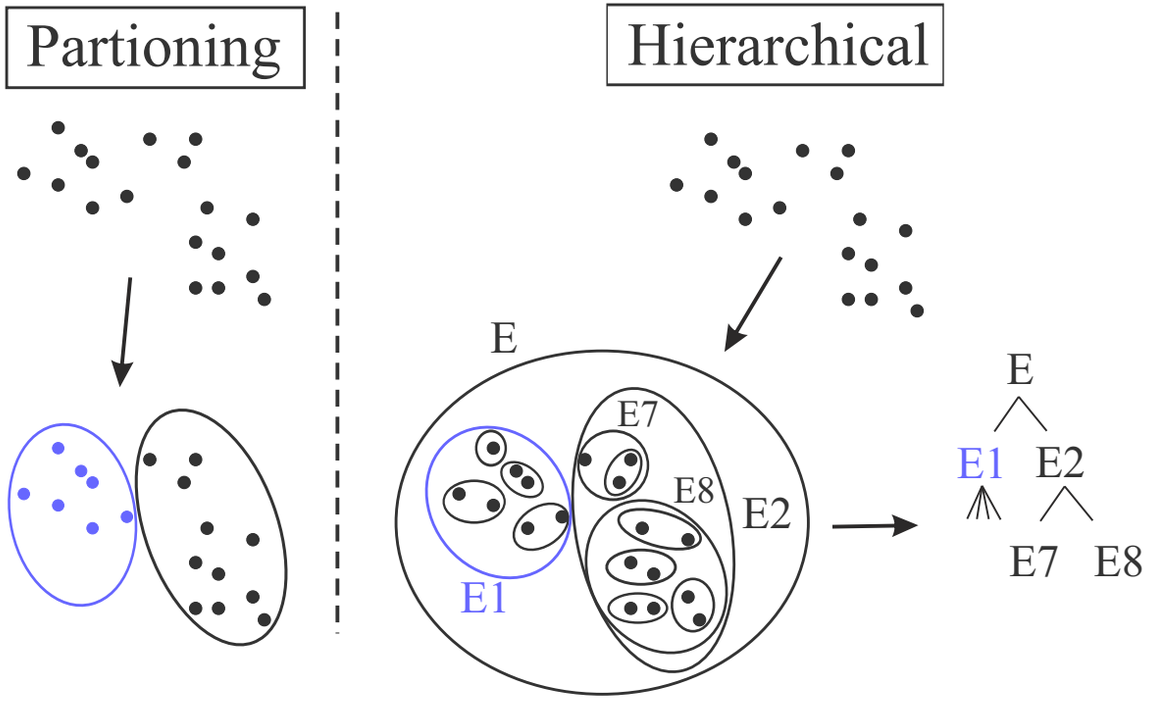
- Deterministic: according to the different measures of similarity, e.g., according to the distance.
- Statistical: a statistical model of the data has to be found from data only.

Motivating illustration



Cluster analysis, two approaches

1. Hierarchical methods – new clusters are sought based on previous clusters either in a top-down or a bottom-up manner.
2. Partitioning methods – clusters are found at one level of the hierarchy (these methods will be of our main concern in this lecture).
   - K-means algorithm. There is a deterministic and a statistical variant.
   - EM algorithm *(EM stands for Expectation-Maximization)*.



Courtesy: Figure; N. Nilsson 1996

Background, supervised learning

- One of the main paradigms of statistical pattern recognition and Bayesian inference is to model the relation between the observable features $x \in \mathcal{X}$ of an object and its hidden state (called also the class label) $y \in \mathcal{Y}$ by a joint probability measure $p(x, y)$.
- This probability measure is often known only up to some parameters Θ.
- In the empirical learning context, it is necessary to estimate these parameters from a training (multi-)set T, which is assumed to represent a sequence of independent realizations of a random variable.
- If the training set T, ideally, contains realizations of pairs (x, y), then the corresponding estimation methods are addressed as supervised learning.

Backround, unsupervised learning

- It is quite common that some of variables y describing the hidden state are hidden. These hidden variables are never observed in the training data.
- Therefore, it is necessary to marginalize over them in order to estimate the unknown parameters Θ.
- Corresponding estimation methods are known as unsupervised learning.
- Moreover, especially in computer vision, the observation x and the hidden state y may have both a complex structure. The hidden state y can be, e.g., a segmentation, a depth map, or a similar object. Consequently, it is often not feasible to provide the complete information y for the realizations in the sample. This means to estimate the parameters in the situation of missing information.
- *The EM algorithm is a method searching for maximum likelihood estimates of the unknown parameters Θ under such conditions.*

  We will start our explanation from the deterministic K-means algorithm. The statistical K-means can be understood a special case of EM-algorithm.

K-means clustering, deterministic, introduction

- The method aims to partition n observations to K clusters.
- Each observation belongs to the cluster, where the nearest centroid serves as its prototype.
- If the distance is taken deterministically then the data space is a partitioned into Voronoi cells.
- The algorithm was proposed by Stuart Lloyd in 1957 (still called Lloyd or Lloyd-Forgy algorithm in the computer science community).
- The term K-means was first used by James MacQueen in 1967.
- The task is computationally NP-hard.
- There are effective heuristic algorithm, which are often used and converge quickly to a local optimum.
- These local minima are usually similar as with EM algorithm used for a mixture of Gaussian probability distributions (to be explicated later).

Distances $dist$ used in K-means clustering

- x_i, x_j are the observations (data points) $\in X$;
  n is the number of observations.
- Distance $dist(x_i, x_j)$ is used to asses 'similarity' among observations.
- Quantitative variables
  - Error $d(x_i, x_j) = |x_i - x_j|$ or $(x_i - x_j)^2$ (Euclidean), etc.
  - Correlation coefficient $d(x_i, x_j) = \rho(x_i, x_j)$
- Ordinal variables, i.e. order is important, e.g. rank. They can be normalized to lie within $\langle 0, 1 \rangle$ (percentage). A quantitative metric can be applied $d(x_i, x_j) = \frac{i - \frac{1}{2}}{M}$, $i = 1, 2, \ldots, M$, where M is the number of equally sized bins.
- Categorical variables $d(x_i, x_j) = dist(x_i, x_j) = \left\{ \begin{array}{ll} 0, & \text{if} \quad x_i = x_j \\ 1, & \text{otherwise} \end{array} \right\}$ or distances must be specified by the user between each pair of categories.

K-means clustering

- A partitioning clustering algorithm.
- Data is partitioned into K clusters.
- The user has to specify:
  1. The number K of clusters;
  2. The distance function.
- Each cluster is represented by its centroid (mean), called also the cluster center.

The Lloyd's algorithm for Euclidean distance

1. Randomly choose K data points (seeds) to be the initial cluster centroids.
2. Assign each data point to the closest centroid.
3. Re-compute the centroids using the current cluster memberships.
4. If a stopping criterion is not met, go to step 2.

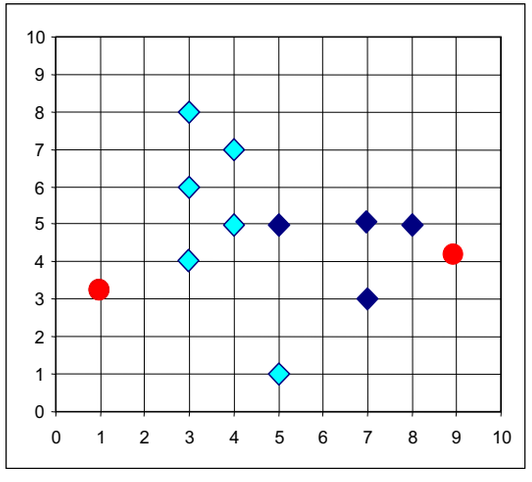
Stopping (convergence) criteria

- No (or minimum) re-assignments of data points to different clusters.
- No (or minimum) change of centroids.
- Minimal decrease in the sum of squared error (SSE),

$$\mathrm{SSE} = \sum_{j=1}^{k} \sum_{x \in C_j} \mathrm{dist}(x, \mu_j) \ ,$$

where C_j is j-th cluster, μ_j is the centroid of C_j (of all points in it); $dist(x, \mu_j)$ is the distance between a point x and centroid μ_j.

K-means clustering algorithm, example



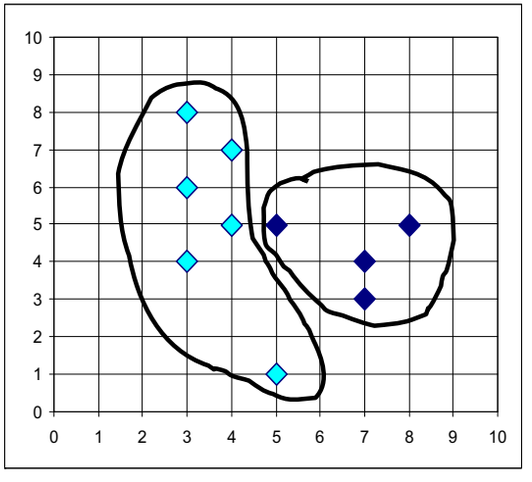
K=2

step 1
inicialization

Arbitrarily choose K object as initial cluster center ●

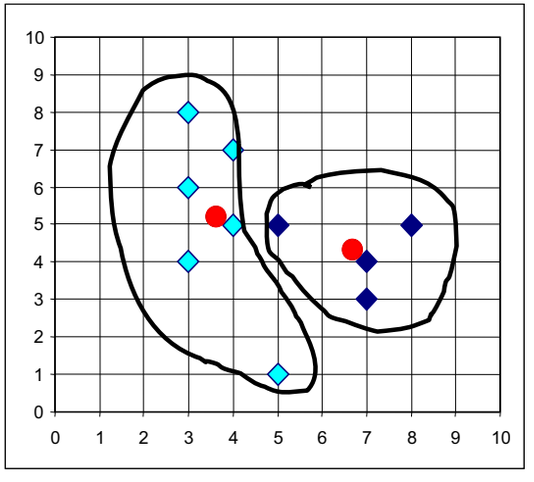
cycle 1
step 2

Assign each objects to most similar center

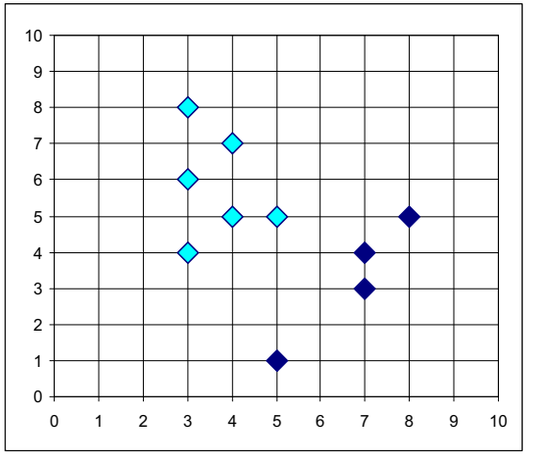


cycle 1
step 3

Update the cluster means

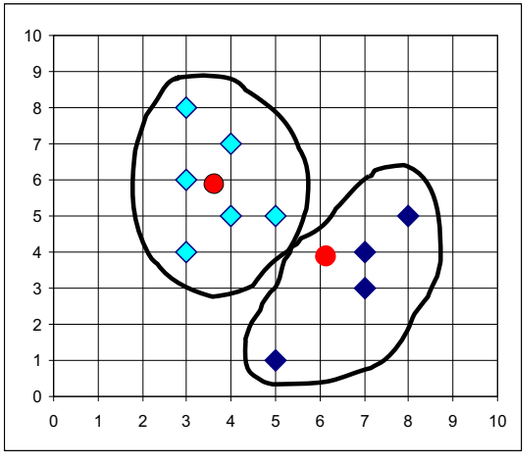


cycle 2, step 2 ↓ reassign



cycle 2
step 3

Update the cluster means



cycle 3, step 2 ↑ reassign, etc.

K-means demo from YouTube.

K-means, computational complexity

Regarding K-means algorithm computational complexity, finding the optimal solution for n observations in d dimensions is (Wikipedia):

- NP-hard in general Euclidean space d even for 2 clusters.
- NP-hard for a general number of clusters K even in the plane.
- If K and d are fixed, the problem can be exactly solved in time $\mathcal{O}(n^{dK+1} \log n)$.

- Variety of heuristic algorithms exists, e.g. for Lloyd's K-means algorithm as was described at Slide 9.

Strengths of K-means algorithm

- Simple. Easy to understand and implement.
- K-means is a very popular clustering algorithm.

Weaknesses of K-means clustering

- The user needs to specify the required number of clusters K.
- The algorithm is sensitive to initial seeds.
- Sensitivity to outliers as initial cluster seeds. How to deal with outliers?
  - Try to identify points located much further away from centroids than other points.
  - Perform random sampling. Since sampling chooses a small subset of the data points only. The chance of selecting an outlier is significantly smaller.
- The algorithm is not suitable for cluster of other shapes than hyper-ellipsoids. There is no good measure for cluster size and/or shape.
- The algorithm can be used only if the mean is defined. For categorical data, the centroid is usually replaced by the most frequent exemplar.

Statistical approach to unsupervised learning

- Observations are treated as random variables.
- The outcome of learning is a statistical model allowing to assign the observation $x \in X$ to the hidden state (or class) $y \in Y$ according to the modeled joint probability distribution $p(x, y)$.
- The class conditional probabilities (called also likelihoods) $p(x|y)$ are derived from the statistical model $p(x, y)$, i.e. the joint probability. They can be used for (Bayesian) decisions similarly as they are used in supervised learning.

- Data compression is an important application of unsupervised learning.

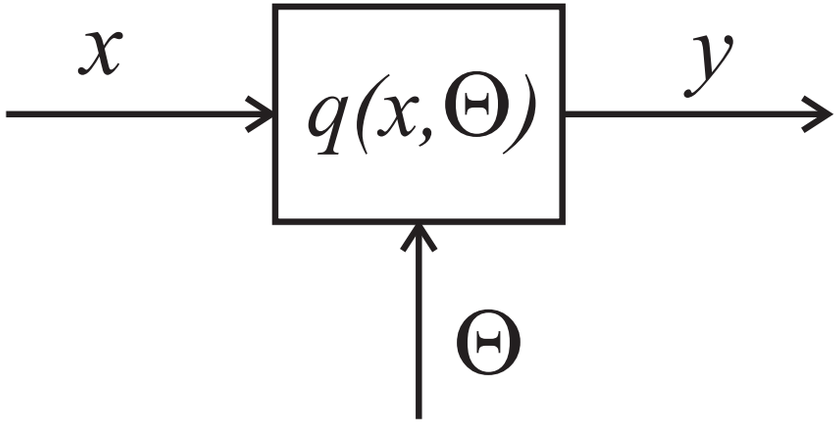
  Here is the link to the vector quantization from signal processing which allows the modeling of probability density function by the distribution prototype vectors. Large set of data points (vectors because they may lie in n-dim vector space) are divided into sets ideally of similar amount of points in it. They are represented by centroids.

Used symbols

- $x \in X$ - observations.
- $y \in Y$ – the hidden state, the outcome of the recognition.
- Θ – the parameter, on which the decision strategy depends (often a vector of parameters).
- $q(x, \Theta)$: $X \to Y$ – the decision strategy (a classifier).
- $x = (x_1, x_2, \ldots, x_n)$ – the sequence of observations is called the training sample T.
- $y = (y_1, y_2, \ldots, y_n)$ – the sequence of recognition results (from a training multi-set).
- In the supervised learning context, the training sequence (multi-set) is $\big((x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\big)$.

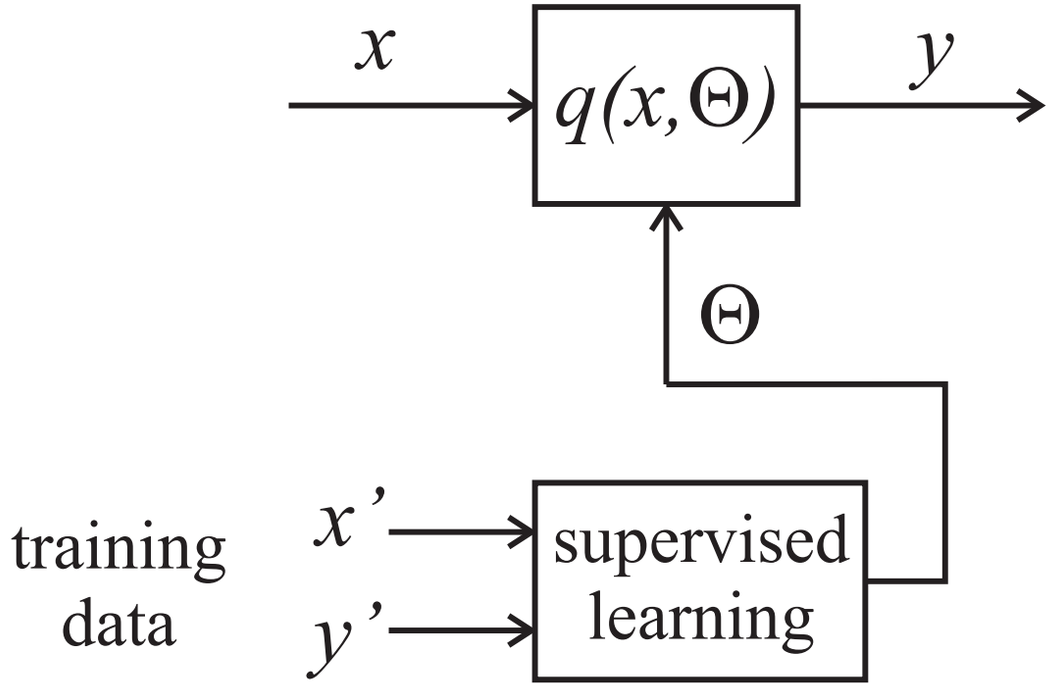
Classifier with adjustable parameters

- We have concentrated to the classifier design so far.
- The decision function q of a classifier depends on the parameter Θ.



$$y = q(x, \Theta)$$

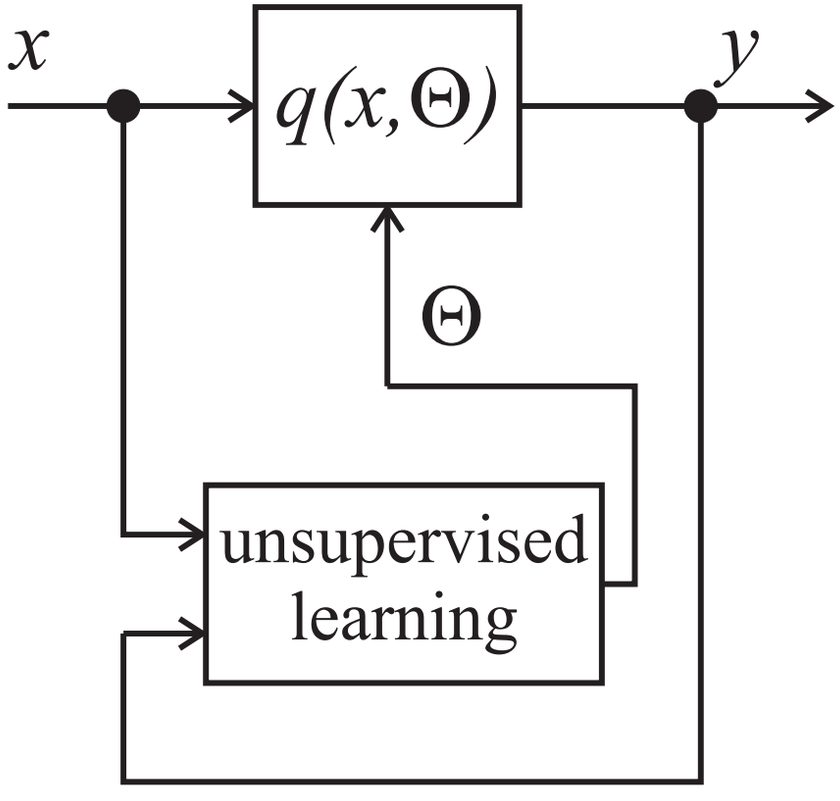
Supervised learning



$$\Theta = \text{learning}(x, y)$$

The decision rule $q(x, \Theta)$ is taught using the training multi-set.

Unsupervised learning



$$\Theta = \text{learning}(x, y)$$

The output of the classifier itself is used for (self)-learning instead of the hidden state y, which would be obtained from the training multi-set in the case of supervised learning.

Unsupervised learning algorithm

Initiatization, i.e. the initial choice of the parameter $\Theta^{t=0}$.

A cycle, where t is the iteration number in the cycle

- Recognition step $y = q(x, \Theta^t)$
- Learning step $\Theta^{t+1} = \text{learning}(x, y)$

The question of the algorithm convergence is an important one.

Use of unsupervised learning

- The data classification is not know in advance.

  Example: Data mining.

- The data classification by a human is often too expensive.

  Example: Speech recognition, in which a complicated Markovian sequence serves as the statistical model. Learning its parameters requires a large amount of training data.

- It is possible to compress large data sets in such a way that they are replaced by much fewer informative representatives.

  Note: It is not needed here, that the representatives have some semantics.

- Unsupervised learning can be used as a method approximating a complex probability distribution by a mixture of simpler distributions, often Gaussians.

Towards statistical learning: K-means algorithm (1)

- Let assume the statistical model

$$p_{x|k}(x|1,\mu_1) = p_{x|k}(x|2,\mu_2) = \ldots = p_{x|k}\left(x|K,\mu_K\right) \text{ , where}$$

  - $p_{x|k}(x|k_j,\mu_j)$ are Gaussian distributions with the unit covariance matrix.
  - The probability distribution parameters are given by the vector $\Theta = \mu_1,\ldots,\mu_K$.

- Recognition is performed according to Bayesian strategy

$$k = \underset{k'}{\operatorname{argmax}}\, p(k'|x) = \underset{k'}{\operatorname{argmin}}\, \|x - \mu_{k'}\|$$

corresponds to the classification based on the nearest neighbor approach.

Towards statistical learning: K-means algorithm (2)

$$
\begin{aligned}
\Theta_{t+1} &= \operatorname*{argmax}_{\Theta} \sum_{i=1}^{n} \log p_{xk}(x_i, k_i) \\
&= \operatorname*{argmax}_{\mu_1,\ldots,\mu_K} \sum_{i=1}^{n} \log \frac{1}{(\sqrt{2\pi})^{n}} \exp\left(-\frac{1}{2}(x_i - \mu_{k_i})^{\top}(x_i - \mu_{k_i})\right) \\
&= \operatorname*{argmin}_{\mu_1,\ldots,\mu_K} \sum_{i=1}^{n} (x_i - \mu_{k_i})^{\top}(x_i - \mu_{k_i}) \\
&= \operatorname*{argmin}_{1} \sum_{i \in I_1} (x_i - \mu_{k_i})^{\top}(x_i - \mu_{k_i}), \ldots, \\
&\qquad \operatorname*{argmin}_{k} \sum_{i \in I_k} (x_i - \mu_{k_i})^{\top}(x_i - \mu_{k_i}) \\
\Rightarrow \quad \mu_j &= \frac{1}{|I_j|} \sum_{i \in I_j} x_i \,, \quad j = 1, \ldots, K
\end{aligned}
$$

A move towards the EM algorithm

We know about probability distributions estimates from previous lectures:

- Parametric estimates, e.g. the maximally likely (ML) estimate.
- Nonparametric estimates, e.g. the Parzen window method or the n-nearest neighbors metods.

We have shown in previous example that the clustering using K-means corresponds to a special case of a statistical model, i.e. the mixture of Gaussian distributions with unitary covariance matrices. This means that these Gaussians have the same distributions with unitary dispersions.

Let us proceed to EM algorithm, which is an unsupervised learning method having a significant theoretical and practical significance.

EM algorithm, informally

- EM algorithm is an iterative estimation procedure from the maximal likelihood (ML) family for the cases, in which a direct ML solution does not exist or it is too complicated;
- EM algorithm decomposes a single complicated ML optimization task into several simpler optimization tasks by introducing a "missing parameter".
- It uses the gradient optimization (the steepest ascent) for finding the ML optimum. This is the reason why it suffers from getting stuck in a local minima.
- There are two main applications of the EM algorithm:
  - Data has incomplete, missing or corrupted values as a result of a faulty observation process.
  - The existence of missing or hidden parameters can simplify the likelihood function, which would otherwise lead to an analytically intractable optimization problem. E.g., teacher's labels are missing.

(The latter item, unsupervised learning, will be of our main interest in this lecture.)

EM algorithm; the definition

- The Expectation Maximization algorithm iteratively maximizes the likelihood of a training sample T with respect to unknown parameters Θ of a probability model under the condition of missing information.
- The training sample T is assumed to represent a set of independent realizations of a random variable defined on the underlying probability space.

Notation remarks on conditional probabilities:

- *We will deal with the conditional probabilities (likelihoods) of the form $p(x|y,\Theta)$, where Θ is a vector of parameters, e.g. of a decision strategy.*
- *We will also write the parameters Θ as a subscript: $p(x|y,\Theta) = p_\Theta(x|y)$ because it simplifies the expressions.*

EM, Theory (1)

- Let Ω be a finite sample space (i.e. all possible outcomes), $\mathcal{F}$ be its power set, and $p_\theta \colon \mathcal{F} \to \mathbb{R}_+$ be a probability measure defined up to unknown parameters $\theta \in \Theta$.
- The task is to estimate parameters θ given observations $X = \{x_1, \ldots, x_n\}$.
- Marginalize over missing hidden states y, i.e. $P(x|\theta) = \sum_y P(x, y|\theta)$.
- Let $X \colon \Omega \to \mathcal{X}$ be a random variable and the data $T = (x_1, x_2, \ldots x_n)$ be a sequence of independent realizations of X.
- The Maximum Likelihood (ML) estimator provides estimates the of the unknown parameter Θ by maximalizing the probability of T

$$\Theta^* = \operatorname*{argmax}_{\Theta} \prod_{i=1}^{n} p_\Theta(\Omega_i) \,,$$

where Ω_i denotes the pre-image (also inverse image) $\{\omega \in \Omega \,|\, X(\omega) = x_i\}$ selecting the corresponding patterns from the input sample space Ω.

Theory (2)

- The likelihood of observed data writes

$$l(\theta) = P(X|\theta) = \prod_{x_i \in X} P(X|\theta) = \prod_{x_i \in X} . \sum_{y} P(x, y|\theta) \, .$$

- The unknown parameters θ^* are obtained by maximizing the log-likelihood $L(\theta) = \log l(\theta)$; $\theta^* = \underset{\Theta}{\operatorname{argmax}} \prod_{i=1}^{n} p_\theta(\Omega_i)$, where $\Omega_i = \{\omega \in \Omega \,|\, X(\omega) = x_i\}$. Ω_i is called the pre-image.
- After logarithm is taken, the optimization task reads

$$\theta^* = \underset{\theta \in \Theta}{\operatorname{argmax}} \; L(x_1, \ldots, x_n) = \underset{\theta \in \Theta}{\operatorname{argmax}} \sum_{i=1}^{n} \log \sum_{\omega \in \Omega_i} p_\Theta(\omega) \qquad (1)$$

EM, Remark about marginalization

In the supervised learning context:

- The sample space Ω is assumed a Cartesian product $\Omega = \mathcal{X} \times \mathcal{Y}$, where $x \in \mathcal{X}$ are observations (measurements) and $y \in \mathcal{Y}$ are hidden states (or class labels more specially).
- Let $X' : \Omega \to (\mathcal{X} \times \mathcal{Y})$ be a random variable and the training multi-set $T = ((x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n))$ be a sequence of independent realizations of X'.

In the unsupervised learning context:

- The hidden state $y \in \mathcal{Y}$ is not known. The set X simply projects onto the first component $X(x, y) = x$ of X'.
- In such a case, the probability $p_\Theta(\Omega_i) = \sum_{y \in \mathcal{Y}} p_\Theta(x_i, y)$ is nothing but the marginalization over all possible y.
- This special case (unsupervised learning) will be considered in this lecture.

EM, Theory (3)

- The optimization task (1) at the slide 28 is often complicated and hardly solvable by standard methods. The objective function is either not concave or parameters Θ are of a rather different natures.
- Suppose, however, that the task of parameter estimation is feasible if the complete information, i.e., a set of realizations of $\omega \in \Omega$, were available.
- This applies in particular if the corresponding simpler objective function

$$\sum_i \log p_\Theta(\omega_i)$$

is concave with respect to parameters Θ or if the task decomposes into simpler independent optimization tasks with respect to individual components of a parameter collection.

EM, Theory (4)

The key idea of the Expectation Maximization algorithm is to exploit this circumstance and to solve the optimization task (1) from slide 28 by the "iterative splitting", i.e. by iterating the following two feasible tasks:

1. Given a current estimate of Θ, estimate (E-step) the missing information, i.e., $p_\Theta(\omega|\Omega_i)$ for each element $x_i \in T$.
2. Given the complete information, solve the corresponding estimation (maximization, M-step) task, resulting in an improved estimate of Θ.

EM, Theory (5)

It is convenient to introduce nonnegative auxiliary variables $\alpha_i(\omega)$, $\omega \in \Omega_i$, for each element x_i of the training sample T fulfilling

$$\sum_{\omega \in \Omega_i} \alpha_i(\omega) = 1\,, \quad \forall i = 1, 2, \ldots n\,. \tag{2}$$

Variables α_i can be seen as posterior probabilities $p(\omega|\Omega_i)$ for $\omega \in \Omega_i$, given a realization x_i. The log-likelihood of a realization x_i can be written when using α_i as

$$\log p_\Theta(\Omega_i) = \sum_{\omega \in \Omega_i} \alpha_i(\omega) \log p_\Theta(\Omega_i)\ , \text{(which follows from Equation (2))} \tag{3}$$

$$= \underbrace{\sum_{\omega \in \Omega_i} \alpha_i(\omega) \log p_\Theta(\omega) - \left(\sum_{\omega \in \Omega_i} \alpha_i(\omega) \log p_\Theta(\omega) \right.}_{=0} \left. - \sum_{\omega \in \Omega_i} \alpha_i(\omega) \log p_\Theta(\Omega_i) \right)$$

$$= \sum_{\omega \in \Omega_i} \alpha_i(\omega) \log p_\Theta(\omega) - \sum_{\omega \in \Omega_i} \alpha_i(\omega) \log \frac{p_\Theta(\omega)}{p_\Theta(\Omega_i)} \tag{4}$$

EM, Theory (6)

The log-likelihood of the training sample T is

$$
\begin{aligned}
L(T,\Omega) &= L(x_1, x_2, \ldots, x_n, \Omega) = \sum_{i=1}^{n} \underbrace{\log p_\Theta(\Omega_i)}_{\text{substitute using (3) from slide 32}} \qquad (5)\\
&= \sum_{i=1}^{n} \sum_{\omega \in \Omega_i} \alpha_i(\omega) \log p_\Theta(\omega) - \sum_{i=1}^{n} \sum_{\omega \in \Omega_i} \alpha_i(\omega) \log \frac{p_\Theta(\omega)}{p_\Theta(\Omega_i)}\\
&= \sum_{i=1}^{n} \sum_{\omega \in \Omega_i} \alpha_i(\omega) \log p_\Theta(\omega) - \sum_{i=1}^{n} \sum_{\omega \in \Omega_i} \alpha_i(\omega) \log p_\Theta(\omega|\Omega_i)
\end{aligned}
$$

- The expression (5) does not depend as a whole on the specific choice of the auxiliary variables α, whereas the minuend and subtrahend do.
- The minuend is the likelihood of a complete data sample if the α are interpreted as the missing information, i.e., posterior probabilities for $\omega \in \Omega_i$ given the observation x_i. The subtrahend will be mentioned later.

EM algorithm

Starting with some reasonable choice for the initial $\Theta^{(0)}$, the likelihood is iteratively increased by alternating the following two steps.

- The (E)xpectation step calculates a new α such that new $\Theta^{(t)}$ becomes a maximizer of the subtrahend. Changing Θ subsequently can only decrease its value.
- The (M)aximization step relies on this and maximizes the minuend only, avoiding to deal with the subtrahend.

- E-step:

$$\alpha_i^{(t)}(\omega) = p_\Theta^{(t)}(\omega) \tag{6}$$

- M-step:

$$\Theta^{(t+1)} = \operatorname*{argmax}_\Theta \sum_{i=1}^{n} \sum_{\omega \in \Omega_i} \alpha_i^{(t)}(\omega) \log p_\Theta(\omega) \tag{7}$$

EM algorithm, the conceptual point of view

- E-step can be seen as the inference. It calculates the missing data in each step t, i.e., the posterior probabilities $p_\Theta^{(t)}(\omega|\Omega_i)$ for each element x_i in the training sample T.
- The M-step utilizes these posterior probabilities in a supervised learning step.
- The names E-step and M-step stem from a rather formal view:
  - The E-step calculates the α and therefore the objective function in Equation (7) at the slide 34, which has the form of an expectation of $\log p_\Theta(\omega)$. The computation of this objective function is sometimes considered to be a part of the E-step.
  - The M-step name comes from maximization of the minuend (supervised learning).

EM, the monotonically decreasing likelihood

- Recall the slide 34. The choice for α in Equation (6) guarantees that the subtrahend in Equation (7) can decrease only whatever the new Θ will be chosen in the subsequent M-step. It follows from a special case of Johan Jensen's inequality (1906)

$$\sum_{\omega \in \Omega_i} p_{\Theta'}(\omega|\Omega_i) \log p_{\Theta'}(\omega|\Omega_i) \leq \sum_{\omega \in \Omega_i} p_{\Theta}(\omega|\Omega_i) \log p_{\Theta}(\omega|\Omega_i)\,,\ \forall \Theta' \neq \Theta$$

- Since the M-step chooses the new Θ to maximize the minuend, the likelihood can only increase or remain unchanged.
- There is an alternative way how to show the monotonicity of the EM algorithm. It considers the maximization of the likelihood and the Jensen's inequality. This alternative explanation is not explicated here.

EM algorithm properties

- EM algorithm maximizes the likelihood function $L_\Theta(T) = \sum_{i=1}^{n} \log\, p_\Theta(x_i)$.
- It is holds that $L(\Theta^0) \leq L(\Theta^1) \leq \ldots \leq L(\Theta^t)$.
- The sequence $L(\Theta^t)$ converges to L^* for $t \to \infty$ (i.e. L is upper bounded).
- L^* is either a local minimum, or the saddle point or the global minimum.
- If the function $\Theta^{t+1} = f(\Theta^t) = L\left(x, R(x, \Theta^t)\right)$ is continuous then the sequence $\Theta^0, \Theta^1, \ldots, \Theta^t$ converges to Θ^* for $t \to \infty$.
- EM algorithm is suitable for MLE (Maximum Likely Estimates).
- There is an analytic solution for simple statistical models, $\frac{\partial L(\Theta)}{\partial \Theta} = 0$.
- We have explained the derivation of the EM algorithm for a discrete probability space and discrete random variables. It can be however generalized for uncountable probability spaces and random variables X with continuous probability density.
- EM is a meta-algorithm, which needs to be adapted to a particular application.

EM algorithm properties; Global convergence?

Is there a case in which EM algorithm converges globally?

- In the case of a special statistical model, i.e. for the conditional independence model and two hidden states only, the EM algorithm converges to the global minimum (proved by M.I. Schlesinger).
- There is a hypothesis that for the conditional independence model it holds also for more than two hidden states.
- The M.I. Schlesinger's result seems to have a theoretical significance only. In practice, the pre-conditions are not met and the EM algorithm does not converge to the global minimum.

A special case
A mixture of probability distributions

- In a special case, when the probability model is decomposed into a mixture of K probability distributions $p_k(x)$, $k = 1, \ldots, K$. $p_\Theta(x) = \sum_{k=1}^{K} p_{\Theta k}(x, k)$
- Think of the individual components as mixture of kernels, except for there is only a few of them as opposed to one kernel per data point in kernel-based density estimation methods.
- The log-likelihood function

$$L_\Theta(T) = \sum_{i=1}^{n} \log\, p_\Theta(x_i) = \sum_{i=1}^{n} \log \underbrace{\sum_{k=1}^{K} p_k(x_i)\, p(k)}_{p_{\Theta k}(x_i)} \tag{8}$$

- In such a case, EM algoritmus can be used for ML estimate of the mixture parameters.

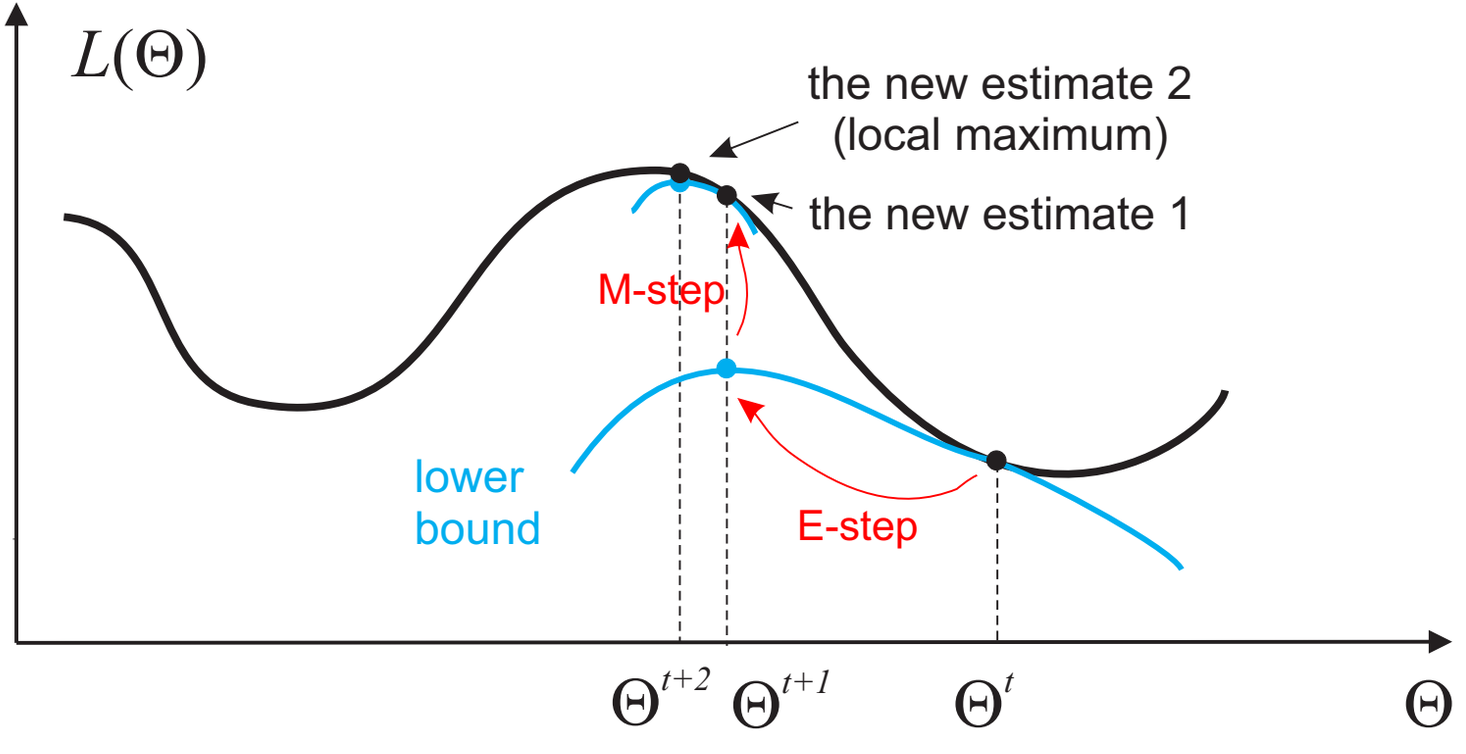
  Example: Parameters estimate for the finite mixture statistical model, often mixture of Gaussians.

EM maximizes the lower bound of L

- EM start from any initial estimate Θ^0.
- The cycle iterates:

  E-step: Estimates the lower bound of the function $L(\Theta)$ in the point Θ^t.

  M-step: finds a new parameter value Θ^{t+1}, which maximizes the estimated lower bound. This value is better suited for optimization.



Even a more special case Gaussian Mixture Model (GMM)

- Let us instantiate the log-likelihood of the general mixture model from Equation (8) at slide 39 by Gaussian distributions and maximize it

$$\Theta^* = \underset{\Theta}{\operatorname{argmax}}\, L_\Theta(T) = \sum_{i=1}^{n} \log\, p_{\Theta k}(x_i) = \sum_{i=1}^{n} \log \sum_{k=1}^{K} p_k(x_i)\, p(k)$$

$$= \sum_{i=1}^{n} \log \sum_{k=1}^{K} \mathcal{N}(x_i|\mu_k, \Sigma_k)\, p(x_k)$$

- We could try to maximize this function by differentiation for a more special case with unitary covariances Σ_k as [Bishop 1995] did. The obtained solution is not a closed form solution and represent a highly nonlinear coupled system of equations.
- We already know that an iterative EM algorithm guarantees to increase the log-likelihood in every iteration.

Example – Gaussian mixture models (GMM)

- Gaussian mixture model is defined by

$$p_\Theta(x,k) = p_\Theta(x|k)\,p_\Theta(k) = \mathcal{N}(x|\mu_k,\Sigma_k)\,p(k)$$

- E-step: calculate

$$\alpha(x_i,k)^t = \frac{\mathcal{N}(x|\mu_k^t,\Sigma_k^t)\,p^t(k)}{\sum_{k=1}^{K}\mathcal{N}(x|\mu_k^t,\Sigma_k^t)\,p^t(k)}$$

- M-step: calculate

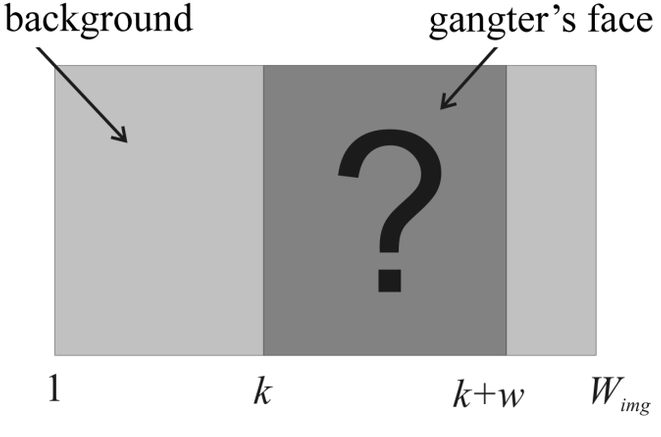
$$\begin{aligned}
p^{t+1}(k) &= \frac{1}{n}\sum_{x_i\in X}\alpha(x_i,k)^t \\
\mu_k^{t+1} &= \frac{\sum_{x_i\in X}\alpha(x_i,k)^t x_i}{\sum_{x_i\in X}\alpha(x_i,k)^t} \\
\Sigma_k^{t+1} &= \frac{\sum_{x_i\in X}\alpha(x_i,k)^t (x_i-\mu_y^t)(x_i-\mu_k^t)^\top}{\sum_{x_i\in X}\alpha(x_i,k)^t}
\end{aligned}$$

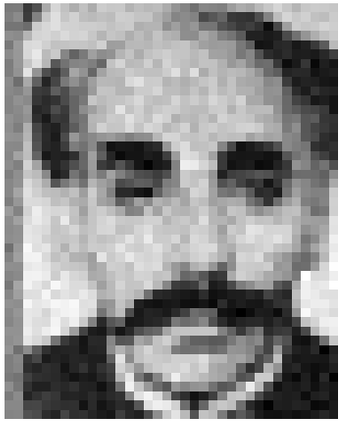
Example – human face image reconstruction (1)

- Each pixel value $x_i(r, c)$ in the image X_i at position (r, c) is observed with the Gaussian noise $\mathcal{N}(0, \sigma)$.
- The probability of observing value $x_i(r, c)$ assuming the leftmost column of a human face f at the position s_i is

$$p(x_i(r,c)|s_i, f, b, \sigma) = \begin{cases} \mathcal{N}(f(r, c - s_i + 1), \sigma) & \text{for } c \in \langle s_i, s_i + w) \\ \mathcal{N}(b, \sigma) & \text{elsewhere} \end{cases},$$

where b is the intensity of the uniform background of width W; $f(r, c')$ are pixels belonging to a face.





Example – human face image reconstruction (2)

- Unobserved data are here the face positions s_i. Parameters of the probability model are face pixels $f(r, c')$, the background intensity b and the noise variation σ.
- Probability of observing set of m images $\mathcal{X} = \{X_1, \ldots, X_m\}$ is

$$
\begin{aligned}
p(\mathcal{X}|f, b, \sigma) &= \prod_{i=1}^{m} \sum_{s} p(X_i, s|f, b, \sigma) = \prod_{i=1}^{m} \sum_{s} p(s)\, p(X_i|s, f, b, \sigma) \\
&= \prod_{i=1}^{m} \sum_{s} p(s) \prod_{r} \prod_{c} p(x_i(r, c)|s)
\end{aligned}
$$

EM relation to K-means clustering

- Consider a classification problem with measurements $x \in X$ and hidden states (classes) $y \in Y$.
- The relation between each measurement x and its class label y can be described using the probability $p_\Theta(x, y)$.
- Having an unlabeled training set of measurements $X = \{x_1, \ldots, x_n\}$, one can use EM algorithm to estimate the probability model and even to classify the observed data without any information from the teacher.
- To classify the data, one can use the output of E-step

$$\alpha(x_i, y)^t = \frac{p_{\Theta^t}(x_i, y)}{\sum_{y \in \mathcal{Y}} p_{\Theta^t}(x_i, y)} = p_{\Theta^t}(y|x_i) \ ,$$

which can be interpreted as the probability of x_i belonging to the class y.

Taking into account known priors, MAP solution

- EM can be used to find MAP (Maximally Aposteriory Probable) solutions for models with defined priors $p(\Theta)$

$$p(\Theta|X) = \frac{p(X|\Theta)\,p(\Theta)}{p(X)} \simeq p(X|\Theta)\,p(\Theta) = p(X,\Theta)$$

- The optimized lower bound is

$$F(\Theta,\alpha) = \sum_{x_i \in X} \sum_{y \in \mathcal{Y}} \alpha(x_i,y) \log \frac{p(x_i,y,\Theta)}{\alpha(x_i,y)}$$

- E-step: calculate

$$\alpha(x_i,y)^t = \frac{p(x_i,y,\Theta^t)}{\sum_{y \in \mathcal{Y}} p(x_i,y,\Theta^t)} = p(y|x_i,\Theta^t)$$

- M-step: calculate

$$\Theta^{t+1} = \operatorname*{argmax}_{\Theta \in \Theta} E_{p(y|x_i,\Theta^t)}(\log p(x_i,y,\Theta)$$