Machine/Statistical Learning

Unsupervised learning
Clustering: K-means, Mixture models and hierarchical approaches

Florent Chatelain, Olivier.JJ.Michel

Grenoble-INP, GISAP-Lab

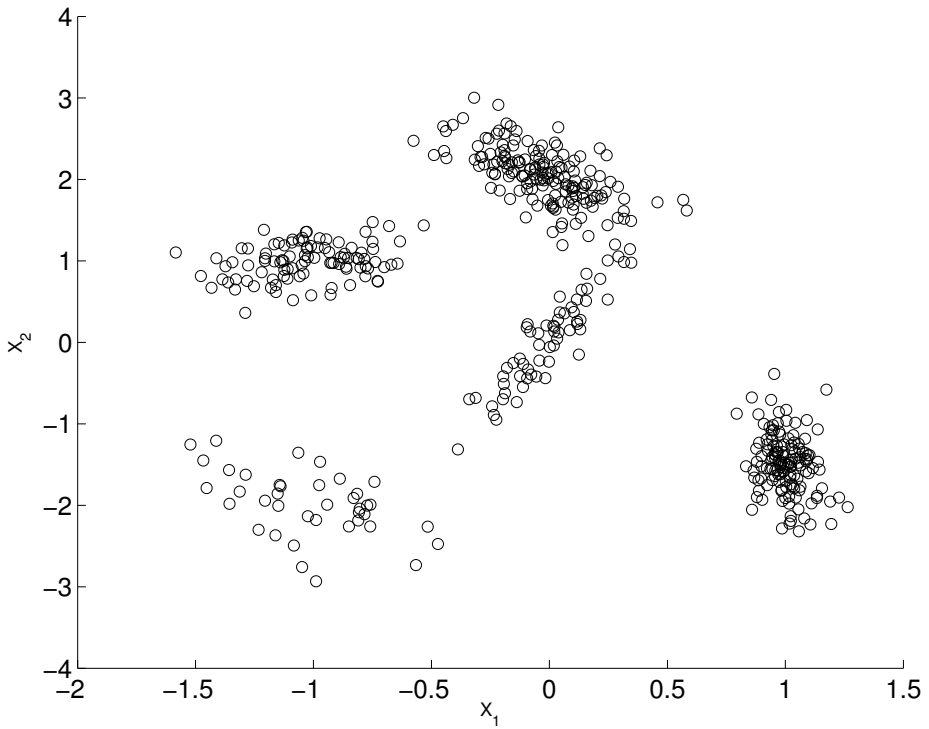
ENSTA, feb. 2020

Unsupervised classification

Assumptions

- $X \in \mathbb{R}^p, \quad Y \in \{1, \ldots, K\} \leftarrow K$ classes
- Training set $(x_1, \ldots, x_n)$ $\leftarrow$ unknown outputs y_i

Example ($p = 2$)

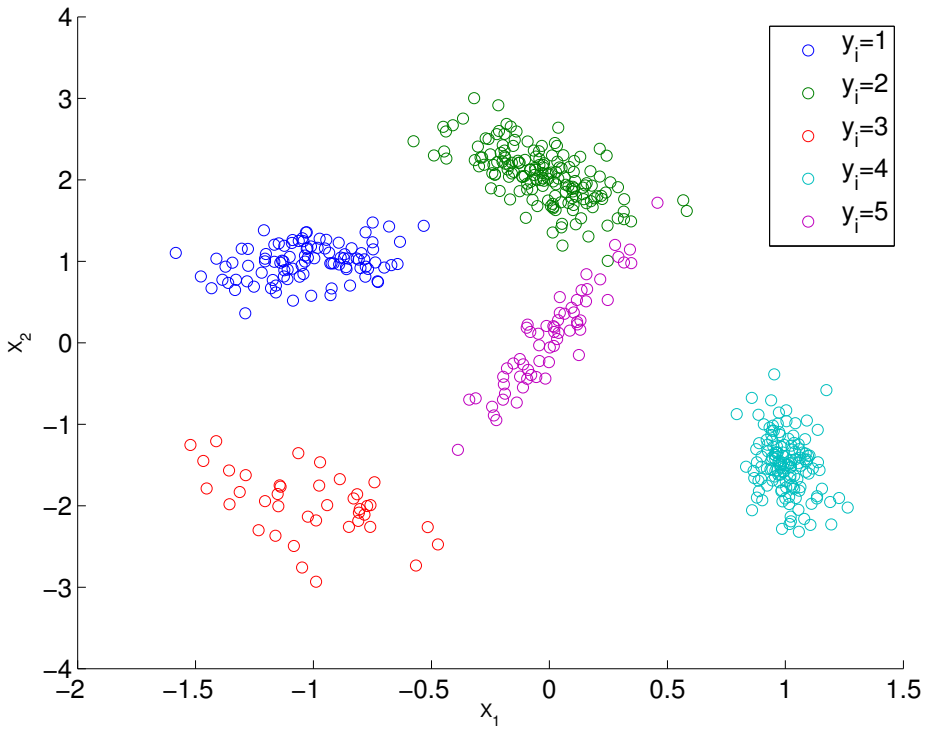


Unsupervised classification : Clustering

Objectives

- grouping *similar* data in the same cluster ← clustering
- ☞ For each x_i, $1 \leq i \leq n$, predict the class variable $Y_i \in \{1, \ldots, K\}$

Example ($p = 2$)



True classes y_i for data x_i ($K = 5$)

Clustering limitations

Combinatorics problem

- Number of partitions into K classes for a sized n dataset : *Stirling number* of the 2nd kind $S(n, K)$
- Number of partitions for a sized n dataset : *Bell number* $B_n = \sum_{k=1}^{n} S(n, k)$

dataset size n	2	5	10	100	200
$S(n, 2)$ ($K = 2$ classes)	1	15	511	6.3×10^{29}	8.0×10^{59}
$S(n, 4)$ ($K = 4$ classes)	0	10	34105	6.7×10^{58}	1.1×10^{119}
B_n	2	52	115975	4.8×10^{115}	6.2×10^{275}

- Remember $\simeq 10^{80}$ atoms in the Universe...

Pb : Exhaustive search (brute-force) not possible in practice

☞ local search around initial solutions/values → sub-optimal

Estimation problem and model selection

- possible parameters are unknown ← estimation
- Number of classes K possibly unknown ← model selection

Mixture of distributions

- Data $X_1, \ldots, X_n$ assumed to be i.i.d. with pdf f
- f is modeled as a *mixture of distributions*

$$f(x) = \sum_{k=1}^{K} \pi_k \phi(x; \theta_k)$$

- $\pi_1, \ldots, \pi_k$ are the relative sizes ($\sum_{k=1}^{K} \pi_k = 1$) of the classes :

$$\Pr(Y_i = k) = \pi_k$$

- density ϕ is the parametric shape of a class,
- parameters $\theta_1, \ldots, \theta_K$ are the *centroids* of the classes/clusters

Latent variable

$Y \in \{1, \ldots, K\}$ indicating the class of the r.v. X

- $Y \sim$ discrete distribution s.t. $\Pr(Y_i = k) = \pi_k, \quad k = 1, \ldots, K$
- $X|Y = k \sim$ distribution with pdf $\phi(\cdot|\theta_k)$

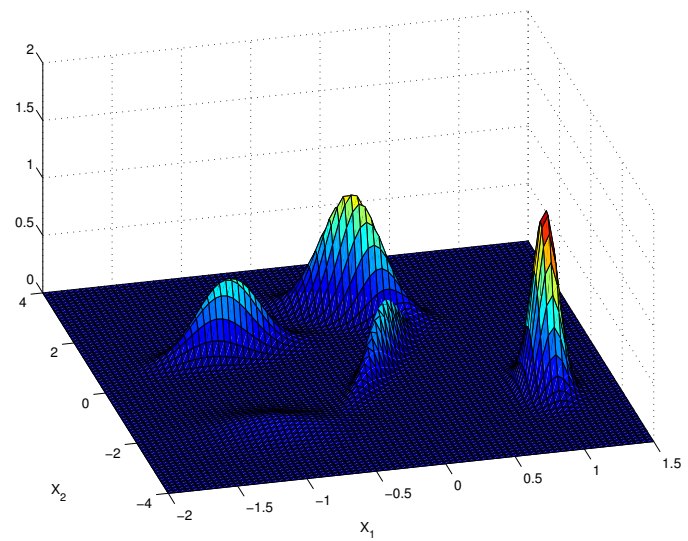
Gaussian mixture model

- Class centroid : $\theta = (\mu \leftarrow \text{mean}, \Sigma \leftarrow \text{covariance matrix})$
- Density ϕ of a class : multivariate normal distribution $\mathcal{N}(\mu, \Sigma)$ pdf

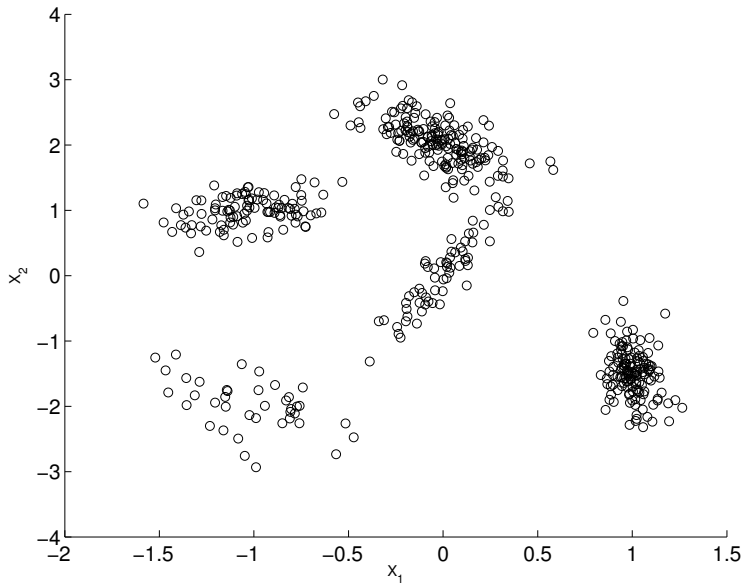
$$\phi(x; \mu, \Sigma) = \left(\det\left(2\pi\Sigma\right)\right)^{-1/2} \exp\left(-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)\right)$$

- Mixture density $\quad f(x) = \sum_{k=1}^{K} \pi_k \phi\left(x; \mu_k, \Sigma_k\right)$

Example ($p = 2$, $K = 5$)



Mixture density f



$n = 500$ realizations

Cost based approximation : K-means

Pb : no simple expression of the Gaussian mixture parameter estimators

☞ several approximations can be conducted to obtain a simple *deterministic* cost criterion

First approximation : euclidean distance

Replace the Mahalanobis distance in the Gaussian density by the simpler euclidean one

$$(x-\mu_k)^T\Sigma_k^{-1}(x-\mu_k) \to \|x-\mu_k\|^2, \quad \text{(i.e. } \Sigma_k = I_p\text{)}\,,$$

☞ cluster centroid for the kth class reduces to $\theta_k = \mu_k \leftarrow$ mean vector

Cost based approximation : K-means (Cont'd)

Pb : no straightforward expression of the Gaussian mixture parameter estimators

☞ several approximations can be conducted to obtain a simple *deterministic* cost criterion

Second approximation : hard thresholding

Binarize the posterior probabilities : for each data point x_i,

$$t_{i,k} \equiv \Pr(Y_i = k | x_i, \theta) = \begin{cases} 1 & \text{if } k = \arg\min_{1 \leq l \leq K} \|x_i - \mu_l\|, \\ 0 & \text{otherwise.} \end{cases}$$

Csq : x_i belongs with certainty to the class whose centroid is the closest

☞ hard thresholding clustering

☞ deterministic model

Cost criterion : K-means clustering

Notations

For a given clustering Y, let

- $n_k = \#\{i \mid Y_i = k\}$ is the size of the kth cluster,
- $\widehat{\mu}_k = \frac{1}{n_k} \sum_{i|Y_i=k} x_i$ is the sample mean of the points assigned in the kth cluster

Under the previous approximations, maximizing the resulting "log-likelihood" reduces to the following optimization problem :

K-means cost criterion

$$\begin{aligned} \text{Minimize} \quad J(Y) &= \sum_{k=1}^{K} \sum_{i=1}^{n} t_{i,k} \|x_i - \widehat{\mu}_k\|^2, \\ &= \sum_{k=1}^{K} \sum_{i|Y_i=k} \|x_i - \widehat{\mu}_k\|^2, \end{aligned}$$

☞ $J(Y)$ is the sum of within-cluster dispersions

Equivalent cost criterion

(negative) Sum of between-cluster dispersions

$$J(Y) = -\sum_{k=1}^{K} n_k \|\widehat{\mu}_k - m\|^2 + \text{constant},$$

where $m = \frac{1}{n}\sum_{i=1}^{n} x_i$ is the total mean.

- ☞ Minimizing the within-cluster dispersion $\Leftrightarrow$ Maximizing the between-cluster dispersion
- ☞ General property of clustering algorithms

Proof : let $S_T = \sum_{j=1}^{n}(x_j - m)^T(x_j - m) = \sum_{k=1}^{K}\sum_{i|Y_i=k}\|x_i - m\|^2$ be the total dispersion.

- Replace x_i by $x_i - \widehat{\mu}_k + \widehat{\mu}_k$, and expand S_T
- Show that $S_T = J(Y) + \sum_{k=1}^{K} n_k\|\widehat{\mu}_k - m\|^2$ (i.e. the cross product equals zero), and conclude by noting that S_T does not depend on Y

K-means : cost criterion optimization

Enlarged optimization problem

$$\min_{Y,\mu} J(Y,\mu) = \sum_{k=1}^{K} \sum_{i|Y_i=k} \underbrace{\|x_i - \mu_k\|^2}_{J_k},$$

- J_K is the quadratic error for the kth cluster

Remarks

- For a given Y, $\min_\mu J(Y,\mu) = J(Y,\widehat{\mu}) \equiv J(Y)$
- For a given μ, exchanging $Y_i = k$ with $Y_i^\star = l$ changes the two quadratic errors

$$\left[\begin{array}{ll} J_k^\star & = J_k - \|x_i - \mu_k\|^2, \\ J_l^\star & = J_l + \|x_i - \mu_l\|^2,, \end{array}\right.$$

Thus $J(Y,\mu)$ is decreased if

$$\begin{array}{lll} & J_l^\star - J_l & \leq J_k - J_k^\star \\ \Leftrightarrow & \|x_i - \mu_l\|^2 & \leq \|x_i - \mu_k\|^2, \\ \Leftrightarrow & x_i \text{ is closer} & \text{(euclidean distance) from the class } l \text{ center,} \end{array}$$

K-means algorithm (LLoyd's algorithm)

- **Require :** K the number of clusters,
- **Initialization :** Set the centroid μ_k, $1 \leq k \leq K$, to a starting value $\mu_k^{(0)}$,
- **For** $t = 1 \to \ldots$ **until convergence** $\left(\text{i.e. } \mu_k^{(t)} = \mu_k^{(t-1)}\right)$
  1. **Assignment step :** assign x_i to the class of the closest center
  $$Y_i^{(t)} = \arg \min_{k=1,\ldots,K} \left\| x_i - \mu_k^{(t-1)} \right\|^2, \quad \text{for } i = 1, \ldots, n$$
  2. **Update step :** update the centroids μ_k, for $k = 1, \ldots, K$
  $$\mu_k^{(t)} = \arg\min_{\mu_k} \sum_{i|Y_i^{(t)}=k} \|x_i - \mu_k\|^2 = \frac{1}{n_k^{(t)}} \sum_{i|Y_i^{(t)}=k} x_i,$$
  i.e. $\mu_k^{(t)}$ is the sample mean of the kth cluster

Convergence of K-means algorithm

Convergence

- each step decreases the criterion,
- there is a (huge) finite number of partitions,

☞ the algorithm converges to a solution (in a finite number of steps)

But no guaranty of the solution optimality (depend on the initialization)...

Stopping criterion

K-means usually very fast for a small/moderate number of clusters K, but

- running time increases with the number of clusters K
- in the worst case, can be very slow to converge even for $K = 2$,

Thus, to shorten the computational time, the algorithm can be stopped when the cost criterion does not decrease significantly

Variants/Improvements of K-means algorithm

Initialization heuristics

- Forgy method
  - pick randomly K observations from the dataset as initial centers,
  - run K-means algorithm with these starting values
  - repeat these 2 steps several times and retain the best (cost sense) clustering
- lot of variants : `Random partitions`, `k-means++`, `power init.`

☞ may lower the computation time of one run,

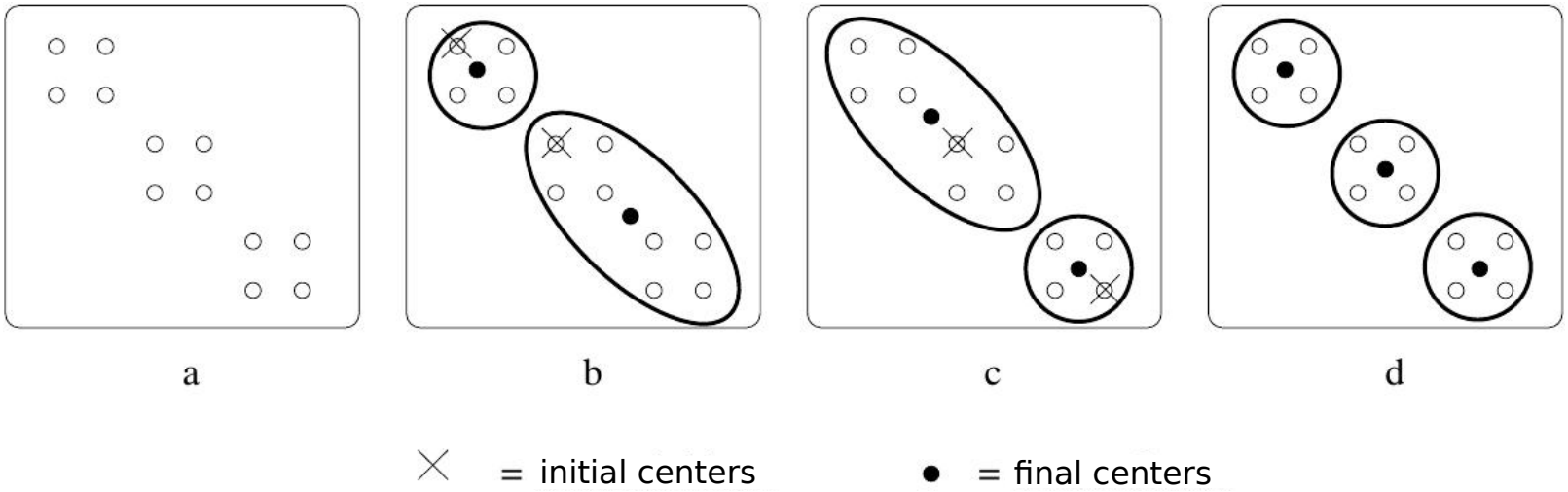
☞ can give some guaranties that the solution is competitive w.r.t. to the optimal one.

Choice of the distance -see also later-

- Standard K-means based on the squared ℓ_2 (euclidean) distance.
- Other distance can be considered : e.g. using ℓ_1 distance yields the K-medians algorithm where the cluster centroid becomes the median

K-means initilization

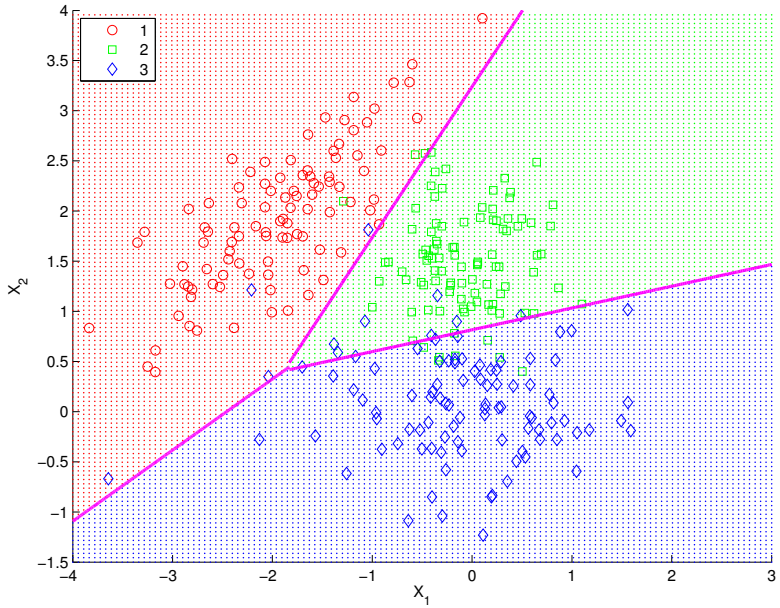
Sensitivity to initialization/data geometry/number of classes



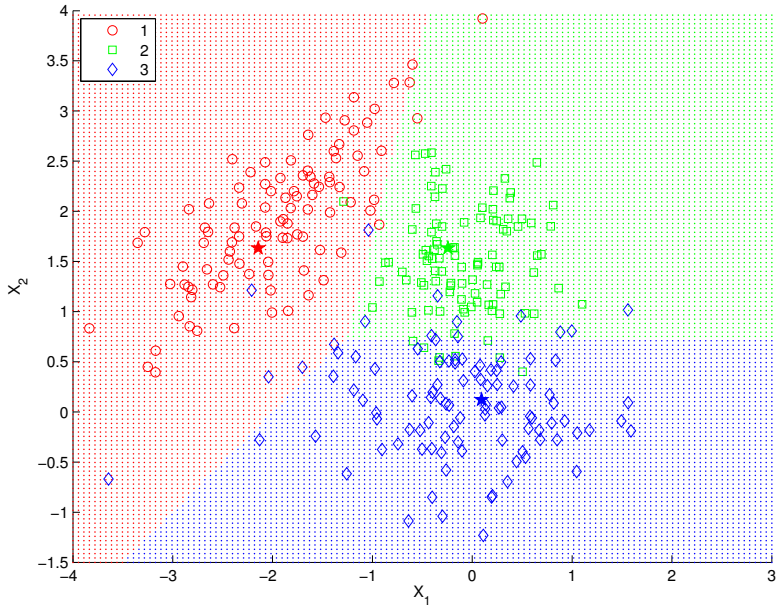
a) set of points $x_i \in \mathbb{R}^p$ $(p = 2)$ to classify, b) and c) two clusterings in $K = 2$ classes with different initial centers, d) clustering in $K = 3$ classes.

K-means

Prediction vs Clustering



LDA (supervised approach)



K-means with $K = 3$ classes

- the points $x_1, \ldots, x_n$ are grouped according to the color of the regions
- Prediction : performance on *new* data is what matters
- Clustering : performance on *current* data is what matters

EM (Expectation-Maximization) algorithm

EM method is a general and important tool of statistical analysis :

- method for finding maximum likelihood (ML) or maximum a posteriori (MAP) estimates of parameters in statistical models, by maximizing iteratively the log-likelihood
- introduction of unobserved latent variables Z to decompose the optimization problem in simpler sub-problems in an iterative way
- EM iteration alternates between performing an expectation (E) step, and a maximization (M) step

EM (Expectation-Maximization) principle

- Z is a latent variable,
- Objective : maximize $\ell(\theta) = \log p(x|\theta)$

Sketch of EM algorithm

- **E step** : compute the expectation of the completed log-likelihood function evaluated using the current estimate for the parameter

$$Q\left(\theta, \theta^{(t-1)}\right) = E_{Z|X,\theta^{(t-1)}}\left[\log p(x, z|\theta)\right],$$
$$= \int p(z|x, \theta^{(t-1)}) \log p(x, z|\theta) dz$$

- **M step** : compute parameters maximizing the expected log-likelihood

$$\theta^{(t)} = \arg\max_{\theta} Q\left(\theta, \theta^{(t-1)}\right),$$

- Repeat until convergence of the $\theta^{(t)}$ sequence

Application of EM to mixture models : E step

Introducing the latent variables Y_i, or equivalently, the binary variables

$$z_{ik} = \begin{cases} 1 & \text{if } Y_i = k, \\ 0 & \text{otherwise,} \end{cases}$$

the likelihood completed with the r.v. z_{ik} reads

$$p(x_1, \ldots, x_n, z|\theta) = \prod_{i=1}^{n} p(x_i, z|\theta) = \prod_{i=1}^{n} \prod_{k=1}^{K} \pi_k \phi(x_i|\theta_k)^{z_{ik}},$$

$$\Rightarrow \quad \log p(x_1, \ldots, x_n, z|\theta) = \sum_{i=1}^{n} \sum_{k=1}^{K} z_{ik} \log \left[\pi_k \phi(x_i|\theta_k)\right],$$

$$\Rightarrow \quad Q\left(\theta, \theta^{(t-1)}\right) = \sum_{i=1}^{n} \sum_{k=1}^{K} \underbrace{E\left[z_{ik}|x_i, \theta^{(t-1)}\right]}_{t_{ik}^{(t-1)}} \log\left(\pi_k \phi(x_i|\theta_k)\right)$$

where $t_{ik}^{(t-1)} = \Pr\left(Y_i = k \,\middle|\, x_i, \theta^{(t-1)}\right) = \dfrac{\pi_k^{(t-1)} \phi\left(x_i|\theta^{(t-1)}\right)}{\sum_{k=1}^{K} \pi_k^{(t-1)} \phi\left(x_i|\theta^{(t-1)}\right)}$

Gaussian mixture models : M step

Find $\theta \equiv \theta^{(t)}$ maximizing $Q\left(\theta, \theta^{(t-1)}\right) = \sum_{i=1}^{n} \sum_{k=1}^{K} t_{ik}^{(t-1)} \log\left[\pi_k \phi(x_i|\theta_k)\right]$

- For any mixture model (i.e. $\forall \phi$) :

$$\pi_k^{(t)} = \frac{1}{n} \sum_{i=1}^{n} t_{ik}^{(t-1)}$$

- For a Gaussian mixture model $\theta = \{\mu_k, \Sigma_k\}$ and

$$\mu_k^{(t)} = \frac{\sum_{i=1}^{n} t_{ik}^{(t-1)} x_i}{\sum_{i=1}^{n} t_{ik}^{(t-1)}},$$

$$\Sigma_k^{(t)} = \frac{\sum_{i=1}^{n} t_{ik}^{(t-1)} \left(x_i - \mu_k^{(t)}\right)\left(x_i - \mu_k^{(t)}\right)^T}{\sum_{i=1}^{n} t_{ik}^{(t-1)}},$$

- empirical averages weighted by the posterior probability in $\theta^{(t-1)}$, $t_{ik}^{(t-1)} \equiv \Pr\left(Y_i = k \,|\, x_i, \theta^{(t-1)}\right)$

☞ soft-thresholding algorithm

EM algorithm for Gaussian mixture models

EM clustering

- Initialize $\pi_k^{(0)}$, $\mu_k^{(0)}$, $\Sigma_k^{(0)}$, for $k = 1, \ldots, K$
- For $t = 1, \ldots$ until convergence

  (E) for $i = 1, \ldots, n$, $k = 1, \ldots, K$, compute $t_{ik}^{(t-1)} \equiv \Pr\left(Y_i = k \,|\, x_i, \theta^{(t-1)}\right)$

  (M) for $k = 1, \ldots, K$, compute $\pi_k^{(t)}$, $\mu_k^{(t)}$, $\Sigma_k^{(t)}$

Prediction/Correction structure

- E step $\Leftrightarrow$ prediction step
- M step $\Leftrightarrow$ update/correction step

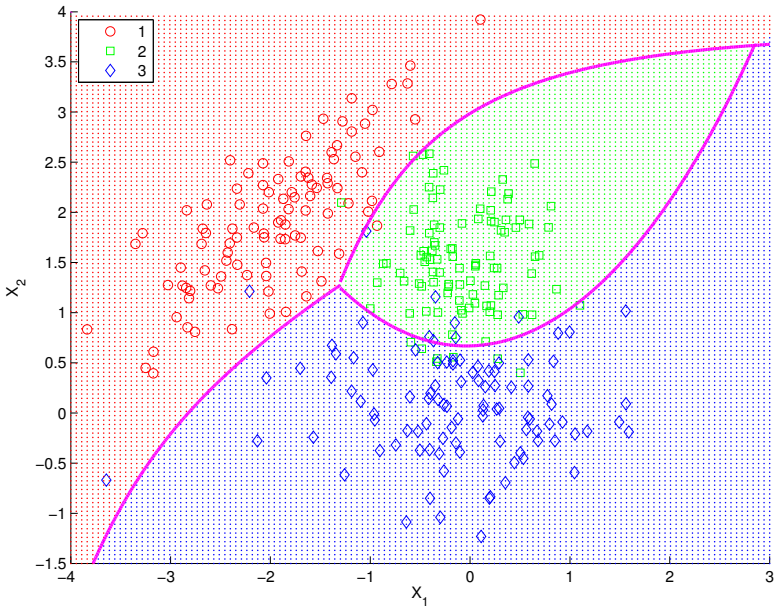
Convergence

- EM : convergence toward a local maximum of the log-likelihood

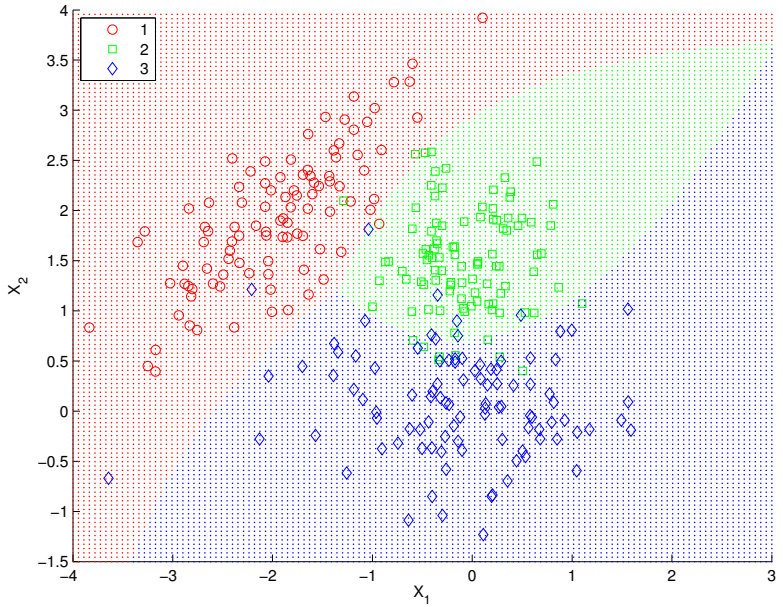
☞ no guaranty of convergence toward the optimal solution (depend on the initial values)..

Gaussian mixture model and EM algorithm

Prediction vs Clustering



QDA (supervised approach)

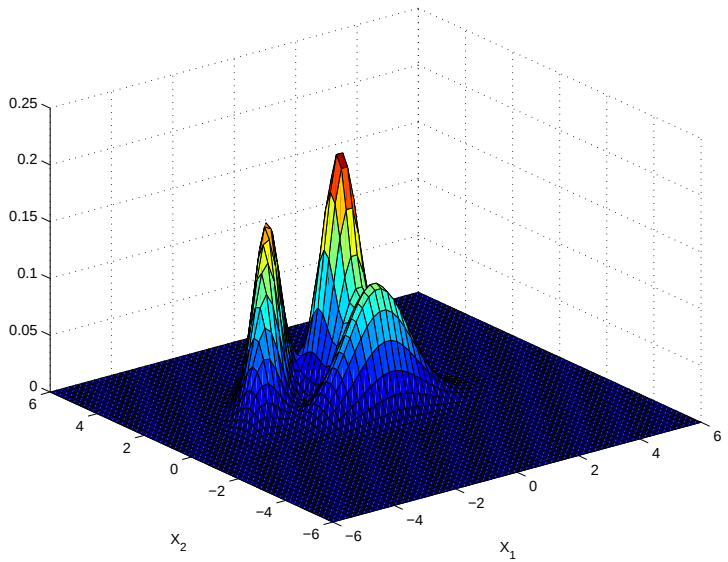


EM with $K = 3$ classes

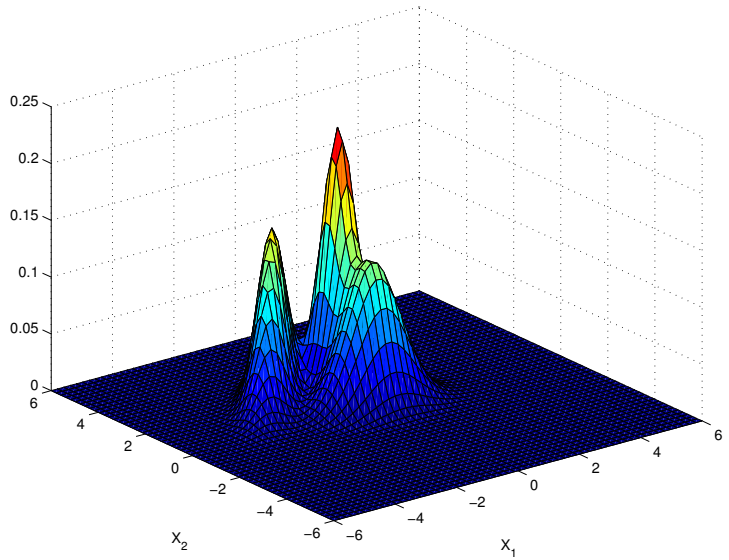
- the points $x_1, \ldots, x_n$ are grouped according to the color of the regions
- Prediction : performance on *new* data is what matters
- Clustering : performance on *current* data is what matters

Gaussian mixture model and EM algorithm

Estimation of the mixture density f



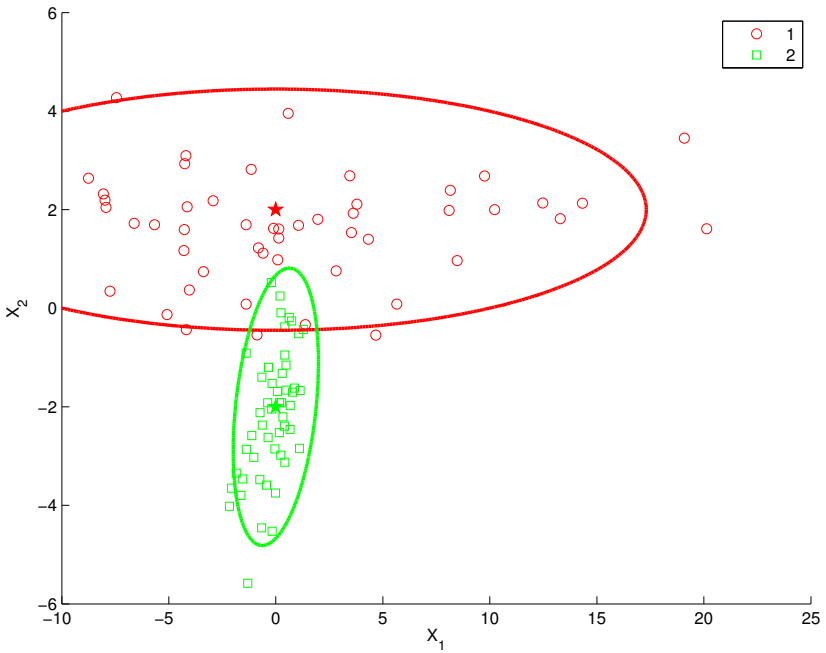
True density of the data points $x_1, \ldots, x_n$



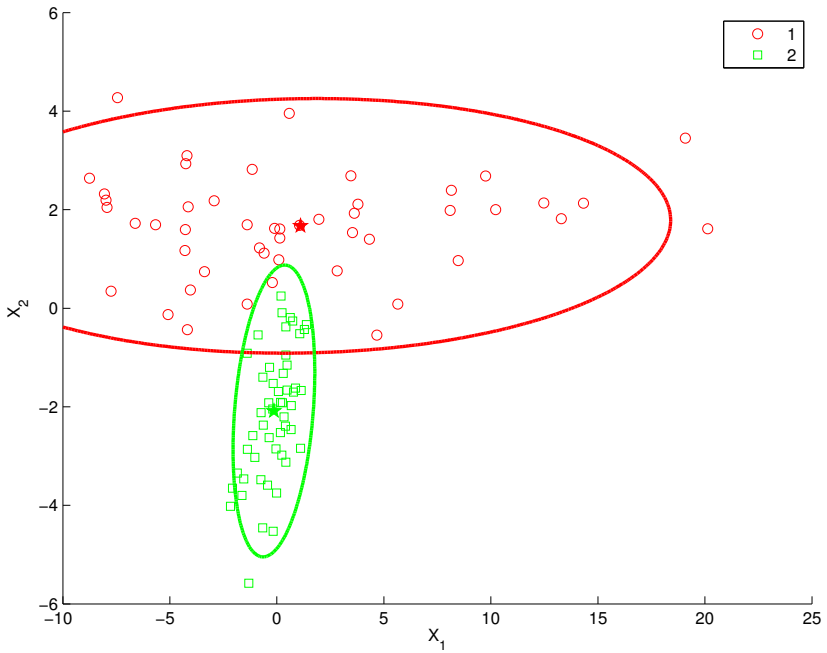
Estimated density with EM ($K = 3$ classes)

Comparison K-means vs Algo EM

2 classes with overlapping and very different dispersions (covariances Σ_k)



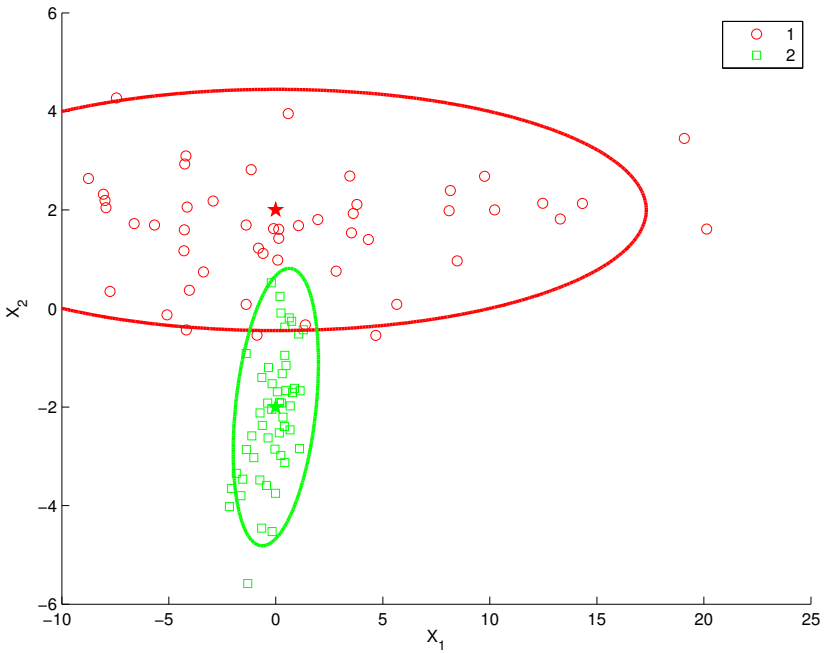
Data $x_1, \ldots, x_n$, classes and true 95% confidence regions



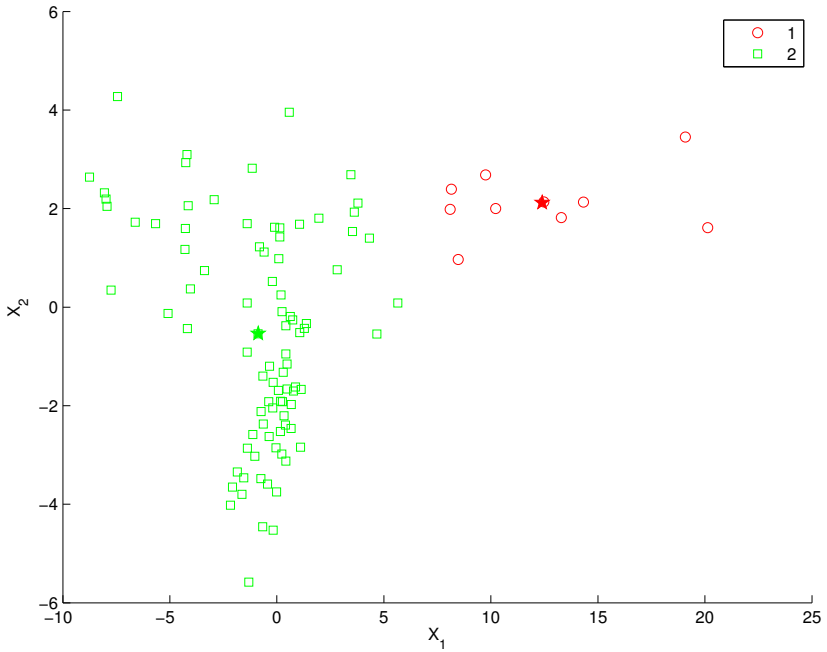
Clustering with EM ($K = 2$) and estimated 95% confidence regions

Comparison K-means vs Algo EM

2 classes with overlapping and very different dispersions (covariances Σ_k)



Data $x_1, \ldots, x_n$, classes and true 95% confidence regions



Classification with K-means ($K = 2$)

Model selection : estimation of K

Minimization of a penalized log-likelihood criterion

$$C(K) = -\hat{l}(x; K) + \text{pen}(K, n)$$

- $\hat{l}(x; K) \equiv l(x; \hat{\theta}_K, K)$ with $\hat{\theta}_K$ the MLE of the model parameters with K classes (profile log-likelihood w.r.t K)

Trade-off between two terms to minimize

- $-\hat{l}(x; K)$: fidelity to the data (likelihood)
- $\text{pen}(K, n)$: low complexity of the model

Model selection : BIC criterion

Bayesian Information Criterion (BIC)

Asymptotic ($n \gg m_K$) criterion for Bayesian models (i.e. with a prior on the model parameters)

$$\text{pen}(K, n) = \frac{1}{2} m_K \log(n)$$

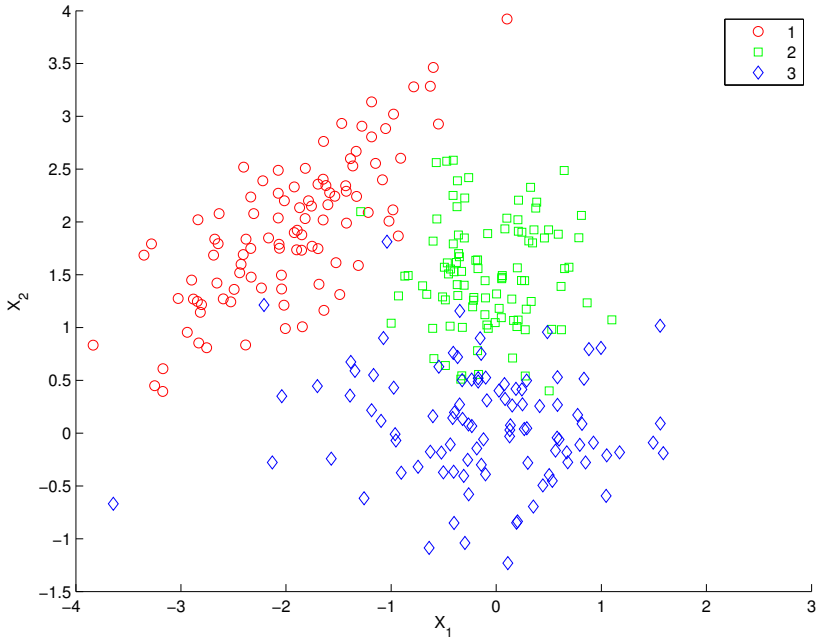
- n is the size of the data
- m_K is the effective number of parameters for the K class model

Equivalent to minimize the following criterion

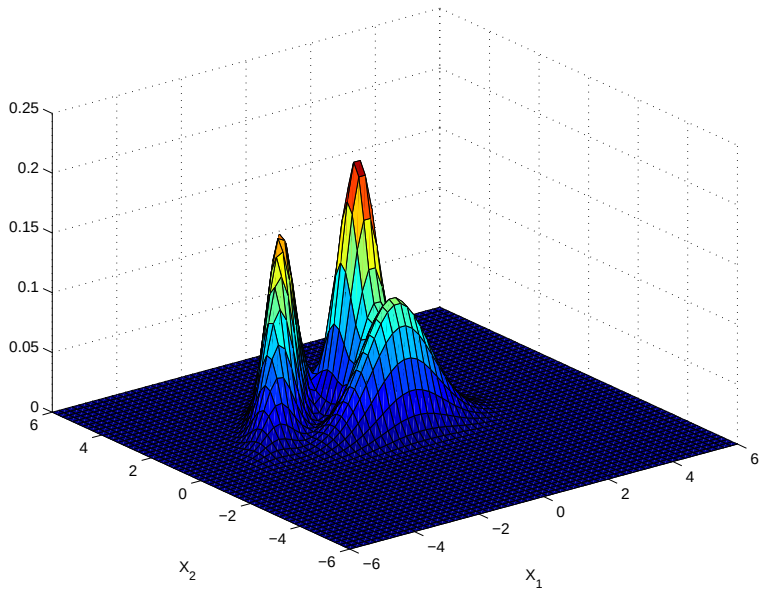
$$\text{BIC}(K) = -2\hat{l}(x; K) + m_K \log(n)$$

Model selection : estimation of K

Example of synthetic data generated according to a mixture of $K = 3$ Gaussians



Dataset $x_1, \ldots, x_n$ ($n = 500$ realizations)

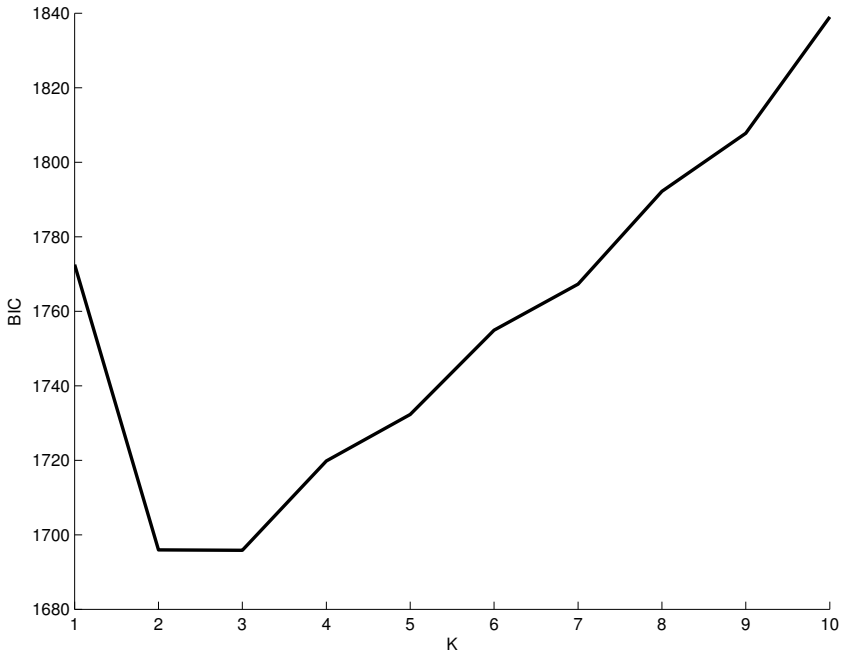


True density f

Model selection : estimation of K

Gaussian mixture : $m_K = \underbrace{K-1}_{\pi_1,\ldots,\pi_{K-1}} + K \times \underbrace{p}_{\mu_k} + K \times \underbrace{\frac{p(p+1)}{2}}_{\Sigma_k}$

BIC criterion w.r.t. K



$\Rightarrow$ $\hat{K} = 2$ or $\hat{K} = 3$ (true value $K = 3$)

Dissimilarity measures

- Dissimilarity measures requires that
  - $d_{ii} = 0$
  - $d_{ij} \leq 0$
  - $d_{ij} = d_{ji}$
- Often $d_{ij} \leq d_{ik} + d_{kj}$ is NOT satisfied $\forall (i, j, k) \in [1, \dots N]^3$

rk1 : d may be sometimes required to be a true distance.

rk2 : From similarity s_{ij} to distance or dissimilarity measure d_{ij} : use any decreasing function e.g.

$$\begin{aligned} d_{ij} &= \max(s_{ij}) - s_{ij} \\ s_{ij} &= \exp(-d_{ij}) \end{aligned}$$

rk3 :

- Dissimilarity measure examples : Euclidean dist, Hamming dist (for categorical variable), Symetrized KL
- Similarity measure example : scalar product, spectral angle, ...

Generalizing KMEANS for alternate dissimilarity measures

Requires to generalize centroids to any dissimilarity measures : introduce Medoids for each cluster (of index k) ; denoting the class label by $y_i = f(x_i)$,
$\forall k \in \{1, \ldots, K\}$

$$\text{Med}_k = \arg \min_{x_j / y_j = k} \sum_{i / y_i = k} d(x_i, x_j)$$

rk1 : The assignment step remains as in the case of centroids.

rk2 : If N is large (more precisely if N_k is large, i.e. the number of points in cluster k), the computation of Med_k may become computationally demanding. Although l_2 norm is most popular, it does not apply for categorical data, where medoids must be introduced.

rk3 : This generalization may be used in order to deal with Kernel trick methods (see Florent's lecture), allowing to deal with non convex clusters.

Kernelized Kmeans –Optionnal–

Let $\kappa : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be a kernel such that $\exists \mathcal{H}$ an Hilbert space and a mapping $\Phi : \mathcal{X} \to \mathcal{H}$, satisfying

$$\forall (x_i, x_j) \in \mathcal{X} \times \mathcal{X}, \kappa(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle_{\mathcal{H}}$$

- Expression the centroid in the KHS

$$c_k \quad = \frac{1}{N_k} \sum_{f(x_i)=k} \Phi(x_i) \Leftrightarrow c_k = \arg\min_z \sum_{x/f(x)=k} ||\Phi(x) - z||^2$$

- Assignment step wrt

$$||\Phi(x) - c_k||^2 = \left\langle \Phi(x) - \frac{1}{N_k} \sum_{f(x_i)=k} \Phi(x_i), \Phi(x) - \frac{1}{N_k} \sum_{f(x_i)=k} \Phi(x_i) \right\rangle$$
$$= \kappa(x, x) - \frac{2}{N_k} \sum_{f(x_i)=k} \kappa(x, x_i) + \frac{1}{N_k^2} \sum_{f(x_i)=k} \sum_{f(x_j)=k} \kappa(x_i, x_j)$$

rk : Assignment does NOT request explicit knowledge of c_k
rk : Usually, as ϕ is not known, the centroids c_k are NOT known

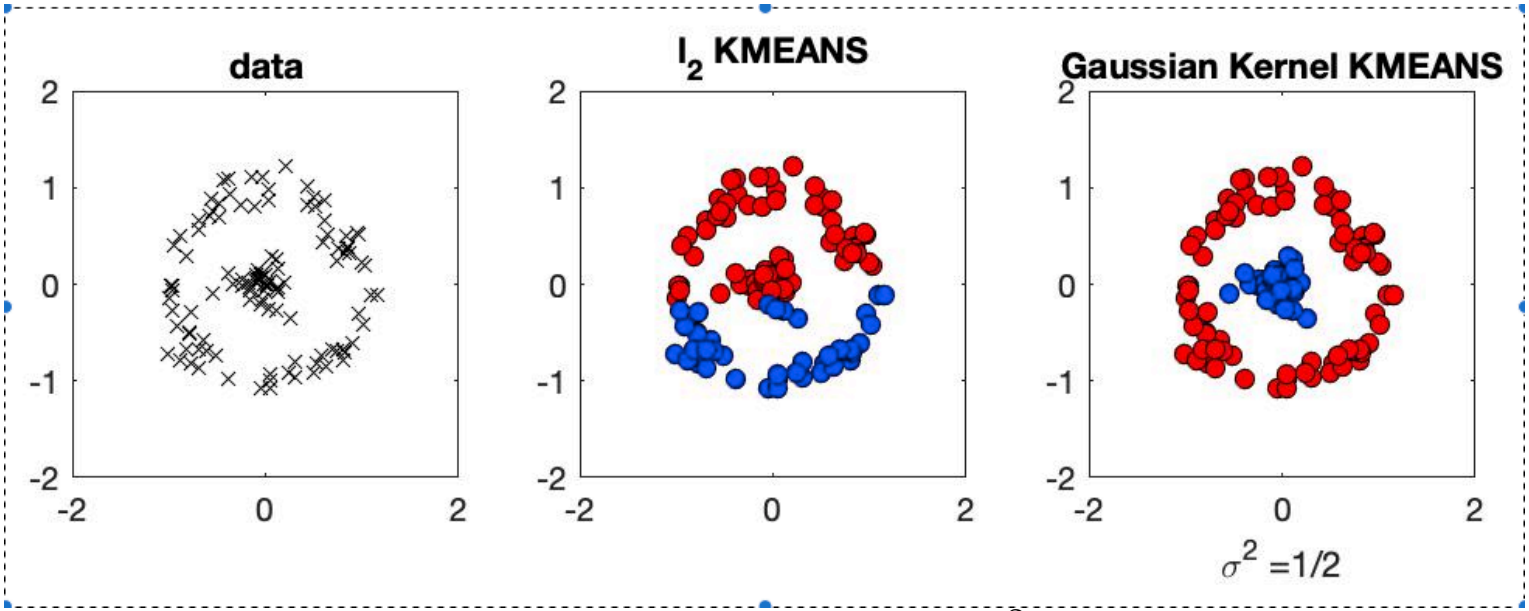
Kernelized Kmeans, cont'd

rk : Kernel Kmeans allows to tackle problems with non convex classes

rk : Kernel Kmeans has increased sensitivity to initial conditions (random initial labelling)

rk : Kernel expression requires some tuning parameter to be set.

Example



$$\kappa(x,y) = \exp(-\frac{||x-y||^2}{2\sigma^2})$$

Evaluating clustering results

Unsupervised framework

$\Longrightarrow$ no ground truth is available, (in general).

• if a probabilistic model is used (such as EM) : likelihood of the test set ?(does not really assess clustering discovered by the model)

• if Deterministic approach (such as Kmeans) : ? $\rightarrow$ how dense (or compact) are the identified clusters, how well separated they are ?

- For l_2 distances, compare within-cluster variance with between-cluster variance (remind that the sum is constant).
- For more general dissimilarity measures, popular quality indices (among others) are
  - Davies Bouldin index
  - Silhouette index

Clustering Quality indices examples

Let C_k denote a cluster, $k \in [1, \ldots, K]$, and $N_k = |C_k|$, Med_k its medoid

Davies Bouldin, DB

- Homogeneity T :

$$T_k = \frac{1}{N_k} \sum_{x \in C_k} d(x, \text{Med}_k) \Rightarrow T = \frac{1}{K} \sum_{k=1}^{K} T_k$$

- Separability S :

$$S_{kl} = d(\text{Med}_k, \text{Med}_l) \Rightarrow S = \frac{2}{K(K-1)} \sum_{k=1}^{K} \sum_{l \neq k}^{K} S_{kl}$$

- DBindex :

$$D_k = \max_{k \neq l} \frac{T_k + T_l}{S_{kl}} \Rightarrow DB = \frac{1}{K} \sum_{k=1}^{K} D_k$$

Clustering Quality indices examples, cont'd

Silhouette index, $\mathcal{S}$

$\mathcal{S}$ is relative to each observation point x_i, whose estimated label is $y_i = k$.

- Average distance to other observations from the same cluster

$$a(x_i) = \frac{1}{N_k - 1} \sum_{j \neq i, j/y_j = k} d(x_i, x_j)$$

- Minimal distance of x_i to the closest cluster

$$b(x_i) = \min_{j \neq i, j/y_j \neq k} \frac{1}{N_{y_j}} \sum_{l/y_l = y_j} d(x_i, x_l)$$

- Silouette

$$\mathcal{S}(x) = \frac{b(x) - a(x)}{max(a(x), b(x))} \Rightarrow \mathcal{S} = \frac{1}{N} \sum_x \mathcal{S}(x)$$

rk1 : if $N_{y_i} = 1$, set $\mathcal{S}(x_i) = 0$
rk2 : $-1 \leq \mathcal{S}(x_i) \leq 1$
rk3 : if $\mathcal{S}(x_i) < 0$, x_i would be better labelled as a member of its neighboring cluster. $\mathcal{S}(x_i) \simeq 0$ if x_i close to the border between clusters.

Clustering quality measure with expert (prior) knowledge

Assume that some labels are known (ground truth $\{y_i, i = 1 \dots N, \}$, $y_i \in \{1, \dots, R\}$ is available) : leads to compare two partitions, i.e. the estimated clustering $(\{f(x_i), i = 1 \dots N)\}$, $f(x_i) \in \{1, \dots, K\}$) with the ground truth partition. Note that labels may take different values for these partitions.

1. RAND index

$$RI = \frac{2}{N(N-1)} \sum_{i=1}^{N} \sum_{j=i+1}^{N} \delta(f(x_i) = f(x_j))\delta(y_i = y_j) + \delta(f(x_i) \neq f(x_j))\delta(y_i \neq y_j)$$

rk1 : This is the proportion of observations pairs that are either from the same known class and have identical estimated labels, or belong to different classes and have different estimated labels.
rk2 : $0 \leq RI \leq 1$

Clustering quality measure with expert knowledge, cont'd

2. Purity index $\mathcal{P}$

Assume a ground truth partition $\{\mathcal{C}_k, k = 1, \ldots, K\}$. Let

$$p_{kl} \overset{def}{=} \frac{N_{kl}}{N_k} = \frac{\#(f(x_i) = k) \text{in } \mathcal{C}_l}{\#(f(x_i) = k)}$$

then

$$P_k \overset{def}{=} \max_l p_{kl}$$

$$\mathcal{P} \overset{def}{=} \sum_{n=1}^{K} \frac{N_k}{N} P_k$$

rk1 : p_{kl} is the proportion of observations whose estimated label is k, that are in class $\mathcal{C}_l$.

rk2 : P_k is this latter proportion, for the class $\mathcal{C}_l$ which contains the more observations with label $f(x_i) = k \rightarrow$ if $\mathcal{C}_l$ matches with cluster k, then $P_k = 1$.

Clustering quality measure with expert knowledge, cont'd

3. (Normalized) Mutual information between two clusterings, $(N)IM$

Let $\mathcal{U} = \{U_1, \ldots, U_R\}$ and $\mathcal{V} = \{V_1, \ldots, V_K\}$

$$p_{UV}(i,j) \overset{def}{=} \mathbb{P}[x \in U_i, x \in V_j] = \frac{|U_i \cap V_j|}{N}$$

$$p_U(i) \overset{def}{=} \frac{|U_i|}{N}$$

then

$$IM(U,V) = \sum_{i=1}^{R} \sum_{j=1}^{K} P_{UV}(i,j) \log \frac{P_{UV}(i,j)}{P_U(i) P_V(j)}$$

or $NIM(U,V) = \frac{2IM(U,V)}{H(U)+H(V)}$, where $H(U) = -\sum_{i=1}^{R} P_U(i) \log P_U(i)$

Hierarchical approaches

Motivations

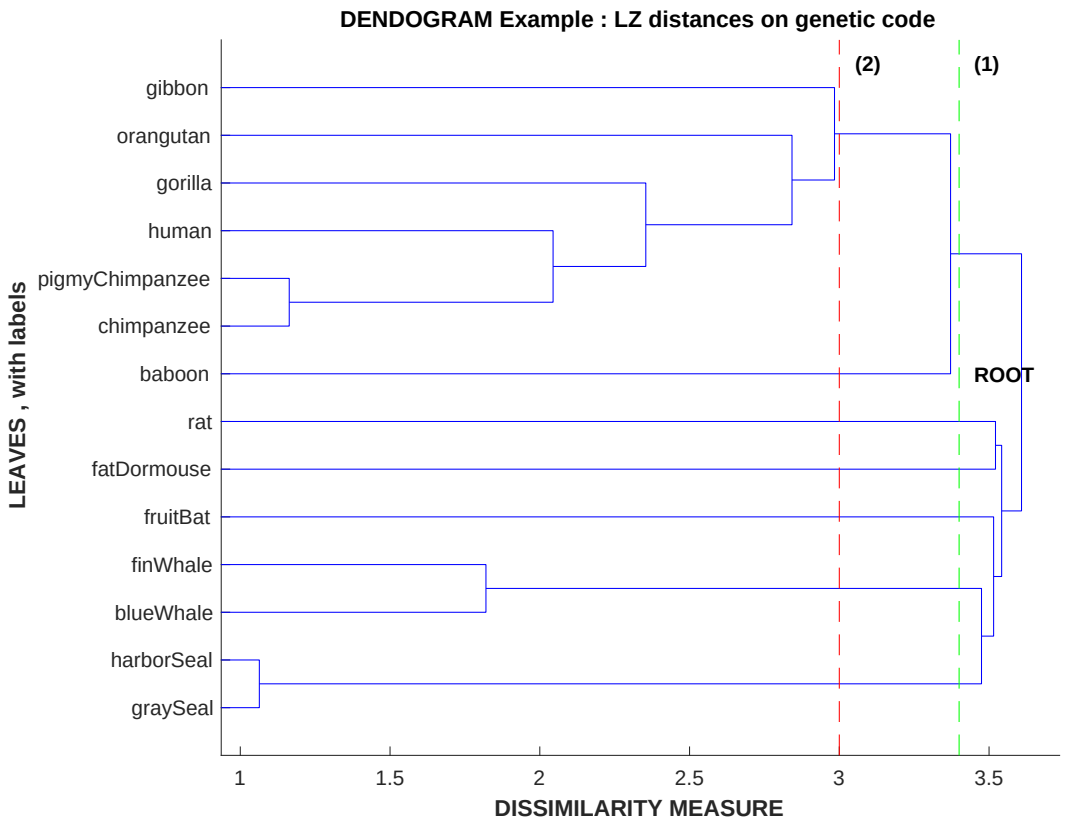
• Recursive approach to partition data at all possible scales, using multi-level hierarchical partitioning

- Each labeling operation does not rely on a single operation, but on a sequence of conditional tests,
- Each operation may use a single or a subset of variables (or characteristics) of the data whereas other methods give the same importance to all variables,
- Allow that different variables are used in different locations in the observation space
- Provide some insights on the relevance of variables for clustering, classification of prediction tasks
- A hierarchical (unsupervised) clustering approach does not require the number of clusters K to be known in advance

Hierarchical clustering : Dendogram

A dendogram is a Tree

- whose root contains all observations
- with N leaves containing a single observation
- where two clusters with the same parents are merged at upper level into a single cluster
- where a cluster s split into two children at lower level.
- $\Rightarrow$ Intermediate nodes contain the relevant information
- the length of a branch is proportional to the dissimilarity between the connected clusters
- Thresholding the dendogram at different levels issues different clustering ((1) or (2))



Dendogram Construction

Divisive, or "top-down"

- Start from root and divide into two cluster wrt a splitting stategy
  - Entropy (*)
  - Variance
  - Davies-Bouldin
  - Silhouette
  - . . . see section on *Clustering Quality indices*

Agglomerative, or "bottom-up"

- At each iteration, find the closest cluster to each others and merge them.
- Iterate until all observations are in a single cluster.

$\Rightarrow$ requires to define closeness measure between clusters

Entropy estimation is difficult in general and uses pdf estimators. Alternate methods use length of quasi additive graphs...

Dendogram Construction, cont'd

Linkage functions : Mostly for agglomerative approaches, measure closeness/distance between clusters

- Requires a distance function $d(.,.)$ on $\{\mathcal{X}\}$ is defined.
  - Single linkage
$$d_{single}(\mathcal{C}_k, \mathcal{C}_l) = \min_{x \in \mathcal{C}_k, x' \in \mathcal{C}_l} d(x, x')$$
  - Complete linkage
$$d_{complete}(\mathcal{C}_k, \mathcal{C}_l) = \max_{x \in \mathcal{C}_k, x' \in \mathcal{C}_l} d(x, x')$$
  - Average linkage
$$d_{average}(\mathcal{C}_k, \mathcal{C}_l) = \frac{1}{|\mathcal{C}_k|} \frac{1}{|\mathcal{C}_l|} \sum_{x \in \mathcal{C}_k} \sum_{x' \in \mathcal{C}_l} d(x, x')$$
  - Centroidal linkage
$$d_{centroidal}(\mathcal{C}_k, \mathcal{C}_l) = d(\frac{1}{|\mathcal{C}_k|} \sum_{x \in \mathcal{C}_k} x, \frac{1}{|\mathcal{C}_l|} \sum_{x' \in \mathcal{C}_l} x')$$

Dendogram Construction, cont'd

Choosing K

- By setting the height of the line or level in the dendogram
- By choosing K to get e.g. the best silhouette coefficient.

Computational cost

As the all set of pairwise distance (must)(*) be computed, computational cost goes like $\mathcal{O}(pN^2)$ if x has p features.
$\Rightarrow$ not well adapted to massive data

(*)