Unsupervised Learning: Clustering

K-means, Mixture models and hierarchical approaches

Nicolas Keriven
CNRS, IRISA, Rennes

ENSTA 2024

Outline

Table of Contents

Unsupervised classification

Assumptions

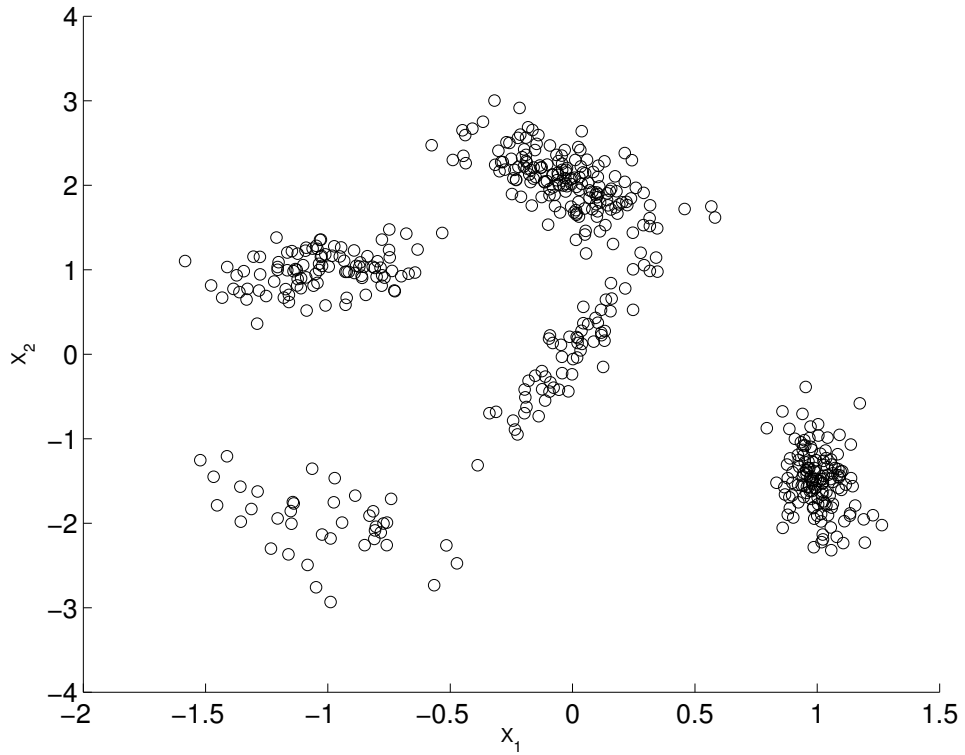
- $x \in \mathbb{R}^p, \quad y \in \{1, \ldots, K\} \leftarrow K$ classes
- Training set $(x_1, \ldots, x_n)$ $\leftarrow$ unknown "outputs" y_i do they even exist ?

Unsupervised classification

Assumptions

- $x \in \mathbb{R}^p$, $y \in \{1, \ldots, K\} \leftarrow K$ classes
- Training set $(x_1, \ldots, x_n)$ ← unknown "outputs" y_i do they even exist ?

Exemple ($p = 2$) :



Unsupervised classification : Clustering

Objectives

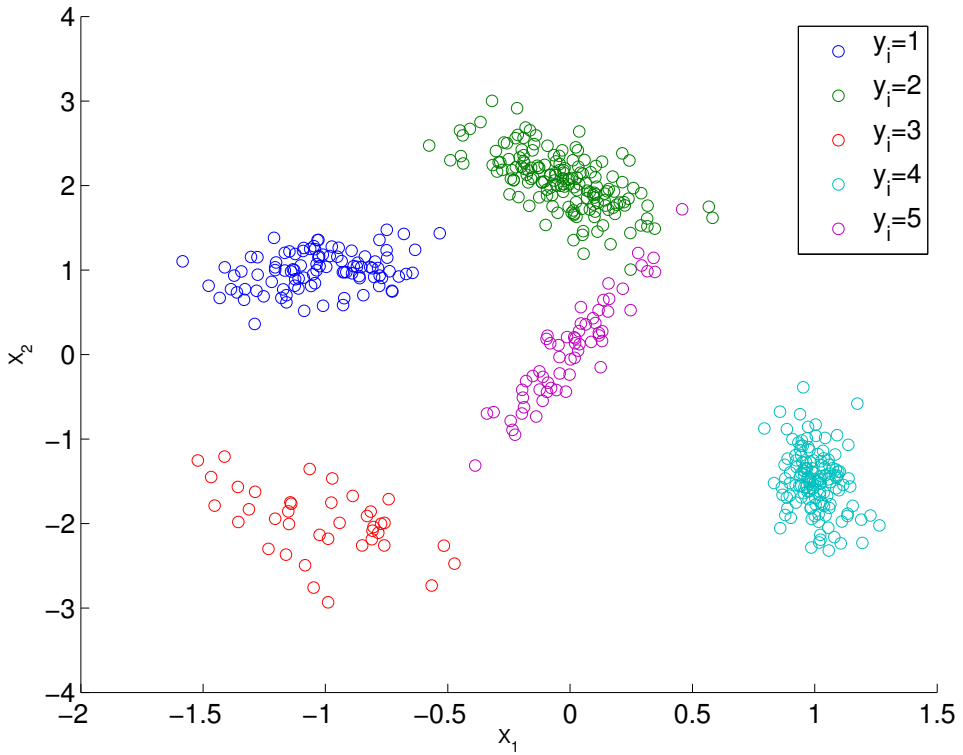
- **Clustering** : grouping *similar* data in the same cluster
- ☞ For each x_i, $1 \leq i \leq n$, predict some class variable $y_i \in \{1, \ldots, K\}$

Unsupervised classification : Clustering

Objectives

- **Clustering** : grouping *similar* data in the same cluster
- ☞ For each x_i, $1 \leq i \leq n$, predict some class variable $y_i \in \{1, \ldots, K\}$

Example ($p = 2$) :



Clustering limitations

Combinatorics problem

- Number of partitions into K classes : *Stirling number of the 2nd kind* $S(n, K)$

Clustering limitations

Combinatorics problem

- Number of partitions into K classes : *Stirling number of the 2nd kind* $S(n, K)$
- Number of partitions (unknown K) : *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}

Clustering limitations

Combinatorics problem

- Number of partitions into K classes : *Stirling number of the 2nd kind* $S(n, K)$
- Number of partitions (unknown K) : *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...
- Exhaustive search (brute-force) not possible in practice

☞ local search around initial solutions/values $\rightarrow$ sub-optimal

Clustering limitations

Combinatorics problem

- Number of partitions into K classes : *Stirling number of the 2nd kind* $S(n, K)$
- Number of partitions (unknown K) : *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...
- Exhaustive search (brute-force) not possible in practice

☞ local search around initial solutions/values $\rightarrow$ sub-optimal

Estimation problem and model selection

- possible "parameters" of the clustering are unknown $\leftarrow$ estimation
- Number of classes K possibly unknown $\leftarrow$ model selection
  - $K = 1$: *underfitting*, $K = n$: *overfitting*

Table of Contents

Mixture of distributions

- Data $x_1, \ldots, x_n$ assumed to be i.i.d. with pdf p_θ
- p_θ is modeled as a mixture of distributions

$$p_\theta(X) = \sum_{k=1}^{K} \pi_k \phi(X, \theta_k)$$

Mixture of distributions

- Data $x_1, \ldots, x_n$ assumed to be i.i.d. with pdf p_θ
- p_θ is modeled as a mixture of distributions

$$p_\theta(X) = \sum_{k=1}^{K} \pi_k \phi(X, \theta_k)$$

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

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

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

Mixture of distributions

- Data $x_1, \ldots, x_n$ assumed to be i.i.d. with pdf p_θ
- p_θ is modeled as a mixture of distributions

$$p_\theta(X) = \sum_{k=1}^{K} \pi_k \phi(X, \theta_k)$$

  - $\pi_1, \ldots, \pi_k$ are the relative sizes of the classes $(\sum_{k=1}^{K} \pi_k = 1)$:
$$\Pr(Y = k) = \pi_k$$
  - density ϕ is the parametric shape of a class,
  - $\theta_1, \ldots, \theta_K$ are the centroids/parameters of the classes

Generative Model with latent variable

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

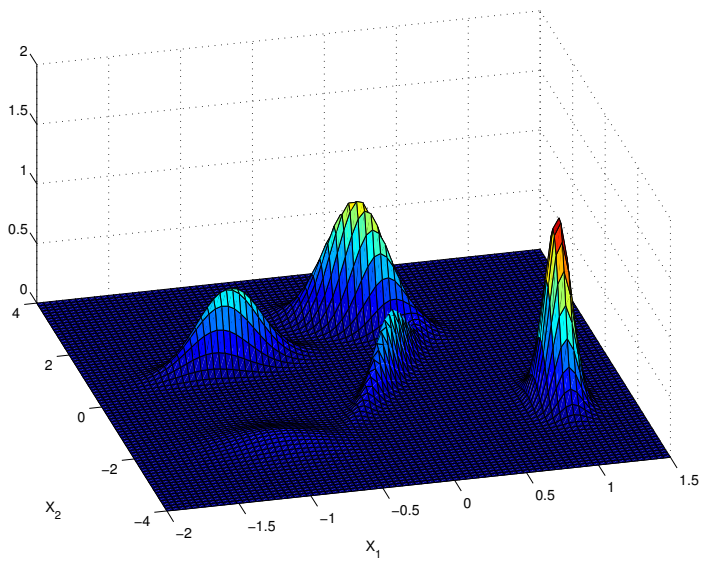
- $Y \sim$ discrete distribution s.t. $\Pr(Y = k) = \pi_k, \quad k = 1, \ldots, K$
- $X|Y = k \sim$ distribution with pdf $\phi(\cdot; \theta_k)$

Gaussian mixture model

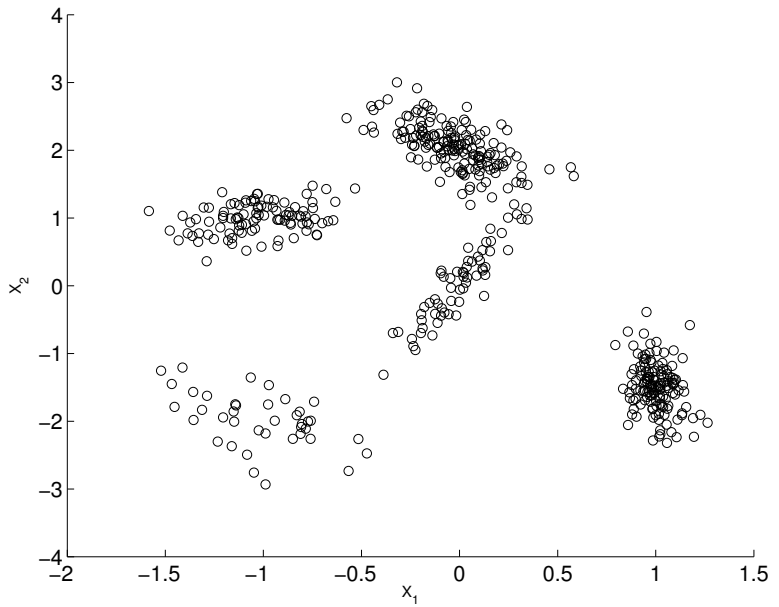
- Class centroid : $\theta = (\mu, \Sigma)$ with μ mean ans Σ covariance
- Density ϕ of a class : multivariate normal distribution $\mathcal{N}(\mu, \Sigma)$ pdf

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

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



Mixture density f



$n = 500$ realizations

Table of Contents

Cost based approximation : K-means

Log-likelihood (to be maximized wrt θ) :

$$\ell(\theta) = \log_\theta p(x_1, \ldots, x_n) = \sum_i \log p_\theta(x_i) = \sum_i \log \left(\sum_k \pi_k \phi(x_i, \mu_k, \Sigma_k) \right)$$

- when $K = 1$, $\hat{\mu}_1 = \bar{x} = \frac{1}{n} \sum_i x_i$, $\hat{\Sigma}_1 = \frac{1}{n-1} \sum_i (x_i - \bar{x})(x_i - \bar{x})^\top$
- no "simple expression" of the parameter estimators when $K > 1$

☞ several approximations can be conducted to obtain a simpler cost criterion

Cost based approximation : K-means

Log-likelihood (to be maximized wrt θ) :

$$\ell(\theta) = \log_\theta p(x_1, \ldots, x_n) = \sum_i \log p_\theta(x_i) = \sum_i \log \left(\sum_k \pi_k \phi(x_i, \mu_k, \Sigma_k) \right)$$

- when $K = 1$, $\quad \hat{\mu}_1 = \bar{x} = \frac{1}{n} \sum_i x_i, \quad \hat{\Sigma}_1 = \frac{1}{n-1} \sum_i (x_i - \bar{x})(x_i - \bar{x})^\top$
- no "simple expression" of the parameter estimators when $K > 1$

☞ several approximations can be conducted to obtain a simpler cost criterion

First approximation : euclidean distance

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

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

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

Cost based approximation : K-means

Log-likelihood (to be maximized wrt θ) :

$$\ell(\theta) = \log_\theta p(x_1, \ldots, x_n) = \sum_i \log p_\theta(x_i) = \sum_i \log \left(\sum_k \pi_k \phi(x_i, \mu_k, \Sigma_k) \right)$$

- when $K = 1$, $\hat{\mu}_1 = \bar{x} = \frac{1}{n} \sum_i x_i$, $\hat{\Sigma}_1 = \frac{1}{n-1} \sum_i (x_i - \bar{x})(x_i - \bar{x})^\top$
- no "simple expression" of the parameter estimators when $K > 1$

☞ several approximations can be conducted to obtain a simpler cost criterion

Second approximation : "hard thresholding"

We keep only the term of the closest centroid :

$$\sum_k \pi_k \phi(x_i; \mu_k) \approx \phi(x_i; \mu_{y_i}) \text{ where } y_i = \arg \min_{1 \leq k \leq K} \|x_i - \mu_k\|$$

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

☞ hard assignment clustering

K-means : cost criterion optimization

Cost criterion : K-means clustering

$$\max_{\theta} \ell(\theta) \Leftrightarrow \min_{\mu} J(\mu) = \sum_{i} \min_{k} (\|x_i - \mu_k\|^2),$$

- This is (still) an NP-hard problem !

K-means : cost criterion optimization

Cost criterion : K-means clustering

$$\max_{\theta} \ell(\theta) \Leftrightarrow \min_{\mu} J(\mu) = \sum_{i} \min_{k}(\|x_i - \mu_k\|^2),$$

- This is (still) an NP-hard problem !

Theorem

The optimal μ_k are the empirical means of the communities $y_i = k$

- If $\mu^\star = \arg\min J(\mu)$ and for all i, $y_i^\star = \arg\min_{1 \leq k \leq K} \|x_i - \mu_k^\star\|$
- Then $\mu_k^\star = \hat{\mu}_k = \frac{1}{\sharp\{y_i^\star = k\}} \sum_{i | y_i^\star = k} x_i$

K-means : cost criterion optimization

Cost criterion : K-means clustering

$$\max_{\theta} \ell(\theta) \Leftrightarrow \min_{\mu} J(\mu) = \sum_i \min_k (\|x_i - \mu_k\|^2),$$

- This is (still) an NP-hard problem !

Theorem

The optimal μ_k are the empirical means of the communities $y_i = k$

- If $\mu^\star = \arg\min J(\mu)$ and for all i, $y_i^\star = \arg\min_{1 \leq k \leq K} \|x_i - \mu_k^\star\|$
- Then $\mu_k^\star = \hat{\mu}_k = \frac{1}{\sharp\{y_i^\star = k\}} \sum_{i | y_i^\star = k} x_i$

Consequence

K-means is, *equivalently*, the search for the best μ or the best clusters y. If

$$J(y) = \sum_i \|x_i - \hat{\mu}_{y_i}\|^2 \qquad \text{where } \hat{\mu}_k = \frac{1}{\sharp\{y_j = k\}} \sum_{j | y_j = k} x_j$$

At the optimum, $J(\mu^\star) = J(y^\star)$, $\mu_k^\star = \frac{1}{\sharp\{y_i^\star = k\}} \sum_{i | y_i^\star = k} x_i$ and $y_i^\star = \arg\min_k \|x_i - \mu_k^\star\|$.

K-means algorithm (LLoyd's algorithm)

Heuristic : *alternate* between

- finding the "best" y for fixed μ
- finding the "best" μ for fixed y

K-means algorithm (LLoyd's algorithm)

Heuristic : *alternate* between

- finding the "best" y for fixed μ
- finding the "best" μ for fixed y

K-means algorithm

- **Require :** K the number of clusters, initial value $\mu_k^{(0)}$ (see after)
- **For** $t = 1, \ldots$ **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
  $$y_i^{(t)} = \arg\min_k \left\| x_i - \mu_k^{(t-1)} \right\|^2, \quad \text{for } i = 1, \ldots, n$$
  2. **Update step :** update the centroids μ_k, for $k = 1, \ldots, K$
  $$\mu_k^{(t)} = \frac{1}{\sharp\{y_i^{(t)} = k\}} \sum_{i | y_i^{(t)} = k} x_i,$$
  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 fixed point (in a finite number of steps)

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

Convergence of K-means algorithm

Convergence

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

☞ the algorithm converges to a fixed point (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

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 guarantees that the solution is close to the optimal one.

Variants/Improvements of K-means algorithm

Initialization heuristics

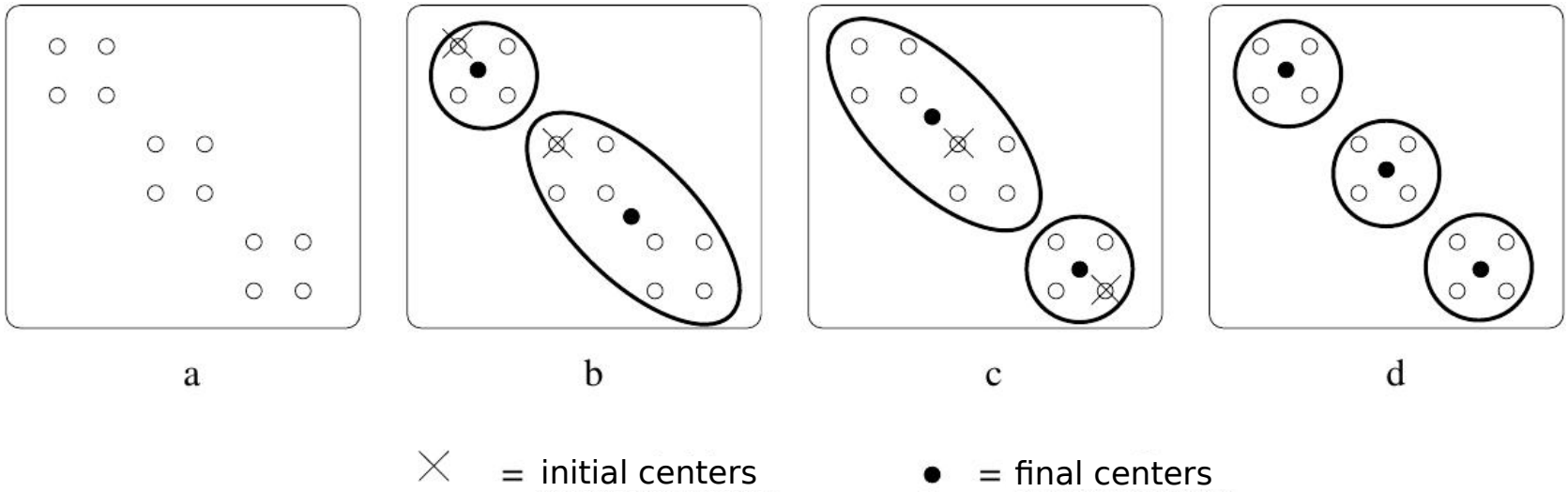
- Forgy method
  - pick randomly K observations from the dataset as initial centers,
  - run K-means algorithm with these starting values
  - repeat these 2 steps several times and retain the best (cost sense) clustering
- lot of variants : `Random partitions`, `k-means++`, `power init.`
  - ☞ may lower the computation time of one run
  - ☞ can give some guarantees that the solution is close 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

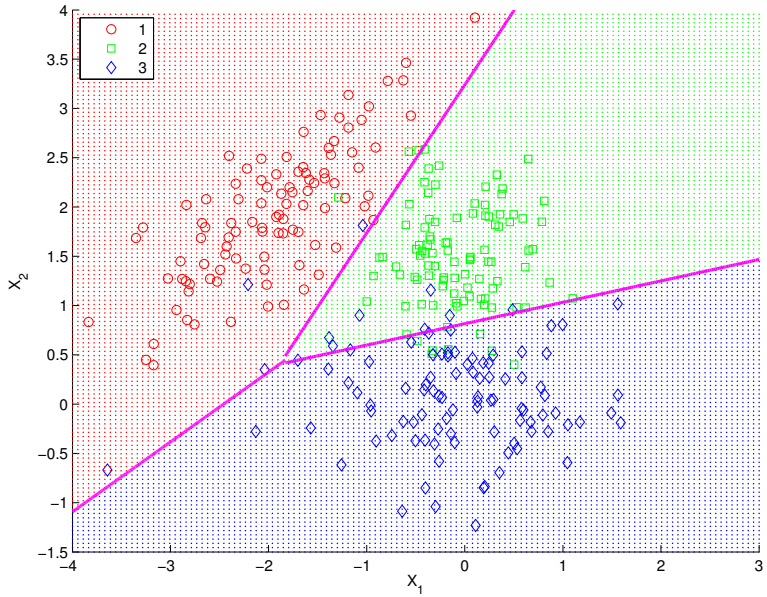
Sensitivity to initialization/data geometry/number of classes



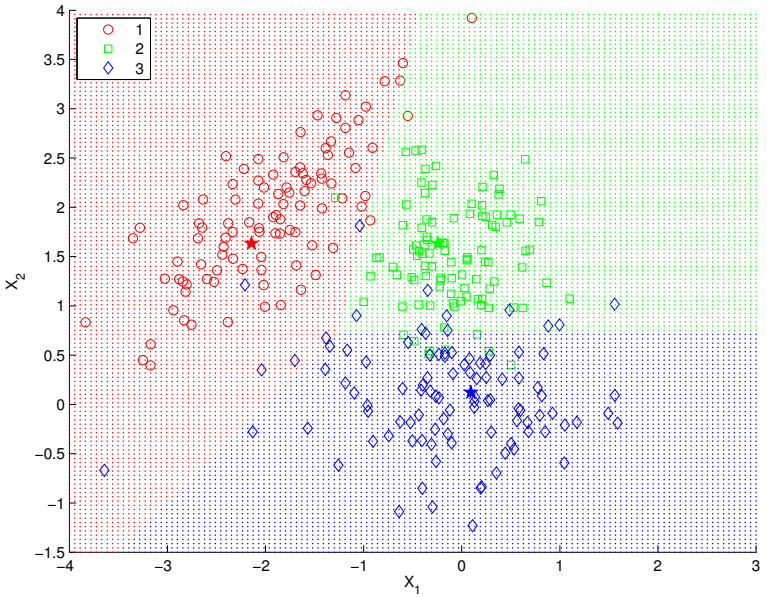
a) set of points $x_i \in \mathbb{R}^p$ ($p = 2$) to classify, b) and c) two clusterings in $K = 2$ classes with different initial centers, d) clustering in $K = 3$ classes.

K-means

Prediction vs Clustering



LDA (supervised approach)



K-means with $K = 3$ classes. Does not use the true y_i !

Table of Contents

EM (Expectation-Maximization) algorithm

- **General** and **important** method for finding maximum likelihood (ML) or maximum a posteriori (MAP) estimates, 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) algorithm

- **General** and **important** method for finding maximum likelihood (ML) or maximum a posteriori (MAP) estimates, 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

Details

- Z is a latent variable, objective : maximize $\ell(\theta) = \log p_\theta(x)$

EM (Expectation-Maximization) algorithm

- **General** and **important** method for finding maximum likelihood (ML) or maximum a posteriori (MAP) estimates, 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

Details

- Z is a latent variable, objective : maximize $\ell(\theta) = \log p_\theta(x)$
- Decompose

$$\ell(\theta) = \sum_i \log p_\theta(x_i) = \sum_i \log p_\theta(x_i, z_i) - \log p_\theta(z_i|x_i)$$

EM (Expectation-Maximization) algorithm

- **General** and **important** method for finding maximum likelihood (ML) or maximum a posteriori (MAP) estimates, 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

Details

- Z is a latent variable, objective : maximize $\ell(\theta) = \log p_\theta(x)$
- Decompose

$$\ell(\theta) = \sum_i \log p_\theta(x_i) = \sum_i \log p_\theta(x_i, z_i) - \log p_\theta(z_i|x_i)$$

$$\ell(\theta) = \underbrace{\sum\nolimits_i \mathbb{E}_{Z|X,\theta^{(t)}} \log p_\theta(x_i, z_i)}_{Q(\theta,\theta^{(t)})} - \sum\nolimits_i \mathbb{E}_{Z|X,\theta^{(t)}} \log p_\theta(z_i|x_i)$$

- where we have taken conditional expectation on Z wrt X and some previous parameter $\theta^{(t)}$, which does not affect the left expression
- we will maximize the first term as a proxy

EM (Expectation-Maximization) principle

Sketch of EM algorithm

- **E step** : compute the expectation of the completed log-likelihood function evaluated using the current estimate for the parameter $\theta^{(t)}$

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

EM (Expectation-Maximization) principle

Sketch of EM algorithm

- **E step** : compute the expectation of the completed log-likelihood function evaluated using the current estimate for the parameter $\theta^{(t)}$

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

- **M step** : compute parameters $\theta^{(t+1)}$ maximizing the expected log-likelihood

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

EM (Expectation-Maximization) principle

Sketch of EM algorithm

- **E step** : compute the expectation of the completed log-likelihood function evaluated using the current estimate for the parameter $\theta^{(t)}$

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

- **M step** : compute parameters $\theta^{(t+1)}$ maximizing the expected log-likelihood

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

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

EM (Expectation-Maximization) principle

Sketch of EM algorithm

- **E step** : compute the expectation of the completed log-likelihood function evaluated using the current estimate for the parameter $\theta^{(t)}$

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

- **M step** : compute parameters $\theta^{(t+1)}$ maximizing the expected log-likelihood

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

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

$\rightarrow$ often implicitely merged in one "single" step

Application of EM to mixture models : E step

Introducing the class latent variables y_i, or equivalently, the binary variables

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

Application of EM to mixture models : E step

Introducing the class 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

$$p_\theta(x, z) = \prod_{i=1}^{n} p_\theta(x_i, z_i) = \prod_{i=1}^{n} \pi_{y_i} \phi(x_i, \theta_{y_i}) = \prod_{i=1}^{n} \prod_{k=1}^{K} [\pi_k \phi(x_i, \theta_k)]^{z_{ik}},$$

Application of EM to mixture models : E step

Introducing the class 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

$$p_\theta(x, z) = \prod_{i=1}^{n} p_\theta(x_i, z_i) = \prod_{i=1}^{n} \pi_{y_i} \phi(x_i, \theta_{y_i}) = \prod_{i=1}^{n} \prod_{k=1}^{K} [\pi_k \phi(x_i, \theta_k)]^{z_{ik}},$$

$$\log p_\theta(x, z) = \sum_{i=1}^{n} \sum_{k=1}^{K} z_{ik} \log(\pi_k \phi(x_i, \theta_k))$$

Application of EM to mixture models : E step

Introducing the class 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

$$p_\theta(x, z) = \prod_{i=1}^{n} p_\theta(x_i, z_i) = \prod_{i=1}^{n} \pi_{y_i} \phi(x_i, \theta_{y_i}) = \prod_{i=1}^{n} \prod_{k=1}^{K} [\pi_k \phi(x_i, \theta_k)]^{z_{ik}},$$

$$\log p_\theta(x, z) = \sum_{i=1}^{n} \sum_{k=1}^{K} z_{ik} \log(\pi_k \phi(x_i, \theta_k))$$

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

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

For Gaussian Mixture Model : $t_{ik}^{(t)} \propto \pi_k^{(t)} \mathcal{N}(x_i, \mu_k^{(t)}, \Sigma_k^{(t)})$

Gaussian mixture models : M step

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

Gaussian mixture models : M step

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

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

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

Gaussian mixture models : M step

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

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

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

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

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

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

  - empirical averages weighted by the posterior probability $t_{ik}^{(t)}$

Gaussian mixture models : M step

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

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

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

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

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

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

  - empirical averages weighted by the posterior probability $t_{ik}^{(t)}$

☞ **"Similar" to K-means, but :**

- Non-Euclidean distance
  $t_{ik}^{(t)} \propto \pi_k^{(t)} \left(\det\left(2\pi\Sigma_k^{(t)}\right)\right)^{-1/2} \exp\left(-\frac{1}{2}(x_i - \mu_k^{(t)})^T (\Sigma_k^{(t)})^{-1} (x_i - \mu_k^{(t)})\right)$
- soft-assignment $t_{ik}^{(t)}$ of the communities

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 $\pi_k^{(t)}$, $\mu_k^{(t)}$, $\Sigma_k^{(t)}$

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 (assign communities weights t_{ik})
- M step $\leftrightarrow$ update/correction step (compute optimal parameters)

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 (assign communities weights t_{ik})
- M step $\leftrightarrow$ update/correction step (compute optimal parameters)

Convergence

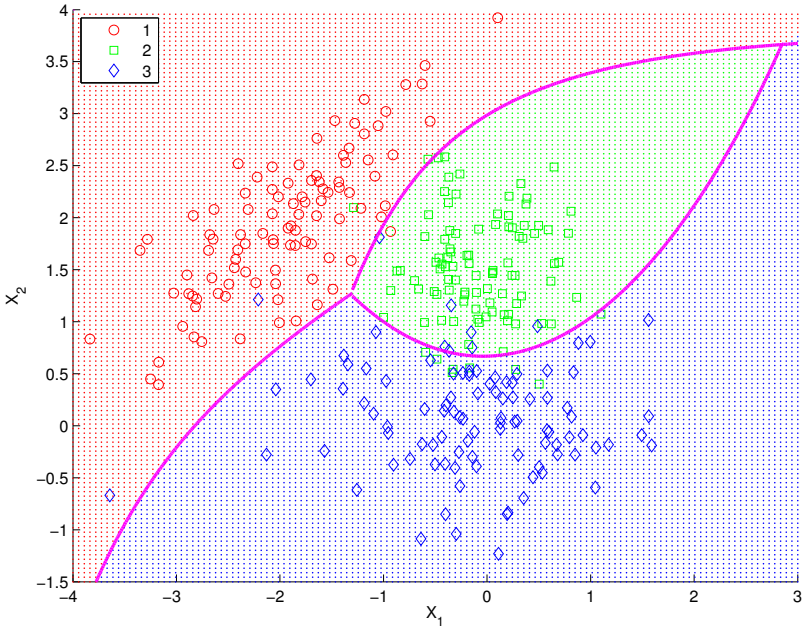
- EM : convergence toward a local maximum of the log-likelihood

☞ **no guarantee** of getting the optimal solution (depend on the initial values)

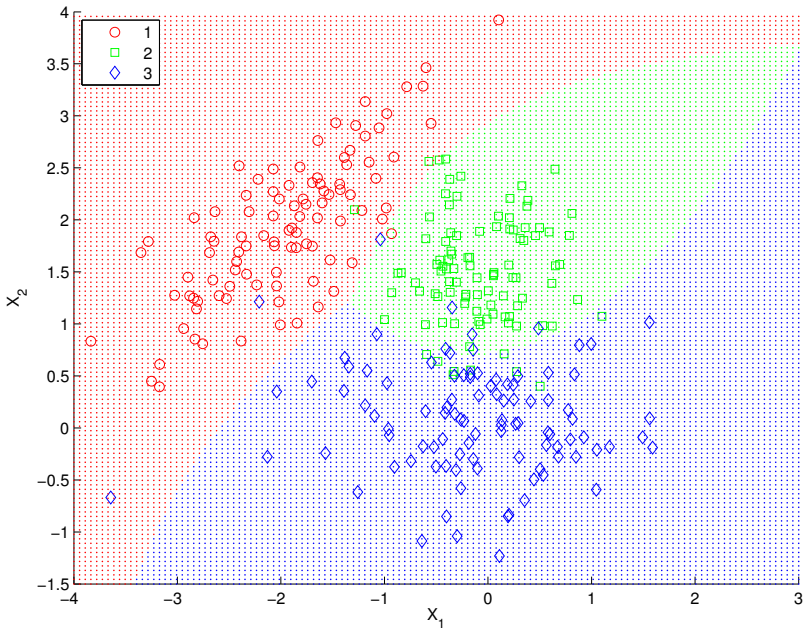
☞ For generic GMM, **no optimal solution** ! ! Can be "degenerate" around a single sample : $\Sigma_k \to 0 \Rightarrow \ell(\theta) \to +\infty$

Gaussian mixture model and EM algorithm

Prediction vs Clustering



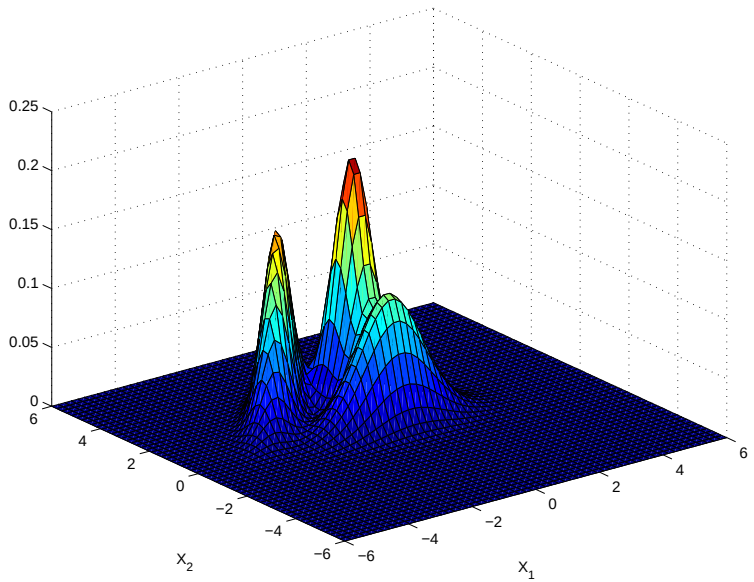
QDA (**supervised** approach)



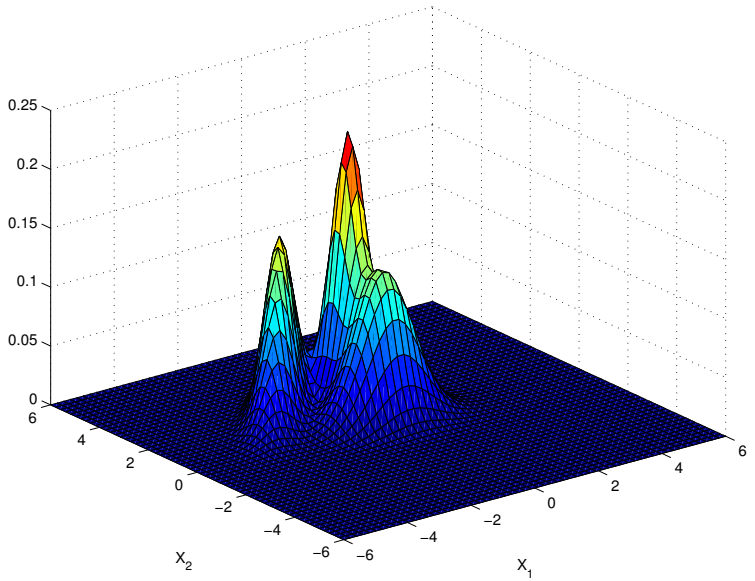
EM with $K = 3$ classes

Gaussian mixture model and EM algorithm

Estimation of the mixture density f



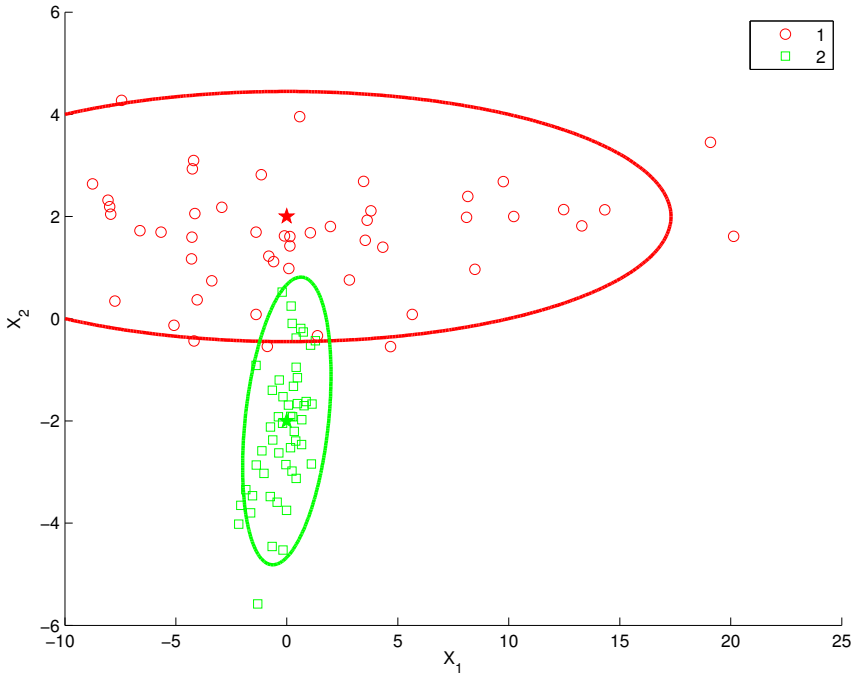
True density of the data points $x_1, \ldots, x_n$



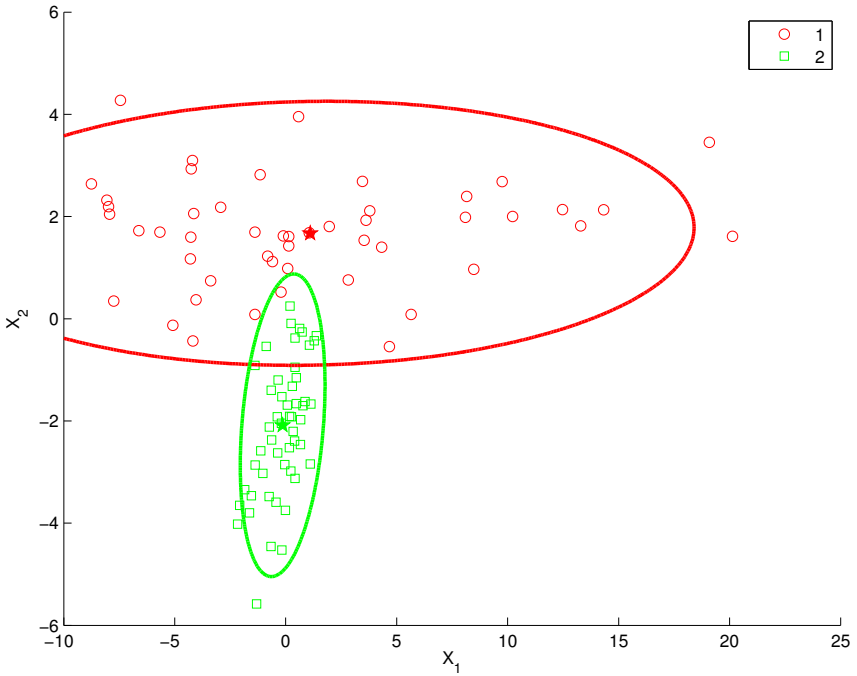
Estimated density with EM ($K = 3$ classes)

Comparison K-means vs Algo EM

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



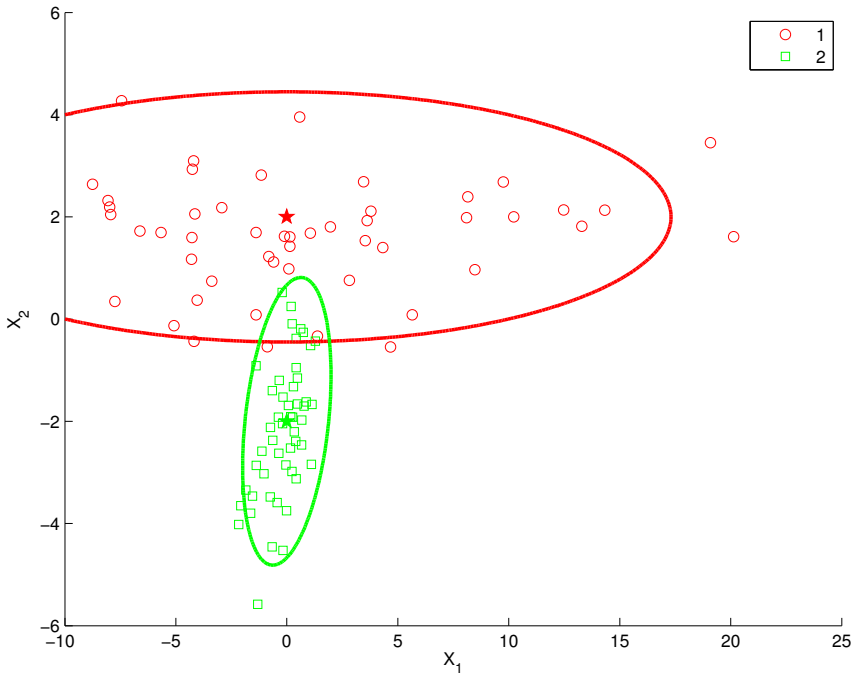
Data $x_1, \ldots, x_n$, classes and true 95% confidence regions



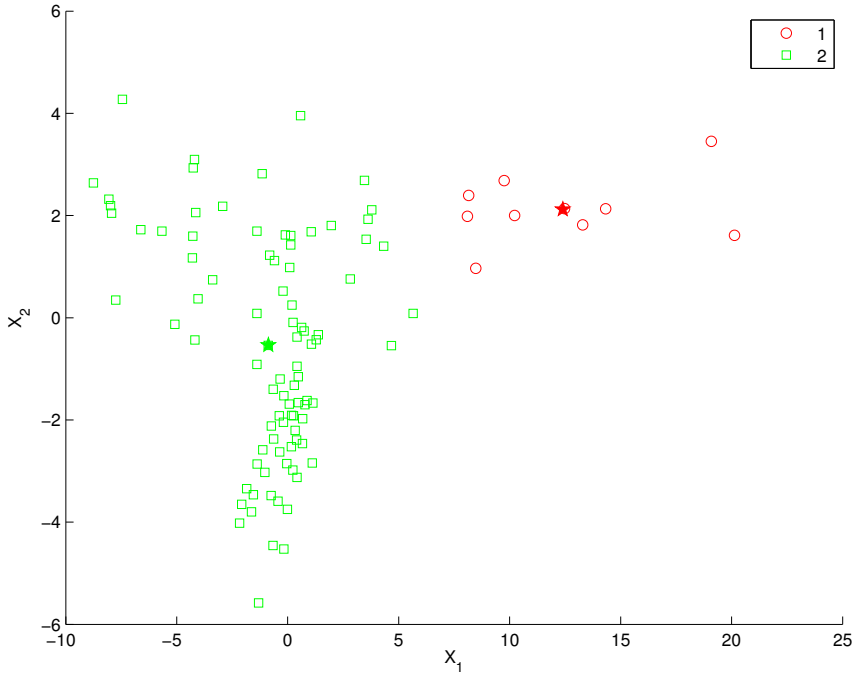
Clustering with EM ($K = 2$) and estimated 95% confidence regions

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 ($K = 2$)

Table of Contents

Model selection : estimation of K

- $K = 1$: underfitting
- $K = n$: overfitting
  - ☞ Solving $\max_{\theta,K} \ell(\theta)$ leads to overfitting : a K-mixture model is also a $K-1$ mixture model (with a $\pi_k = 0$), so the optimal ℓ is increasing in K

Model selection : estimation of K

- $K = 1$: underfitting
- $K = n$: overfitting
  - ☞ Solving $\max_{\theta,K} \ell(\theta)$ leads to overfitting : a K-mixture model is also a $K-1$ mixture model (with a $\pi_k = 0$), so the optimal ℓ is increasing in K

Minimization of a penalized log-likelihood criterion

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

- $\hat{\ell}(K, x) \equiv \ell(\hat{\theta}_K)$ 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

- $-\hat{\ell}(K, x)$: fidelity to the data (likelihood) $\rightarrow$ tends to increase K
- $\text{pen}(K, n)$: low complexity of the model $\rightarrow$ tends to decrease K

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

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)

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

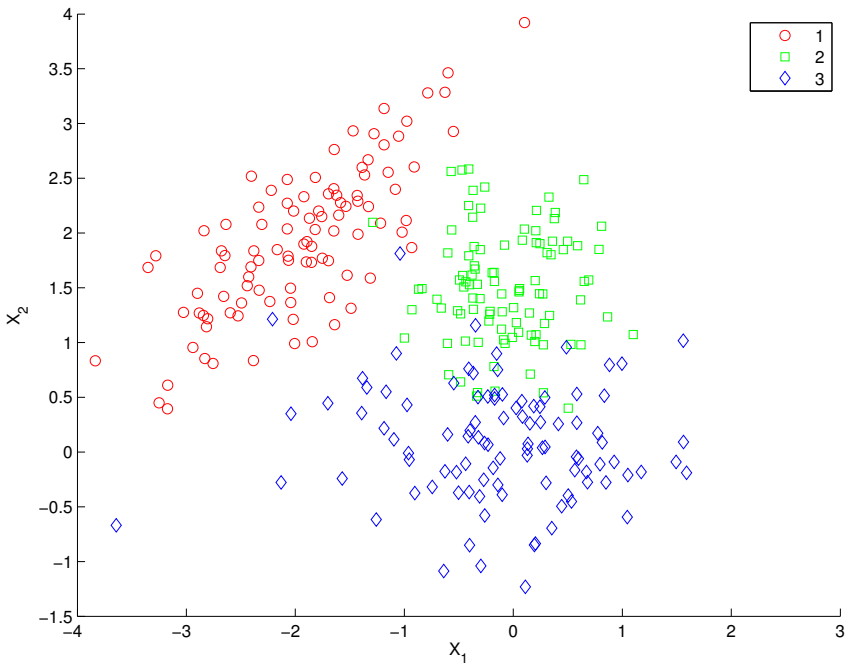
- n is the size of the data
- m_K is the effective number of parameters for the K class model

Equivalent to minimize the following criterion

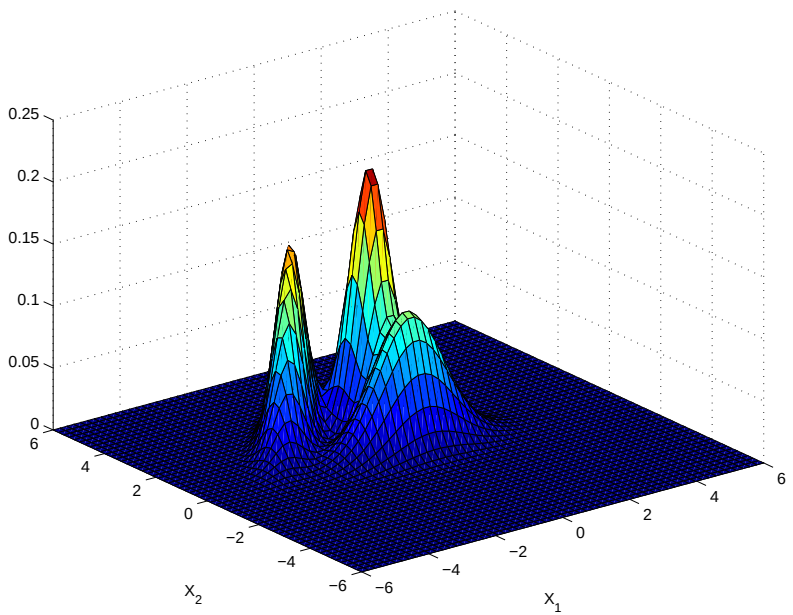
$$\mathrm{BIC}(K) = -2\hat{\ell}(K, x) + m_K \log(n)$$

Model selection : estimation of K

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



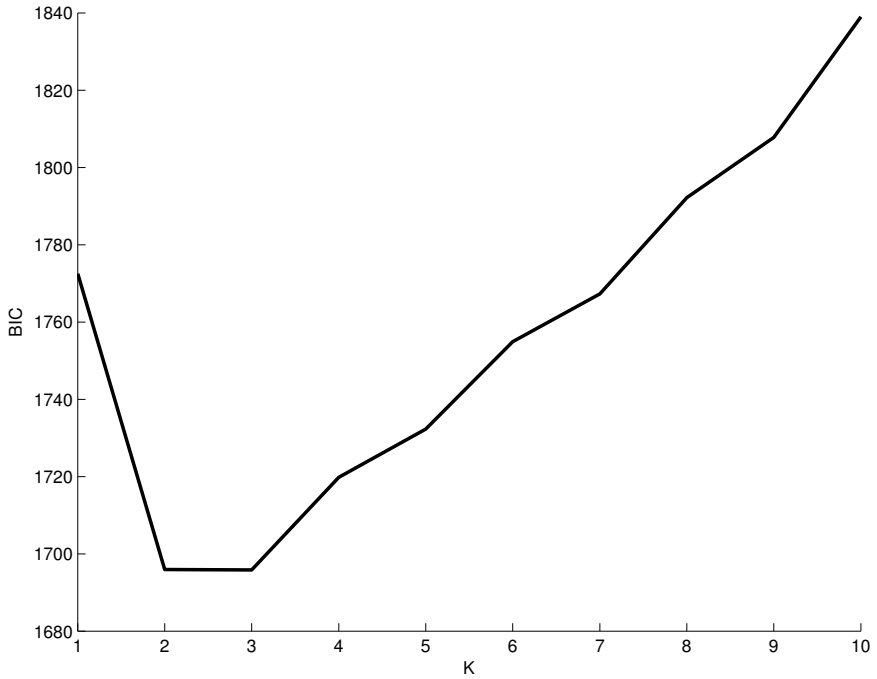
Dataset $x_1, \ldots, x_n$ ($n = 500$ realizations)



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 $K = 3$)

Table of Contents

Kernel Kmeans

Recall :

$$J(y) = \sum_i \|x_i - \hat{\mu}_{y_i}\|^2 \qquad \text{where } \hat{\mu}_k = \frac{1}{\sharp\{y_j = k\}} \sum_{j|y_j=k} x_j$$

- Can be computed using only the distances $\|x_i - x_j\|^2$, aka their inner products
- Can be kernelized ! $k(x, z) = \langle \phi(x), \phi(z) \rangle_{\mathcal{H}}$

Kernel Kmeans

Recall :

$$J(y) = \sum_i \|x_i - \hat{\mu}_{y_i}\|^2 \qquad \text{where } \hat{\mu}_k = \frac{1}{\sharp\{y_j = k\}} \sum_{j|y_j=k} x_j$$

- Can be computed using only the distances $\|x_i - x_j\|^2$, aka their inner products
- Can be kernelized ! $k(x, z) = \langle \phi(x), \phi(z) \rangle_{\mathcal{H}}$
- Expression the centroid in the RKHS (generally not explicit !)

$$\mu_k^{(t)} = \frac{1}{n_k^{(t)}} \sum_{i|y_i^{(t)}=k} \phi(x_i) \text{ with } n_k^{(t)} = \sharp\{y_i^{(t)} = k\}$$

Kernel Kmeans

Recall :

$$J(y) = \sum_i \|x_i - \hat{\mu}_{y_i}\|^2 \qquad \text{where } \hat{\mu}_k = \frac{1}{\sharp\{y_j = k\}} \sum_{j|y_j=k} x_j$$

- Can be computed using only the distances $\|x_i - x_j\|^2$, aka their inner products
- Can be kernelized ! $k(x, z) = \langle \phi(x), \phi(z) \rangle_{\mathcal{H}}$
- Expression the centroid in the RKHS (generally not explicit !)

$$\mu_k^{(t)} = \frac{1}{n_k^{(t)}} \sum_{i|y_i^{(t)}=k} \phi(x_i) \text{ with } n_k^{(t)} = \sharp\{y_i^{(t)} = k\}$$

- Assignement step $y_i^{(t+1)} = \arg\min_k \|\phi(x_i) - \mu_k^{(t)}\|^2$ with

$$\begin{aligned} ||\phi(x) - \mu_k^{(t)}||^2 &= \left\langle \phi(x) - \frac{1}{n_k^{(t)}} \sum_{i|y_i^{(t)}=k} \phi(x_i), \phi(x) - \frac{1}{n_k^{(t)}} \sum_{i|y_i^{(t)}=k} \phi(x_i) \right\rangle_{\mathcal{H}} \\ &= k(x,x) - \frac{2}{n_k^{(t)}} \sum_{i|y_i^{(t)}=k} k(x, x_i) + \frac{1}{(n_k^{(t)})^2} \sum_{i,j|y_i^{(t)}, y_j^{(t)}=k} k(x_i, x_j) \end{aligned}$$

Kernel Kmeans

Recall :

$$J(y) = \sum_i \|x_i - \hat{\mu}_{y_i}\|^2 \qquad \text{where } \hat{\mu}_k = \frac{1}{\sharp\{y_j=k\}} \sum_{j|y_j=k} x_j$$

- Can be computed using only the distances $\|x_i - x_j\|^2$, aka their inner products
- Can be kernelized ! $k(x,z) = \langle \phi(x), \phi(z) \rangle_{\mathcal{H}}$
- Expression the centroid in the RKHS (generally not explicit !)

$$\mu_k^{(t)} = \frac{1}{n_k^{(t)}} \sum_{i|y_i^{(t)}=k} \phi(x_i) \text{ with } n_k^{(t)} = \sharp\{y_i^{(t)} = k\}$$

- Assignement step $y_i^{(t+1)} = \arg\min_k \|\phi(x_i) - \mu_k^{(t)}\|^2$ with

$$\begin{aligned} ||\phi(x) - \mu_k^{(t)}||^2 &= \left\langle \phi(x) - \frac{1}{n_k^{(t)}} \sum_{i|y_i^{(t)}=k} \phi(x_i), \phi(x) - \frac{1}{n_k^{(t)}} \sum_{i|y_i^{(t)}=k} \phi(x_i) \right\rangle_{\mathcal{H}} \\ &= k(x,x) - \frac{2}{n_k^{(t)}} \sum_{i|y_i^{(t)}=k} k(x,x_i) + \frac{1}{(n_k^{(t)})^2} \sum_{i,j|y_i^{(t)},y_j^{(t)}=k} k(x_i,x_j) \end{aligned}$$

- Assignment does NOT request explicit knowledge of μ_k (usually NOT known)
- Computationally more demanding : $O(N^2)$ instead of $O(NK)$

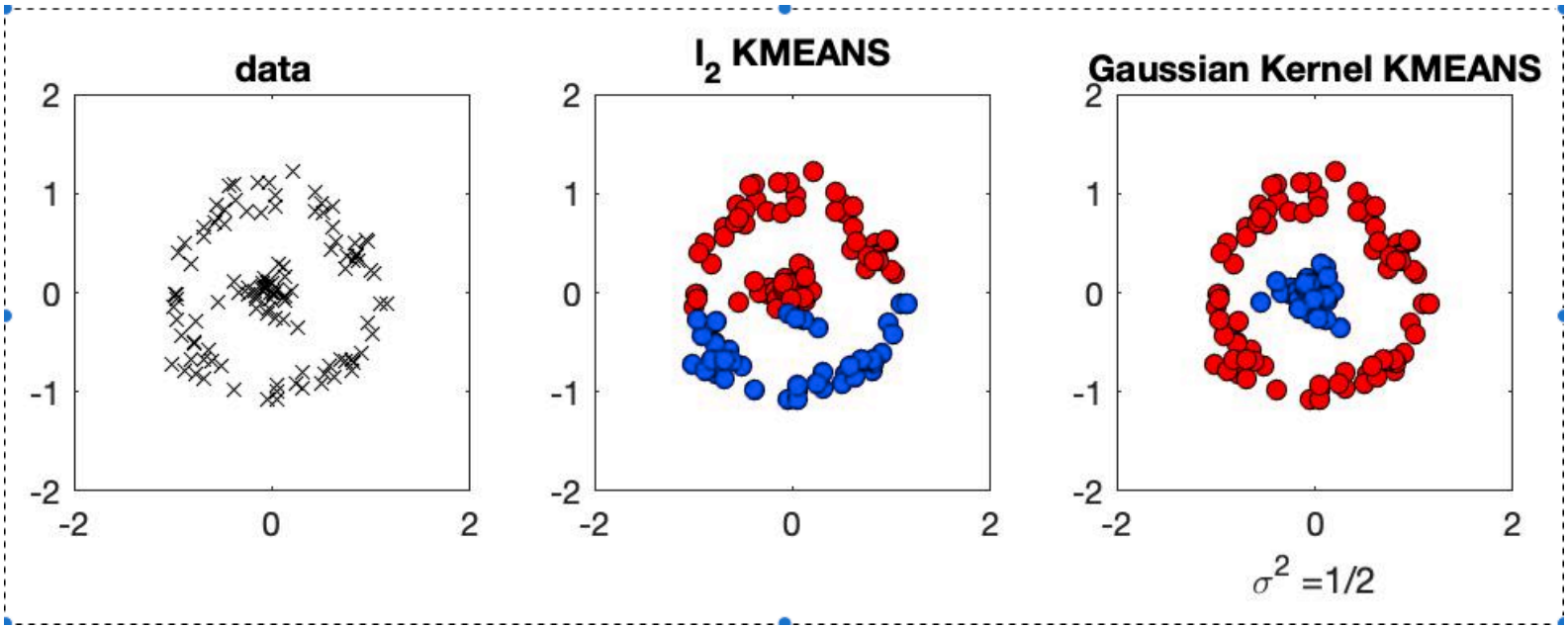
Kernel Kmeans

- Kernel K-means allows to tackle problems with non convex classes
- Kernel K-means has increased sensitivity to initial conditions (random initial labelling instead of centroids)
- Kernel expression requires some tuning parameter to be set.

Kernel Kmeans

- Kernel K-means allows to tackle problems with non convex classes
- Kernel K-means has increased sensitivity to initial conditions (random initial labelling instead of centroids)
- Kernel expression requires some tuning parameter to be set.

Example :



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

Dissimilarity measures

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

Dissimilarity measures

- Dissimilarity measures requires that
  - $d_{ii} = 0$
  - $d_{ij} \geq 0$
  - $d_{ij} = d_{ji}$
- Often $d_{ij} \leq d_{ik} + d_{kj}$ is NOT satisfied $\forall (i, j, k) \in [1, \ldots N]^3$
- d may be sometimes required to be a true distance

Dissimilarity measures

- Dissimilarity measures requires that
  - $d_{ii} = 0$
  - $d_{ij} \geq 0$
  - $d_{ij} = d_{ji}$
- Often $d_{ij} \leq d_{ik} + d_{kj}$ is NOT satisfied $\forall (i, j, k) \in [1, \ldots N]^3$
- d may be sometimes required to be a true distance
- From similarity s_{ij} to distance or dissimilarity measure d_{ij} : use any decreasing function e.g.

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

Dissimilarity measures

- Dissimilarity measures requires that
  - $d_{ii} = 0$
  - $d_{ij} \geq 0$
  - $d_{ij} = d_{ji}$
- Often $d_{ij} \leq d_{ik} + d_{kj}$ is NOT satisfied $\forall (i, j, k) \in [1, \dots N]^3$
- d may be sometimes required to be a true distance
- From similarity s_{ij} to distance or dissimilarity measure d_{ij} : use any decreasing function e.g.

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

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

Generalizing K-MEANS for alternate dissimilarity measures

How to generalize μ ? No notion of sum for categorical x_i.

Generalizing K-MEANS for alternate dissimilarity measures

How to generalize μ ? No notion of sum for categorical x_i.

Remark : μ can be defined using only the distances

$$\hat{\mu}_k = \min_{\mu \in \mathbb{R}^p} \sum_{x_i | y_i = k} \|x_i - \mu\|^2$$

Generalizing K-MEANS for alternate dissimilarity measures

How to generalize μ ? No notion of sum for categorical x_i.
Remark : μ can be defined using only the distances

$$\hat{\mu}_k = \min_{\mu \in \mathbb{R}^p} \sum_{x_i | y_i = k} \|x_i - \mu\|^2$$

Pbm : The d_{ij} are defined only between the x_i, x_j

Generalizing K-MEANS for alternate dissimilarity measures

How to generalize μ ? No notion of sum for categorical x_i.
Remark : μ can be defined using only the distances

$$\hat{\mu}_k = \min_{\mu \in \mathbb{R}^p} \sum_{x_i | y_i = k} \|x_i - \mu\|^2$$

Pbm : The d_{ij} are defined only between the x_i, x_j
Def : Medoids for each cluster (of index k) :

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

- take the x_j that is at average distance of its community as a proxy for "the mean" μ
- The assignment step remains the same : $y_i = \arg\min_k d_{i,j_k}$ where $\mathrm{Med}_k = x_{j_k}$

Generalizing K-MEANS for alternate dissimilarity measures

How to generalize μ ? No notion of sum for categorical x_i.
Remark : μ can be defined using only the distances

$$\hat{\mu}_k = \min_{\mu \in \mathbb{R}^p} \sum_{x_i | y_i = k} \|x_i - \mu\|^2$$

Pbm : The d_{ij} are defined only between the x_i, x_j
Def : Medoids for each cluster (of index k) :

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

- take the x_j that is at average distance of its community as a proxy for "the mean" μ
- The assignment step remains the same : $y_i = \arg\min_k d_{i,j_k}$ where $\mathrm{Med}_k = x_{j_k}$

Remark

- If N is large, 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.

Table of Contents

Evaluating clustering results

Unsupervised framework

- no ground truth is available, (in general).

Evaluating clustering results

Unsupervised framework

- 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)

Evaluating clustering results

Unsupervised framework

- 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 (thm : the sum is constant).
  - For general dissimilarity measures, popular quality indices (among others)
    - 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 :

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

- Separability S :

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

- DBindex (lower is better) :

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

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

$$b(x_i) = \min_{l \neq y_i} \frac{1}{N_l} \sum_{j | y_j = l} d(x_i, x_j)$$

- Silhouette (higher is better)

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

  - if $N_{y_i} = 1$, set $\mathcal{S}(x_i) = 0$
  - $-1 \leq \mathcal{S}(x_i) \leq 1$
  - 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 $\hat{y}$ with the ground truth partition. Note that labels may take different values for these partitions : permutations, different number of 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 $\hat{y}$ with the ground truth partition. Note that labels may take different values for these partitions : permutations, different number of clusters...

1. RAND index (higher is better)

$$RI = \frac{2}{n(n-1)} \sum_{i=1}^{n} \sum_{j=i+1}^{n} \delta(\hat{y}_i = \hat{y}_j)\delta(y_i = y_j) + \delta(\hat{y}_i \neq \hat{y}_j)\delta(y_i \neq y_j)$$

- 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.
- $0 \leq RI \leq 1$

Clustering quality measure with expert knowledge

2. Purity index $\mathcal{P}$ (higher is better)

Let

$$p_{kl} \overset{def}{=} \frac{n_{kl}}{n_k} = \frac{\#(\hat{y}_i = k) \cap (y_i = l)}{\#(\hat{y}_i = k)}$$

and

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

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

- p_{kl} is the proportion of observations whose estimated label is k and true label is l
- P_k is this latter proportion, for the class $Y_i = l$ which contains the more observations with label $\hat{Y}_i = k \rightarrow$ if the cluster k matches with one true community, then $P_k = 1$

Clustering quality measure with expert knowledge

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 $H(U) = -\sum_{i=1}^{R} P_U(i) \log P_U(i)$

Table of Contents

Hierarchical approaches

Motivations

Recursive approach to partition data at all possible scale, using multi-level hierarchical partitioning... Many variants ! Application dependent.

Hierarchical approaches

Motivations

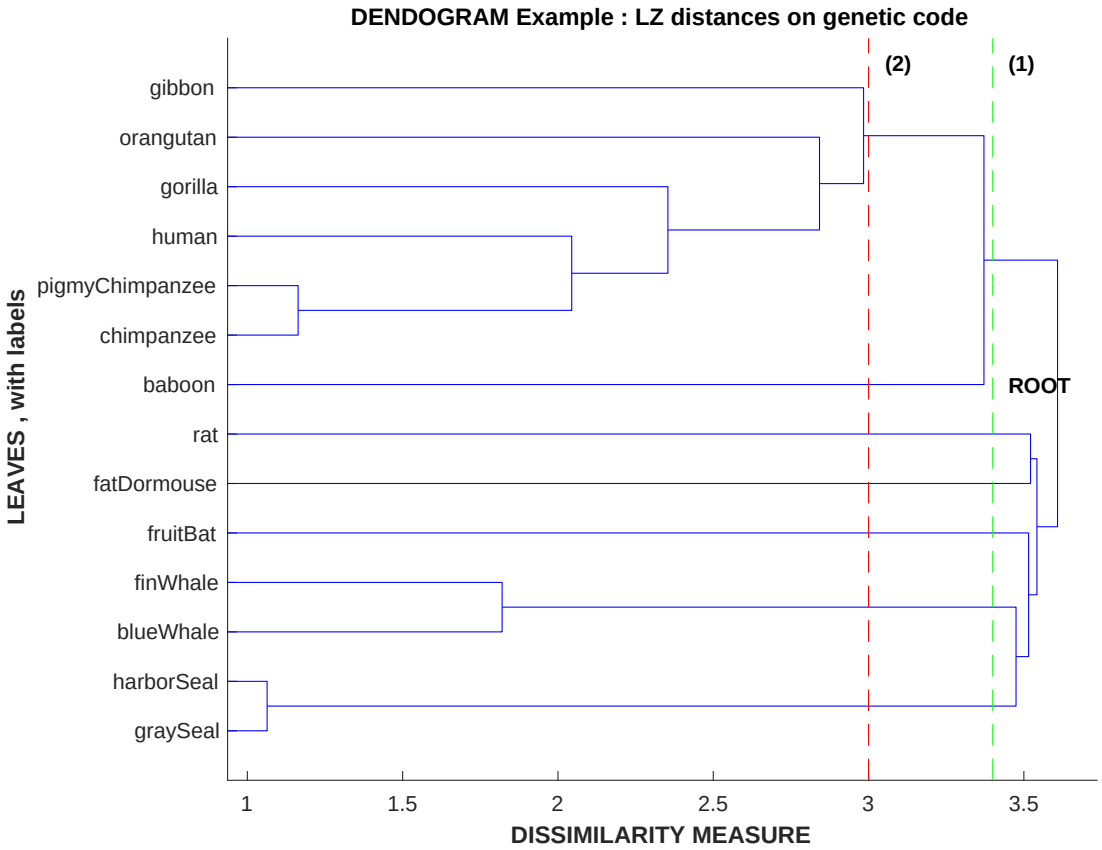
Recursive approach to partition data at all possible scale, using multi-level hierarchical partitioning... Many variants ! Application dependent.

- 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 is 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*

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

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*

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

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

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