K-means et mélanges de gaussiennes

Master parcours SSD - UE Apprentissage Statistique I

Pierre Mahé - bioMérieux & Université de Grenoble-Alpes

Rapppels : clustering

Clustering - qualité

But du clustering :

- déterminer des ensembles de points proches ...
- ... qui soient distants les uns des autres

Clustering - qualité

But du clustering :

- déterminer des ensembles de points proches ...
- ... qui soient distants les uns des autres

Fonction objective (à minimiser) = dispersion "intra" cluster

$$W(C) = \sum_{k=1}^{K} \sum_{i:C(i)=k} \sum_{j:C(j)=k} d(x_i, x_j)$$

- K = nombre de clusters
- C = clustering : $C(i) = k \Leftrightarrow x_i \in$ cluster k
- $d(x, y)$ = distance/disimilarité entre x et y

Clustering - qualité

But du clustering :

- déterminer des ensembles de points proches ...
- ... qui soient distants les uns des autres

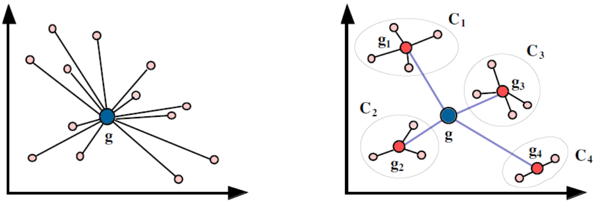
Fonction objective (à minimiser) = dispersion "intra" cluster

$$W(C) = \sum_{k=1}^{K} \sum_{i:C(i)=k} \sum_{j:C(j)=k} d(x_i, x_j)$$

- K = nombre de clusters
- C = clustering : $C(i) = k \Leftrightarrow x_i \in$ cluster k
- $d(x, y)$ = distance/disimilarité entre x et y

$\Rightarrow$ problème combinatoire, présence de minima locaux.

Clustering - qualité

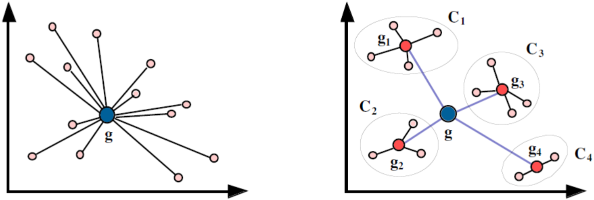


- dispersion totale = dispersion intra + dispersion inter :

$$\sum_{i,j=1}^{n} d(x_i, x_j) = \sum_{k=1}^{K} \sum_{i:C(i)=k} \left(\sum_{j:C(j)=k} d(x_i, x_j) + \sum_{j:C(j)\neq k} d(x_i, x_j) \right)$$

$$= \sum_{k=1}^{K} \sum_{i:C(i)=k} \sum_{j:C(j)=k} d(x_i, x_j) + \sum_{k=1}^{K} \sum_{i:C(i)=k} \sum_{j:C(j)\neq k} d(x_i, x_j)$$

Clustering - qualité



- dispersion totale = dispersion intra + dispersion inter :

$$\sum_{i,j=1}^{n} d(x_i, x_j) = \sum_{k=1}^{K} \sum_{i:C(i)=k} \Big(\sum_{j:C(j)=k} d(x_i, x_j) + \sum_{j:C(j)\neq k} d(x_i, x_j) \Big)$$

$$= \sum_{k=1}^{K} \sum_{i:C(i)=k} \sum_{j:C(j)=k} d(x_i, x_j) + \sum_{k=1}^{K} \sum_{i:C(i)=k} \sum_{j:C(j)\neq k} d(x_i, x_j)$$

- dispersion intra = dispersion totale - dispersion inter

$\Rightarrow$ minimiser intra équivalent à maximiser inter

Clustering et optimisation

Question : minimiser

$$W(C) = \sum_{k=1}^{K} \sum_{i:C(i)=k} \sum_{j:C(j)=k} d(x_i, x_j)$$

Clustering et optimisation

Question : minimiser

$$W(C) = \sum_{k=1}^{K} \sum_{i:C(i)=k} \sum_{j:C(j)=k} d(x_i, x_j)$$

Approche directe (gloutonne) :

1. considérer toutes les partitions possibles
2. retenir la meilleure

Clustering et optimisation

Question : minimiser

$$W(C) = \sum_{k=1}^{K} \sum_{i:C(i)=k} \sum_{j:C(j)=k} d(x_i, x_j)$$

Approche directe (gloutonne) :

1. considérer toutes les partitions possibles
2. retenir la meilleure

Problème combinatoire : le nombre de partitions augmente exponentiellement avec n

- e.g., 10 observations / 4 clusters : 34105 partitions possibles
- formellement :

$$\frac{1}{K!} \sum_{k=1}^{K} (-1)^{K-k} C_k^K k^n$$

Clustering et optimisation

Stratégie : approximer le critère à optimiser

$$W(C) = \sum_{k=1}^{K} \sum_{i:C(i)=k} \sum_{j:C(j)=k} d(x_i, x_j)$$

Clustering et optimisation

Stratégie : approximer le critère à optimiser

$$W(C) = \sum_{k=1}^{K} \sum_{i:C(i)=k} \sum_{j:C(j)=k} d(x_i, x_j)$$

Algorithme *k*-means :

- s'appuie sur la distance Euclidienne : $d(x_i, x_j) = ||x_i - x_j||^2$
- considère le critère :

$$W_{SS}(C) = \sum_{k=1}^{K} \sum_{i:C(i)=k} ||x_i - \mu_k||^2, \quad \text{où} \quad \mu_k = \frac{1}{n_k} \sum_{i:C(i)=k} x_i.$$

$\Rightarrow$ Objectif : minimiser la distance aux centres des clusters

- $\{\mu_k\}_{k=1,\ldots,K}$ = centroïdes

Algorithme k-means et optimisation

Avec la distance Euclidienne on a (au sein du cluster k) :

$$\sum_{i:C(i)=k} ||x_i - \mu_k||^2 = \frac{1}{2} \times \frac{1}{n_k} \sum_{i:C(i)=k} \sum_{j:C(j)=k} ||x_i - x_j||^2$$

Algorithme k-means et optimisation

Avec la distance Euclidienne on a (au sein du cluster k) :

$$\sum_{i:C(i)=k} ||x_i - \mu_k||^2 = \frac{1}{2} \times \frac{1}{n_k} \sum_{i:C(i)=k} \sum_{j:C(j)=k} ||x_i - x_j||^2$$

Le critère considéré par k-means est donc :

$$\begin{aligned} W_{SS}(C) &= \sum_{k=1}^{K} \frac{1}{2} \times \frac{1}{n_k} \sum_{i:C(i)=k} \sum_{j:C(j)=k} ||x_i - x_j||^2 \\ &= \boldsymbol{\frac{1}{2}} \sum_{k=1}^{K} \boldsymbol{\frac{1}{n_k}} \sum_{i:C(i)=k} \sum_{j:C(j)=k} d(x_i, x_j) \end{aligned}$$

Algorithme k-means et optimisation

Avec la distance Euclidienne on a (au sein du cluster k) :

$$\sum_{i:C(i)=k} ||x_i - \mu_k||^2 = \frac{1}{2} \times \frac{1}{n_k} \sum_{i:C(i)=k} \sum_{j:C(j)=k} ||x_i - x_j||^2$$

Le critère considéré par k-means est donc :

$$\begin{aligned} W_{SS}(C) &= \sum_{k=1}^{K} \frac{1}{2} \times \frac{1}{n_k} \sum_{i:C(i)=k} \sum_{j:C(j)=k} ||x_i - x_j||^2 \\ &= \mathbf{\frac{1}{2}} \sum_{k=1}^{K} \mathbf{\frac{1}{n_k}} \sum_{i:C(i)=k} \sum_{j:C(j)=k} d(x_i, x_j) \end{aligned}$$

$\Rightarrow$ $W_{SS}(C)$ = (total) within sum of squares

Algorithme k-means

k-means : objectif

Objectif : pour un nombre de clusters K fixé, minimiser

$$W_{SS}(C) = \sum_{k=1}^{K} \sum_{i:C(i)=k} ||x_i - \mu_k||^2,$$

où $\mu_k = \dfrac{1}{n_k} \sum_{i:C(i)=k} x_i$.

k-means : objectif

Objectif : pour un nombre de clusters K fixé, minimiser

$$W_{SS}(C) = \sum_{k=1}^{K} \sum_{i:C(i)=k} ||x_i - \mu_k||^2,$$

où $\mu_k = \frac{1}{n_k} \sum_{i:C(i)=k} x_i$.

Terminologie : $W_{SS}(C) = \sum_{k=1}^{K} W_{SS}^k(C)$

- $W_{SS}(C) =$ (total) within sum of squares
- $W_{SS}^k(C) =$ within sum of squares du cluster k

Algorithme k-means

1. Initialisation : affecter les points aléatoirement aux clusters
2. Itérer la procédure suivante :
   2.1 calculer les centroïdes des clusters :

   $$\mu_k^{(t)} = \frac{1}{n_k} \sum_{i:C(i)^{(t)}=k} x_i$$

   2.2 affecter chaque point au cluster dont le centroïde est le plus proche :

   $$C(i)^{(t+1)} = \arg \min_{k=1,\dots K} ||x_i - \mu_k^{(t)}||$$

Algorithme k-means

1. Initialisation : affecter les points aléatoirement aux clusters
2. Itérer la procédure suivante :
   2.1 calculer les centroïdes des clusters :

$$\mu_k^{(t)} = \frac{1}{n_k} \sum_{i:C(i)^{(t)}=k} x_i$$

   2.2 affecter chaque point au cluster dont le centroïde est le plus proche :

$$C(i)^{(t+1)} = \arg \min_{k=1,\ldots K} ||x_i - \mu_k^{(t)}||$$

Critère d'arrêt :

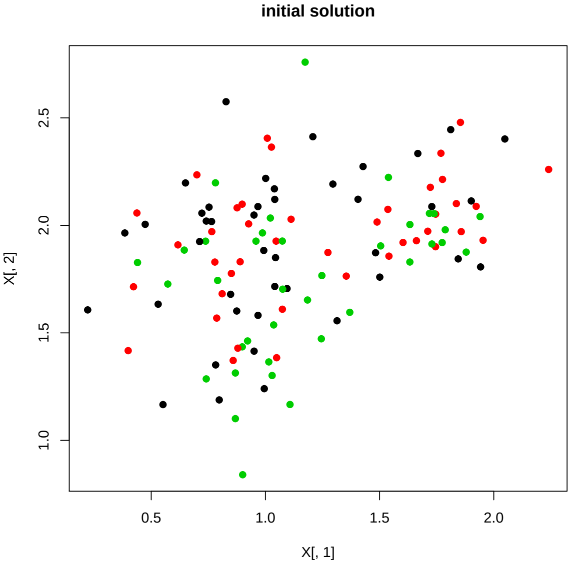
- convergence : les affectations ne changent plus
  - i.e., $C(i)^{(t+1)} = C(i)^{(t)}, \ \forall i = 1, ..., n.$
- nombre maximum d'itérations atteint

Algorithme k-means - remarques

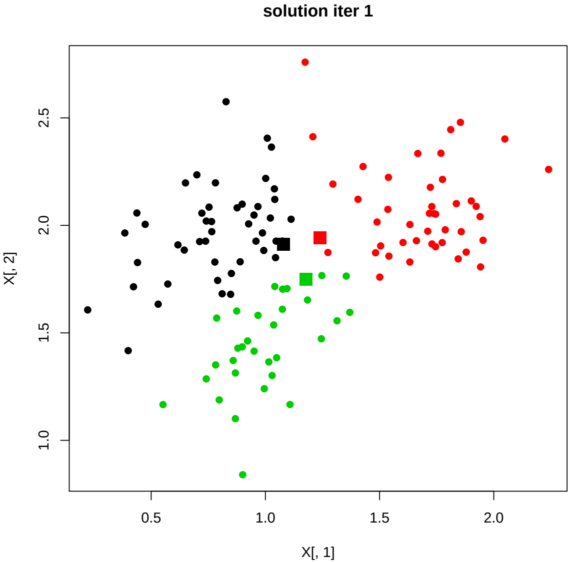
Deux remarques importantes :

1. Pré-requis : nombre de clusters K
2. L'algorithme converge
   - i.e., on a bien $C(i)^{(t+1)} = C(i)^{(t)}$ pour un t fini

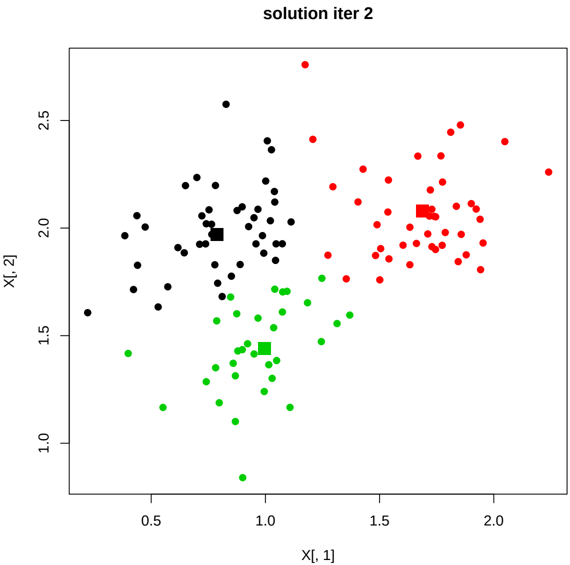
Algorithme k-means - illustration



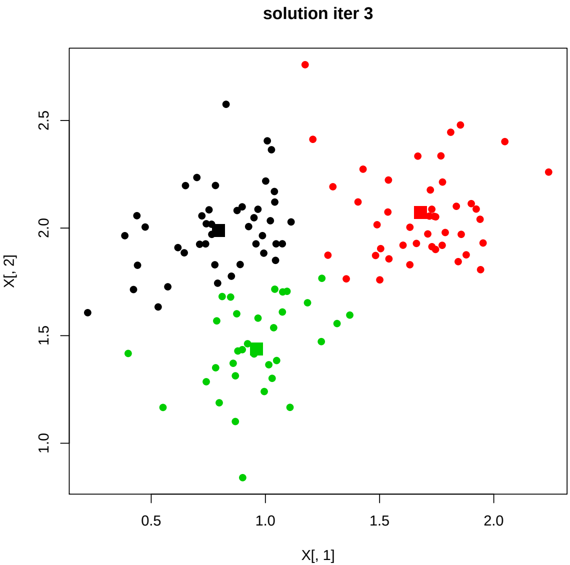
Algorithme k-means - illustration



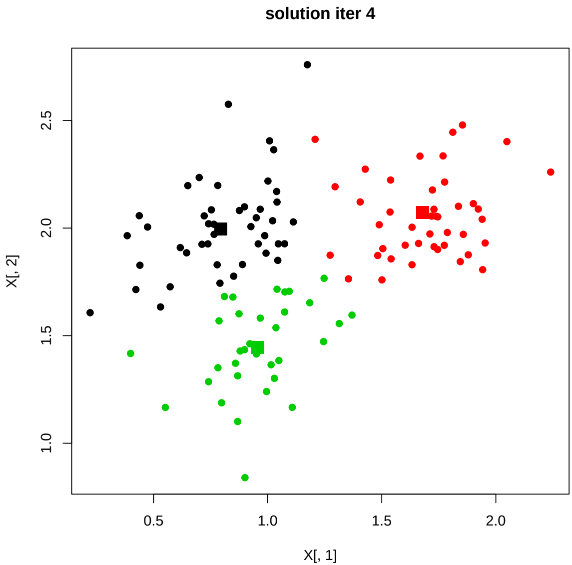
Algorithme k-means - illustration



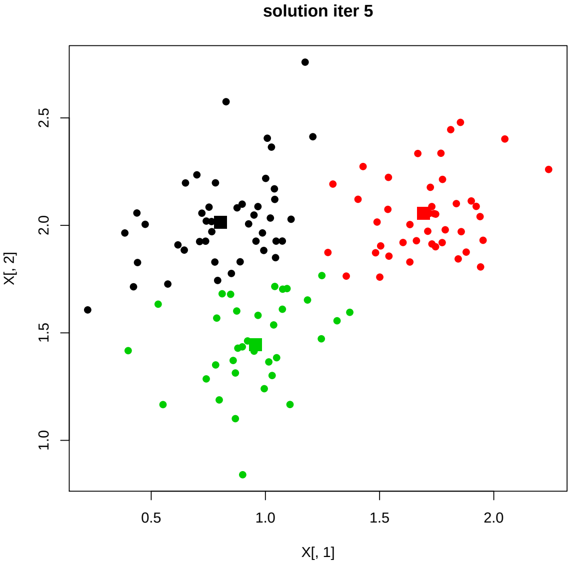
Algorithme k-means - illustration



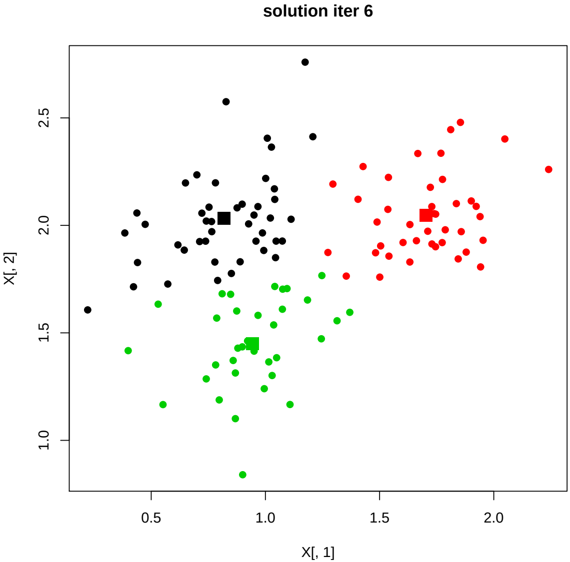
Algorithme k-means - illustration



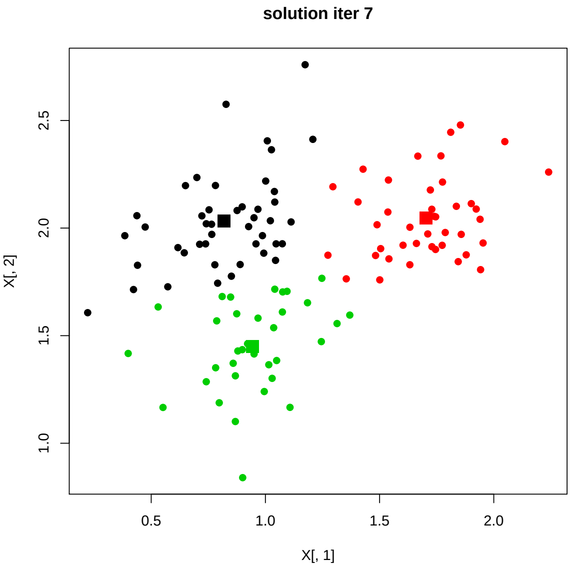
Algorithme k-means - illustration



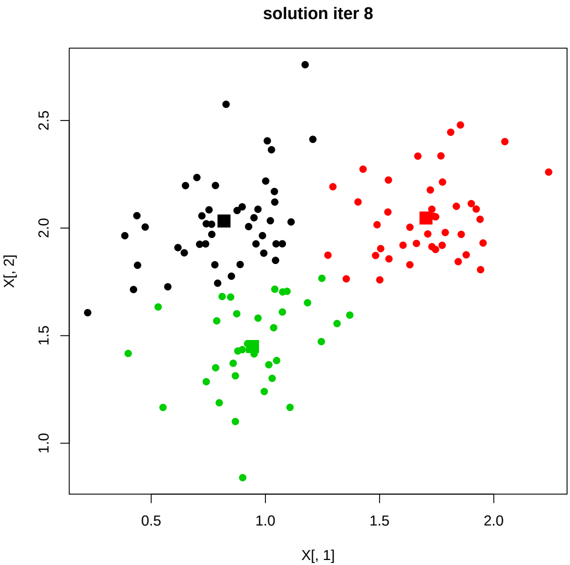
Algorithme k-means - illustration



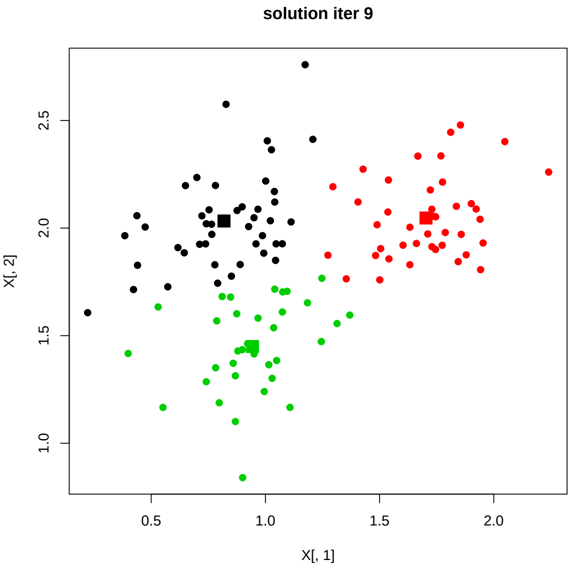
Algorithme k-means - illustration



Algorithme k-means - illustration

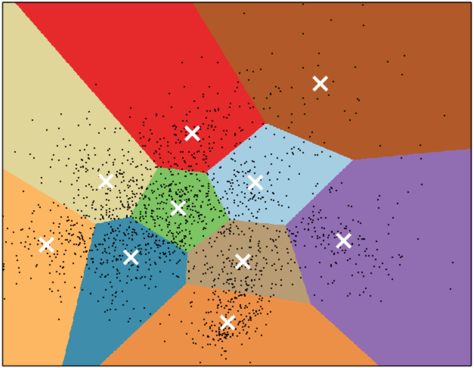


Algorithme k-means - illustration



Algorithme k-means - illustration

A convergence, solution = diagramme de Voronoï



- partition de l'espace définie par les centroïdes.

Algorithme k-means

Questions ouvertes :

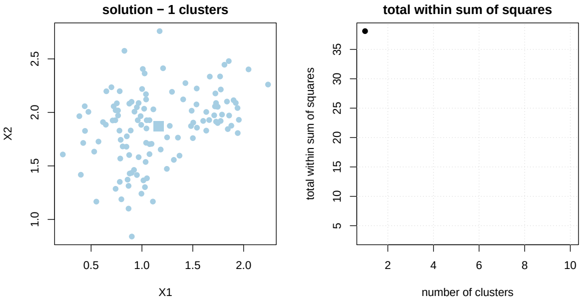
1. choix du nombre de clusters
2. stabilité de la solution
3. critère de distance

Algorithme k-means - choix du nombre de clusters

Comment choisir le nombre de clusters ?

Première idée : utiliser le critère $W_{SS}(C)$

- "total within sum of squares"

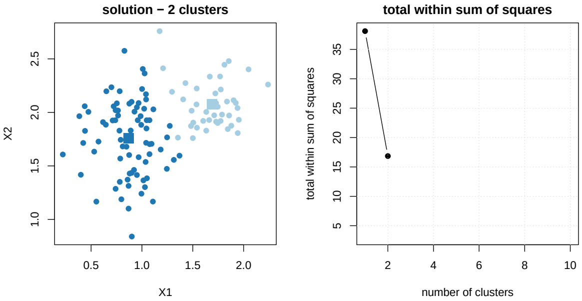


Algorithme k-means - choix du nombre de clusters

Comment choisir le nombre de clusters ?

Première idée : utiliser le critère $W_{SS}(C)$

- "total within sum of squares"

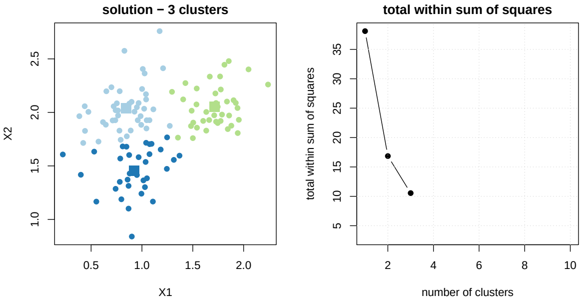


Algorithme k-means - choix du nombre de clusters

Comment choisir le nombre de clusters ?

Première idée : utiliser le critère $W_{SS}(C)$

- "total within sum of squares"

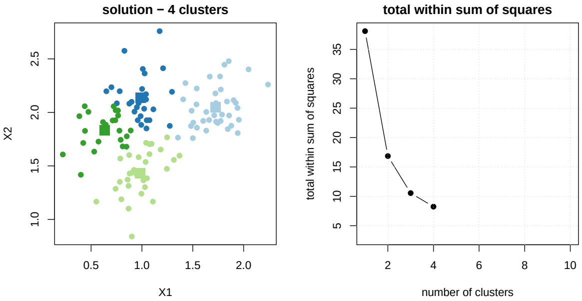


Algorithme k-means - choix du nombre de clusters

Comment choisir le nombre de clusters ?

Première idée : utiliser le critère $W_{SS}(C)$

- "total within sum of squares"

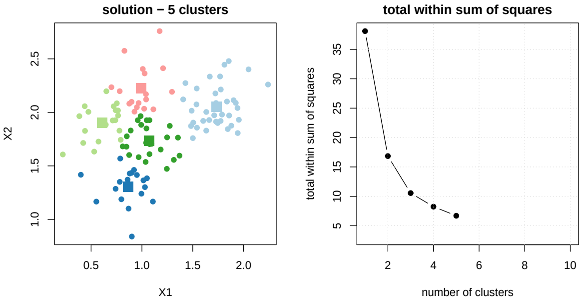


Algorithme k-means - choix du nombre de clusters

Comment choisir le nombre de clusters ?

Première idée : utiliser le critère $W_{SS}(C)$

- "total within sum of squares"

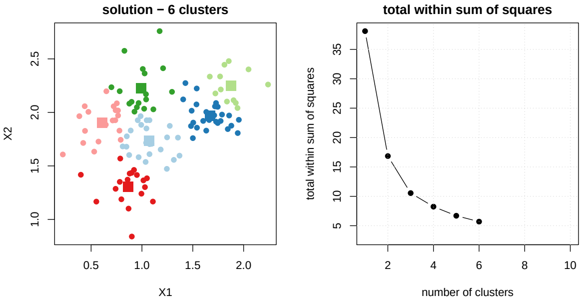


Algorithme k-means - choix du nombre de clusters

Comment choisir le nombre de clusters ?

Première idée : utiliser le critère $W_{SS}(C)$

- "total within sum of squares"

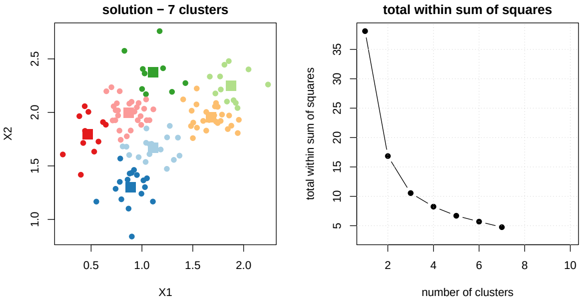


Algorithme *k*-means - choix du nombre de clusters

Comment choisir le nombre de clusters ?

Première idée : utiliser le critère $W_{SS}(C)$

- "total within sum of squares"

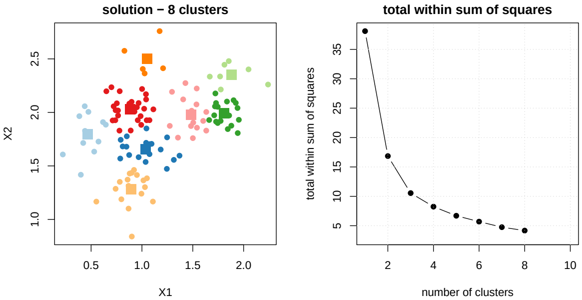


Algorithme k-means - choix du nombre de clusters

Comment choisir le nombre de clusters ?

Première idée : utiliser le critère $W_{SS}(C)$

- "total within sum of squares"

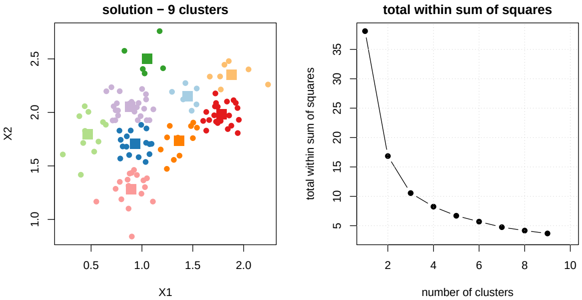


Algorithme k-means - choix du nombre de clusters

Comment choisir le nombre de clusters ?

Première idée : utiliser le critère $W_{SS}(C)$

- "total within sum of squares"

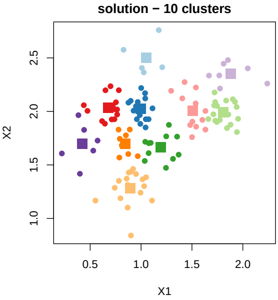


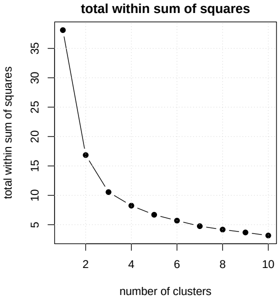
Algorithme k-means - choix du nombre de clusters

Comment choisir le nombre de clusters ?

Première idée : utiliser le critère $W_{SS}(C)$

- "total within sum of squares"





Algorithme k-means - choix du nombre de clusters

Le critère $W_{SS}(C)$ de "within sum of squares" décroît avec K
$\Rightarrow$ pas un bon critère pour choisir le nombre de clusters
$\Rightarrow$ permet de comparer différents clustering à K fixé.

Algorithme k-means - choix du nombre de clusters

Le critère $W_{SS}(C)$ de "within sum of squares" décroît avec K

$\Rightarrow$ pas un bon critère pour choisir le nombre de clusters

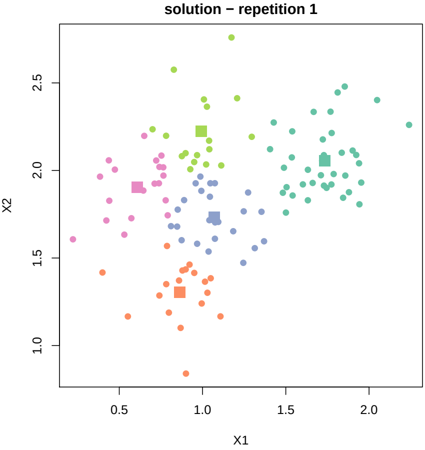
$\Rightarrow$ permet de comparer différents clustering à K fixé.

(au moins) 2 solutions :

1. approche empirique basée sur le graphe $W_{SS}(C) = f(K)$
   - méthode "du coude" (elbow method)
   - $\sim$ choisir le nombre de composantes dans une ACP
2. se placer dans un cadre probabiliste
   - compromis vraisemblance / complexité
   - voir mélanges de gaussiennes.

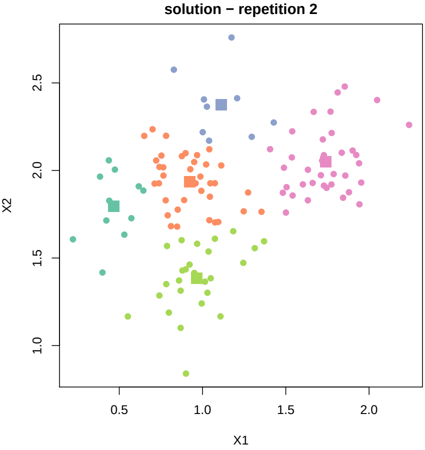
Algorithme k-means - stabilité de la solution

Instabilité du résultat :



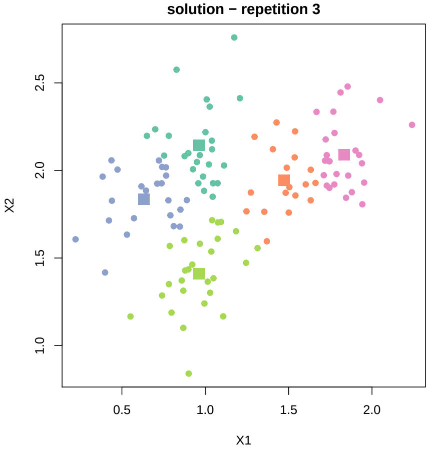
Algorithme k-means - stabilité de la solution

Instabilité du résultat :



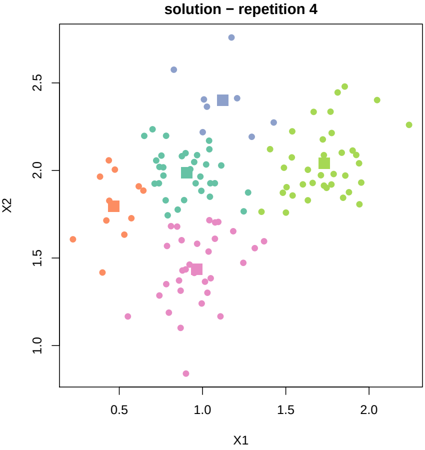
Algorithme k-means - stabilité de la solution

Instabilité du résultat :



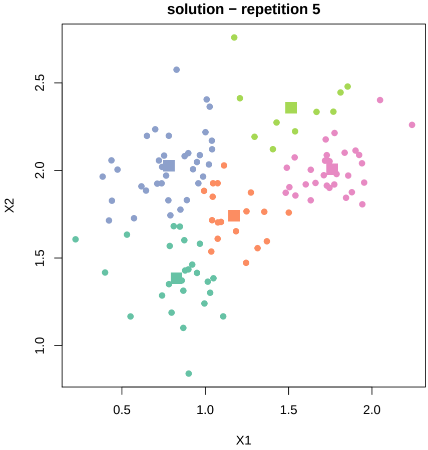
Algorithme k-means - stabilité de la solution

Instabilité du résultat :



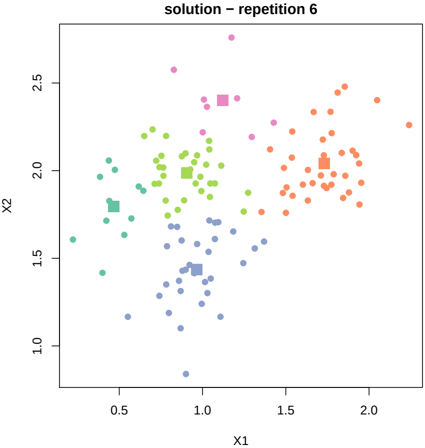
Algorithme k-means - stabilité de la solution

Instabilité du résultat :



Algorithme k-means - stabilité de la solution

Instabilité du résultat :



Algorithme k-means - stabilité de la solution

Instabilité : pourquoi ?

- critère objectif non convexe : solution = minimum local
- aspect stochastique lié à l'initialisation de l'algorithme
- (+ identifiabilité de la solution)
  - $\sim$ la couleur des clusters

Instabilité : que faire ?

1. répéter la procédure
2. choisir la meilleure solution en terme de W_{SS}
   - (NB : on est à K fixé)

Algorithme k-means - critère de distance

k-means : intrinsèquement lié à la distance Euclidienne

- ré-écriture & simplification du problème via centroïdes

Algorithme k-means - critère de distance

k-means : intrinsèquement lié à la distance Euclidienne

- ré-écriture & simplification du problème via centroïdes

Pourquoi considérer d'autres critères de distance ?

- données vectorielles mais autre notion de similarité
- travailler à partir d'une matrice de distance / similarité

Algorithme k-means - critère de distance

k-means : intrinsèquement lié à la distance Euclidienne

- ré-écriture & simplification du problème via centroïdes

Pourquoi considérer d'autres critères de distance ?

- données vectorielles mais autre notion de similarité
- travailler à partir d'une matrice de distance / similarité

Une solution : algorithme des k-médoïdes

- médoïde : **instance** représentative d'un cluster
- (e.g., avec la plus petite distance moyenne aux autres)

Algorithme k-means - critère de distance

k-means : intrinsèquement lié à la distance Euclidienne

- ré-écriture & simplification du problème via centroïdes

Pourquoi considérer d'autres critères de distance ?

- données vectorielles mais autre notion de similarité
- travailler à partir d'une matrice de distance / similarité

Une solution : algorithme des k-médoïdes

- médoïde : **instance** représentative d'un cluster
- (e.g., avec la plus petite distance moyenne aux autres)

$\Rightarrow$ médoïde = observation ; centroïde = moyenne
$\Rightarrow$ nécessaire pour travailler sur une matrice de distance

Algorithme des k-médoïdes

1. Initialisation : affecter les points aléatoirement aux clusters
2. Itérer la procédure suivante :
   2.1 calculer les medoïdes des clusters :

$$m_k^{(t)} = \arg \min_{i:C(i)^{(t)}=k} \sum_{j:C(j)^{(t)}=k} d(x_i, x_j)$$

   2.2 affecter chaque point au cluster dont le médoïde est le plus proche :

$$C(i)^{(t+1)} = \arg \min_{k=1,\ldots K} d(x_i, m_k^{(t)})$$

Critère d'arrêt :

- convergence : les affectations ne changent plus
  - i.e., $C(i)^{(t+1)} = C(i)^{(t)}, \ \ \forall i = 1, ..., n.$
- nombre maximum d'itérations atteint

Algorithme des k-médoïdes

k-means vs k-médoïdes : identification des représentants :

- k-means :

$$\mu_k = \frac{1}{n_k} \sum_{i:C(i)=k} x_i$$

- k-médoïdes :

$$m_k = \arg \min_{i:C(i)=k} \sum_{j:C(j)=k} d(x_i, x_j)$$

$\Rightarrow$ identification des médoïdes = un problème d'optimisation

$\Rightarrow$ plus complexe (et long) à mettre en oeuvre

Algorithme k-means - mise en oeuvre

Algorithme k-means : très simple à programmer.

Fonction `kmeans` dans `R`.

Algorithme k-médoïdes : fonction `pam` du package `cluster`.

Voir TP.

Algorithme k-means et traitement d'image

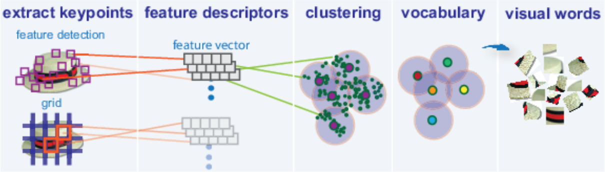
Segmentation / compression :

Figure: Image tirée de Bishop (2006)

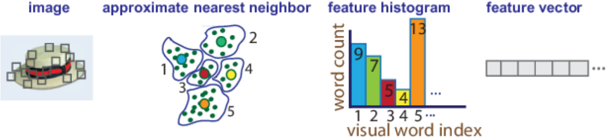
Algorithme k-means et traitement d'image

Catégorisation d'images et représentation mots visuels [1] :

- apprentissage des "mots visuels" :



- représentation d'une image :



- (puis approches supervisées)

1. images tirées de la documentation de Matlab

Modèles de mélange

k-means & modèles de mélange

Avantages k-means :

- simple à mettre en oeuvre
- utile pour de nombreuses applications

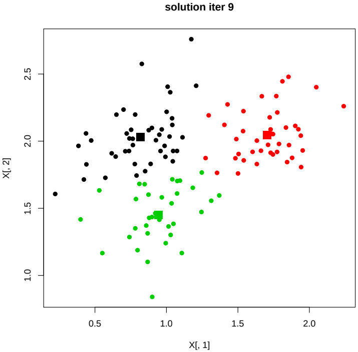
Limites :

- clustering "hard" : pas d'incertitude dans l'affectation
- pas de critère objectif pour choisir K

Modèles de mélange : extension probabiliste de k-means

- (par mélange de gaussiennes)

Clustering "hard" ?



$\Rightarrow$ Pas de distinction entre les points :

- proches du centre du cluster
- proches de la frontière avec les autres clusters

Modèles de mélange

Modèle de mélange : modèle probabiliste

- prenant en compte des sous-populations...
- ...sans qu'elles soient spécifiées à l'avance

Modèles de mélange

Modèle de mélange : modèle probabiliste

- prenant en compte des sous-populations...
- ...sans qu'elles soient spécifiées à l'avance

Formellement :

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

où :

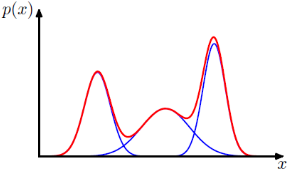
1. $\{f(x; k)\}$ sont les K densités
2. $\{\pi_k\}$ sont les proportions de mélange : $\sum_{k=1}^{K} \pi_k = 1$

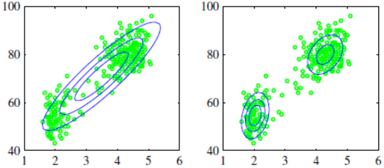
Mélanges de gaussiennes

Mélange de Gaussiennes à K composantes :

$$p(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x; \mu_k, \sigma_k), \qquad \text{si } x \in \mathbb{R}$$

$$p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \mathcal{MN}(\mathbf{x}; \boldsymbol{\mu}_k, \Sigma_k), \qquad \text{si } \mathbf{x} \in \mathbb{R}^p, p > 1.$$





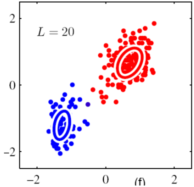
Modèle de mélange & clustering

Variables aléatoires mises en jeu (pour chaque observation) :

1. $\mathbf{x} \in \mathbb{R}^p$: observations (multivariées)
2. $\{z_k\}_{k=1,\dots,K}$: appartenance aux clusters
   - $z_k \in \{0, 1\}$

Hypothèse : distributions gaussiennes au sein des clusters :

$$p(\mathbf{x}|z_1 = 1) = \mathcal{MN}(\mathbf{x}; \boldsymbol{\mu}_1, \Sigma_1)$$
$$p(\mathbf{x}|z_2 = 1) = \mathcal{MN}(\mathbf{x}; \boldsymbol{\mu}_2, \Sigma_2)$$



$\Rightarrow$ Clustering : basé sur $p(z_k = 1|\mathbf{x})$.

Modèle de mélange & clustering

On peut ré-écrire le modèle général comme :

$$\begin{aligned} p(\mathbf{x}) &= \sum_{k=1}^{K} p(\mathbf{x}; z_k = 1) \\ &= \sum_{k=1}^{K} p(z_k = 1) p(\mathbf{x}|z_k = 1) \\ &= \sum_{k=1}^{K} \pi_k \mathcal{MN}(\mathbf{x}; \boldsymbol{\mu}_k, \Sigma_k) \end{aligned}$$

- $p(\mathbf{x}|z_k = 1)$: densité au sein des classes
- $p(z_k = 1)$: loi a priori
- $p(\mathbf{x}; z_k = 1)$: loi jointe

$\Rightarrow$ on passe de $p(\mathbf{x}|z_k = 1)$ à $p(z_k = 1|\mathbf{x})$ par la loi de Bayes.

Intermède...

Formule de Bayes :

$$p(A = a_1 | B = b) = \frac{p(A = a_1; B = b)}{p(B = b)}$$

Intermède...

Formule de Bayes :

$$p(A = a_1 | B = b) = \frac{p(A = a_1 ; B = b)}{p(B = b)}$$

$$= \frac{p(B = b | A = a_1) p(A = a_1)}{p(B = b)}$$

Intermède...

Formule de Bayes :

$$
\begin{aligned}
p(A = a_1 | B = b) &= \frac{p(A = a_1; B = b)}{p(B = b)} \\
&= \frac{p(B = b | A = a_1) p(A = a_1)}{p(B = b)} \\
&= \frac{p(B = b | A = a_1) p(A = a_1)}{\sum_i p(B = b; A = a_i)}
\end{aligned}
$$

Intermède...

Formule de Bayes :

$$p(A = a_1 | B = b) = \frac{p(A = a_1; B = b)}{p(B = b)}$$

$$= \frac{p(B = b | A = a_1) p(A = a_1)}{p(B = b)}$$

$$= \frac{p(B = b | A = a_1) p(A = a_1)}{\sum_i p(B = b; A = a_i)}$$

$$= \frac{p(B = b | A = a_1) p(A = a_1)}{\sum_i p(B = b | A = a_i) p(A = a_i)}$$

Intermède...

Formule de Bayes :

$$p(A = a_1 | B = b) = \frac{p(A = a_1; B = b)}{p(B = b)}$$

$$= \frac{p(B = b | A = a_1) p(A = a_1)}{p(B = b)}$$

$$= \frac{p(B = b | A = a_1) p(A = a_1)}{\sum_i p(B = b; A = a_i)}$$

$$= \frac{p(B = b | A = a_1) p(A = a_1)}{\sum_i p(B = b | A = a_i) p(A = a_i)}$$

$\Rightarrow$ on peut "inverser" $p(B|A)$ en $p(A|B)$ (et vice versa)

Modèle de mélange & clustering

Grâce à la formule de Bayes, on peut écrire :

$$p(z_k = 1|\mathbf{x}) = \frac{p(\mathbf{x}; z_k = 1)}{p(\mathbf{x})}$$

Modèle de mélange & clustering

Grâce à la formule de Bayes, on peut écrire :

$$p(z_k = 1|\mathbf{x}) = \frac{p(\mathbf{x}; z_k = 1)}{p(\mathbf{x})}$$
$$= \frac{p(\mathbf{x}|z_k = 1)p(z_k = 1)}{p(\mathbf{x})}$$

Modèle de mélange & clustering

Grâce à la formule de Bayes, on peut écrire :

$$
\begin{aligned}
p(z_k = 1|\mathbf{x}) &= \frac{p(\mathbf{x}; z_k = 1)}{p(\mathbf{x})} \\
&= \frac{p(\mathbf{x}|z_k = 1)p(z_k = 1)}{p(\mathbf{x})} \\
&= \frac{p(\mathbf{x}|z_k = 1)p(z_k = 1)}{\sum_{i=1}^{K} p(\mathbf{x}|z_i = 1)p(z_i = 1)}
\end{aligned}
$$

Modèle de mélange & clustering

Grâce à la formule de Bayes, on peut écrire :

$$\begin{aligned} p(z_k = 1|\mathbf{x}) &= \frac{p(\mathbf{x}; z_k = 1)}{p(\mathbf{x})} \\ &= \frac{p(\mathbf{x}|z_k = 1)p(z_k = 1)}{p(\mathbf{x})} \\ &= \frac{p(\mathbf{x}|z_k = 1)p(z_k = 1)}{\sum_{i=1}^{K} p(\mathbf{x}|z_i = 1)p(z_i = 1)} \end{aligned}$$

⇒ affection probabiliste d'une observation $\mathbf{x}$ aux clusters :

$$p(z_k = 1|\mathbf{x})$$

Modèle de mélange & clustering

Grâce à la formule de Bayes, on peut écrire :

$$
\begin{aligned}
p(z_k = 1|\mathbf{x}) &= \frac{p(\mathbf{x}; z_k = 1)}{p(\mathbf{x})} \\
&= \frac{p(\mathbf{x}|z_k = 1)p(z_k = 1)}{p(\mathbf{x})} \\
&= \frac{p(\mathbf{x}|z_k = 1)p(z_k = 1)}{\sum_{i=1}^{K} p(\mathbf{x}|z_i = 1)p(z_i = 1)}
\end{aligned}
$$

$\Rightarrow$ affection probabiliste d'une observation $\mathbf{x}$ aux clusters :

$$p(z_k = 1|\mathbf{x})$$

Remarques :

- on connaît $p(\mathbf{x}|z_k = 1)$ et $p(z_k = 1)$
- on vérifie facilement que $\sum_{k=1}^{K} p(z_k = 1|\mathbf{x}) = 1$

Clustering "hard" vs clustering "soft"

Algorithme *k*-means :

$$C(\mathbf{x}) = \arg \min_{k=1,\ldots,K} ||\mathbf{x} - \mu_k||^2$$

Clustering "hard" vs clustering "soft"

Algorithme k-means :

$$C(\mathbf{x}) = \arg\min_{k=1,\ldots,K} ||\mathbf{x} - \mu_k||^2$$

Peut s'écrire comme :

$$p(z_i = 1|\mathbf{x}) = \mathbb{1}\left(i = \arg\min_{k=1,\ldots,K} ||\mathbf{x} - \mu_k||^2\right)$$

Clustering "hard" vs clustering "soft"

Algorithme k-means :

$$C(\mathbf{x}) = \arg\min_{k=1,\dots,K} ||\mathbf{x} - \mu_k||^2$$

Peut s'écrire comme :

$$p(z_i = 1|\mathbf{x}) = \mathbb{1}\left(i = \arg\min_{k=1,\dots,K} ||\mathbf{x} - \mu_k||^2\right)$$

soit :

$$p(z_i = 1|\mathbf{x}) = \begin{cases} 1 & \text{si } \mu_i \text{ est le plus proche centroïde} \\ 0 & \text{sinon} \end{cases}$$

$\Rightarrow$ affection "hard" au cluster le plus proche

Mélanges de Gaussiennes - estimation

Pour une observation $\mathbf{x}$ on a donc le modèle :

$$\begin{aligned} p(\mathbf{x}) &= \sum_{k=1}^{K} \pi_k \mathcal{MN}(\mathbf{x}; \boldsymbol{\mu}_k, \Sigma_k) \\ &= \sum_{k=1}^{K} p(z_k = 1) p(\mathbf{x} | z_k = 1) \end{aligned}$$

Mélanges de Gaussiennes - estimation

Pour une observation $\mathbf{x}$ on a donc le modèle :

$$
\begin{aligned}
p(\mathbf{x}) &= \sum_{k=1}^{K} \pi_k \mathcal{MN}(\mathbf{x}; \boldsymbol{\mu}_k, \Sigma_k) \\
&= \sum_{k=1}^{K} p(z_k = 1) p(\mathbf{x} | z_k = 1)
\end{aligned}
$$

$\Rightarrow$ Paramètres à estimer : $\left\{\pi_k, \mu_k, \Sigma_k\right\}$, pour $k = 1, ..., K$.

Mélanges de Gaussiennes - estimation

Pour une observation $\mathbf{x}$ on a donc le modèle :

$$\begin{aligned} p(\mathbf{x}) &= \sum_{k=1}^{K} \pi_k \mathcal{MN}(\mathbf{x}; \boldsymbol{\mu}_k, \Sigma_k) \\ &= \sum_{k=1}^{K} p(z_k = 1) p(\mathbf{x} | z_k = 1) \end{aligned}$$

$\Rightarrow$ Paramètres à estimer : $\left\{\pi_k, \mu_k, \Sigma_k\right\}$, pour $k = 1, ..., K$.

Principe = maximiser la vraisemblance :

$$p(D) = \prod_{i=1}^{n} p(\mathbf{x}_i)$$

à partir du jeu de données $D = \{\mathbf{x}_i\},\ i = 1, ..., n$.

Mélanges de Gaussiennes - estimation

Vraisemblance : $p(D) = \prod_{i=1}^{n} p(\mathsf{x}_i)$, où $D = \{\mathsf{x}_i\},\ i = 1, ..., n.$

Mélanges de Gaussiennes - estimation

Vraisemblance : $p(D) = \prod_{i=1}^{n} p(\mathsf{x}_i)$, où $D = \{\mathsf{x}_i\},\ i = 1, ..., n$.

Log-vraisemblance :

$$\ln p(D) = \sum_{i=1}^{n} \ln p(\mathsf{x}_i)$$

Mélanges de Gaussiennes - estimation

Vraisemblance : $p(D) = \prod_{i=1}^{n} p(\mathrm{x}_i)$, où $D = \{\mathrm{x}_i\},\ i = 1, ..., n.$

Log-vraisemblance :

$$\begin{aligned} \ln p(D) &= \sum_{i=1}^{n} \ln p(\mathrm{x}_i) \\ &= \sum_{i=1}^{n} \ln \Big(\sum_{k=1}^{K} p(\mathrm{x}_i | z_k^{(i)} = 1) p(z_k^{(i)} = 1) \Big) \end{aligned}$$

Mélanges de Gaussiennes - estimation

Vraisemblance : $p(D) = \prod_{i=1}^{n} p(\mathbf{x}_i)$, où $D = \{\mathbf{x}_i\}, \; i = 1, ..., n.$

Log-vraisemblance :

$$\begin{aligned}
\ln p(D) &= \sum_{i=1}^{n} \ln p(\mathbf{x}_i) \\
&= \sum_{i=1}^{n} \ln \Big(\sum_{k=1}^{K} p(\mathbf{x}_i | z_k^{(i)} = 1) p(z_k^{(i)} = 1) \Big) \\
&= \sum_{i=1}^{n} \ln \Big(\sum_{k=1}^{K} \mathcal{MN}(\mathbf{x}_i; \mu_k, \Sigma_k) \pi_k \Big)
\end{aligned}$$

Mélanges de Gaussiennes - estimation

Vraisemblance : $p(D) = \prod_{i=1}^{n} p(\mathbf{x}_i)$, où $D = \{\mathbf{x}_i\},\ i = 1, ..., n.$

Log-vraisemblance :

$$\begin{aligned}
\ln p(D) &= \sum_{i=1}^{n} \ln p(\mathbf{x}_i) \\
&= \sum_{i=1}^{n} \ln \left(\sum_{k=1}^{K} p(\mathbf{x}_i | z_k^{(i)} = 1) p(z_k^{(i)} = 1) \right) \\
&= \sum_{i=1}^{n} \ln \left(\sum_{k=1}^{K} \mathcal{MN}(\mathbf{x}_i; \mu_k, \Sigma_k) \pi_k \right)
\end{aligned}$$

$\Rightarrow$ à maximiser selon $\left\{\pi_k, \mu_k, \Sigma_k\right\}$:

1. on dérive $\ln p(D)$ selon $\left\{\pi_k, \mu_k, \Sigma_k\right\}$
2. on annule les dérivées

Mélanges de Gaussiennes - estimation

On obtient les équations suivantes :

$$\pi_k = \frac{1}{n} \sum_{i=1}^{n} p(z_k^{(i)} = 1 | \mathbf{x}_i)) = \frac{n'_k}{n} \tag{1}$$

$$\mu_k = \frac{1}{n'_k} \sum_{i=1}^{n} p(z_k^{(i)} = 1 | \mathbf{x}_i)\, \mathbf{x}_i \tag{2}$$

$$\Sigma_k = \frac{1}{n'_k} \sum_{i=1}^{n} p(z_k^{(i)} = 1 | \mathbf{x}_i)\, (\mathbf{x}_i - \mu_k)(\mathbf{x}_i - \mu_k)^T \tag{3}$$

$\Rightarrow$ chaque observation $\mathbf{x}_i$ contribue à l'estimation de l'ensemble des $\{\pi_k, \mu_k, \Sigma_k\}$ à hauteur de $p(z_k^{(i)} = 1 | \mathbf{x}_i))$.

Mélanges de Gaussiennes - estimation

Interprétation : différence avec k-means

$$\pi_k = \frac{1}{n}\sum_{i=1}^{n} p(z_k^{(i)} = 1|\mathbf{x}_i)) = \frac{n'_k}{n}$$

$$\neq \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}(C(i) = k) = \frac{n_k}{n}$$

$\Rightarrow$ k-means

Mélanges de Gaussiennes - estimation

Interprétation : différence avec k-means

$$\pi_k = \frac{1}{n}\sum_{i=1}^{n} p(z_k^{(i)} = 1|\mathbf{x}_i)) = \frac{n'_k}{n}$$

$$\neq \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}(C(i) = k) = \frac{n_k}{n} \qquad \Rightarrow k\text{-means}$$

$$\mu_k = \frac{1}{n'_k}\sum_{i=1}^{n} p(z_k^{(i)} = 1|\mathbf{x}_i)\,\mathbf{x}_i$$

$$\neq \frac{1}{n_k}\sum_{i:C(i)=k} \mathbf{x}_i = \frac{1}{n_k}\sum_{i=1}^{n} \mathbb{1}(C(i) = k)\mathbf{x}_i \quad \Rightarrow k\text{-means}$$

Mélanges de Gaussiennes - estimation

Interprétation : différence avec k-means

$$\pi_k = \frac{1}{n}\sum_{i=1}^{n} p(z_k^{(i)} = 1|\mathbf{x}_i)) = \frac{n'_k}{n}$$

$$\neq \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}(C(i) = k) = \frac{n_k}{n} \qquad \Rightarrow k\text{-means}$$

$$\mu_k = \frac{1}{n'_k}\sum_{i=1}^{n} p(z_k^{(i)} = 1|\mathbf{x}_i)\,\mathbf{x}_i$$

$$\neq \frac{1}{n_k}\sum_{i:C(i)=k} \mathbf{x}_i = \frac{1}{n_k}\sum_{i=1}^{n} \mathbb{1}(C(i) = k)\mathbf{x}_i \qquad \Rightarrow k\text{-means}$$

$\Rightarrow$ chaque observation $\mathbf{x}_i$ contribue à l'estimation de l'ensemble des $\{\pi_k, \mu_k, \Sigma_k\}$ à hauteur de $p(z_k^{(i)} = 1|\mathbf{x}_i))$.

Mélanges de Gaussiennes - estimation

Remarque : de la même manière

$$\Sigma_k = E\left[(\mathbf{x} - \mu_k)(\mathbf{x} - \mu_k)^T\right]$$

⇒ estimateur "classique" :

$$\Sigma_k = \frac{1}{n_k} \sum_{i=1}^{n_k} (\mathbf{x}_i - \mu_k)(\mathbf{x}_i - \mu_k)^T$$

⇒ modèle de mélange :

$$\Sigma_k = \frac{1}{n'_k} \sum_{i=1}^{n} p(z_k^{(i)} = 1|\mathbf{x}_i)\,(\mathbf{x}_i - \mu_k)(\mathbf{x}_i - \mu_k)^T$$

(NB : Σ_k pas utilisé dans k-means)

Mélanges de Gaussiennes - estimation

Problème : les équations suivantes :

$$\pi_k = \frac{1}{n}\sum_{i=1}^{n} p(z_k^{(i)} = 1|\mathbf{x}_i)) = \frac{n'_k}{n}$$

$$\mu_k = \frac{1}{n'_k}\sum_{i=1}^{n} p(z_k^{(i)} = 1|\mathbf{x}_i)\,\mathbf{x}_i$$

$$\Sigma_k = \frac{1}{n'_k}\sum_{i=1}^{n} p(z_k^{(i)} = 1|\mathbf{x}_i)\,(\mathbf{x}_i - \mu_k)(\mathbf{x}_i - \mu_k)^T$$

et $p(z_k^{(i)} = 1|\mathbf{x}_i) = \dfrac{\mathcal{MN}(\mathbf{x}_i; \mu_k, \Sigma_k)\pi_k}{\sum_{j=1}^{K}\mathcal{MN}(\mathbf{x}_i; \mu_j, \Sigma_j)\pi_j}$

sont liées (couplées) :

- il nous faut $p(z_k^{(i)} = 1|\mathbf{x}_i)$ pour avoir (π_k, μ_k, Σ_k)
- il nous faut $\{\pi_k, \mu_k, \Sigma_k\}$ pour avoir $p(z_k^{(i)} = 1|\mathbf{x}_i)$

Mélanges de Gaussiennes - estimation

Solution = procédure itérative :

1. estimer $p(z_k^{(i)} = 1|\mathbf{x}_i)$

$$p(z_k^{(i)} = 1|\mathbf{x}_i) = \frac{\mathcal{MN}(\mathbf{x}_i;\mu_k,\Sigma_k)\pi_k}{\sum_{j=1}^{K} \mathcal{MN}(\mathbf{x}_i;\mu_j,\Sigma_j)\pi_j}$$

2. mettre à jour les estimations de $\{\pi_k, \mu_k, \Sigma_k\}$

$$\pi_k = \frac{1}{n}\sum_{i=1}^{n} p(z_k^{(i)} = 1|\mathbf{x}_i)) = \frac{n'_k}{n}$$

$$\mu_k = \frac{1}{n'_k}\sum_{i=1}^{n} p(z_k^{(i)} = 1|\mathbf{x}_i)\,\mathbf{x}_i$$

$$\Sigma_k = \frac{1}{n'_k}\sum_{i=1}^{n} p(z_k^{(i)} = 1|\mathbf{x}_i)\,(\mathbf{x}_i - \mu_k)(\mathbf{x}_i - \mu_k)^T$$

Mélanges de Gaussiennes - estimation

Solution = procédure itérative :

1. estimer $p(z_k^{(i)} = 1|\mathbf{x}_i)$

$$p(z_k^{(i)} = 1|\mathbf{x}_i) = \frac{\mathcal{MN}(\mathbf{x}_i;\mu_k,\Sigma_k)\pi_k}{\sum_{j=1}^{K}\mathcal{MN}(\mathbf{x}_i;\mu_j,\Sigma_j)\pi_j}$$

2. mettre à jour les estimations de $\left\{\pi_k, \mu_k, \Sigma_k\right\}$

$$\pi_k = \frac{1}{n}\sum_{i=1}^{n} p(z_k^{(i)} = 1|\mathbf{x}_i)) = \frac{n'_k}{n}$$

$$\mu_k = \frac{1}{n'_k}\sum_{i=1}^{n} p(z_k^{(i)} = 1|\mathbf{x}_i)\,\mathbf{x}_i$$

$$\Sigma_k = \frac{1}{n'_k}\sum_{i=1}^{n} p(z_k^{(i)} = 1|\mathbf{x}_i)\,(\mathbf{x}_i - \mu_k)(\mathbf{x}_i - \mu_k)^T$$

⇒ un cas particulier de l'algorithme EM

- étape 1 = "Expectation", étape 2 = "Maximization"

⇒ ∼ une version probabiliste de k-means

Mélanges de Gaussiennes - illustration

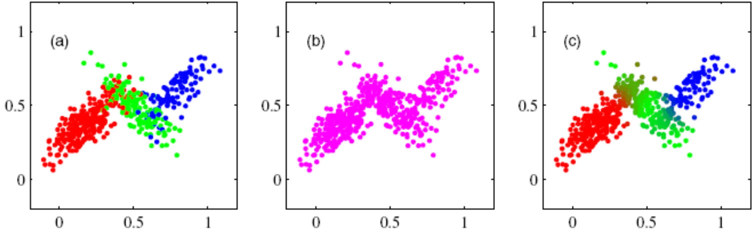
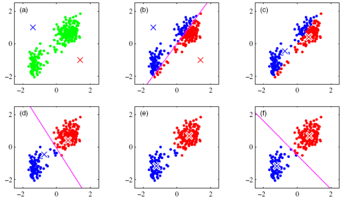


Figure: Figure tirée de Bishop (2006)

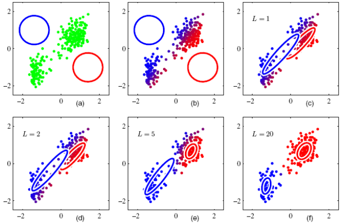
- Gauche : simulation des données (3 clusters)
- Milieu : données d'entrée (cadre non supervisé)
- Droite : résultat (couleur en fonction de $p(z_k = 1|\mathbf{x})$)

Algorithme EM - illustration

k-means :



Mélange de Gaussiennes & EM :



(images tirées de Bishop (2006))

Choisir le nombre de clusters

La vraisemblance augmente avec K :

$$p(D) = \prod_{i=1}^{n} p(\mathbf{x}_i) = \prod_{i=1}^{n} \sum_{k=1}^{K} \pi_k \mathcal{MN}(\mathbf{x}_i; \mu_k, \Sigma_k),$$

- (de la même manière que l'erreur diminue avec k-means)

Choisir le nombre de clusters

La vraisemblance augmente avec K :

$$p(D) = \prod_{i=1}^{n} p(\mathbf{x}_i) = \prod_{i=1}^{n} \sum_{k=1}^{K} \pi_k \mathcal{MN}(\mathbf{x}_i; \mu_k, \Sigma_k),$$

- (de la même manière que l'erreur diminue avec k-means)

Critères de sélection de modèles = compromis :

1. **qualité** du modèle : $p(D)$ = maximum de vraisemblance
2. **complexité** du modèle : m = nombre de paramètres

Choisir le nombre de clusters

La vraisemblance augmente avec K :

$$p(D) = \prod_{i=1}^{n} p(\mathbf{x}_i) = \prod_{i=1}^{n} \sum_{k=1}^{K} \pi_k \mathcal{MN}(\mathbf{x}_i; \mu_k, \Sigma_k),$$

- (de la même manière que l'erreur diminue avec k-means)

Critères de sélection de modèles = compromis :

1. **qualité** du modèle : $p(D)$ = maximum de vraisemblance
2. **complexité** du modèle : m = nombre de paramètres

Classiquement :

- Bayesian Information Criterion $= -2 \ln p(D) + \ln(n) m$
- Akaike Information Criterion $= -2 \ln p(D) + 2m$

$\Rightarrow$ à minimiser

Choisir le nombre de clusters

Bayesian Information Criterion :

$$\text{BIC} = -2 \ln p(D) + \ln(n) m$$

⇒ à minimiser : meilleur modèle = BIC le plus faible

- à vraisemblance égale on préfère le modèle le plus simple
- paramètre supplémentaire si améliore $\ln p(D)$ de $\ln(n)/2$

Choisir le nombre de clusters

Bayesian Information Criterion :

$$\text{BIC} = -2 \ln p(D) + \ln(n) m$$

$\Rightarrow$ à minimiser : meilleur modèle = BIC le plus faible

- à vraisemblance égale on préfère le modèle le plus simple
- paramètre supplémentaire si améliore $\ln p(D)$ de $\ln(n)/2$

Stratégie :

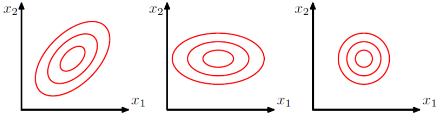
1. faire varier K
2. caculer la vraisemblance par EM
3. calculer le BIC
4. retenir le modèle de BIC minimum

Choisir le nombre de clusters

Attention : c'est bien le nombre de paramètres qui entre en jeu dans le BIC, et pas le nombre de clusters

$\Rightarrow$ 1 cluster = 1 Gaussienne multivariée :

- p paramètres pour μ_k
- de 1 à $\frac{p(p+1)}{2}$ paramètres pour Σ_k
- $+$ 1 paramètre pour π_k



(a) (b) (c)

$\Rightarrow$ en pratique, considérer différentes structures de covariance

Mélanges de Gaussiennes - mise en oeuvre

Package `Mclust` de `R`.

La fonction `mclust` :

1. estime les différents modèles
   - nombre de clusters
   - structure de covariance
2. calcule les BIC associés
3. sélectionne le meilleur modèle

La fonction `plot.mclust` propose différentes sorties graphiques

- BIC, clustering, incertitude...

Conclusion

Conclusion

Clustering et critères de dispersion

Algorithme k-means

- distance Euclidienne, diagramme de Voronoï
- algorithme itératif, minimum local
- k-means vs k-médoïdes
- applications en imagerie

Modèles de mélange

- extension probabiliste de k-means, "soft" clustering
- maximum de vraisemblance, algorithme EM
- critère BIC et sélection de modèle
- **NB : vont bien au delà du clustering**

TP : k-means (+ modèles de mélanges).

Références

Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.