Classification

Dhafer Malouche

ESSAI
dhafer.malouche@me.com

2011-2012

Plan

Objectif :

- $E = \{w_1, w_2, \ldots, w_n\}$ n individus.
- $x^1, \ldots, x^d$ d variables observées, quantitatives et/ou qualitatives
- $x_i^j = x^j(w_i)$ l'observation de w_i % à x^j.
- Question : Existe-t-il des groupes homogènes d'individus naturellement constitués dans E, et combien ?

Définition d'une distance

On définit sur E l'application suivante

$$d : E \times E \to \mathbb{R}$$

qui peut vérifier

1. Symétrie : $d(w_i, w_{i'}) = d(w_{i'}, w_i)$ pour tout $i, i' = 1, \ldots, n$.
2. Positivité : $d(w_i, w_{i'}) \geq 0$, pour tout $i, i' = 1, \ldots, n$.
3. Inégalité triangulaire

$$d(w_i, w_{i'}) \leq d(w_i, w_{i''}) + d(w_{i''}, w_{i'}), \quad \forall\ w_i,\ w_{i'},\ w_{i''}.$$

4. Réfléxivité : $d(w_i, w_{i'}) = 0 \iff w_i = w_{i'}$.

d est une distance si 1 et 2 sont satisfaites, et d est une distance métrique si de plus 3 et 4 sont satisfaites.

Distances pour des observations quantitatives

- On suppose que nos données sont exclusivement composées d'observations de variables quantitatives.
- La distance euclidienne (norme L_2) : si $x_i = x(w_i) = (x_i^1, \ldots, x_i^d)'$ et $x_{i'} = x(w_{i'}) = (x_{i'}^1, \ldots, x_{i'}^d)'$, alors

$$d(w_i, w_{i'}) = \sqrt{\sum_{j=1}^{d} (x_i^j - x_{i'}^j)^2}.$$

- Propriétés : Clusters hypersphériques, clusters composés avec la d.e. sont invariants par rotation et translation, transformations linéaires peuvent causer des distortions dans les clusters, variables avec une grande variance ou des grandes valeurs peuvent dominer les autres variables.
- Nécessité d'une normalisation des données.

D'autres types de distances :

- Minkowski :

$$d_p(w_i, w_{i'}) = \left(\sum_{j=1}^{d} (x_i^j - x_{i'}^j)^p \right)^{1/p}.$$

  $p = 2$ on obtient la distance euclidienne.

- Distance L_∞ :

$$d_\infty(w_i, w_{i'}) = \max_{1 \leq j \leq d} |x_i^j - x_{i'}^j|.$$

- Distance de Mahalanobis

$$d(w_i, w_{i'}) = (x_i - x_{i'})' S^{-1} (x_i - x_{i'})$$

  où S est matrice variance des données. Les clusters ont une forme hyperellipsoidale et invariants par une transformation linéaire. Si $S = I$, alors elle est équivalente à une distance euclidienne.

Algorithme ascendant (agrégatif) hiérarchique

1. $k = n$ le nombre de classes
2. Chaque classes contient une observation : n classes $C^1, \ldots, C^n$.
3. Calculer la matrice de distances (entre classes).
4. Considérer les deux classes C^i et C^j telles que

$$d(C^i, C^j) = \min d(C^{i'}, C^{j'})$$

.

5. Agréger C^i et C^j dans une seule classe C^{ij}. Donc $k = k - 1$.
6. $k = 1$? Si oui STOP, sinon, aller à 3.

Méthodes d'agrégations

$$d(C^l, C^{ij}) = \alpha_i d(C^l, C^i) + \alpha_j d(C^l, C^j) + \beta d(C^i, C^j) + \gamma \left| d(C^l, C^i) - d(C^l, C^j) \right|$$

Algorithme de classification	α_i	α_j	β	γ
Single Linkage	$1/2$	$1/2$	0	$-1/2$
Complete Linkage	$1/2$	$1/2$	0	$1/2$
Centroïd Linkage	$\frac{n_i}{n_i+n_j}$	$\frac{n_j}{n_i+n_j}$	$-\frac{n_i n_j}{n_i+n_j}$	0
Méthode de Ward	$\frac{n_i+n_l}{n_i+n_j+n_l}$	$\frac{n_j+n_l}{n_i+n_j+n_l}$	$-\frac{n_l}{n_i+n_j+n_l}$	0

Méthodes d'agrégations

- *Single Linkage* : $d(C^l, C^{ij}) = \min(d(C^l, C^i), d(C^l, C^j))$
- *Complete Linkage* : $d(C^l, C^{ij}) = \max(d(C^l, C^i), d(C^l, C^j))$
- *Centroïd Linkage* : $d(C^l, C^{ij}) = d(m_l, m_{ij})$ où m_l et m_{ij} sont resp. les centres de gravité de C^l et C^{ij}.
- Méthode de Ward : elle a pour objectif de minimiser l'accroissement dans la variance intra-classe :

$$E = \sum_{l=1}^{k} \sum_{x_i \in C^l} d(x_i, m_l)^2.$$

Algorithme descendant (divisif) hiérarchique

1. k=0, Toutes sont dans la même classe C_0
2. Considérer C_i la classe de diamètre maximale ($d(C_i) = \max_{w,w' \in C_i} d(w, w')$).Toutes sont dans la même classe C_i,
3. Introduire $C_j = \emptyset$.
   1. Pour chaque $w \in C_i$, calculer

$$d\left(w, C_i \setminus \{w\}\right) = \frac{1}{|C_i| - 1} \sum_{w' \in C_i \setminus \{w\}} d(w, w').$$

   2. Calculer la différence $\Delta(w, C_i, C_j) = d\left(w, C_i \setminus \{w\}\right) - d(w, C_j)$
4. Si $w_m = \text{argmax}_{w \in C_i} d\left(w, C_i \setminus \{w\}\right)$ et $\Delta(w_m, C_i, C_j) > 0$ alors $w_m \ni C_j$
5. si $k < n$ alors $k = k + 1$ revient à 1/ sinon STOP.

- **Données** Soient $w_1, \ldots, w_n$ n individus pour lesquels on a observé d variables quantitatives $x^1, \ldots, x^d$. Donc chaque w_i est caractérisé par un vecteur d'observations $x_i' = (x_i^1, \ldots, x_i^d) \in \mathbb{R}^d$.
- **Objectif** A partir d'une fonction critère J donnée, partitionner $w^1, \ldots, w_n$ en K classes (K est fixé d'avance) minimisant ou maximisant la fonction J
- Méthode exhaustive : calculer pour toute les partitions possible C, le critère $J(C)$ et la solution est alors

$$C_{sol} = \text{argmax} J(C).$$

- Solution impossible si n devient grand. En effet le nombre de partitions possible est égal à

$$P(n, K) = \frac{1}{K!} \frac{m = 1}{K} (-1)^{K-m} C_K^m m^N.$$

Pour $n = 30$, et $K = 3$, $P(n, K)$ est de l'ordre de 2×10^{14}.

$K-$means

- Critère à minimiser : la somme des carrées des erreurs (SSE) :

$$J(C) = \sum_{c \in C} \sum_{w \in c} d(w, m(c))$$

où $m(c)$ est le centre de gravité de c.

- Algorithme :
  1. Initialise $K-$ centres $m_1, \ldots, m_K$ et
  2. Affecter chaque individu vers la classe la plus proche $C = \{c_1, \ldots, c_K\}$:

$$w \in c_l \text{ si } d(w, m_l) = \min_{l'=1,\ldots,K} d(w, m_{l'}).$$

  3. Re-calculer les centres de gravité des nouvelles classes,
  4. Répéter 2 et 3 jusqu'à convergence.

Mesure de la validité d'une classification

Deux types de mesures :

- Indices de validité interne : à partir d'un résultat de classification et de l'information intrinsèque dans la base de données on mesure la qualité du résultat :
  - indice de connectivité
  - indice de silhouette
  - indice de *Dunn*
- Mesure de la stabilité de la classification : elle mesure la consistence du résultat d'une classification en la comparant avec une classification obtenue en supprimant chaque fois une colonne (une variable) :
  - proportion moyenne de chevauchement
  - distance moyenne
  - distance moyenne entre les moyennes

Notations

Soit

- n le nombre des observations,
- d le nombre des variables,
- $x_1, \ldots, x_n$ les vecteurs d'observations à classer,
- $\mathcal{C} = \{C_1, \ldots, C_k\}$ une classification à tester,
- $C(i)$ est la classe contenant x_i, i.e., $C(i) \ni x_i$.

Pour tout x_i, et $j \in \{1, \ldots, n\} \setminus \{i\}$: $x_i^{(j)}$ est le $j^{\text{ième}}$ voisin le plus proche de x_i :

$$d(x_i, x_{(1)}) \leq d(x_i, x_{(2)}) \leq \ldots d(x_i, x_{(j)}) \leq \ldots \leq d(x_i, x_{(n-1)})$$

On pose pour tout $i, j \in \{1, \ldots, n\}$

$$\delta(x_i, x_i^{(j)}) = \begin{cases} 0 & \text{si } x_i,\ x_i^{(j)} \in C(i) \\ 1/j & \text{sinon} \end{cases}$$

```
> data(ruspini)
> X=ruspini[1:5,]
> dist(X)
```

```
          1         2         3         4
2 10.049876
3  8.485281  6.403124
4 24.515301 14.560220 18.027756
5  9.848858 16.124515 10.440307 28.284271
```

Donc $x_1^{(1)} = x_3$, $x_1^{(2)} = x_5$, $x_1^{(3)} = x_2$, $x_1^{(4)} = x_4$.
Si x_1, $x_3 \in C_1$ et x_2, x_4, $x_5 \in C_2$ alors
$\delta(x_1, x_3) = \delta(x_1, x_1^{(1)}) = 0$ et $\delta(x_1, x_1^{(4)}) = 1/4$.

Exercice

Comparer $\delta(x_i, x_i^{(j)})$ et $\delta(x_j, x_j^{(i)})$.

Indices de Connectivité

Pour une classification $\mathcal{C}$,

$$\text{Conn}(\mathcal{C}) = \sum_{i,j=1,\ldots,n} \delta(x_i, x_i^{(j)})$$

Conn($\mathcal{C}$) prend ses valeurs dans $[0, +\infty[$ et c'est un cœfficient qui doit être minimisée :

Conn($\mathcal{C}$) proche de zéro $\iff$ une bonne connectivité à l'intérieure des classes.

`R : clValid package, connectivity`

Largeur de la silhouette

- C'est la moyenne de toutes les valeurs de la silhouette.
- Valeur de la silhouette pour une observation x_i :

$$S(i) = \frac{b_i - a_i}{\max(b_i, a_i)}$$

où

- a_i est la distance moyenne entre x_i et tous les autres $x_j \in C(i)$.

$$a_i = \frac{1}{n(C(i))} \sum_{x \in C(i)} d(x_i, x).$$

- b_i est la distance moyenne entre x_i et les observations dans la classe la pus proche de x_i :

$$b_i = \min_{C \in \mathcal{C} \setminus C(i)} \frac{1}{n(C)} \sum_{x \in C} d(x_i, x).$$

Largeur de la silhouette

- Définie par

$$S(\mathcal{C}) = \frac{1}{n} \sum_i S(i)$$

- La largeur de la silhouette est un nombre qui est toujours compris entre -1 et $+1$.
- S doit être maximisée.
- une valeur de $S(i)$ négative indique que cet individu n'est pas dans "la bonne classe" et il pourrait être déplacé vers la classe la plus proche.
- `R, package cluster, silhouette`

Indice de *Dunn*

L'indice de *Dunn* est le ratio entre la plus petite distance entre les individus n'appartenant pas aux mêmes classes par rapport à la plus grande distance intra-classe :

$$D(\mathcal{C}) = \frac{\min_{C,\, C' \in \mathcal{C},\, C \neq C'} \left(\min_{x \in C,\, x' \in C'} d(x, x') \right)}{\max_{C \in \mathcal{C}} \operatorname{diam}(C)}$$

où

$$\operatorname{diam}(C) = \max_{x,\, x' \in C} d(x, x')$$

L'indice de *Dunn* prend que des valeurs strictement positives et il doit être maximisé.

`R, package clValid, dunn`

Travaux pratiques

- Installer les packages, `clValid`, `cluster` et `clusterGeneration` et charger les dans votre environnement `R`
- A l'aide de la commande simClustDesign générer un jeu de données ayant déjà la répartition réelle des individus. On commencera par un jeu de données de dimension 2, on fera varier l'argument `sepVal` mesurant le degré de "séparabilité" entre les classes.
- Explorer les commandes `plot2DProjection` et `nearestNeighborSepVal`
- Créer des différents jeux de données, calculer les indices internes connectivité, silhouette et Dunn et conclure.

Proportion moyenne de chevauchement

- *Average Proportion of Non-overlap* (APN)
- l'APN mesure la proportion moyenne des individus qui changent de classes quand on effectue une classification en supprimant une variable de la base des données.
- *Calcul* :
  - Soit $\mathcal{C}^0 = \{C_1^0, \ldots, C_K^0\}$ une classification obtenue en utilisant toute la base de données.
    Pour un individu x_i, on note $C(i)^0$ la classe contenant x_i.
  - L'APN se calcule de la façon suivante

$$\text{APN}(K) = \frac{1}{nd} \sum_{i=1}^{n} \sum_{l=1}^{d} \left(1 - \frac{|C(i)^l \cap C(i)^0|}{|C(i)^0|} \right)$$

  - $\text{APN}(K) \in [0, 1]$, des valeurs proches de zéro indiquent une forte consistence ou stabilité dans la classification originale $\mathcal{C}^0$.

Proportion moyenne de chevauchement

- *Average Proportion of Non-overlap* (APN)
- l'APN mesure la proportion moyenne des individus qui changent de classes quand on effectue une classification en supprimant une variable de la base des données.
- *Calcul* :
  - Soit $\mathcal{C}^l = \{C_1^l, \ldots, C_K^l\}$ une classification obtenue en utilisant la base privée de la $l-$ ième variable.
    Pour un individu x_i, on note $C(i)^l$ la classe contenant x_i.
  - L'APN se calcule de la façon suivante

$$\text{APN}(K) = \frac{1}{nd} \sum_{i=1}^{n} \sum_{l=1}^{d} \left(1 - \frac{|C(i)^l \cap C(i)^0|}{|C(i)^0|} \right)$$

  - $\text{APN}(K) \in [0, 1]$, des valeurs proches de zéro indiquent une forte consistence ou stabilité dans la classification originale $\mathcal{C}^0$.

Trois autres indices de stabilité

- Distance moyenne, *Average Distance* (AD)

$$AD(K) = \frac{1}{nd} \sum_{i=1}^{n} \sum_{l=1}^{d} \frac{1}{|C(i)^l| \, |C(i)^0|} \left(\sum_{x \in C(i)^0, x' \in C(i)^l} \text{dist}(x, x') \right)$$

  $\text{AD}(K) \in [0, +\infty[$ et des valeurs de $\text{AD}(K)$ proches de zéro indique une forte consistence.

- Distance moyenne entre les moyennes, *Average Distance between means* (ADM)

$$ADM(K) = \frac{1}{nd} \sum_{i=1}^{n} \sum_{l=1}^{d} \text{dist} \left(\overline{x}_{C(i)^l}, \overline{x}_{C(i)^0} \right)$$

  $\text{ADM}(K) \in [0, +\infty[$ et des valeurs de $\text{ADM}(K)$ proches de zéro indique une forte consistence.

Trois autres indices de stabilité

- Facteur de mérite, *Figure of Merit* (FOM) : pour chaque $l \in \{1, \ldots, d\}$, on calcule d'abord

$$\mathrm{FOM}(l, K)^2 = \frac{1}{n} \sum_{C^l \in \mathcal{C}^l} \sum_{x \in C^l} \mathrm{dist}(x, \overline{x}_{C^l})$$

et l'indice FOM est alors égal à

$$\mathrm{FOM}(K) = \sqrt{\frac{n}{n-K} \sum_{l=1}^{d} \mathrm{FOM}(l, K)}$$

$\mathrm{FOM}(K) \in [0, +\infty[$ et des valeurs de $\mathrm{FOM}(K)$ proches de zéro indique une forte consistence.

Travaux pratiques

1. Utilisant la commande `simClustDesign` du package `clusterGeneration` un jeu de données avec un nombre de classes fixe et un nombre de variables égal à 5 (dont une est une variable “bruitée”).
2. L’objectif de ce TP est d’utiliser les indices de validation définis ci-dessus pour retrouver le nombre de classes après avoir effectué une classification selon les méthodes : hiérarchiques, k-means et diana.
3. Explorer la commande `clValid` du package `clValid` qui permet de calculer à la fois les indices internes et de stabilités.
4. Représenter le résultat dans un graphique, donner la méthode adéquate pour retrouver la classification réelle des individus.

Indices externes

- $\mathcal{P}$ une partition existante de l'ensemble des individus E, $\mathcal{P} = \{G_1, \ldots, G_r\}$.
- $\mathcal{C} = \{C_1, \ldots, C_k\}$ une classification obtenue à l'aide d'un algorithme de classification (*hiérarchique*, *kmeans*...)
- Objectif : comparer la classification $\mathcal{C}$ et la partition $\mathcal{P}$
- Définition de quelques indices de validité externes : package `clv` sour `R`

Calcul des indices externes

Soient deux individus x_i et $x_{i'}$. Quatre cas sont possibles

Cas 1 $\exists C \in \mathcal{C}$ et $\exists G \in \mathcal{P}$ tel que

$$x_i,\, x_{i'} \in C \text{ et } x_i,\, x_{i'} \in G$$

Cas 2 $\exists C \in \mathcal{C}$ et $\exists G \neq G' \in \mathcal{P}$ tel que

$$x_i,\, x_{i'} \in C \text{ et } x_i \in G,\, x_{i'} \in G'$$

Cas 3 $\exists C \neq C' \in \mathcal{C}$ et $\exists G \in \mathcal{P}$ tel que

$$x_i \in C,\, x_{i'} \in C' \text{ et } x_i,\, x_{i'} \in G$$

Cas 4 $\exists C \neq C' \in \mathcal{C}$ et $\exists G \neq G' \in \mathcal{P}$ tel que

$$x_i \in C,\, x_{i'} \in C' \text{ et } x_i \in G,\, x_{i'} \in G'$$

Calcul des indices externes

Notons par $\mathcal{A}_l$ l'ensemble des paires $(x_i, x_{i'})$ vérifiant le cas l et par $a_l = |\mathcal{A}_l|$. Trois indices :

- Indice de Rand

$$\text{Rand} = \frac{a_1 + a_4}{a_1 + a_2 + a_3 + a_4}$$

- Indice de Jaccard

$$\text{Jaccard} = \frac{a_1}{a_1 + a_2 + a_3}.$$

- Indice de Folkes et Mallows

$$\text{FM} = \sqrt{\frac{a_1}{a_1 + a_3} \frac{a_1}{a_1 + a_2}}$$

Usage sous R

1. Charger `clv`
2. Effectuer une classification à l'aide d'une des algorithmes de classification.
3. Utiliser la commande std.ext pour calculer a_1, a_2, a_3 et a_4
4. Utiliser les commandes `clv.Rand`, `clv.Jaccard` et `clv.Folkes.Mallows` pour calculer respectivement les indices de Rand, Jaccard et FM. L'argument de ses principales fonctions est l'objet obtenu à l'aide de `std.ext`

Application :

- Appliquer toutes ces commandes sur des données simulées
- Créer une procédure permettant de déterminer les variables minimales construisant les classe.

Travaux pratiques sous R

1. Construire une fonction R permettant de calculer tous les indices de validité étudiés.
2. Construire une fonction R permettant de permuter la position d'un individu d'une classe à l'autre.
3. Construire une fonction R permettant de détecter la stabilité des individus dans une classification.
4. Créer des fonctions R permettant d'illustrer tous les résultats des fonctions précédentes à l'aide de graphiques.

l'EM-algorithme

- Soit $\underline{X} = (X_1, \ldots, X_n) \sim g(x \mid \theta)$ un $n-$échantillon de vecteurs aléatoire dans $\mathbb{R}^d$. On suppose

$$g(x \mid \theta) = \int f(x, z \mid \theta) dz$$

- Objectif : calculer $\widehat{\theta} = \text{argmax}_\theta L(\underline{x} \mid \theta) = \text{argmax}_\theta \prod_{i=1}^{n} g(x_i \mid \theta)$ où $\underline{x} = (x_1, \ldots, x_n)$ est une réalisation de $\underline{X}$.
- On pose $(X, Z) \sim f(x, z \mid \theta)$, et la densité de $Z \mid \theta, X = x$ est

$$k(z \mid \theta, x) = \frac{f(x, z \mid \theta)}{g(x \mid \theta)}.$$

l'EM-algorithme

- Donc

$$\log g(x \mid \theta) = \log f(x, z \mid \theta) - \log k(z \mid \theta, x)$$

- Pour un θ_0 fixé,

$$\begin{aligned} \log g(x \mid \theta) k(z \mid \theta_0, x) &= \log f(x, z \mid \theta) k(z \mid \theta_0, x) \\ &\quad - \log k(z \mid \theta, x) k(z \mid \theta_0, x) \quad (1) \end{aligned}$$

- En intégrant (1) par rapport à z, on obtient

$$\log g(x \mid \theta) = \mathbb{E}_{\theta_0}(\log f(x, Z \mid \theta)) - \mathbb{E}_{\theta_0}(\log k(Z \mid x, \theta))$$

l'EM-algorithme

- La fonction vraisemblance s'écrit alors

$$\begin{aligned}\log L(\underline{x} \mid \theta) &= \sum_{i=1}^{n} \log g(x_i \mid \theta) \\ &= \underbrace{\mathbb{E}_{\theta_0}(\log L^c(\underline{x}, Z \mid \theta))}_{Q(\theta|\underline{x},\theta_0)} - \sum_{i=1}^{n} \mathbb{E}_{\theta_0}(\log k(Z \mid x_i, \theta))\end{aligned}$$

  où L^c est appelée la vraisemblance *complète*.

- Ainsi, dans la recherche du maximum de $\log L(\underline{x} \mid \theta)$, dans l'EM-algorithme on maximize, d'une façon itérative, $Q(\theta \mid \underline{x}, \theta_0)$.

l'EM-algorithme

1. Considérons une valeur initiale $\widehat{\theta}_{(0)}$
2. Calculer l'espérance *(E−step)*

$$Q(\theta \mid \underline{x}, \widehat{\theta}_{(m)}) = \mathbb{E}_{\widehat{\theta}_{(m)}}(\log L^c(\underline{x}, Z \mid \theta))$$

où la moyenne a été calculée par rapport à $k(z \mid \widehat{\theta}_{(m)}, x)$ et $m = 0$

3. Maximiser la fonction $Q(\theta \mid \underline{x}, \widehat{\theta}_{(m)})$ en θ *(M−step)* :

$$\widehat{\theta}_{(m+1)} = \text{argmax}_\theta Q(\theta \mid \underline{x}, \widehat{\theta}_{(m)})$$

et $m = m + 1$.

4. Répéter les étapes 2-3 jusqu'à le point fix soit atteint : $\widehat{\theta}_{(m+1)} = \widehat{\theta}_{(m)}$

l'EM-algorithme, dans la classification

- On suppose que les données $\underline{x} = (x_1, \ldots, x_n) \in (\mathbb{R}^d)^n$ est un $n-$échantillon qu'on suppose issu d'une loi de probabilité

$$g(x \mid \theta) = \sum_{l=1}^{k} \alpha_l g_l(x \mid \theta_l)$$

où $\alpha_l \in]0,1[$, $l = 1, \ldots, k$ et $\sum_{l=1}^{k} \alpha_l = 1$ et $g_l(x \mid \theta_l)$ est la densité d'une loi Gaussienne multivariée :

$$f_l(x \mid \theta_l) = \frac{1}{(2\pi)^{d/2}|\Sigma_l|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu_l)'\Sigma_l^{-1}(x-\mu_l)\right)$$

- Donc $X \sim g(x \mid \theta)$ et Z est la variable à valeurs dans $\{1, \ldots, k\}$, où k est la nombre de classes et

$$\mathbb{P}(Z = l) = \alpha_l, \quad \forall\, l = 1, \ldots, k.$$

Mise en œuvre sous `R` : le package `mclust`

- Description détaillé du package dans
  - C. Fraley and A. E. Raftery (2006). MCLUST Version 3 for R : Normal Mixture Modeling and Model-Based Clustering, Technical Report no. 504, Department of Statistics, University of Washington
  - C. Fraley and A. E. Raftery (2002). Model-based clustering, discriminant analysis, and density estimation. Journal of the American Statistical Association 97 :611-631
- Utiliser la commande `Mclust`,
- `Mclust` applique l'EM-algorithme (avec des différentes paramétrisations), et utilise le BIC comme critère de sélection du modèle : chercher le modèle M_k correspondant au BIC maximal.

$$\text{BIC} = 2 \log \text{Vraisemblance} - \text{Nbre paramètres} \times \log(n)$$

Paramétrisations...

identifier	Model	HC	EM	Distribution	Volume	Shape	Orientation
E		•	•	(univariate)	equal		
V		•	•	(univariate)	variable		
EII	λI	•	•	Spherical	equal	equal	NA
VII	$\lambda_k I$	•	•	Spherical	variable	equal	NA
EEI	λA		•	Diagonal	equal	equal	coordinate axes
VEI	$\lambda_k A$		•	Diagonal	variable	equal	coordinate axes
EVI	λA_k		•	Diagonal	equal	variable	coordinate axes
VVI	$\lambda_k A_k$		•	Diagonal	variable	variable	coordinate axes
EEE	$\lambda D A D^T$	•	•	Ellipsoidal	equal	equal	equal
EEV	$\lambda D_k A D_k^T$		•	Ellipsoidal	equal	equal	variable
VEV	$\lambda_k D_k A D_k^T$		•	Ellipsoidal	variable	equal	variable
VVV	$\lambda_k D_k A_k D_k^T$	•	•	Ellipsoidal	variable	variable	variable