

## Mixture model

- $\log(P(\cdot)) = \log \left( \prod_{i=1}^n P(x_i | z_{ik}=1) \right)$   
 $= \sum_{i=1}^n \log \left( \prod_{k=1}^K P(x_i | z_{ik}=1)^{z_{ik}} \right) \leq -\sum_{i=1}^n \sum_{k=1}^K z_{ik} \log \frac{P(z_{ik}=1)}{P(x_i | z_{ik}=1)}$   
 $= \sum_{i=1}^n \sum_{k=1}^K z_{ik} \log \left( \prod_{k=1}^K P(x_i | \theta_k) \right)$   
 $\quad \quad \quad \mathbb{P}(z=k)$   
 $\Leftrightarrow \text{vector de proportions}$

- $t_{ik} = E_{z_{ik}|x_i} [z_{ik} = 1]$   
 $= \mathbb{P}(z_{ik} = 1 | x_i)$   
 $= \frac{\mathbb{P}(z_{ik} = 1 \cap x_i)}{\mathbb{P}(x_i)}$   
 $= \frac{\mathbb{P}(z_{ik} = 1) \mathbb{P}(x_i | z_{ik} = 1)}{\sum_{l=1}^K \mathbb{P}(z_{il} = 1) \mathbb{P}(x_i | z_{il} = 1)} \quad \leftarrow \text{Bayes}$   
 $= \frac{\pi_k f(x_i; \theta_k)}{\sum_{l=1}^K \pi_l f(x_i; \theta_l)}$

$$\begin{aligned}
 \bullet Q(\theta^* | \theta) &= E_{Z|X} [\log(P_\theta(X \cap Z) | X \cap \theta^*)] \\
 &= E_{Z|X} \left[ \sum_{i=1}^n \sum_{k=1}^K z_{ik} \log(\pi_k P_\theta(z_i; | z_{ik}=1)) \right] \\
 &= \sum_{i=1}^n \sum_{k=1}^K \underbrace{E_{Z|X} [z_{ik}=1 \cap \theta^*]}_{\epsilon_{ik}^q} \log(\pi_k P(x_i; | z_{ik}=1)) \\
 &= \sum_{i=1}^n \sum_{k=1}^K \epsilon_{ik}^q \log(\pi_k P(x_i; | \theta^*))
 \end{aligned}$$

### Cours :

→ la mixture d'une distribution uniforme n'est pas identifiable

CAR

$$\text{si } f(x) = \frac{1}{2} (f_1(x) + f_2(x))$$

$$f_1(x) = U_{[0,2]} \text{ et } f_2(x) = U_{[1,3]}$$

ou

$$f_1(x) = U_{[0,3]} \text{ et } f_2(x) = U_{[-1,2]}$$

→ le k-means algorithme assume que

↳ les proportions de la mixture sont toutes égales

↳ la matrice de cov est proportionnelle à l'identité

↳ les composantes de la mixture sont normales

→ l'EN-algo calcule l'espérance de la log-vrai  
complétée

→ Le M-step de l'EN-algo maximise l'espérance de la log vraisemblance sachant X.

- à l'ETI algo trouve un maximum local
- une finite mixture est une densité
- BIC : utilisé pour le model de sélection
- ICL : utilisé pour déterminer une entroy.

### ACP

l'ACP cherche de nouvelles variables qui sont des combinaisons linéaires des anciennes variables et qui sont linéairement II

- Calcul du pourcentage de la variance expliquée par les  $m^{es}$  axes

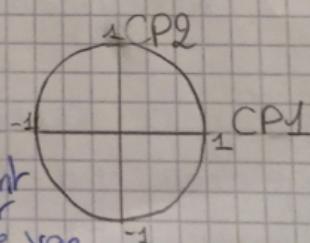
Note:  $S: \sum_i^n VP_i = nb de var$

$$1. S = \sum_{i=1}^n VP_i \quad n > m \Leftrightarrow var sont réduits$$

$$2. \% axe m = \frac{\sum_{i=1}^m VP_i}{S}$$

- Plan principal du cercle de CORRELATION  $\Rightarrow$

Bien le tracer complètement  
il suffit de faire une flèche partant de  $(0; 0)$  vers  $(CP_1; CP_2)$  pour chaque var



## Cours

- ACP est utilisée pour identifier de nouvelles variables indépendantes
- des composantes principales sont un nouveau système de coordonnées
- La variance totale du jeu de données =
  - ↳ à la somme des var
  - ↳ au nb de var si elles sont réduites
- les axes principaux sont:
  - ↳ des vecteurs couvrant un sous-espace où les données sont projetées
  - ↳ les vecteurs propres de la matrice Var/cov emploie
  - ↳ des vecteurs orthogonaux
- Le pourcentage de la variance projetée sert à choisir le nb de composantes
- ACP vise à trouver une représentation des données
- ACP :
  - ↳ permet une décomp de la variance totale
  - ↳ maximise la variance projetée
  - ↳ minimise la variance dans l'espace orthogonal à l'espace de projection
- Il n'y a aucune condition à respecter v.à.v du nb de var / individu (et inversement)

## Classical Scaling

PCoA

- $D^2(i, j) = \sum_{k=1}^p (x_{ik} - x_{jk})^2$   
 $\uparrow$   
 Matrice de distance  
 $= \sum_{k=1}^p (x_{ik}^2 + x_{jk}^2 - 2x_{ik}x_{jk})$   
 $= \mathbb{1}(n)\text{diag}(XX^T) - 2XX^T + \text{diag}(XX^T)\mathbb{1}(n)$

- Soit  $J$  la matrice de centrage

$$J = I - \frac{1}{n}\mathbb{1}(n,n)$$

- Centrage de  $X$  en colonnes

$$\Rightarrow JX = X - \frac{1}{n}\mathbb{1}(n,n)X$$

- Centrage de  $X$  en ligne

$$\Rightarrow X^t J = X^t - \frac{1}{n} X^t \mathbb{1}(n,n)$$

PB: On connaît  $D^2$  et on veut  $X$ :

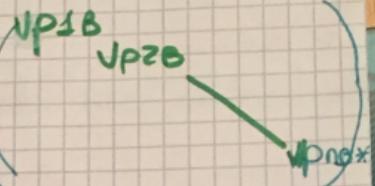
$$\begin{aligned} \Rightarrow -\frac{1}{2}JD^2J &= -\frac{1}{2}J\text{diag}(XX^t)\mathbb{1}(n)J - \\ &\quad \frac{1}{2}J\mathbb{1}(n)\text{diag}(XX^t)J + \\ &\quad JXX^tJ \\ &= XX^t \end{aligned}$$

- On trouve les coordonnées multidimensionnel scaling à partir de  $B = XX^t$  avec la décomposition spectrale de  $B$

Rappel:

$\text{decomp}(B) = \text{Vecteurs propres de } B$

$t$  (vect. prop. de  $B$ )



$$B = C \Lambda C^t$$

$$= (C \Lambda^{1/2})(C \Lambda^{1/2})^t$$

$$= X X^t$$

## Algorithme

$\Delta^2$ : la matrice du carré des dissimilarités (= distance)

① Calcul de  $\Delta^2$

② Calcul de  $B_{\Delta^2} = -\frac{1}{2} J \Delta^2 J$

③ Décomposition spectrale de  $B_{\Delta^2} = C \Lambda C^t$

④ On choisit  $m$  la dimension de représentation de la solution

\*  $X = C^+ \Lambda_+^{1/2}$  a)  $\Lambda_+$  est la matrice des  $m$  plus grande vp classée par ordre décroissant sur le diag

b)  $C^+$  est la matrice des  $m$  vect. propre associé

c) La solution est \*

## k-means

### Algorithm:

1. Initialisation des centres aléatoirement

2. 1 Calcul, on choisit  $c_{ik}$  qui minimise  $I_w$

$$\forall k, \hat{\mu}_k = \underset{M_k}{\operatorname{argmin}} I_w \quad \text{variance intra-classes}$$

$$\hat{\mu} = \frac{1}{\sum c_{ik}} \sum_{i=1}^n c_{ik} \underline{x_i}$$

2. 2 On assigne chaque  $x_i$  à la classe dont il est le plus proche du centre

$$\hat{C} = \underset{C}{\operatorname{argmin}} I_w(C, \hat{\mu}_1, \dots, \hat{\mu}_k)$$

## k-medoid

### Algorithm:

1. Init : On sélectionne k medoid

2. Tant qu' on n'a pas convergence  
 { (=> "tant que le coût de la configuration" décroît)

Affecter chaque individu à la classe dont le médoid est le plus proche  
 Recalculer les médoides des classes  
 } à partir des individu rattachés

HAC

## Algorithm

Init: → chaque élément constitue une classe  
→ une matrice de dissimilarité est calculé entre toutes les classes (tous les éléments)

Tant que nb de classes > 1)

{  
→ on merge les 2 classes les plus proches de D

→ on calcule (selon la distance choisie)  
la distance entre la nouvelle classe et les autres

}

## Distances possibles

→ Single link (lien min):  $D(h, h') = \min(d(x_i, y_j) | x_i \in h, y_j \in h')$

→ Complete link (lien max):  $D(h, h') = \max(d(x_i, y_j) | x_i \in h, y_j \in h')$

→ Average link :  $D(h, h') = \frac{\sum_{i=1}^{n_h} \sum_{j=1}^{n_{h'}} d(x_i, x_j)}{n_h \cdot n_{h'}}$

→ Ward criterion:  $D(h, h') = \frac{n_h \cdot n_{h'}}{n_h + n_{h'}} \|m_h - m_{h'}\|^2$

↳ celle qu'on privilégie en général

## Spectral clustering

Le partitionnement spectral est une méthode de partitionnement en  $k$  groupes reposant sur la minimisation d'un critère de type "côute".

Côute normalisée bidirectionnelle

1. Calcul de la matrice de similitude  $W$  + calcul du degré de la matrice ( $D$ ) qui est diagonale avec  $D(i,i) = \sum_{j \in V} W(i,j)$

2. Résoudre  $(D - W)y = \lambda Dy$   
matrice  
laplacienne

3. Utiliser le vect prop avec la seconde plus petite vp pour (bipartitionner?) le graphe en 2.

## Kernel PCA

$$K(\underline{x}, \underline{x}') = \langle \underline{x}, \underline{x}' \rangle_{\mathbb{R}^d}$$

$$\text{i) } \langle \underline{x}, \underline{x}' \rangle = \langle \underline{x}', \underline{x} \rangle$$

$$\text{ii) } \sum_{ij} a_i a_j \langle \underline{x}_i, \underline{x}_j \rangle = \sum_{i=1}^n \sum_{j=1}^n \langle a_i \underline{x}_i, a_j \underline{x}_j \rangle \\ = \left\langle \sum_{i=1}^n a_i \underline{x}_i, \sum_{j=1}^n a_j \underline{x}_j \right\rangle$$

Algorithm:

1. On choisit un kernel  $K$
2. On construit la kernel matrice normalisée des données

$$\tilde{K} = K - \frac{2}{n} \mathbf{1}_{(n,n)} K + \frac{1}{n^2} \mathbf{1}_{(n,n)} K$$

3. On résout le vp pb:

$$\tilde{K} \underline{\alpha_i} = \lambda_i \underline{\alpha_i}$$

4. Pour chaque point (ancien et nouveau)  
on peut faire la projection:

$$\underline{v}_j = \sum_{i=1}^n \alpha_i x_i K(x, x_i) \quad j=1, \dots, d$$