

Formation Machine Learning

Apprentissage Non Supervisé

Bassem Ben Hamed

ENET'Com, Université de Sfax

17 février 2026

Plan de la présentation

1 Introduction à l'apprentissage non supervisé

2 Réduction de dimensionnalité

- PCA et SVD
- UMAP et t-SNE

3 Clustering

- Méthode K-means avec métriques
- Regroupement hiérarchique avec dendrogrammes
- GMM avec BIC et AIC
- DBSCAN avec paramètres

4 Études comparatives des 4 méthodes

Plan de la section

1 Introduction à l'apprentissage non supervisé

2 Réduction de dimensionnalité

- PCA et SVD
- UMAP et t-SNE

3 Clustering

- Méthode K-means avec métriques
- Regroupement hiérarchique avec dendrogrammes
- GMM avec BIC et AIC
- DBSCAN avec paramètres

4 Études comparatives des 4 méthodes

Qu'est-ce que l'apprentissage non supervisé ?

Définition

Méthodes d'analyse de données qui découvrent des **structures cachées** dans des données **non étiquetées**, sans supervision externe.

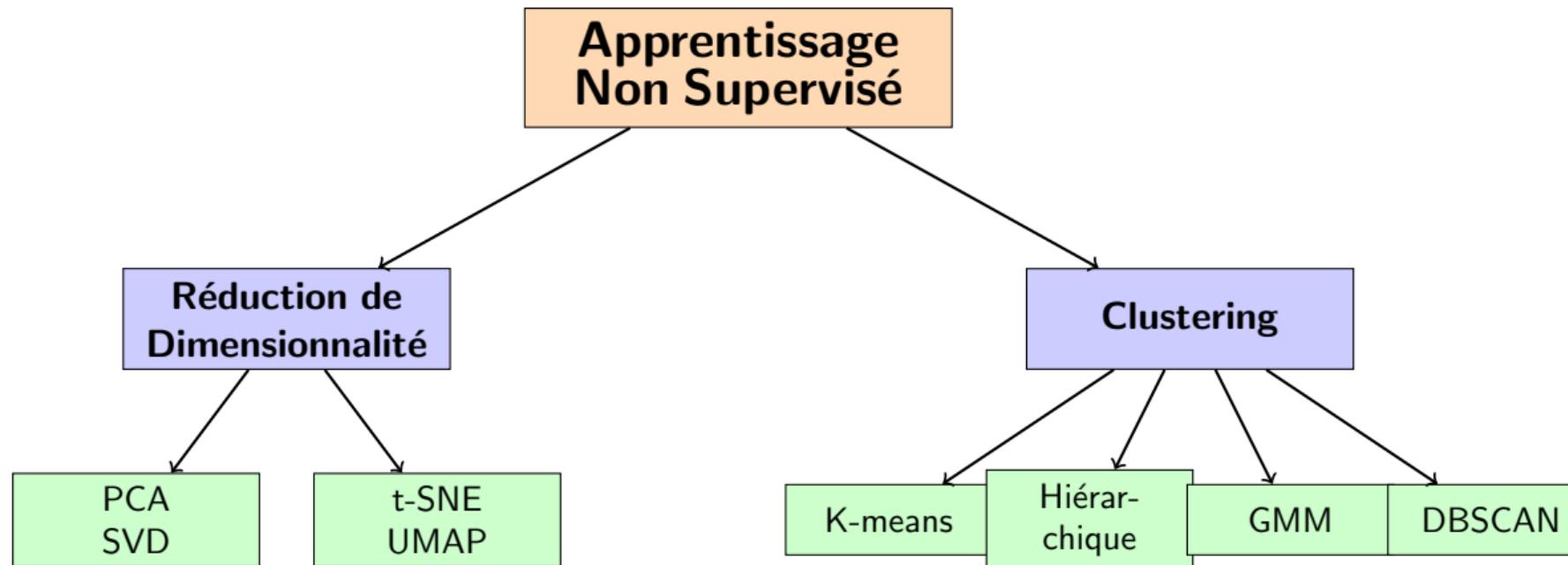
Caractéristiques :

- Pas de labels/cibles
- Exploration autonome
- Découverte de patterns
- Réduction de complexité

Objectifs :

- Réduire dimensionnalité
- Découvrir groupes naturels
- Déetecter anomalies
- Extraire caractéristiques

Taxonomie des méthodes



Plan de la section

1 Introduction à l'apprentissage non supervisé

2 Réduction de dimensionnalité

- PCA et SVD
- UMAP et t-SNE

3 Clustering

- Méthode K-means avec métriques
- Regroupement hiérarchique avec dendrogrammes
- GMM avec BIC et AIC
- DBSCAN avec paramètres

4 Études comparatives des 4 méthodes

Principe fondamental

Identifier les **directions de variance maximale** dans les données et projeter sur ces directions

Objectifs multiples :

- ① **Réduction** : d variables $\rightarrow k$ composantes ($k < d$)
- ② **Maximiser** la variance capturée
- ③ **Décorrélérer** les variables
- ④ **Éliminer** la redondance

Notations fondamentales :

- $X \in \mathbb{R}^{n \times d}$: matrice des données
- $\bar{x} = \frac{1}{n} \sum_{i=1}^n x_i$: vecteur des moyennes
- Σ : matrice de covariance

PCA : Centrage des données

Étape 1 - Centrage obligatoire :

$$X_c = X - \mathbf{1}_n \bar{x}^T$$

où $\mathbf{1}_n$ est un vecteur de n uns.

Pour chaque observation :

$$x_{c,i} = x_i - \bar{x}$$

Après centrage :

$$\frac{1}{n} \sum_{i=1}^n x_{c,i,j} = 0 \quad \forall j$$

La moyenne de chaque variable devient nulle.

Importance

Sans centrage, la première composante pointe généralement vers la moyenne globale et non vers la direction de variance maximale.

PCA : Matrice de covariance

Définition :

$$\Sigma = \frac{1}{n} X_c^T X_c \in \mathbb{R}^{d \times d}$$

Élément par élément :

$$\Sigma_{jk} = \text{Cov}(X_j, X_k) = \frac{1}{n} \sum_{i=1}^n x_{c,ij} \cdot x_{c,ik}$$

Propriétés fondamentales :

- Symétrique : $\Sigma = \Sigma^T$
- Semi-définie positive : $\forall v \in \mathbb{R}^d, v^T \Sigma v \geq 0$
- Diagonale : $\Sigma_{jj} = \text{Var}(X_j)$
- Hors-diagonale : $\Sigma_{jk} = \text{Cov}(X_j, X_k)$ pour $j \neq k$

Trace (variance totale) :

$$\text{tr}(\Sigma) = \sum_{j=1}^d \Sigma_{jj} = \sum_{j=1}^d \text{Var}(X_j)$$

PCA : Décomposition spectrale

Théorème spectral :

$$\Sigma = V \Lambda V^T$$

où :

- $V = [v_1, v_2, \dots, v_d]$: matrice orthogonale des vecteurs propres
- $V^T V = I$ (orthonormalité)
- $\Lambda = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_d)$: valeurs propres
- Convention : $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_d \geq 0$

Équation caractéristique :

$$\Sigma v_j = \lambda_j v_j$$

Interprétation géométrique :

- v_j : direction de la j -ème composante principale
- λ_j : variance capturée par la j -ème composante
- Les vecteurs propres forment une base orthonormée

PCA : Problème d'optimisation

Première composante principale :

$$\max_{w_1} \text{Var}(X_c w_1) = \max_{w_1} w_1^T \Sigma w_1$$

sous contrainte $\|w_1\| = 1$

Lagrangien :

$$\mathcal{L}(w_1, \lambda) = w_1^T \Sigma w_1 - \lambda(w_1^T w_1 - 1)$$

Condition d'optimalité :

$$\frac{\partial \mathcal{L}}{\partial w_1} = 2\Sigma w_1 - 2\lambda w_1 = 0 \quad \Rightarrow \quad \Sigma w_1 = \lambda w_1$$

Solution optimale :

$$w_1 = v_1 \quad (\text{vecteur propre associé à } \lambda_1)$$

Variance maximale :

$$w_1^T \Sigma w_1 = w_1^T (\lambda_1 w_1) = \lambda_1 w_1^T w_1 = \lambda_1$$

PCA : Composantes principales suivantes

Deuxième composante :

$$\max_{w_2} w_2^T \Sigma w_2$$

sous contraintes $\|w_2\| = 1$ et $w_2^T w_1 = 0$

Généralisation : La k -ième composante principale est :

$$w_k = v_k \quad (\text{vecteur propre associé à } \lambda_k)$$

avec les contraintes d'orthogonalité :

$$w_i^T w_j = \delta_{ij} = \begin{cases} 1 & \text{si } i = j \\ 0 & \text{si } i \neq j \end{cases}$$

Propriété fondamentale : Les composantes principales forment une base orthonormée qui diagonalise la matrice de covariance.

PCA : Transformation et projection

Matrice de projection :

$$W = [v_1, v_2, \dots, v_k] \in \mathbb{R}^{d \times k}$$

avec $W^T W = I_k$ (orthonormalité)

Scores (coordonnées dans le nouvel espace) :

$$Z = X_c W \in \mathbb{R}^{n \times k}$$

Pour une observation x_i :

$$z_i = x_{c,i}^T W = \begin{bmatrix} x_{c,i}^T v_1 \\ x_{c,i}^T v_2 \\ \vdots \\ x_{c,i}^T v_k \end{bmatrix}$$

Chaque composante :

$$z_{ij} = x_{c,i}^T v_j = \sum_{l=1}^d x_{c,il} v_{jl}$$

PCA : Reconstruction des données

Reconstruction approchée :

$$\hat{X}_c = ZW^T = X_c WW^T$$

Cas particuliers :

- Si $k = d$: $\hat{X}_c = X_c$ (reconstruction exacte)
- Si $k < d$: $\hat{X}_c \approx X_c$ (approximation)

Données non centrées :

$$\hat{X} = \hat{X}_c + \mathbf{1}_n \bar{x}^T = X_c WW^T + \mathbf{1}_n \bar{x}^T$$

Erreur de reconstruction totale :

$$E = \sum_{i=1}^n \|x_{c,i} - \hat{x}_{c,i}\|^2 = n \sum_{j=k+1}^d \lambda_j$$

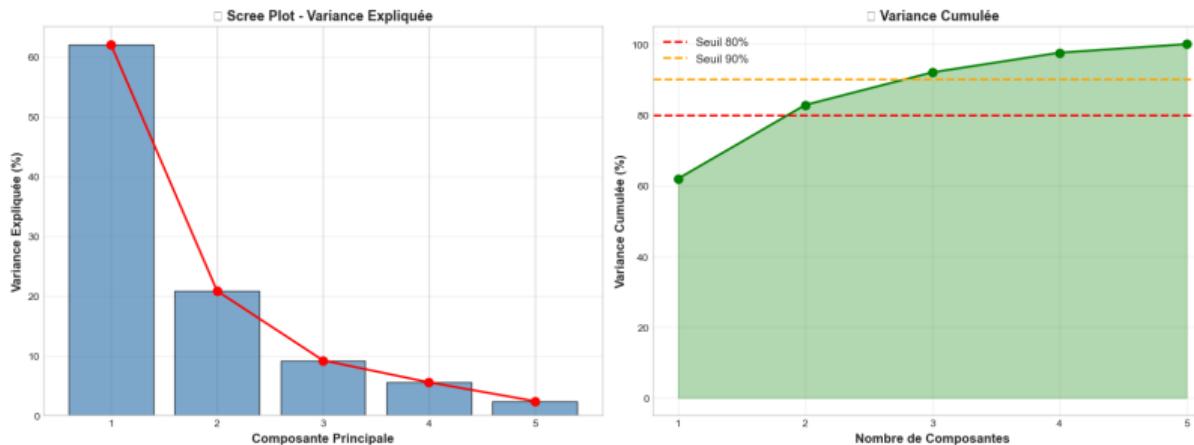
Interprétation

L'erreur est la somme des valeurs propres des composantes ignorées

PCA : Variance expliquée

Proportion de variance expliquée :

$$R^2(k) = \frac{\sum_{j=1}^k \lambda_j}{\sum_{j=1}^d \lambda_j}$$



Choix de k par seuil :

$$k^* = \min\{k : R^2(k) \geq 0.9\}$$

PCA : Méthodes de choix de k

Méthode 1 : Seuil de variance

$$k^* = \min\{k : R^2(k) \geq \text{seuil}\}$$

Seuils typiques : 80%, 90%, 95%

Méthode 2 : Règle de Kaiser

$$k^* = |\{\lambda_j : \lambda_j > \bar{\lambda}\}|$$

où $\bar{\lambda} = \frac{1}{d} \sum_{j=1}^d \lambda_j$

Ne garder que les composantes avec $\lambda_j > \bar{\lambda}$

Méthode 3 : Scree plot

Identifier le "coude" dans le graphique des valeurs propres

Méthode 4 : Validation croisée

Minimiser l'erreur de reconstruction sur un ensemble de validation

PCA : Bilan

Avantages

- Simple et intuitif
- Rapide : $O(nd^2)$
- Base théorique solide
- Décorrélation automatique
- Réduction du bruit
- Interprétation géométrique
- Nombreuses implémentations

Limitations

- Linéaire uniquement
- Sensible aux outliers
- Sensible à l'échelle
- Perte d'interprétabilité
- Suppose corrélations linéaires
- Variables importantes \neq variance élevée

Prétraitement crucial

Toujours normaliser avant PCA :

$$x_{s,ij} = \frac{x_{ij} - \bar{x}_j}{\sigma_j}$$

t-SNE : t-Distributed Stochastic Neighbor Embedding

Principe

Préserver les **similarités locales** entre points proches dans un espace de basse dimension

Philosophie :

- Points proches dans l'espace original doivent rester proches
- Points éloignés peuvent être plus ou moins éloignés
- Focus sur la structure locale

Applications principales :

- Visualisation de données haute dimension
- Exploration de structures de clusters
- Analyse qualitative des données

Note importante

t-SNE est **non-déterministe** : plusieurs exécutions donnent des résultats différents

t-SNE : Espace de haute dimension

Probabilités conditionnelles :

$$p_{j|i} = \frac{\exp(-\|x_i - x_j\|^2/2\sigma_i^2)}{\sum_{k \neq i} \exp(-\|x_i - x_k\|^2/2\sigma_i^2)}$$

où σ_i est déterminé par la **perplexité** (hyperparamètre)

Perplexité :

$$\text{Perplexité} = 2^{H(P_i)}$$

où $H(P_i) = -\sum_j p_{j|i} \log_2 p_{j|i}$ est l'entropie

Symétrisation :

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n}$$

Interprétation : p_{ij} mesure la similarité entre x_i et x_j

t-SNE : Espace de basse dimension

Distribution de Student (1 degré de liberté) :

$$q_{ij} = \frac{(1 + \|y_i - y_j\|^2)^{-1}}{\sum_{k \neq I} (1 + \|y_k - y_I\|^2)^{-1}}$$

où y_i sont les projections en basse dimension

Pourquoi Student-t ?

- Queue plus lourde que la gaussienne
- Permet aux points éloignés de rester éloignés
- Évite le "crowding problem" (encombrement ou congestion)

Fonction objectif - Divergence KL :

$$C = KL(P||Q) = \sum_{i \neq j} p_{ij} \log \frac{p_{ij}}{q_{ij}}$$

Objectif : Minimiser C pour que $Q \approx P$

t-SNE : Gradient et optimisation

Gradient de la fonction de coût :

$$\frac{\partial C}{\partial y_i} = 4 \sum_j (p_{ij} - q_{ij})(y_i - y_j)(1 + \|y_i - y_j\|^2)^{-1}$$

Forces attractives et répulsives :

- Si $p_{ij} > q_{ij}$: force attractive (rapprocher y_i et y_j)
- Si $p_{ij} < q_{ij}$: force répulsive (éloigner y_i et y_j)

Descente de gradient avec momentum :

$$y_i^{(t+1)} = y_i^{(t)} - \eta \frac{\partial C}{\partial y_i} + \alpha(t)(y_i^{(t)} - y_i^{(t-1)})$$

où :

- η : taux d'apprentissage
- $\alpha(t)$: coefficient de momentum

t-SNE : Hyperparamètres

Perplexité (principal hyperparamètre) :

- Contrôle le nombre effectif de voisins considérés
- Valeurs typiques : 5-50
- **Recommandation : 30**
- Petit dataset ($n < 1000$) : perplexité = 5-10
- Grand dataset : perplexité = 30-50

Autres paramètres :

- Nombre d'itérations : 1000-5000
- Taux d'apprentissage : 10-1000
- Early exaggeration : multiplie p_{ij} au début

Attention

t-SNE est computationnellement coûteux : $O(n^2)$

Fondement théorique

Basé sur la **théorie des variétés Riemanniennes** et la topologie algébrique

Avantages sur t-SNE :

- Plus **rapide** : $O(n \log n)$ vs $O(n^2)$
- Préserve mieux la **structure globale**
- Scalable aux grands datasets
- Résultats plus stables

Hypothèse clé : Les données sont échantillonnées uniformément sur une variété Riemannienne localement connectée

UMAP : Construction du graphe haute dimension

Pour chaque point, trouver le plus proche voisin :

$$\rho_i = \min_{j \in N_k(i)} d(x_i, x_j)$$

où $N_k(i)$ sont les k plus proches voisins de i

Probabilités conditionnelles :

$$p_{j|i} = \exp\left(-\frac{\max(0, d(x_i, x_j) - \rho_i)}{\sigma_i}\right)$$

où σ_i est déterminé pour maintenir un nombre effectif de voisins

Symétrisation par union probabiliste :

$$p_{ij} = p_{j|i} + p_{i|j} - p_{j|i} \cdot p_{i|j}$$

Cette formule représente la probabilité que i et j soient connectés (au moins une direction)

UMAP : Espace de basse dimension

Distribution paramétrique :

$$q_{ij} = \left(1 + a\|y_i - y_j\|^{2b}\right)^{-1}$$

où a et b sont des hyperparamètres :

- Typiquement : $a \approx 1.929$, $b \approx 0.7924$
- Déterminés par ajustement de courbe

Fonction objectif - Entropie croisée :

$$C = \sum_{i,j} \left[p_{ij} \log \frac{p_{ij}}{q_{ij}} + (1 - p_{ij}) \log \frac{1 - p_{ij}}{1 - q_{ij}} \right]$$

Deux termes :

- **Attraction** : $p_{ij} \log \frac{p_{ij}}{q_{ij}}$
- **Répulsion** : $(1 - p_{ij}) \log \frac{1 - p_{ij}}{1 - q_{ij}}$

UMAP : Optimisation

Gradient :

$$\frac{\partial C}{\partial y_i} = \sum_j \left[\frac{-2ab p_{ij} \|y_i - y_j\|^{2b-2}}{1 + a \|y_i - y_j\|^{2b}} - \frac{2ab(1 - p_{ij}) \|y_i - y_j\|^{2b-2}}{(1 + a \|y_i - y_j\|^{2b})^2} \right] (y_i - y_j)$$

Échantillonnage négatif :

- Ne calcule pas toutes les paires de points
- Échantillonne aléatoirement les points éloignés
- \Rightarrow Complexité $O(n \log n)$ au lieu de $O(n^2)$

Optimisation stochastique :

- Descente de gradient stochastique
- Mise à jour par mini-batches
- Convergence rapide

t-SNE vs UMAP : Comparaison détaillée

Aspect	t-SNE	UMAP
Distribution basse dim	Student-t (queue lourde)	Paramétrique ajustable
Fonction de coût	Divergence KL	Entropie croisée
Forces répulsives	Toutes les paires	Échantillonnage négatif
Complexité	$O(n^2)$	$O(n \log n)$
Structure globale	Moins bien préservée	Mieux préservée
Structure locale	Excellente	Très bonne
Déterminisme	Non	Non
Vitesse	(lent)	(rapide)
Fondement	Probabilités	Variétés Riemanniennes

Recommandation :

- **t-SNE** : Visualisation, focus structure locale, petits datasets
- **UMAP** : Grands datasets, prétraitement, structure globale

Plan de la section

1 Introduction à l'apprentissage non supervisé

2 Réduction de dimensionnalité

- PCA et SVD
- UMAP et t-SNE

3 Clustering

- Méthode K-means avec métriques
- Regroupement hiérarchique avec dendrogrammes
- GMM avec BIC et AIC
- DBSCAN avec paramètres

4 Études comparatives des 4 méthodes

K-means : Principe et objectif

Objectif

Partitionner n observations en k clusters en minimisant la variance intra-cluster

Fonction objectif (WCSS) :

$$J = \sum_{i=1}^k \sum_{x \in C_i} \|x - \mu_i\|^2$$

où :

- C_i : cluster i
- $\mu_i = \frac{1}{|C_i|} \sum_{x \in C_i} x$: centroïde du cluster i
- $\|\cdot\|$: norme euclidienne

Distance euclidienne :

$$d(x, \mu) = \sqrt{\sum_{j=1}^d (x_j - \mu_j)^2}$$

K-means : Algorithme de Lloyd

```
1: Initialisation : Choisir  $k$  centroïdes  $\mu_1^{(0)}, \dots, \mu_k^{(0)}$ 
2:   (K-means++, aléatoire, ou autre)
3: repeat
4:   Étape E (Affectation) :
5:     for chaque point  $x_n$  do
6:        $c_n = \arg \min_{i \in \{1, \dots, k\}} \|x_n - \mu_i\|^2$ 
7:     end for
8:   Étape M (Mise à jour) :
9:     for chaque cluster  $i$  do
10:       $\mu_i = \frac{1}{|C_i|} \sum_{x \in C_i} x$ 
11:    end for
12:  until Convergence :  $|\Delta J| < \epsilon$  ou affectations stables
```

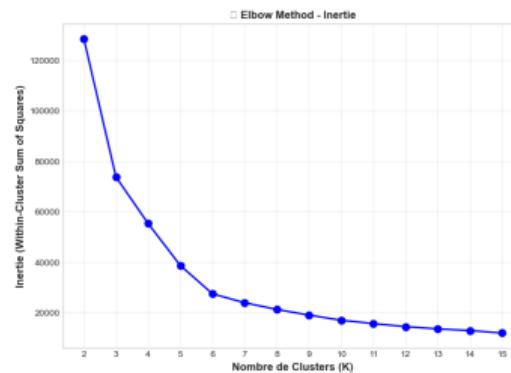
Complexité : $O(n \cdot k \cdot d \cdot t)$ où t = nombre d'itérations

Convergence : Garantie vers un minimum local

Méthode du coude (Elbow Method)

Principe : Tracer l'inertie en fonction de k

$$\text{Inertie}(k) = J(k) = \sum_{i=1}^k \sum_{x \in C_i} \|x - \mu_i\|^2$$



Limitation

Le coude n'est pas toujours évident \Rightarrow KneeLocator

KneeLocator : Détection automatique

Méthode : Distance maximale à la ligne droite

Étape 1 - Normalisation :

$$x'_i = \frac{x_i - \min(x)}{\max(x) - \min(x)}, \quad y'_i = \frac{y_i - \min(y)}{\max(y) - \min(y)}$$

Étape 2 - Ligne de référence L :

$$L : ax + by + c = 0$$

où $a = y'_n - y'_1$, $b = x'_1 - x'_n$, $c = x'_n \cdot y'_1 - x'_1 \cdot y'_n$

Étape 3 - Distance perpendiculaire :

$$d_i = \frac{|ax'_i + by'_i + c|}{\sqrt{a^2 + b^2}}$$

Étape 4 - Point de coude :

$$k^* = \arg \max_{i \in \{1, \dots, n\}} d_i$$

Score de Silhouette

Pour chaque point x_i dans le cluster C_p :

Cohésion intra-cluster : Distance moyenne aux autres points du même cluster

$$a(i) = \frac{1}{|C_p| - 1} \sum_{x_j \in C_p, j \neq i} d(x_i, x_j)$$

Séparation inter-cluster : Distance moyenne au cluster le plus proche

$$b(i) = \min_{C_q \neq C_p} \left\{ \frac{1}{|C_q|} \sum_{x_k \in C_q} d(x_i, x_k) \right\}$$

Coefficient de silhouette individuel :

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}$$

Score global :

$$S = \frac{1}{n} \sum_{i=1}^n s(i) \in [-1, 1]$$

Score de Silhouette : Visualisation

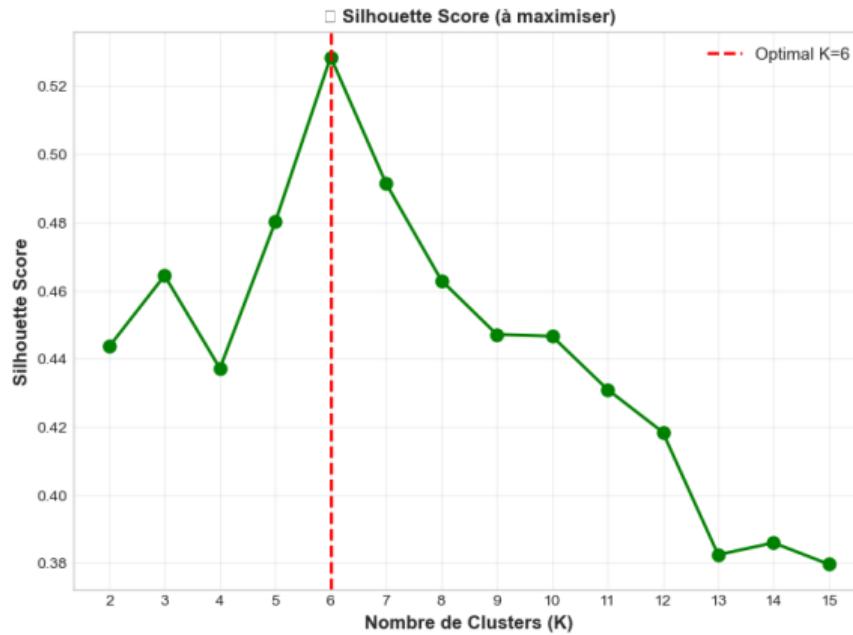


Figure – Score de silhouette pour chaque k .

Score de Silhouette : Interprétation

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}} \in [-1, 1]$$

Forme alternative :

$$s(i) = \begin{cases} \frac{b(i)-a(i)}{b(i)} & \text{si } a(i) < b(i) \\ 0 & \text{si } a(i) = b(i) \\ \frac{b(i)-a(i)}{a(i)} & \text{si } a(i) > b(i) \end{cases}$$

Interprétation :

- $s(i) \approx 1$: Point bien classé ($a(i) \ll b(i)$)
- $s(i) \approx 0$: Point sur frontière ($a(i) \approx b(i)$)
- $s(i) < 0$: Point mal classé ($a(i) > b(i)$)

Complexité : $O(n^2)$ - coûteux pour grands datasets

Index Davies-Bouldin

Dispersion intra-cluster : Distance moyenne au centroïde (compacté)

$$S_j = \frac{1}{|C_j|} \sum_{x \in C_j} d(x, \mu_j)$$

Séparation inter-cluster : Distance entre centroïdes

$$M_{ij} = d(\mu_i, \mu_j) = \|\mu_i - \mu_j\|$$

Ratio de similarité :

$$R_{ij} = \frac{S_i + S_j}{M_{ij}}$$

Plus R_{ij} est petit, plus les clusters sont distincts

Index Davies-Bouldin :

$$DB = \frac{1}{k} \sum_{i=1}^k \max_{j \neq i} R_{ij} \in [0, +\infty)$$

Objectif : Minimiser $DB \rightarrow 0$ (optimal)

Index Davies-Bouldin : Visualisation

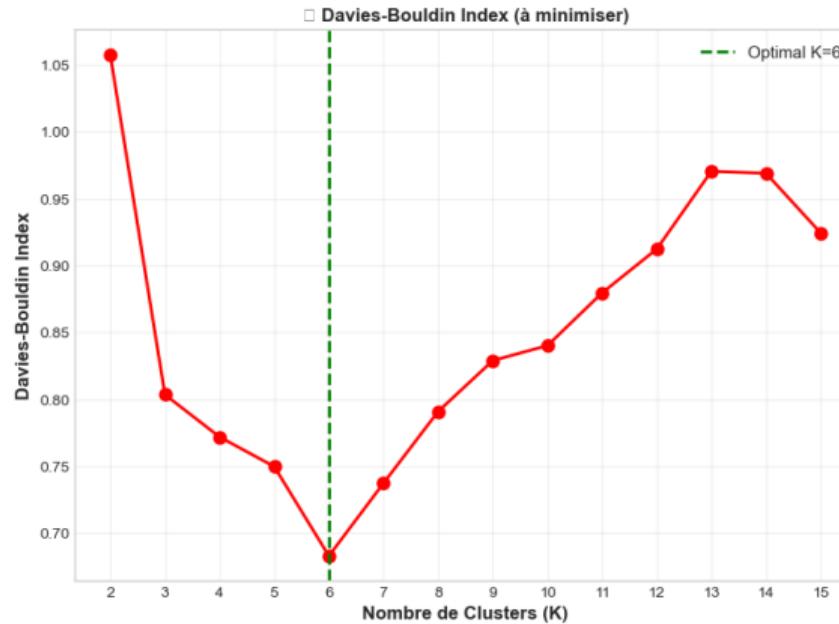


Figure – Index Davies-Bouldin pour chaque k .

Score Calinski-Harabasz (Variance Ratio)

Variance inter-cluster (Between) : Mesure la séparation entre clusters

$$SS_B = \sum_{j=1}^k |C_j| \cdot \|\mu_j - \mu\|^2$$

Variance intra-cluster (Within) : Mesure la compacité des clusters

$$SS_W = \sum_{j=1}^k \sum_{x \in C_j} \|x - \mu_j\|^2$$

Score Calinski-Harabasz :

$$CH = \frac{SS_B}{SS_W} \cdot \frac{n - k}{k - 1}$$

Forme F-statistique :

$$CH = \frac{SS_B/(k - 1)}{SS_W/(n - k)}$$

Objectif : Maximiser $CH \rightarrow +\infty$

Score Calinski-Harabasz : Visualisation

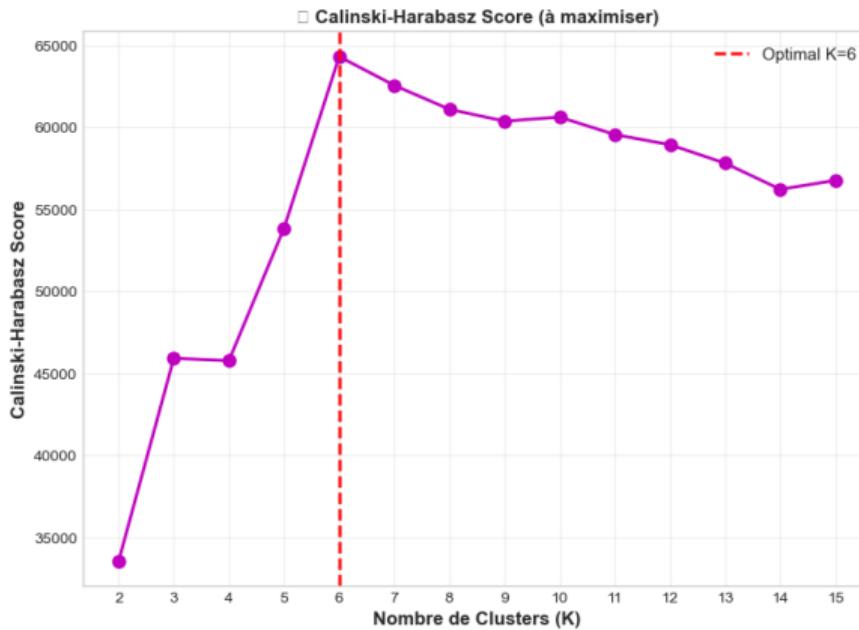


Figure – Score Calinski-Harabasz pour chaque k .

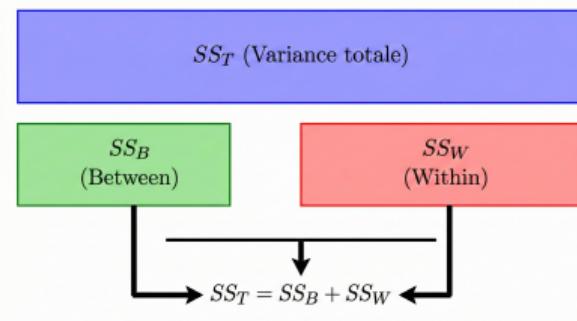
Décomposition ANOVA de la variance

Variance totale :

$$SS_T = \sum_{i=1}^n \|x_i - \mu\|^2$$

Propriété fondamentale :

$$SS_T = SS_B + SS_W$$



Interprétation :

$$CH = \frac{\text{Séparation}}{\text{Compacité}} \cdot \text{facteur de normalisation}$$

Comparaison des métriques de clustering

Métrique	Objectif	Domaine	Optimal	Complexité
Silhouette	Maximiser	$[-1, 1]$	$\rightarrow 1$	$O(n^2)$
Davies-Bouldin	Minimiser	$[0, +\infty)$	$\rightarrow 0$	$O(nk)$
Calinski-Harabasz	Maximiser	$[0, +\infty)$	$\rightarrow +\infty$	$O(nk)$

Principe commun

$$\text{Bon clustering} = \frac{\text{Séparation entre clusters}}{\text{Compacité des clusters}}$$

Formulations équivalentes :

- **Silhouette** : $\frac{b-a}{\max\{a,b\}}$ (Séparation - Compacité) / max
- **Davies-Bouldin** : $\frac{S_i+S_j}{M_{ij}}$ (Compacité) / Séparation
- **Calinski-Harabasz** : $\frac{SS_B}{SS_W}$ Séparation / Compacité

Choix de la métrique

Utiliser Silhouette quand :

- Dataset petite/moyenne taille ($n < 10,000$)
- Besoin d'identifier points mal classés individuellement
- Clusters de formes variées

Utiliser Davies-Bouldin quand :

- Dataset moyenne/grande taille
- Clusters approximativement convexes
- Besoin de rapidité de calcul

Utiliser Calinski-Harabasz quand :

- Dataset de grande taille
- Clusters gaussiens ou convexes
- Choix automatique du nombre de clusters

Recommandation : Calculer les **trois métriques** et vérifier leur concordance

$$S \uparrow, DB \downarrow, CH \uparrow \Rightarrow \text{Clustering cohérent}$$

Clustering Hiérarchique : Principe

Objectif

Construire une **hiérarchie** de clusters sans spécifier k à l'avance

Deux approches :

Ascendant (CAH)

- Bottom-up
- Départ : n clusters
- Fusion progressive
- Plus utilisé
- Déterministe

Descendant (DIANA)

- Top-down
- Départ : 1 cluster
- Division progressive
- Moins utilisé
- Plus complexe

Propriétés :

- Hiérarchie imbriquée
- Pas de recouvrement
- Résultat : **dendrogramme**

Algorithme CAH (ascendant)

- 1: **Initialisation :**
- 2: Chaque observation = un cluster (n clusters)
- 3: Calculer matrice de distances $D = [d_{ij}]_{n \times n}$
- 4: **while** nombre de clusters > 1 **do**
- 5: **Fusion** : Trouver les deux clusters les plus proches

$$C_i^*, C_j^* = \arg \min_{i \neq j} d(C_i, C_j)$$

- 6: Fusionner : $C_{new} = C_i^* \cup C_j^*$
- 7: Enregistrer hauteur : $h = d(C_i^*, C_j^*)$
- 8: **Mise à jour** : Recalculer distances avec C_{new}
- 9: **end while**
- 10: **Sortie** : Dendrogramme complet

Complexité : $O(n^2 \log n)$ optimisée, $O(n^3)$ naïve

Méthodes de liaison (Linkage)

1. Liaison simple (Single linkage) : Distance minimale - Tend à créer clusters allongés (effet de chaîne)

$$d_{simple}(C_i, C_j) = \min_{x \in C_i, y \in C_j} d(x, y)$$

2. Liaison complète (Complete linkage) : Distance maximale - Favorise clusters compacts

$$d_{complete}(C_i, C_j) = \max_{x \in C_i, y \in C_j} d(x, y)$$

3. Liaison moyenne (UPGMA) : Compromis entre simple et complète

$$d_{average}(C_i, C_j) = \frac{1}{|C_i| \cdot |C_j|} \sum_{x \in C_i} \sum_{y \in C_j} d(x, y)$$

4. Méthode de Ward (la plus utilisée) :

$$d_{Ward}(C_i, C_j) = \frac{|C_i| \cdot |C_j|}{|C_i| + |C_j|} \|\mu_i - \mu_j\|^2$$

Méthode de Ward : Détails

Critère : minimiser l'augmentation de variance

Formulation par inertie :

$$d_{Ward}(C_i, C_j) = I(C_i \cup C_j) - I(C_i) - I(C_j)$$

où l'inertie d'un cluster :

$$I(C) = \sum_{x \in C} \|x - \mu_C\|^2$$

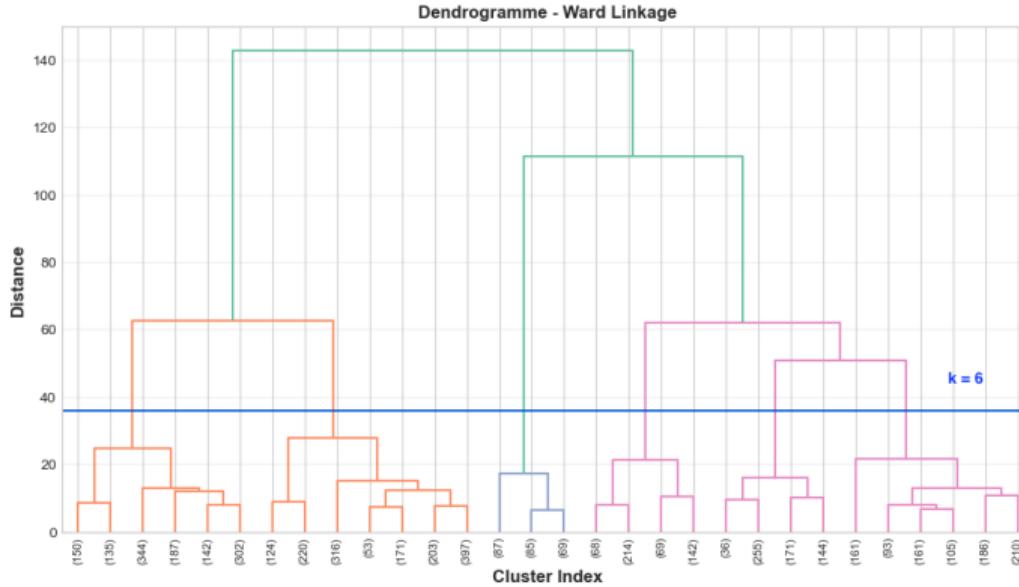
Formule de mise à jour (Lance-Williams) :

$$d(C_k, C_i \cup C_j) = \frac{(|C_i| + |C_k|)d(C_k, C_i) + (|C_j| + |C_k|)d(C_k, C_j) - |C_k|d(C_i, C_j)}{|C_i| + |C_j| + |C_k|}$$

Propriétés :

- Tend à créer clusters de taille égale
- Équivalent à K-means (minimise variance intra-cluster)
- Méthode la plus populaire en pratique

Dendrogramme : Visualisation



Interprétation :

- Hauteur = dissimilarité de fusion
- Coupe horizontale = choix de k
- Grands sauts = bonne séparation

Détermination du nombre de clusters

Méthode 1 : Observation visuelle

- Identifier les grands "sauts" verticaux
- Couper avant le plus grand saut

Méthode 2 : Variation des hauteurs

$$\Delta h^{(t)} = h^{(t)} - h^{(t-1)}, \quad k^* = \arg \max_t \Delta h^{(t)}$$

Méthode 3 : Coefficient cophénétique

$$c = \text{corr}(d_{ij}, h_{ij})$$

Mesure la fidélité du dendrogramme aux distances originales

- $c > 0.8$: Excellente représentation
- $c > 0.7$: Bonne représentation
- $c < 0.7$: Représentation discutable

Méthode 4 : Indices de qualité

Gaussian Mixture Model (GMM)

Hypothèse fondamentale

Les données proviennent d'un **mélange de K distributions gaussiennes**

Modèle de mélange :

$$p(x) = \sum_{k=1}^K \pi_k \mathcal{N}(x|\mu_k, \Sigma_k)$$

où :

- π_k : poids de la composante k ($\sum_{k=1}^K \pi_k = 1$, $\pi_k \geq 0$)
- $\mu_k \in \mathbb{R}^d$: moyenne de la composante k
- $\Sigma_k \in \mathbb{R}^{d \times d}$: covariance de la composante k

Gaussienne multivariée :

$$\mathcal{N}(x|\mu, \Sigma) = \frac{1}{(2\pi)^{d/2} |\Sigma|^{1/2}} \exp \left(-\frac{1}{2} (x - \mu)^T \Sigma^{-1} (x - \mu) \right)$$

GMM vs K-means

Aspect	K-means	GMM
Assignation	Dure (hard) Un cluster/point	Douce (soft) Probabilités
Forme clusters	Sphérique	Elliptique
Distance	Euclidienne	Probabiliste
Covariance	Identique ($\sigma^2 I$)	Variable (Σ_k)
Modèle	Géométrique	Génératif
Incertitude	Non	Oui

Exemple d'assignation douce

K-means : "Point x appartient au cluster 1"

GMM : "Point x : 70% cluster 1, 25% cluster 2, 5% cluster 3"

Avantage GMM : Quantification de l'incertitude

GMM : Responsabilité (Soft Assignment)

Probabilité qu'un point x_i provienne de la composante k :

$$\gamma_{ik} = P(z_i = k | x_i, \theta) = \frac{\pi_k \mathcal{N}(x_i | \mu_k, \Sigma_k)}{\sum_{j=1}^K \pi_j \mathcal{N}(x_i | \mu_j, \Sigma_j)}$$

Règle de Bayes :

$$\gamma_{ik} = \frac{P(x_i | z_i = k) \cdot P(z_i = k)}{P(x_i)} = \frac{\text{vraisemblance} \times \text{prior}}{\text{évidence}}$$

Propriétés :

- $\gamma_{ik} \in [0, 1]$
- $\sum_{k=1}^K \gamma_{ik} = 1$ (normalisation)
- γ_{ik} = "responsabilité" de la composante k pour x_i

Interprétation : Degré d'appartenance floue à chaque cluster

Algorithme EM (Expectation-Maximization)

Objectif : Maximiser la log-vraisemblance

$$\mathcal{L}(\theta) = \sum_{i=1}^n \log p(x_i|\theta) = \sum_{i=1}^n \log \sum_{k=1}^K \pi_k \mathcal{N}(x_i|\mu_k, \Sigma_k)$$

Problème : Somme à l'intérieur du log \Rightarrow difficile à optimiser

Solution : Problème circulaire résolu par alternance E-M

- Pour estimer les **paramètres**, il faut connaître les **assignations**
- Pour calculer les **assignations**, il faut connaître les **paramètres**

Propriété fondamentale :

$$\mathcal{L}(\theta^{(t)}) \geq \mathcal{L}(\theta^{(t-1)})$$

Convergence monotone vers un maximum local

Algorithme EM : Étapes

1: **Initialisation** : $\theta^{(0)} = \{\pi_k^{(0)}, \mu_k^{(0)}, \Sigma_k^{(0)}\}$

2: (K-means, aléatoire, ou K-means++)

3: **repeat**

4: **Étape E (Expectation)** : Calculer responsabilités

$$\gamma_{ik}^{(t)} = \frac{\pi_k^{(t-1)} \mathcal{N}(x_i | \mu_k^{(t-1)}, \Sigma_k^{(t-1)})}{\sum_{j=1}^K \pi_j^{(t-1)} \mathcal{N}(x_i | \mu_j^{(t-1)}, \Sigma_j^{(t-1)})}$$

5: **Étape M (Maximization)** : Mettre à jour paramètres

6: Calculer : $N_k^{(t)} = \sum_{i=1}^n \gamma_{ik}^{(t)}$

7: $\pi_k^{(t)} = N_k^{(t)} / n$

8: $\mu_k^{(t)} = \frac{1}{N_k^{(t)}} \sum_{i=1}^n \gamma_{ik}^{(t)} x_i$

9: $\Sigma_k^{(t)} = \frac{1}{N_k^{(t)}} \sum_{i=1}^n \gamma_{ik}^{(t)} (x_i - \mu_k^{(t)}) (x_i - \mu_k^{(t)})^T$

10: **until** $|\mathcal{L}^{(t)} - \mathcal{L}^{(t-1)}| < \epsilon$

EM : Étape M - Dérivations

Taille effective du cluster k :

$$N_k^{(t)} = \sum_{i=1}^n \gamma_{ik}^{(t)}$$

Mise à jour des poids (avec Lagrangien) :

$$\mathcal{L} = \sum_{k=1}^K N_k \log \pi_k - \lambda \left(\sum_{k=1}^K \pi_k - 1 \right), \quad \frac{\partial \mathcal{L}}{\partial \pi_k} = \frac{N_k}{\pi_k} - \lambda = 0 \Rightarrow \pi_k = \frac{N_k}{\lambda}$$

Avec $\sum_k \pi_k = 1$: $\lambda = n \Rightarrow \pi_k^{(t)} = \frac{N_k^{(t)}}{n}$

Mise à jour des moyennes et des covariances :

$$\mu_k^{(t)} = \frac{\sum_i \gamma_{ik}^{(t)} x_i}{\sum_i \gamma_{ik}^{(t)}} = \frac{1}{N_k^{(t)}} \sum_{i=1}^n \gamma_{ik}^{(t)} x_i, \quad \Sigma_k^{(t)} = \frac{1}{N_k^{(t)}} \sum_{i=1}^n \gamma_{ik}^{(t)} (x_i - \mu_k^{(t)}) (x_i - \mu_k^{(t)})^T$$

Critère d'Information d'Akaike (AIC)

Formule :

$$AIC = -2\mathcal{L}(\hat{\theta}) + 2p$$

où :

- $\mathcal{L}(\hat{\theta})$: log-vraisemblance maximisée
- p : nombre de paramètres libres

Pour GMM avec covariances complètes :

$$p = \underbrace{(K - 1)}_{\text{poids}} + \underbrace{K \cdot d}_{\text{moyennes}} + \underbrace{K \cdot \frac{d(d + 1)}{2}}_{\text{covariances}} = K \left(1 + d + \frac{d(d + 1)}{2} \right) - 1$$

Principe de parcimonie :

- $-2\mathcal{L}$: terme d'ajustement (à minimiser)
- $2p$: pénalité de complexité

Objectif : Minimiser AIC $\Rightarrow K^* = \arg \min_K AIC(K)$

Critère d'Information Bayésien (BIC)

Formule :

$$BIC = -2\mathcal{L}(\hat{\theta}) + p \log(n)$$

où n = nombre d'observations

Comparaison des pénalités :

Critère	Pénalité	$n = 100$	$n = 1000$
AIC	$2p$	$2p$	$2p$
BIC	$p \log(n)$	$\approx 4.6p$	$\approx 6.9p$

Conséquences :

- Pour $n > e^2 \approx 7.4$: BIC pénalise plus que AIC
- BIC favorise des modèles **plus simples**
- BIC est **asymptotiquement consistant**
- AIC minimise l'erreur de prédiction

Objectif : **Minimiser** BIC $\Rightarrow K^* = \arg \min_K BIC(K)$

AIC vs BIC : Philosophies

AIC

- **Philosophie** : Prédiction optimale
- Minimise l'erreur attendue
- Privilégie la complexité
- Meilleur pour : petits/moyens datasets
- Fondement : Théorie de l'information (divergence KL)
- Peut sur-ajuster pour grands n

BIC

- **Philosophie** : Sélection du vrai modèle
- Approximation bayésienne
- Privilégie la parcimonie
- Meilleur pour : grands datasets
- Fondement : Évidence bayésienne
- Asymptotiquement consistant

Recommandation pratique

- Calculer **les deux critères**
- Si concordance : confiance élevée
- Si divergence : examiner visuellement les données
- Compléter avec Silhouette, DB, CH

DBSCAN : Density-Based Clustering

Principe

Identifier les clusters comme des régions de **haute densité** séparées par des régions de **faible densité**

Avantages uniques :

- Déetecte des clusters de **formes arbitraires**
- Nombre de clusters **automatique** (pas besoin de k)
- Détection explicite du **bruit (noise/outliers)**
- Pas de centroïdes requis
- Déterministe

Deux paramètres :

- ① ε (**eps**) : rayon du voisinage
- ② **MinPts (min_samples)** : nombre minimum de points pour un point cœur

Philosophie : Un cluster = région dense + connectée

Voisinage ε :

$$N_\varepsilon(p) = \{q \in D : d(p, q) \leq \varepsilon\}$$

Types de points :

① **Point cœur (Core point) :**

$$|N_\varepsilon(p)| \geq \text{MinPts}$$

Se trouve dans une région dense

② **Point frontière (Border point) :**

$$|N_\varepsilon(p)| < \text{MinPts} \text{ ET } \exists q \in \text{Core} : p \in N_\varepsilon(q)$$

À la périphérie d'un cluster

③ **Point de bruit (Noise) :**

$$|N_\varepsilon(p)| < \text{MinPts} \text{ ET } \forall q \in \text{Core} : p \notin N_\varepsilon(q)$$

Outlier isolé

DBSCAN : Relations entre points

Directement accessible par densité : Relation **non symétrique** en général

$$p \rightsquigarrow_{\varepsilon} q \text{ si } p \in \text{Core ET } q \in N_{\varepsilon}(p)$$

Accessible par densité (chaîne) : Relation **transitive**

$$p \rightsquigarrow_{\varepsilon}^* q$$

S'il existe une chaîne p_1, \dots, p_n avec :

$$p_1 = p, \ p_n = q, \ p_i \rightsquigarrow_{\varepsilon} p_{i+1} \ \forall i$$

Connexité par densité : Relation **symétrique et transitive**

$$p \leftrightarrow_{\varepsilon} q$$

S'il existe o tel que : $p \rightsquigarrow_{\varepsilon}^* o$ ET $q \rightsquigarrow_{\varepsilon}^* o$

Définition formelle d'un cluster :

$$C = \{q \in D : \exists p \in \text{Core}(C), q \rightsquigarrow_{\varepsilon}^* p\}$$

Algorithme DBSCAN

```
1: Initialisation : Tous points non visités,  $C = 0$ 
2: for chaque point  $p$  non visité do
3:   Marquer  $p$  comme visité
4:    $N_{\varepsilon}(p) \leftarrow$  voisins dans rayon  $\varepsilon$ 
5:   if  $|N_{\varepsilon}(p)| < \text{MinPts}$  then
6:     Marquer  $p$  comme bruit (provisoirement)
7:   else
8:      $C \leftarrow C + 1$  (nouveau cluster)
9:     Créer cluster  $C_C$  avec  $p$ 
10:    Étendre cluster :
11:    for all  $q \in N_{\varepsilon}(p)$  do
12:      if  $q$  non visité then
13:        Marquer  $q$  visité
14:         $N_{\varepsilon}(q) \leftarrow$  voisins de  $q$ 
15:        if  $|N_{\varepsilon}(q)| \geq \text{MinPts}$  then
16:          Ajouter  $N_{\varepsilon}(q)$  à la liste
17:        end if
18:      end if
19:      if  $q$  pas dans un cluster then
20:        Ajouter  $q$  à  $C_C$ 
21:      end if
22:    end for
23:  end if
24: end for
```

DBSCAN : Choix de ε (eps)

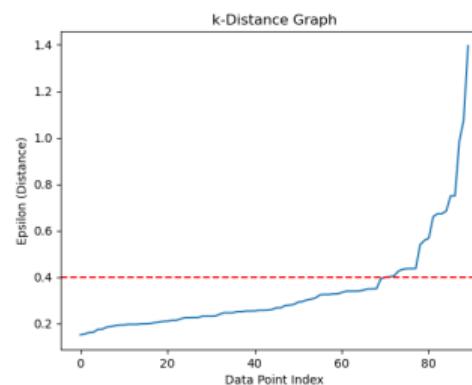
Méthode du graphe k-distance

Étape 1 : Pour chaque point, calculer distance au k -ième voisin

$$d_k(p) = d(p, \text{NN}_k(p))$$

où $k = \text{MinPts}$

Étape 2 : Trier en ordre décroissant et tracer



Étape 3 : Identifier le coude $\rightarrow \varepsilon^*$

Utiliser KneeLocator !

DBSCAN : Choix de MinPts

Règle générale :

$$\text{MinPts} \geq d + 1, \quad \text{où } d = \text{dimensionnalité}$$

Règle pratique courante :

$$\text{MinPts} = 4 \text{ pour } d = 2$$

Données bruitées :

$$\text{MinPts} \geq 2d$$

Grands datasets :

$$\text{MinPts} \approx \ln(n)$$

- MinPts **trop petit** : Nombreux petits clusters, sensible bruit
- MinPts **trop grand** : Petits clusters ignorés
- ε **trop petit** : Beaucoup de bruit
- ε **trop grand** : Tout devient un cluster

DBSCAN : Avantages et Limitations

Avantages

- Formes arbitraires
- K automatique
- Détection de bruit
- Déterministe
- Pas de centroïdes
- Efficace avec index spatial
- Complexité $O(n \log n)$

Limitations

- Densités variables
- Sensible à ε , MinPts
- Haute dimensionnalité
- Pas de hiérarchie
- Points frontière ambigus
- Choix paramètres non trivial

Alternative : HDBSCAN - Hierarchical DBSCAN (données avec **densités variables**)

- Un seul paramètre (`min_cluster_size`)
- Construit une hiérarchie de densités
- Plus robuste et automatique

Plan de la section

1 Introduction à l'apprentissage non supervisé

2 Réduction de dimensionnalité

- PCA et SVD
- UMAP et t-SNE

3 Clustering

- Méthode K-means avec métriques
- Regroupement hiérarchique avec dendrogrammes
- GMM avec BIC et AIC
- DBSCAN avec paramètres

4 Études comparatives des 4 méthodes

Tableau comparatif global

Critère	K-means	Hiérarchique	GMM	DBSCAN
Assignation	Dure	Dure	Douce	Dure
Forme clusters	Sphérique	Variable	Elliptique	Arbitraire
K prédefini	Oui	Non	Oui	Non
Complexité	$O(nKdi)$	$O(n^2 \log n)$	$O(nKdi)$	$O(n \log n)$
Initialisation	Sensible	Déterministe	Très sensible	Non
Bruit	Ignoré	Ignoré	Ignoré	Détecté
Scalabilité	***	*	**	**
Hiérarchie	Non	Oui	Non	Non
Probabilités	Non	Non	Oui	Non
Vitesse	***	*	**	**
Mémoire	$O(nk)$	$O(n^2)$	$O(nk)$	$O(n)$

Légende : n = observations, k ou K = clusters, d = dimensions, i = itérations

Fonctions objectives comparées

K-means - Minimiser inertie :

$$J_{KM} = \sum_{k=1}^K \sum_{x \in C_k} \|x - \mu_k\|^2$$

Hiérarchique (Ward) - Minimiser augmentation variance :

$$\Delta(C_i, C_j) = \frac{|C_i||C_j|}{|C_i| + |C_j|} \|\mu_i - \mu_j\|^2$$

GMM - Maximiser log-vraisemblance :

$$\mathcal{L}_{GMM} = \sum_{i=1}^n \log \sum_{k=1}^K \pi_k \mathcal{N}(x_i | \mu_k, \Sigma_k)$$

DBSCAN - Pas de fonction objective explicite

- Basé sur la densité et la connectivité
- Définition topologique/géométrique
- Pas d'optimisation numérique

Quand utiliser K-means ?

Idéal si

- $n > 50,000$ (grande échelle)
- Clusters **sphériques**
- Tailles similaires
- **Vitesse critique**
- K connu approximativement
- Données numériques continues
- Besoin de centroïdes explicites

Éviter si

- Clusters non convexes ou allongés
- Tailles très différentes
- Présence importante de bruit
- K totalement inconnu
- Données catégorielles

Applications : Segmentation client, compression d'images, quantification vectorielle, initialisation GMM

Quand utiliser le Clustering Hiérarchique ?

Idéal si

- $n < 5,000$ (petite/moyenne échelle)
- **K totalement inconnu**
- Besoin de **dendrogramme**
- Structure hiérarchique naturelle
- Exploration exploratoire
- Interprétabilité importante
- Analyse de relations

Éviter si

- $n > 10,000$ (trop lent)
- Ressources mémoire limitées ($O(n^2)$)
- Besoin de rapidité
- Données très bruitées
(surtout single linkage)

Applications : Taxonomie biologique, analyse de gènes, hiérarchie d'entreprise, phylogénétique

Quand utiliser les GMM ?

Idéal si

- Besoin de **probabilités d'appartenance**
- Clusters **elliptiques**
- Modèle **génératif** nécessaire
- Données gaussiennes (approximativement)
- Quantification de l'incertitude
- Différentes formes et orientations
- Génération de nouvelles données

Éviter si

- $n > 100,000$
(sauf optimisations)
- Distributions fortement non-gaussiennes
- Besoin de rapidité maximale
- K totalement inconnu

Applications : Reconnaissance vocale, détection d'anomalies, classification floue, modélisation de distributions

Quand utiliser DBSCAN ?

✓ Idéal si

- Clusters de **formes arbitraires**
- Présence importante de **bruit / outliers**
- **K totalement inconnu**
- Densité relativement uniforme dans les clusters
- Données spatiales / géographiques
- Pas de centroïdes requis
- Détection d'anomalies nécessaire

55 Éviter si

- Clusters de **densités très variables**
- Haute dimensionnalité ($d > 20$)
- Besoin de centroïdes explicites
- Paramètres difficiles à estimer
- Données uniformément denses

Applications : Analyse spatiale, trajectoires GPS, segmentation d'images, détection de fraude, zones urbaines

Pipeline recommandé en pratique

① Prétraitement (crucial)

- Normalisation : Z-score ou Min-Max
- Gestion valeurs manquantes
- Si $d > 50$: Réduction (PCA, UMAP)
- Détection outliers préliminaire

② Exploration initiale

- Si $n < 5k$: Hiérarchique (dendrogramme)
- Si $n > 5k$: K-means avec plusieurs K
- Visualisation 2D/3D (t-SNE ou UMAP)

③ Affinement

- Si formes complexes : DBSCAN
- Si besoin probabilités : GMM
- Si sphérique : K-means (optimal)

④ Validation rigoureuse

- Silhouette, DB, CH
- BIC/AIC (GMM)
- Visualisation + expertise métier

Synthèse et recommandations finales

Choix optimal = Contexte + Données + Objectifs

Aucune méthode n'est universellement meilleure

Critères de décision (checklist) :

- ① **Taille** : $n < 5k$, $5k < n < 50k$, $n > 50k$?
- ② **Forme** : sphérique, elliptique, arbitraire ?
- ③ **K connu** : oui, approximativement, totalement inconnu ?
- ④ **Bruit** : présent (DBSCAN) ou absent ?
- ⑤ **Probabilités** : nécessaires (GMM) ou non ?
- ⑥ **Hiérarchie** : pertinente (Hiérarchique) ou non ?
- ⑦ **Vitesse** : critique (K-means) ou secondaire ?

Approche combinée

Combiner plusieurs méthodes donne souvent les meilleurs résultats :

- Explorer avec Hiérarchique → Affiner avec K-means

Points clés de la présentation :

① Réduction de dimensionnalité

- PCA : linéaire, variance maximale, rapide
- t-SNE : non-linéaire, visualisation, structure locale
- UMAP : rapide, structure globale + locale

② Clustering : 4 familles complémentaires

- K-means : rapide, sphérique, grande échelle
- Hiérarchique : dendrogramme, exploration, K flexible
- GMM : probabiliste, elliptique, modèle génératif
- DBSCAN : formes arbitraires, détection bruit, K auto

③ Validation multi-critères

- Métriques : Silhouette, Davies-Bouldin, Calinski-Harabasz
- Critères de modèle : AIC, BIC
- Visualisation + expertise métier

L'apprentissage non supervisé révèle les structures cachées dans vos données !



Références principales

Ressources en ligne :

- Scikit-learn : scikit-learn.org
- Stanford CS229 : cs229.stanford.edu
- Python Machine Learning : docs.python.org

Livres recommandés :

- Bishop - "Pattern Recognition and Machine Learning"
- Murphy - "Machine Learning : A Probabilistic Perspective"
- Hastie et al. - "The Elements of Statistical Learning"

Merci pour votre attention !

Questions ?

Les formulations mathématiques complètes sont disponibles dans les documents de référence