

Analyse et visualisation de données

Machine learning

Plan du cours

- Rappel sur les notions de bases en machine learning
- Apprentissage non supervisé
 - Clustering (hiérarchique, KMeans)
 - Réduction de dimensionnalité (PCA)
- Apprentissage supervisé
 - Toujours plus d'algorithmes
 - Gradient boosting
 - Réseaux de neurones
 - Feature selection
 - Explicabilité

Evaluation

- Contrôle continu
 - Exercices notés
 - Contrôles intermédiaires
 - Contrôles surprises
 - Examen final

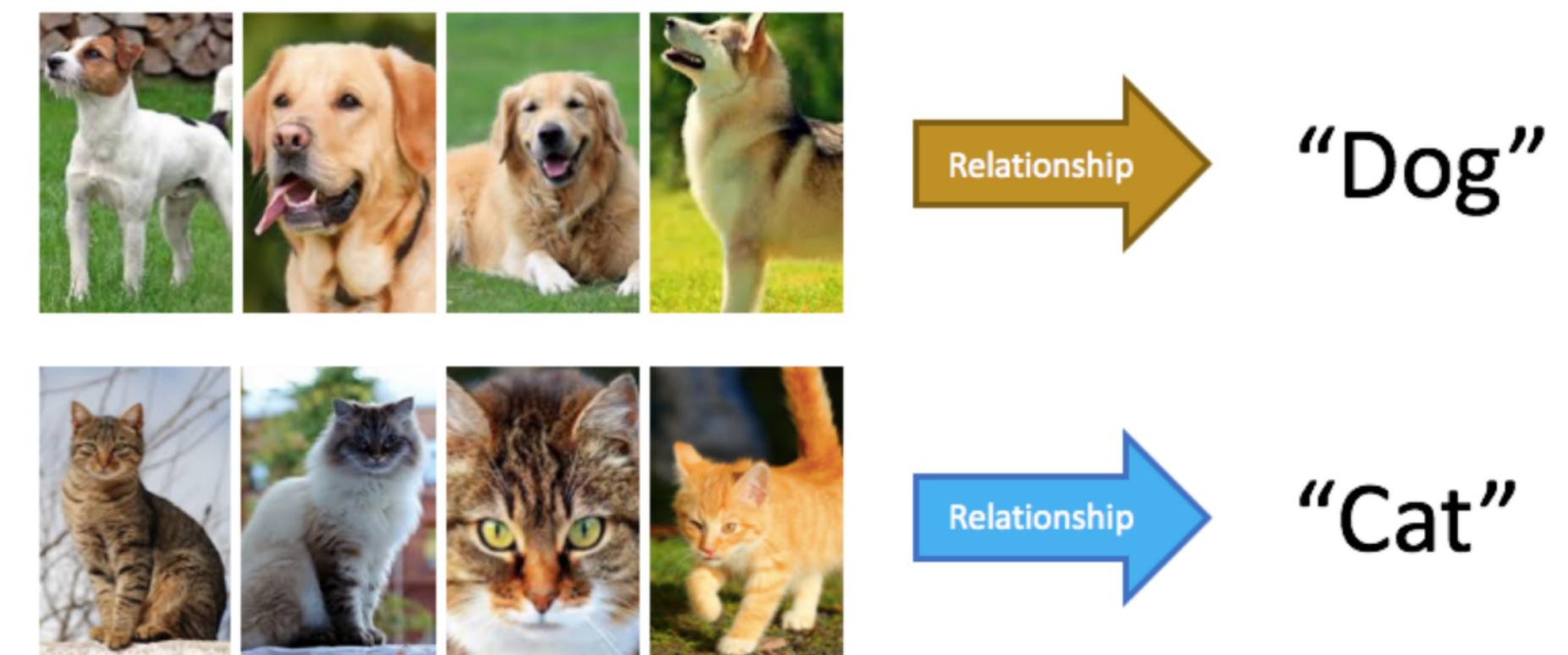
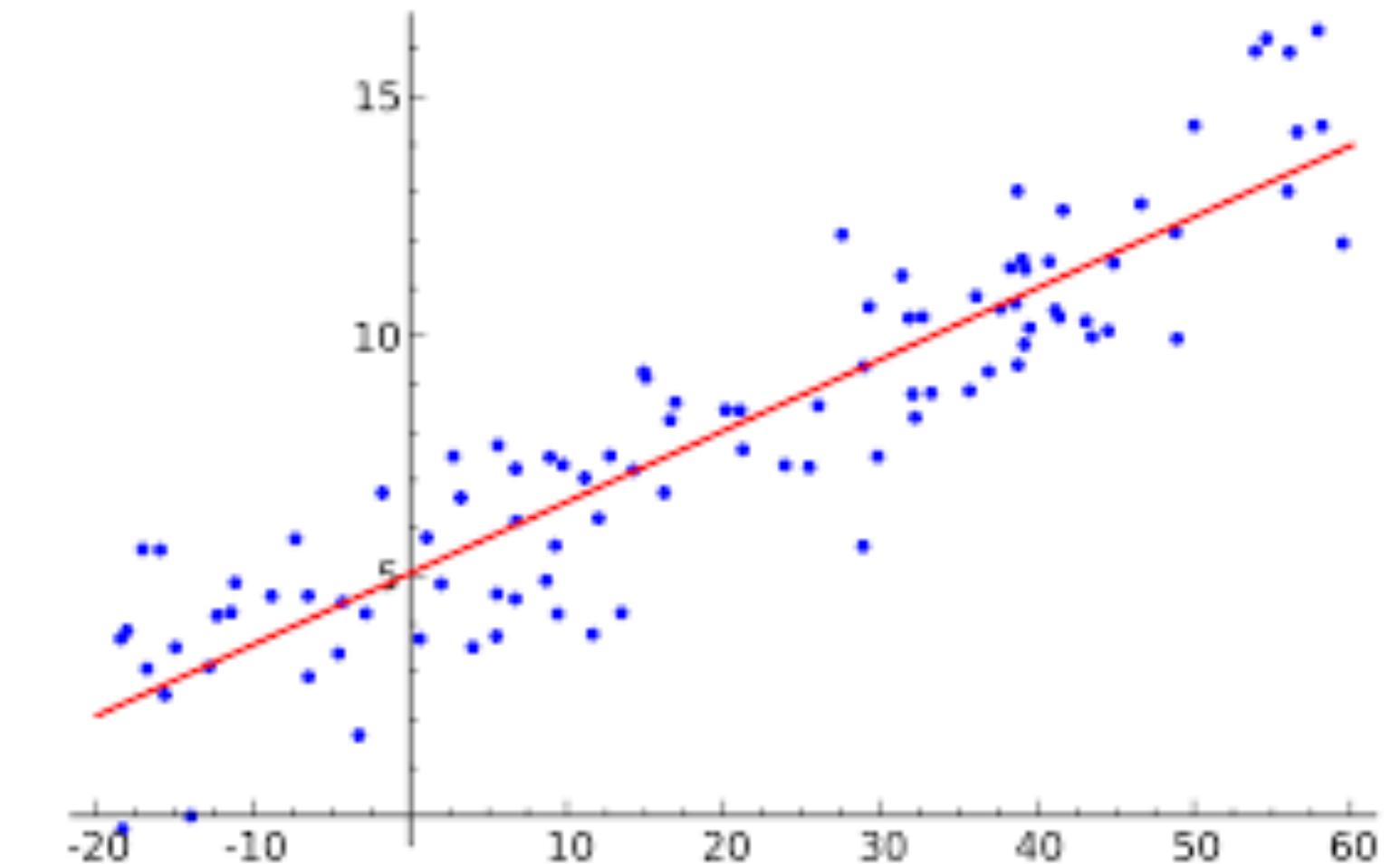
Rappel

Supervisé vs Non supervisé

Apprentissage supervisé

Définition

- Principe
 - Etant donné un ensemble de N exemples d'entraînement $\{(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}, i = 1..N\}$, on cherche une fonction de prédiction telle que $y = f(x)$
 - L'apprentissage est supervisé car les labels y sont connus
- Exemples
 - Classification d'images
 - Prédiction météo (température), stock, etc.

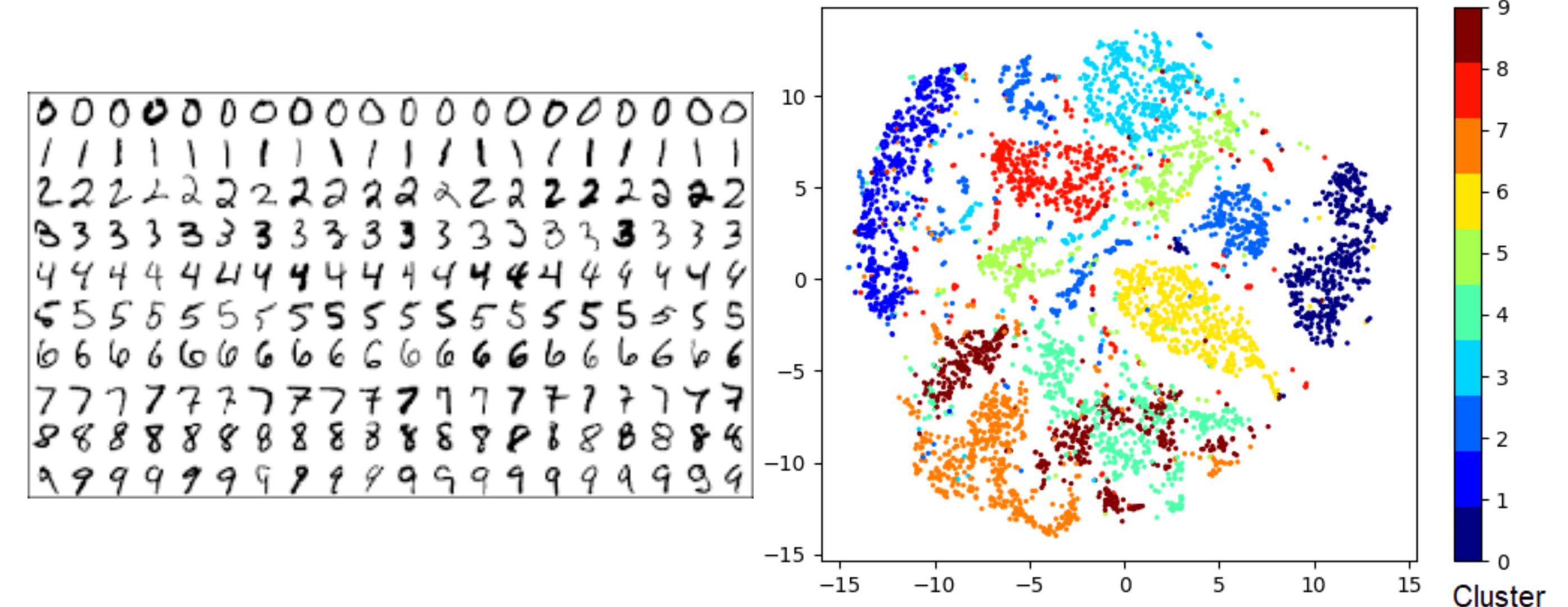


Apprentissage non-supervisé

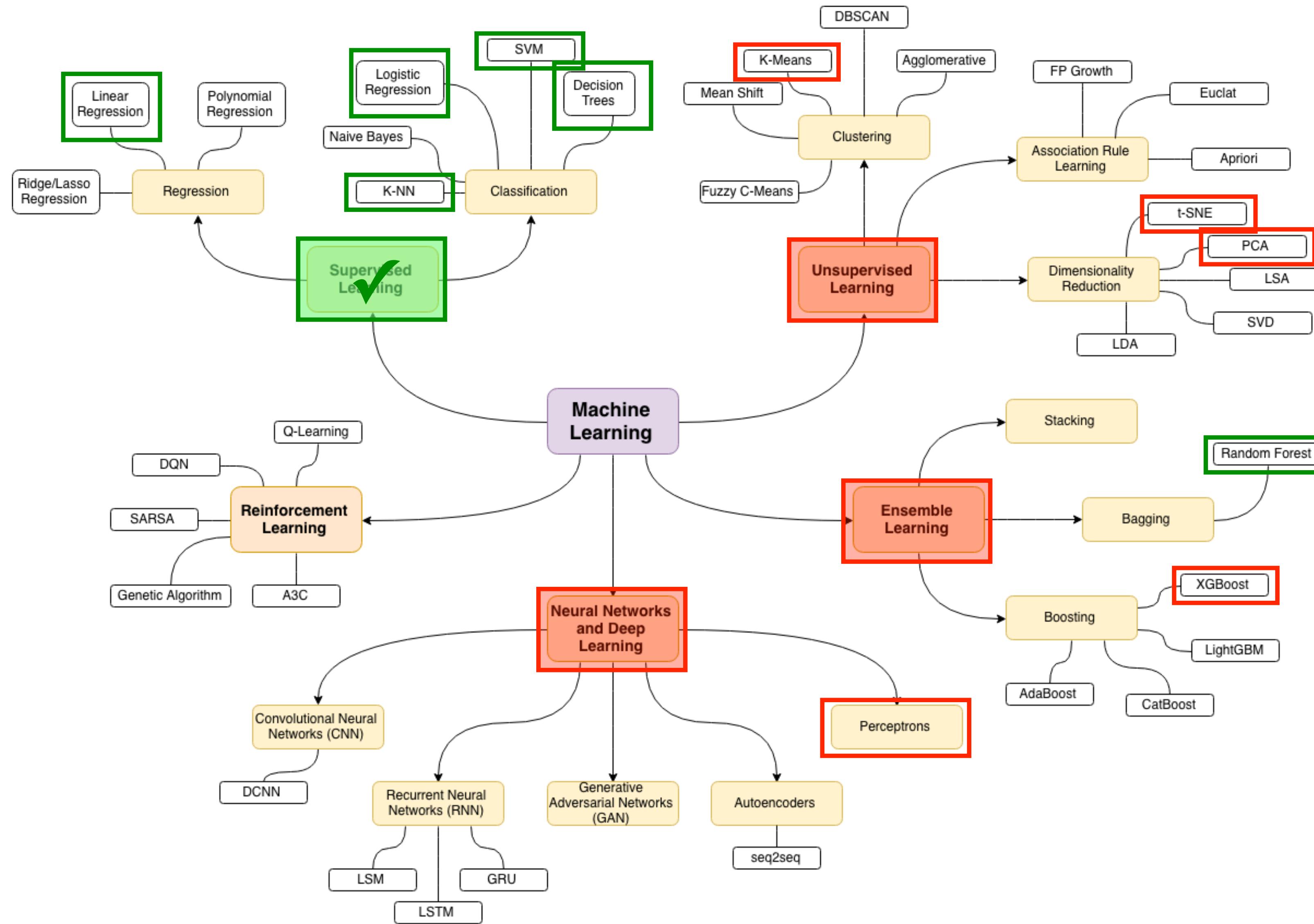
Définition

- Principe :
 - Seul les caractéristiques $\{x_i \in \mathcal{X}, i = 1..N\}$ sont disponibles.
 - On cherche à décrire comment les données sont **organisées** et extraire les **sous-ensemble homogènes**.
- Exemple :
 - Reconnaissance des formes
 - Analyse spatiales
 - Catégorisation de documents
 - Catégorisation d'usage (web, personnalisation, etc.)

0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4
5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5
6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6
7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7
8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8
9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9



Ce qu'on a vu et ce qu'on va voir



Apprentissage non supervisé

Clustering & Réduction de dimensionnalité

Clustering

Clustering - Plan

- Introduction
 - Notion de dissimilarité
 - Qualité des clusters
- Méthodes de clustering
 - Clustering hiérarchique
 - K-Means

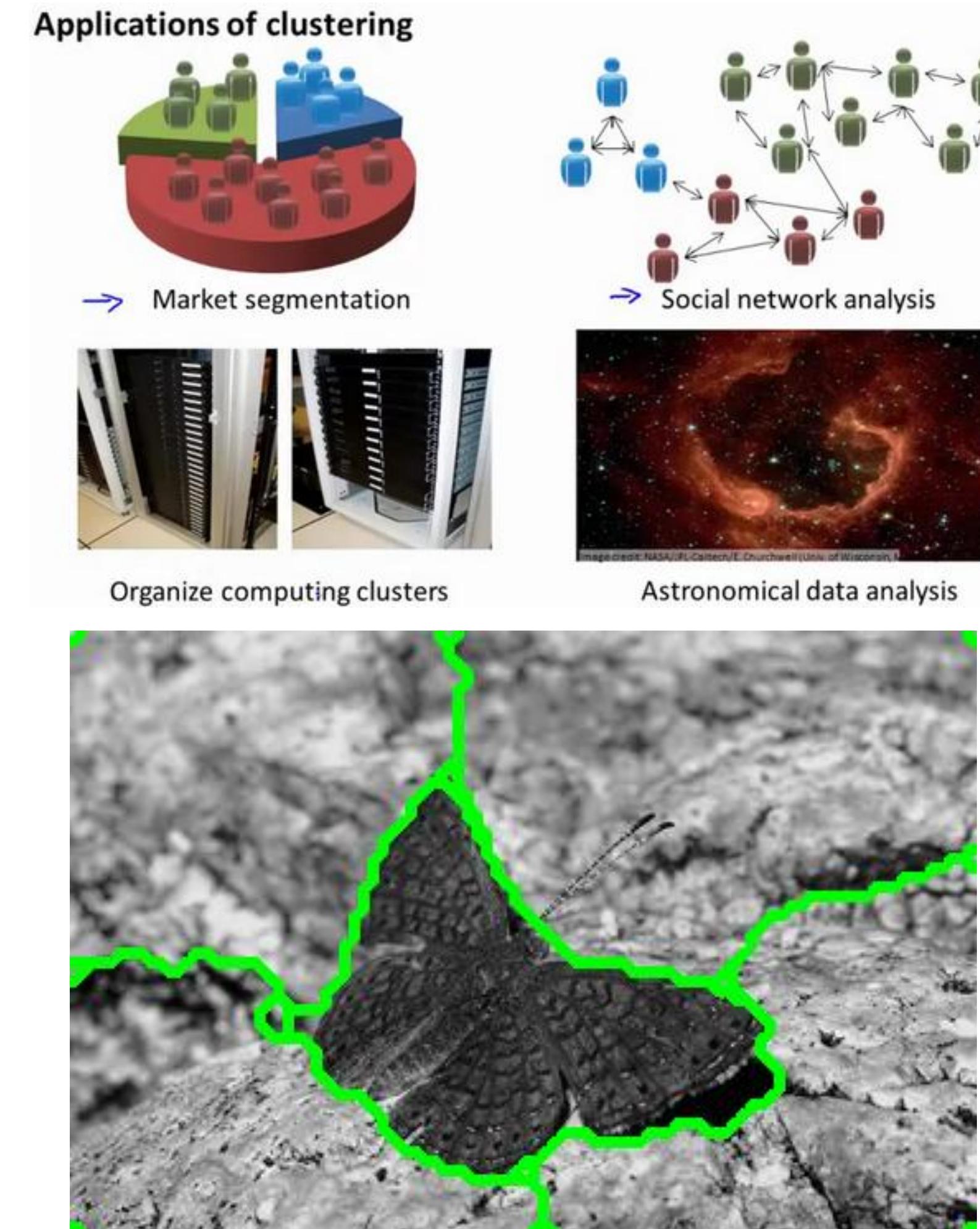
Introduction

- Apprentissage non-supervisé
- Soit un ensemble d'exemples d'entraînement $\mathcal{D} = \{x_i \in \mathbb{R}^d\}_{i=1}^N$
 - Objectif : Structurer les données dans des **catégories homogènes**
 - Grouper les données dans des **clusters** tel que les **données d'un cluster soient aussi similaires que possible**
- Exemple : clustering d'images



Quelques applications

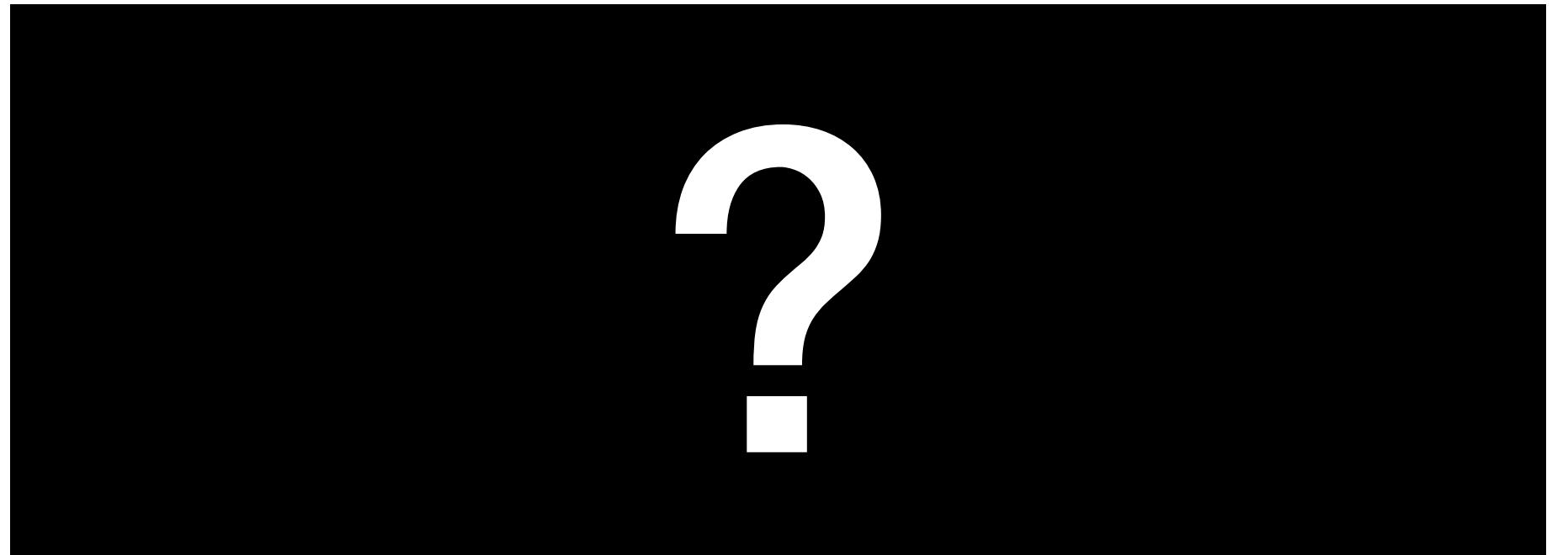
- Fouille de texte
 - Synthèse
 - Classement automatique en dossier
- Fouille de graphes
 - Sous-réseau sociaux
- Bioinformatique
 - Similarité de gènes
- Marketing
 - Segmentation de client
 - Profilage d'utilisateurs
- Segmentation d'images
 - Zones homogènes d'une image



Qu'est-ce que le clustering ?

- Algorithmes/méthodes permettant de trouver rassembler des informations similaires
 - Comment définir la similarité / dissimilarité entre exemples ?
 - Comment caractériser un cluster ?
 - Combien de clusters doit-on trouver ?
 - Quel algorithme utiliser ?
 - Comment évaluer le résultat obtenu ?

What is a natural grouping among these objects?



Mesure de dissimilarité

- La dissimilarité est une fonction de 2 paramètres (x_1, x_2) retournant une valeur dans \mathbb{R}_+ telle que :
 - $D(x_1, x_2) = D(x_2, x_1) \geq 0$ et
 - $D(x_1, x_2) = 0 \implies x_1 = x_2$
- Mesures de dissimilarités : distance entre 2 points x_1 et x_2
 - Distance de Minkowski: $D(x_1, x_2)^q = \sum_{j=1}^d |x_{1,j} - x_{2,j}|^q$
 - Distance euclidienne au carré ($q=2$): $D(x_1, x_2)^2 = \sum_{j=1}^d |x_{1,j} - x_{2,j}|^2 = (x_1 - x_2)^\top (x_1 - x_2)$
 - Distance de Manhattan ($q=1$): $D(x_1, x_2) = \sum_{j=1}^d |x_{1,j} - x_{2,j}|$

Dissimilitude entre clusters

- Distance $D(\mathcal{C}_1, \mathcal{C}_2)$ entre 2 clusters \mathcal{C}_1 and \mathcal{C}_2

- Diamètre minimum (plus proche voisin)

$$D_{min}(\mathcal{C}_1, \mathcal{C}_2) = \min\{D(x_i, x_j), x_i \in \mathcal{C}_1, x_j \in \mathcal{C}_2\}$$

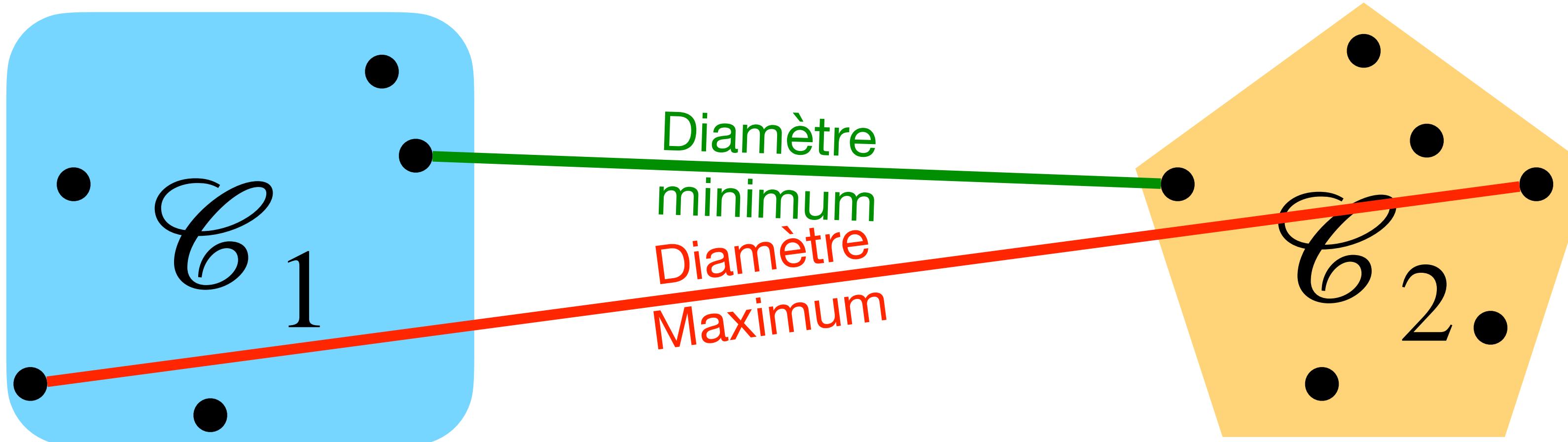
- Diamètre maximum

$$D_{max}(\mathcal{C}_1, \mathcal{C}_2) = \max\{D(x_i, x_j), x_i \in \mathcal{C}_1, x_j \in \mathcal{C}_2\}$$

Dissimilitude entre clusters

Distance $D(\mathcal{C}_1, \mathcal{C}_2)$ entre 2 clusters \mathcal{C}_1 and \mathcal{C}_2

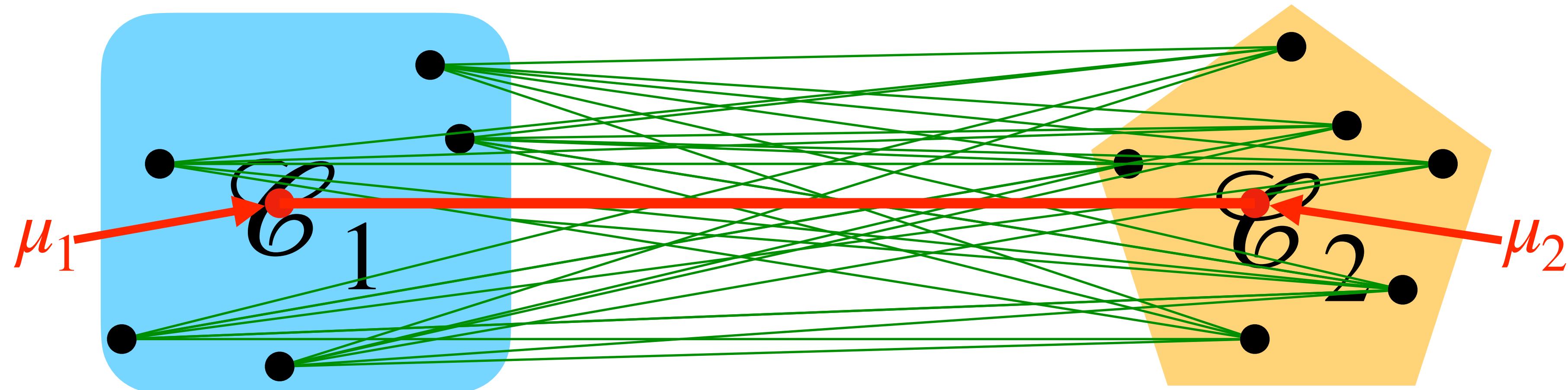
- Diamètre minimum: $D_{min}(\mathcal{C}_1, \mathcal{C}_2) = \min\{D(x_i, x_j), x_i \in \mathcal{C}_1, x_j \in \mathcal{C}_2\}$
- Diamètre maximum: $D_{max}(\mathcal{C}_1, \mathcal{C}_2) = \max\{D(x_i, x_j), x_i \in \mathcal{C}_1, x_j \in \mathcal{C}_2\}$



Dissimilitude entre clusters

Distance $D(\mathcal{C}_1, \mathcal{C}_2)$ entre 2 clusters \mathcal{C}_1 and \mathcal{C}_2

- Distance moyenne: $D_{moy}(\mathcal{C}_1, \mathcal{C}_2) = \frac{\sum_{x_i \in \mathcal{C}_1} \sum_{x_j \in \mathcal{C}_2} D(x_i, x_j)}{n_1 n_2}$
- Distance de Ward (entre les centres): $D_{Ward}(\mathcal{C}_1, \mathcal{C}_2) = \sqrt{\frac{n_1 n_2}{n_1 + n_2}} D(\mu_1, \mu_2)$

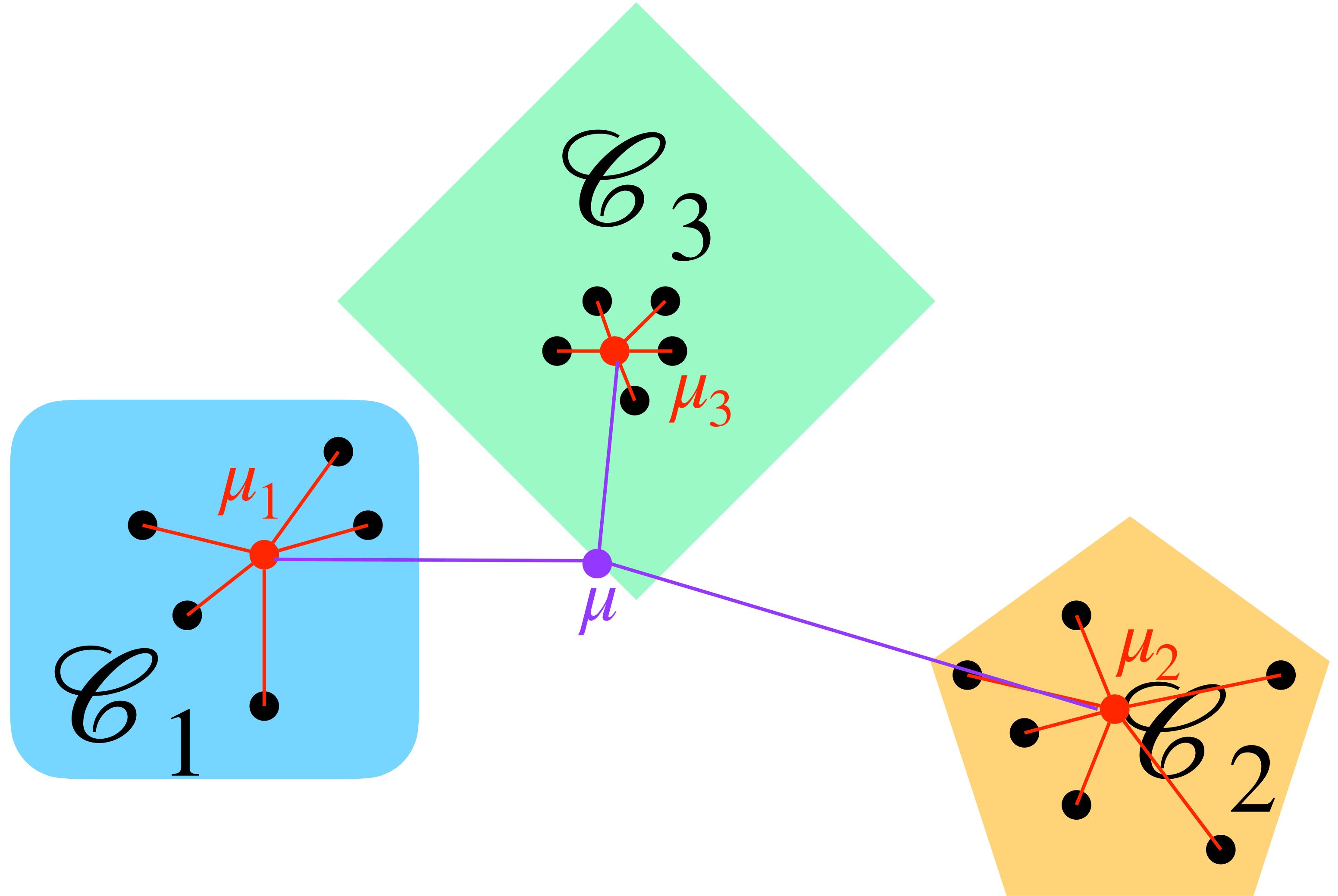


Caractéristiques d'un bon clustering

- Chaque cluster \mathcal{C}_l est caractérisé par :
 - Son centroïde : $\mu_l = \frac{1}{n_l} \sum_{i \in \mathcal{C}_l} x_i$ avec $n_l = \text{card}(\mathcal{C}_l)$
 - Son inertie : $J_l = \sum_{i \in \mathcal{C}_l} D^2(x_i, \mu_l)$
- Inertie : Distance moyenne intra-cluster $J_{intra} = \sum_l J_l$ avec $n_{\mathcal{C}} = \text{Card}(\mathcal{C})$
- Distorsion : Distance inter-cluster $J_{inter} = \sum_l n_l D^2(\mu_l, \mu)$ avec $\mu = \frac{1}{N} \sum_i x_i$

Illustration

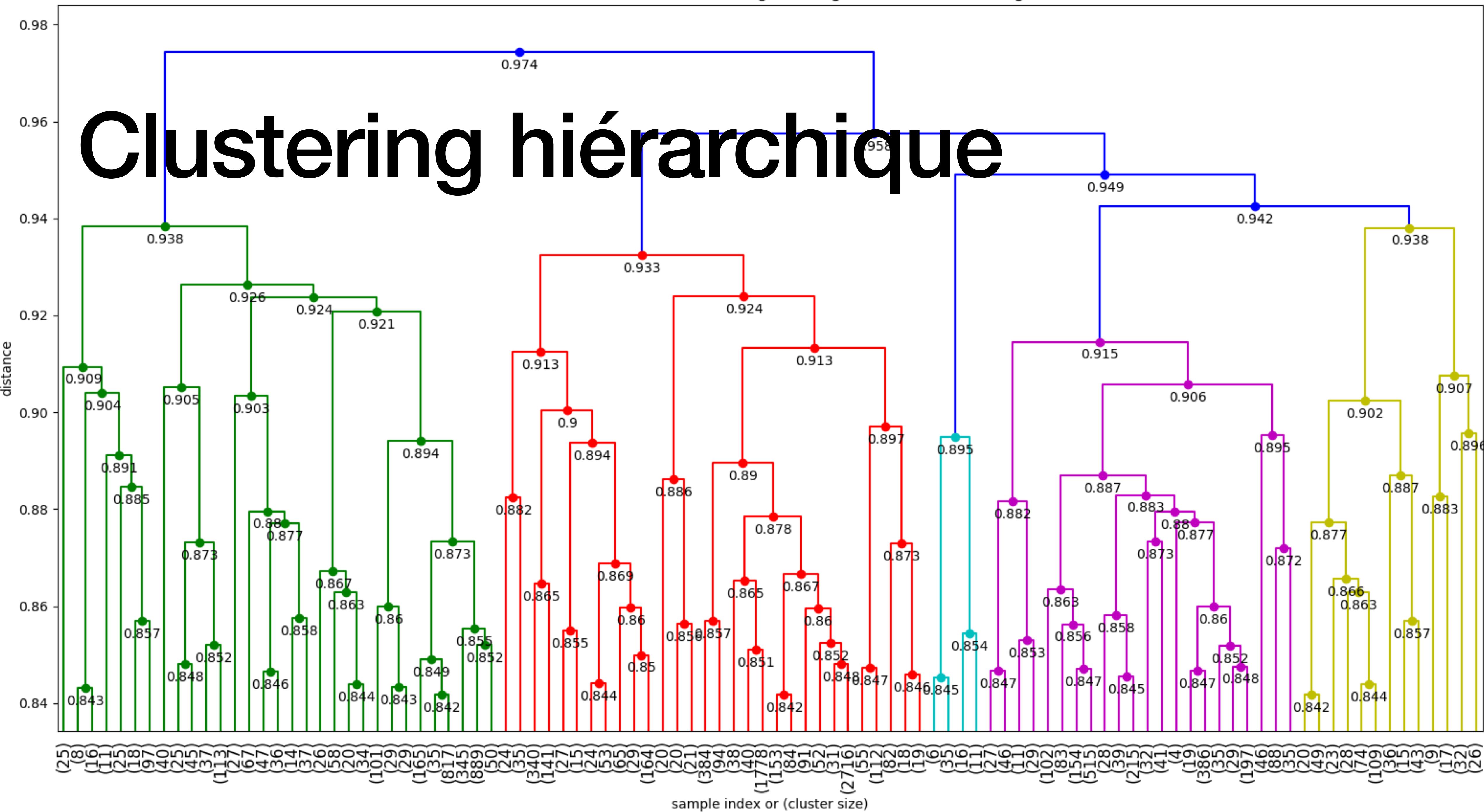
Inertie
Distorsion



- Un bon clustering est un clustering qui :
 - Minimise l'inertie
 - Maximise la distorsion

Truncated Hierarchical Clustering Dendrogram (method = average)

Clustering hiérarchique



Clustering hiérarchique

Principe de l'algorithme par agglomération

- Chaque point ou cluster est aggloméré au cluster/point le plus proche
- Algorithme

Initialisation

Chaque point du dataset représente un cluster

Chaque cluster est ajouté à la liste de clusters $L_{\mathcal{C}}$

Répéter

Calculer la distance entre chaque cluster de la liste $L_{\mathcal{C}}$

Choisir les 2 clusters \mathcal{C}_i et \mathcal{C}_j ayant la distance minimale

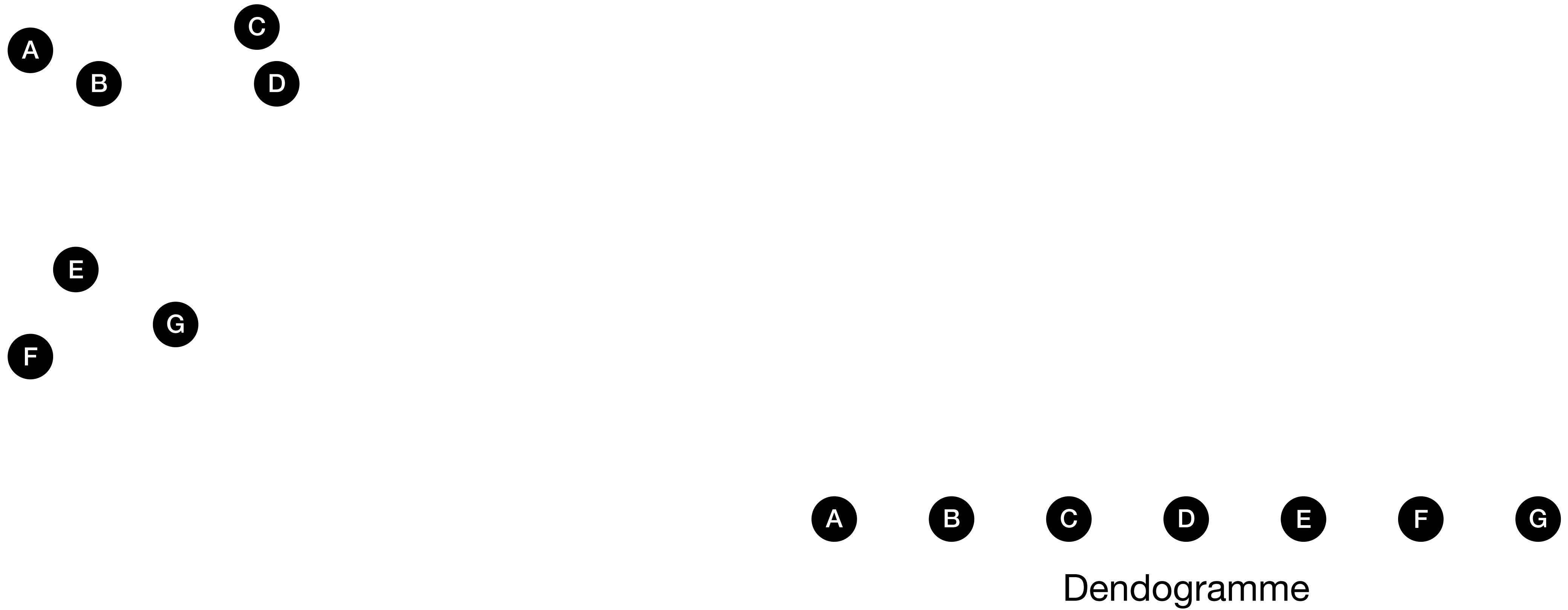
Fusionner les clusters \mathcal{C}_i et \mathcal{C}_j en un nouveau cluster \mathcal{C}_k

Supprimer \mathcal{C}_i et \mathcal{C}_j à $L_{\mathcal{C}}$

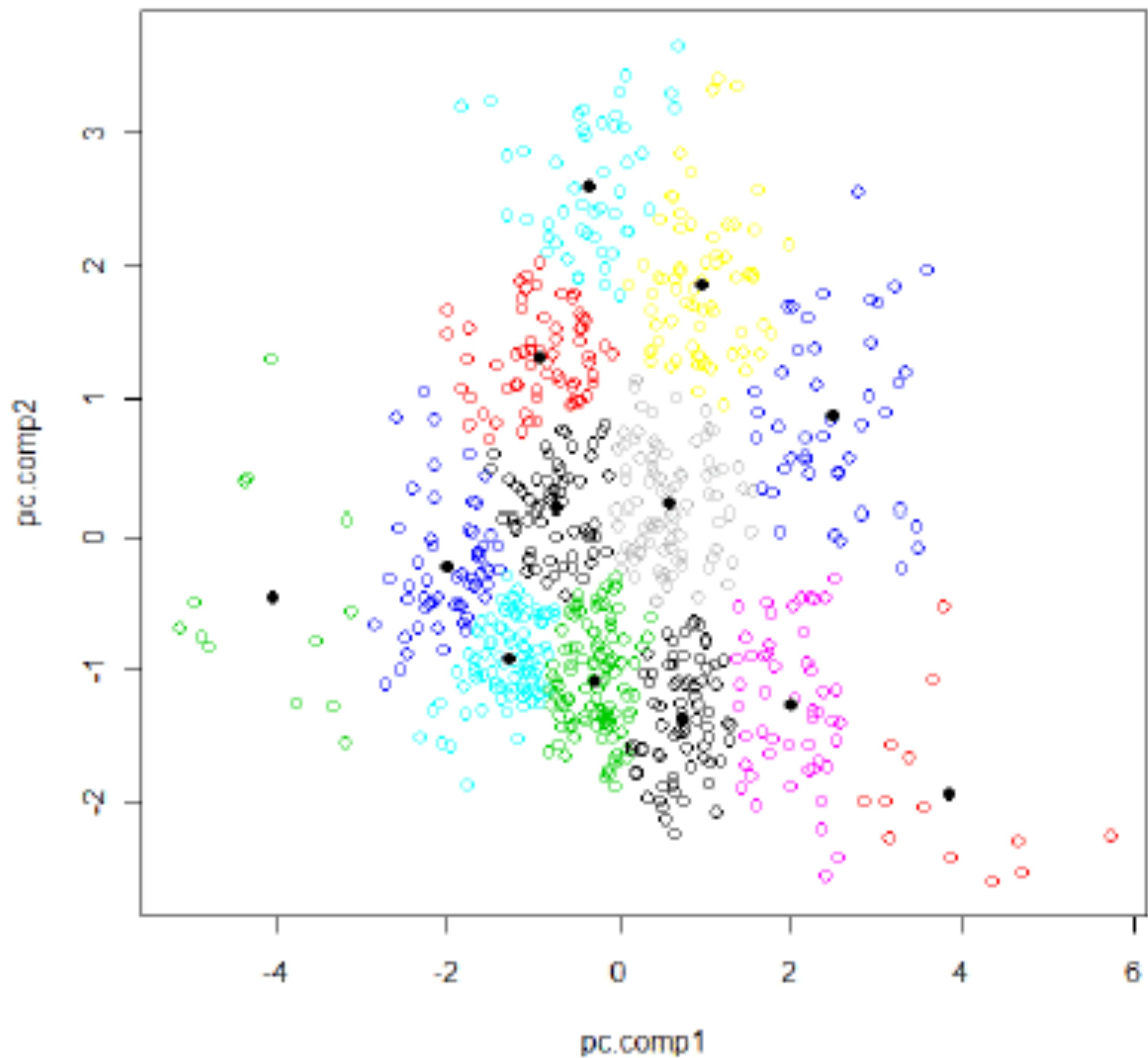
Ajouter \mathcal{C}_k à $L_{\mathcal{C}}$

Jusqu'à ce que $Card(L_{\mathcal{C}}) = 1$

Illustration



K-Means



K-Means

Principe et algorithme

- Trouver la meilleure partition en K clusters d'un ensemble de données
- Algorithme

Initialisation

Soit un dataset $D = \{x_i \in \mathbb{R}^d\}_{i=1}^N$

Choisir les centroïdes μ_k de clusters \mathcal{C}_k pour $k = 1..K, K < N$

Répéter

$x_i, i = 1..N$ est affecté au cluster \mathcal{C}_k dont le centroïde μ_k est le plus proche

Recalculer les centroïdes μ_k des clusters \mathcal{C}_k pour $k = 1..K$

Jusqu'à convergence

Illustration

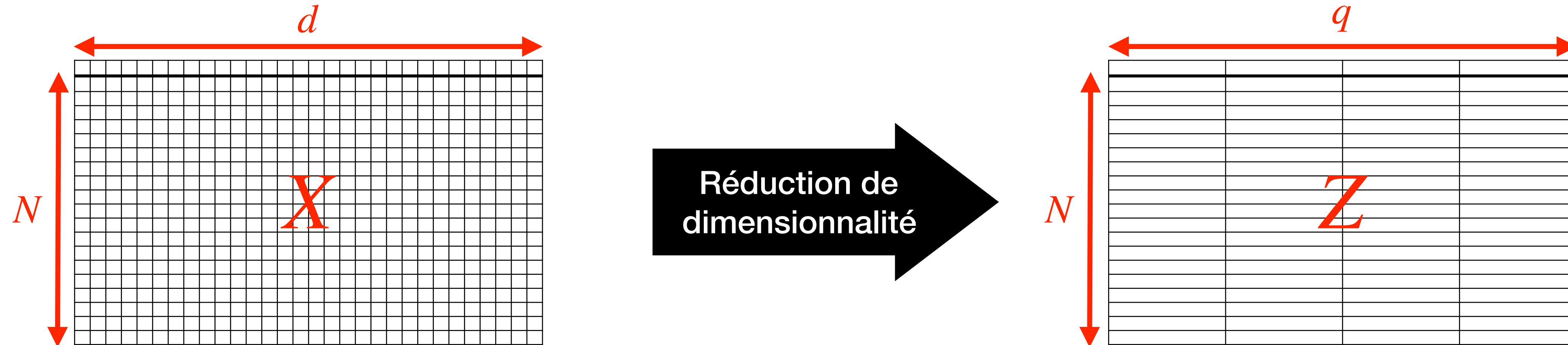
- Voir le notebook jupyter

Réduction de dimensionnalité

Réduction de dimensionnalité

Objectifs

- Soit $X \in \mathbb{R}^{N \times d}$ un dataset (N exemples ayant chacun d dimensions)
- Trouver une projection de X dans $Z \in \mathbb{R}^{N \times q}$ avec $q < d$



Utilisations Visualisation

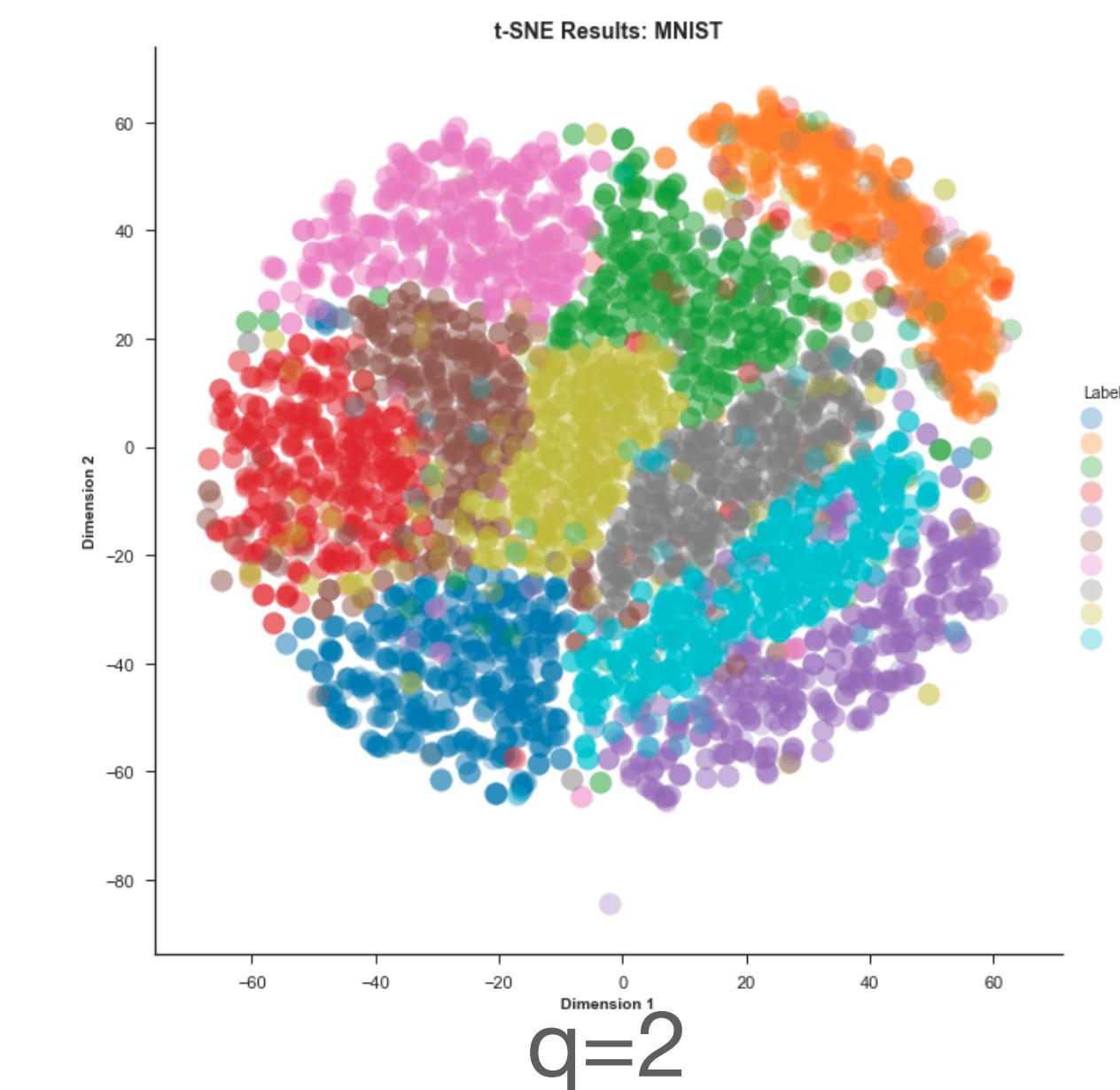


Perte de données

- Afficher les données
- Identifier les données aberrantes
- Visualiser les données en fonction de leur labels (si disponibles)

0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4
5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5
6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6
7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7
8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8
9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9

$$d = 28 \times 28 = 784$$



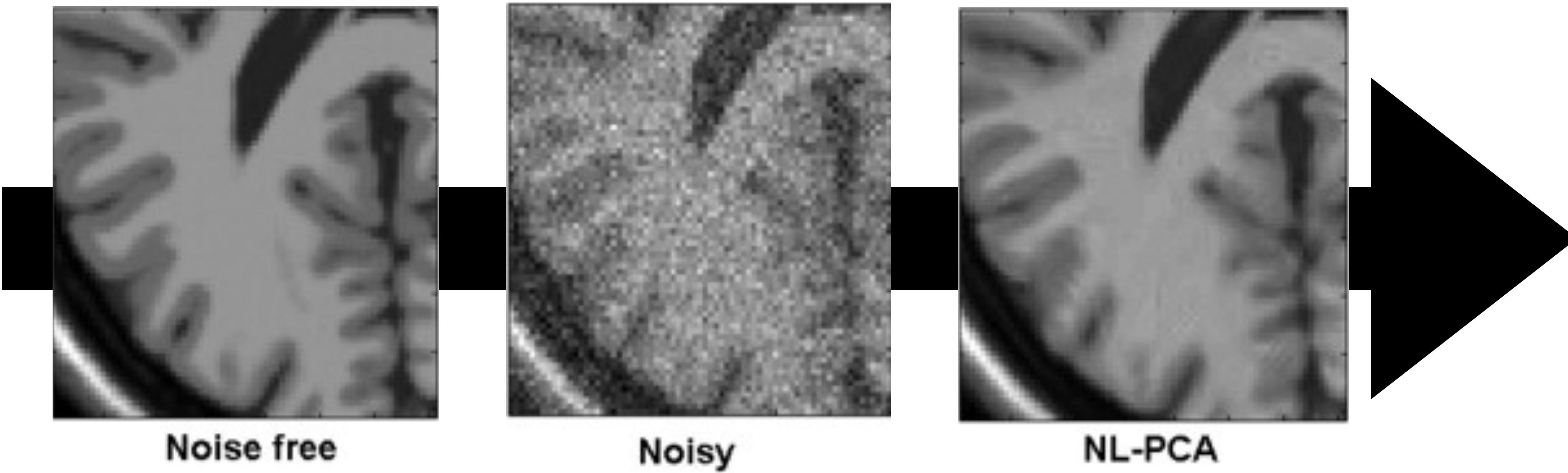
Utilisations

Représentation des données



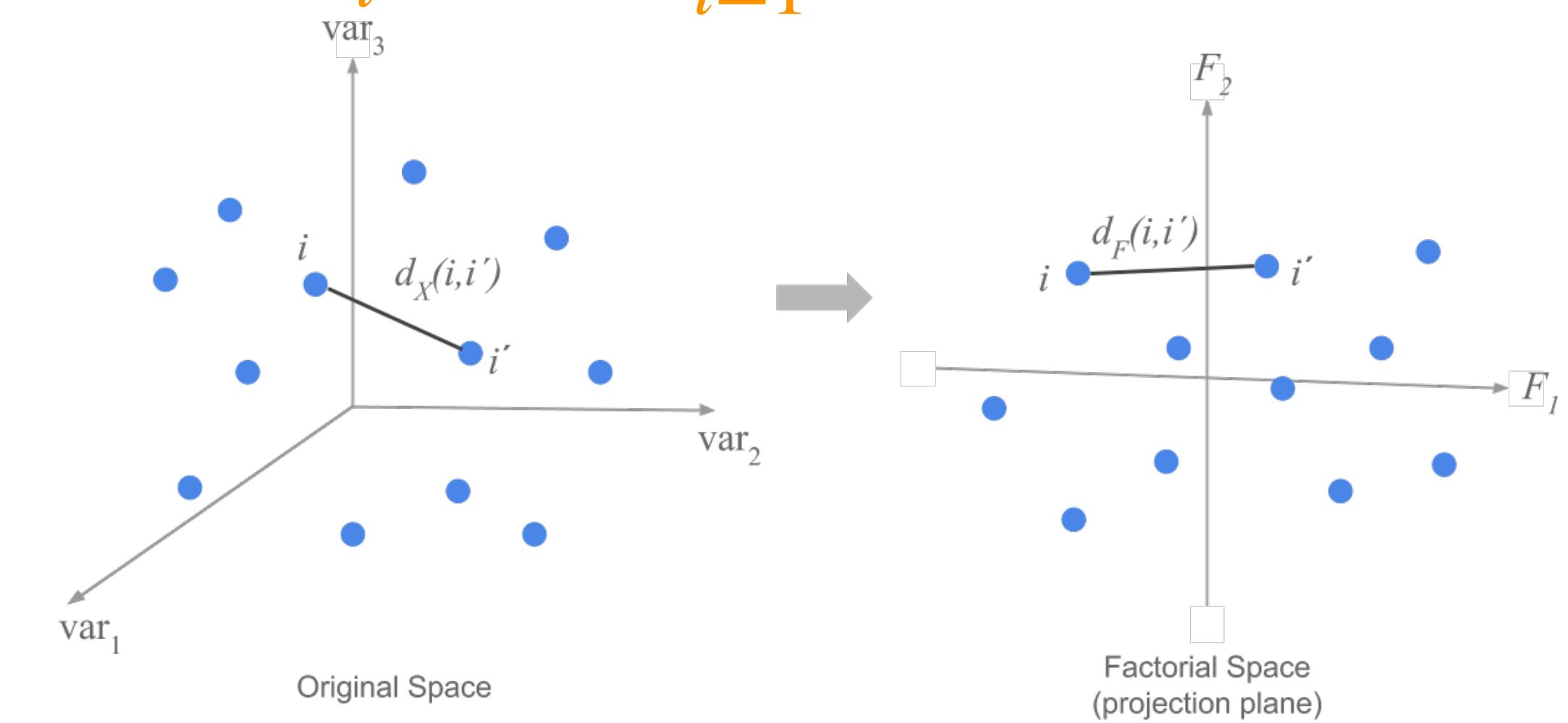
Perte de données

- Réduire les temps de calcul
- Réduire le bruit



Principes

- On veut projeter les exemples $\{x_i \in \mathbb{R}^d\}_{i=1}^N$ dans $\{z_i \in \mathbb{R}^q\}_{i=1}^N$ tout en préservant au mieux la topologie :
 - La distance entre les exemples
 - Leur voisinage (organisation topologique)
- 2 types de méthodes :
 - Réduction linéaire : Analyse en Composantes Principales (ACP ou PCA en anglais)
 - Réduction non-linéaire : Stochastic Neighbor Embedding (SNE)



Algorithme de l'Analyse en Composantes Principales

Etape 1 : standardisation des données

- Objectif :
 - Standardiser les domaines de valeurs de chacune des variables du dataset
 - Minimiser la variance des variables initiales pour la suite de l'algorithme
- Mathématiquement :

$$\{x_i \in \mathbb{R}^d\}_{i=1}^N \rightarrow \{x_{ij} = \frac{x_{ij} - \bar{x}_j}{\sigma_j}, j = 1, \dots, d\}$$

Algorithme de l'Analyse en Composantes Principales

Etape 2 : Calcul de la matrice de covariance

- Objectif : trouver les corrélations entre variables
- Covariance de deux variables $x \in \mathbb{R}^N$ et $y \in \mathbb{R}^N$

$$\sigma(x, y) = \sigma(y, x) = \frac{1}{N-1} \sum_{i=1}^N (x_i - \bar{x})(y_i - \bar{y})$$

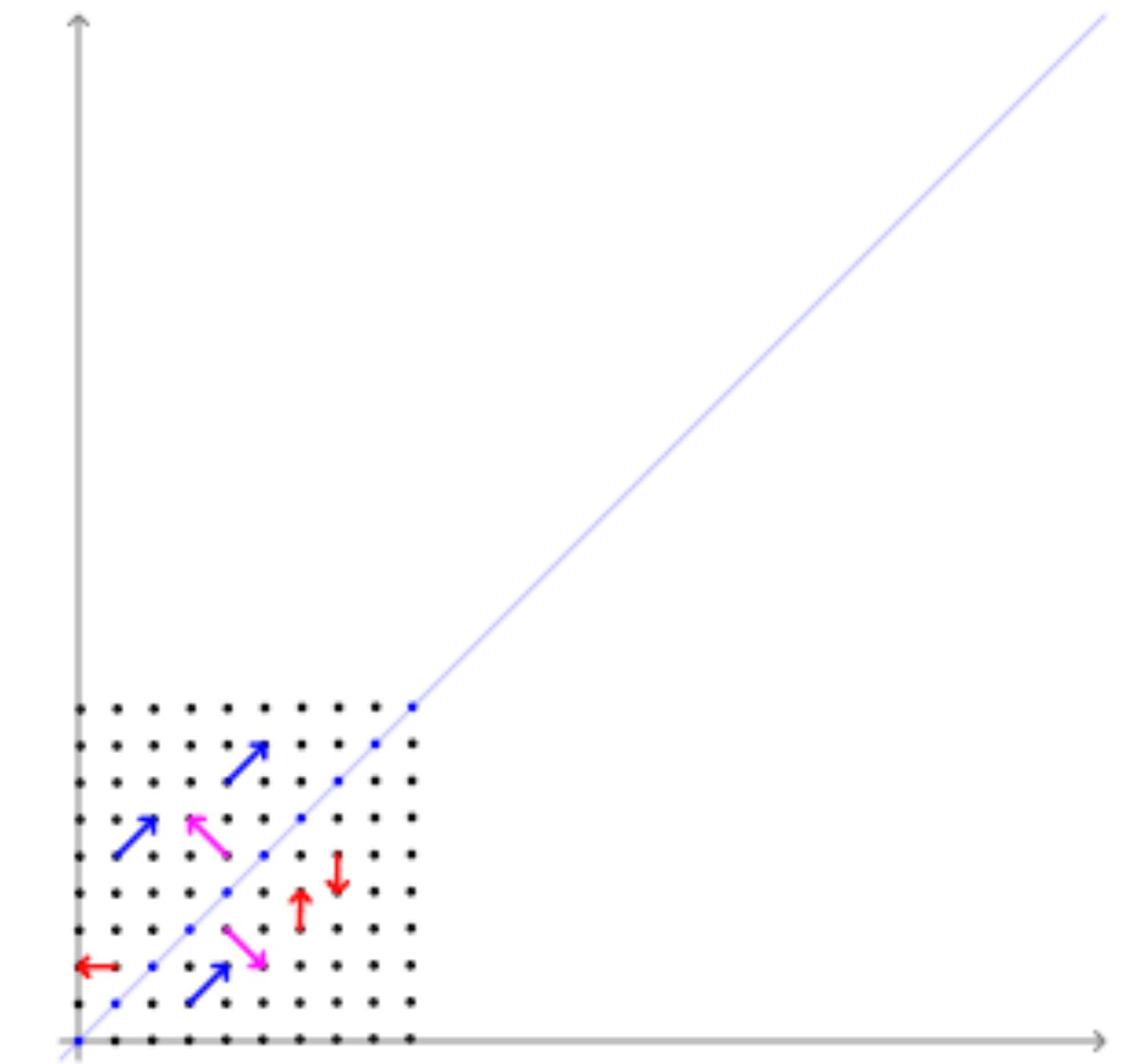
- Matrice de covariance en 3 variables $(x, y, z) \in \mathbb{R}^3$
- Importance signe des covariances
 - Si $\sigma(x, y) > 0$, quand x augmente, y augmente aussi
 - Si $\sigma(x, y) < 0$, quand x augmente, y diminue aussi

$$C = \begin{pmatrix} \sigma(x, x) & \sigma(x, y) & \sigma(x, z) \\ \sigma(y, x) & \sigma(y, y) & \sigma(y, z) \\ \sigma(z, x) & \sigma(z, y) & \sigma(z, z) \end{pmatrix}$$

Algorithme de l'Analyse en Composantes Principales

Etape 3 : Calcul des vecteurs et valeurs propre

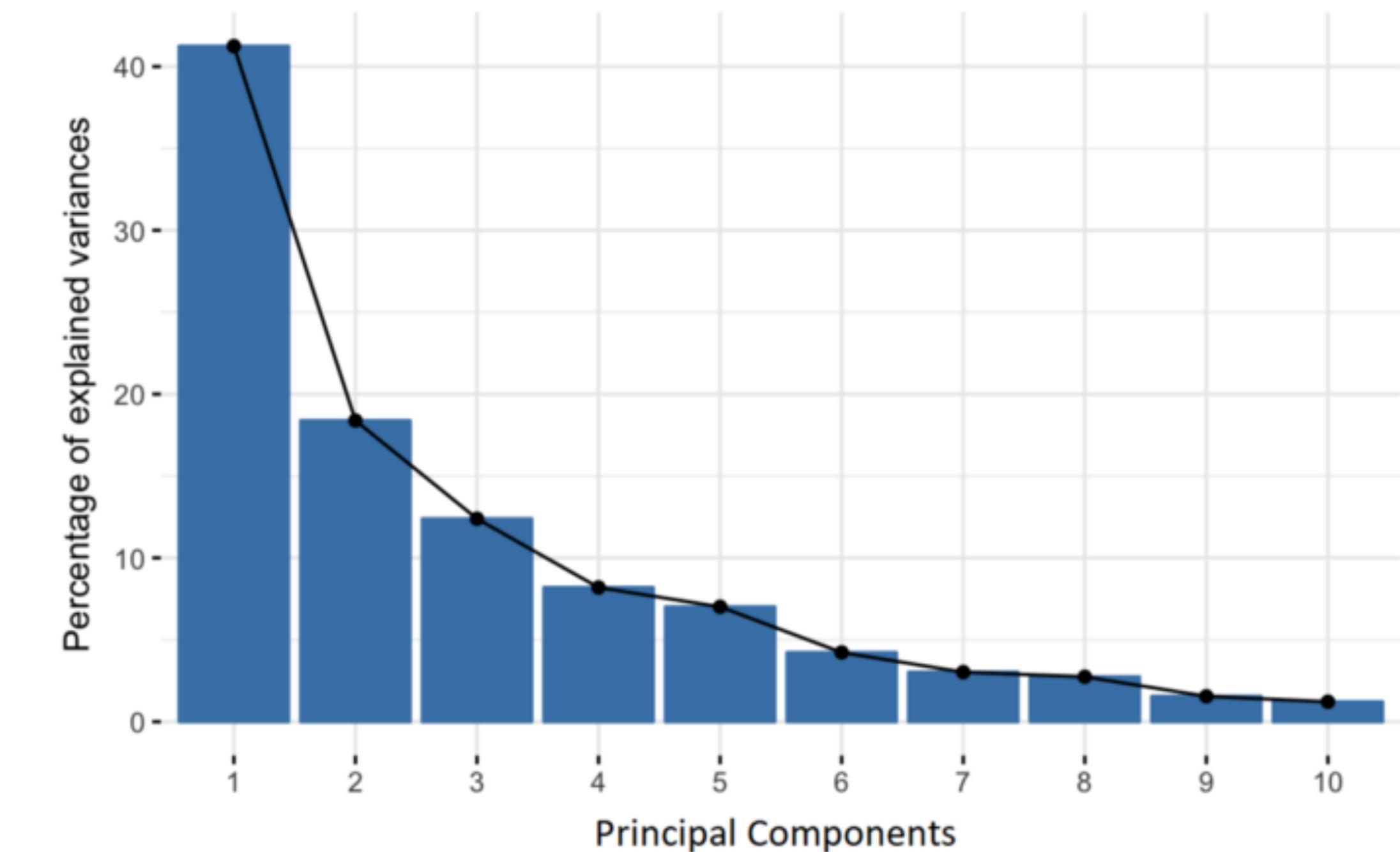
- Notion de vecteurs et valeurs propre
 - Concepts algébrique permettant de transformer linéairement une matrice
 - On cherche les vecteurs propres $X \in \mathbb{R}^d$ et leur valeur propre $\lambda \in \mathbb{R}$ tels quels $AX = \lambda X$ avec $A \in \mathbb{R}^{d \times d}$ une matrice carrée
 - En résumé, on cherche à résoudre le système d'équation $(A - \lambda \text{Id})X = 0$ avec $\text{Id} \in \mathbb{R}^d$ la matrice identité



Algorithme de l'Analyse en Composantes Principales

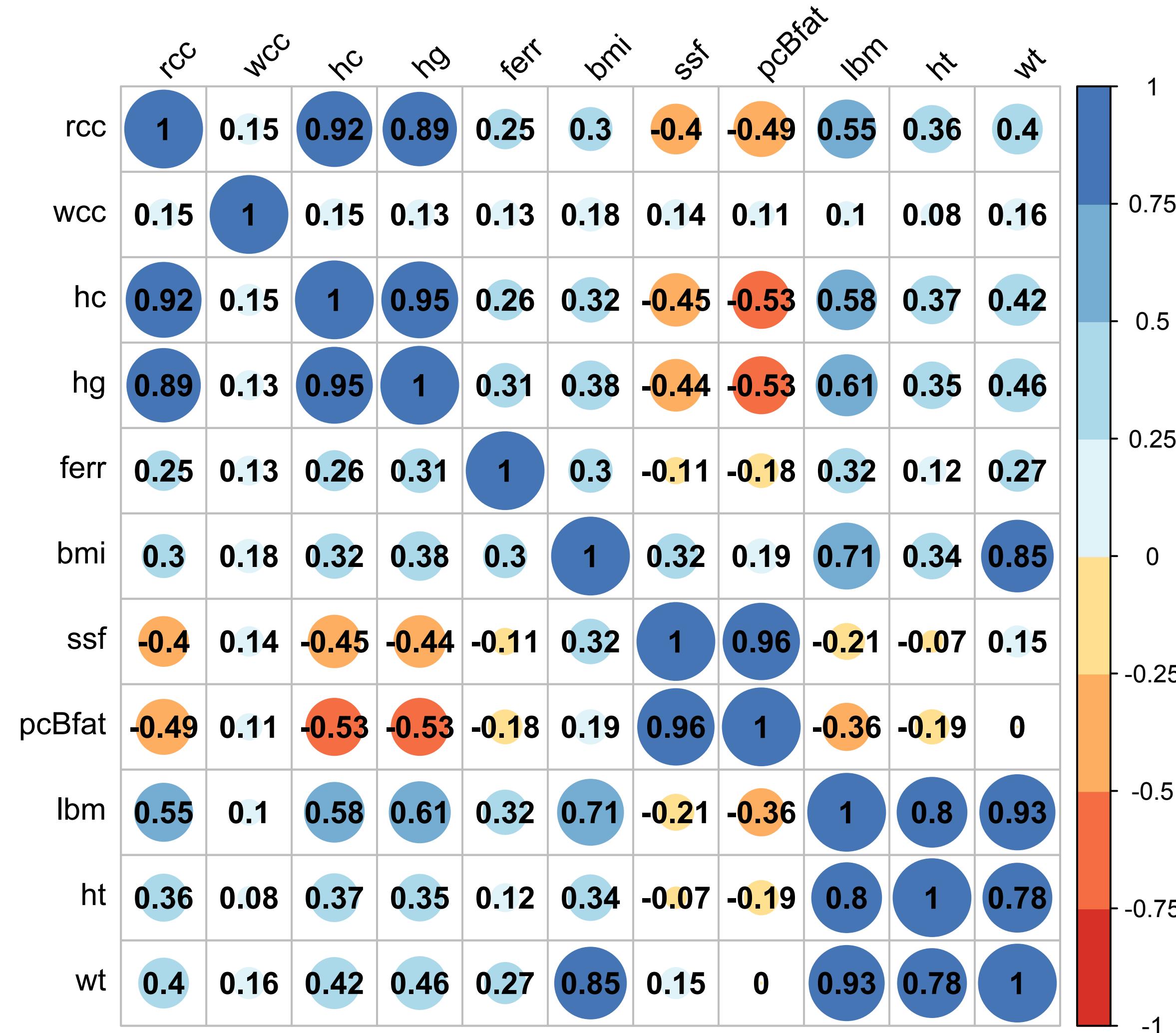
Etape 3 : Calcul des vecteurs et valeurs propre de la matrice de covariance

- L'application à la matrice de covariance permet de trouver les **composantes principales** des données
- Composantes principales
 - Construction linéaire des variables initiales
 - Les nouvelles variables (composantes principales) sont aussi peu corrélées que possible
 - Le maximum des informations des variables initiales sont “compressées” dans les premières composantes



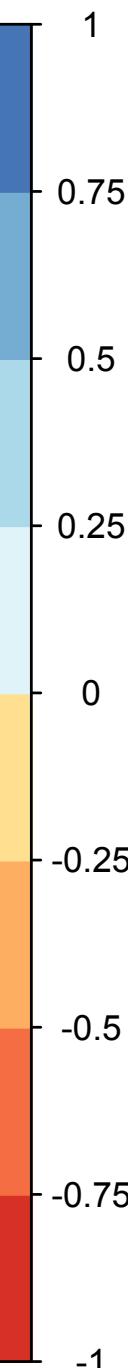
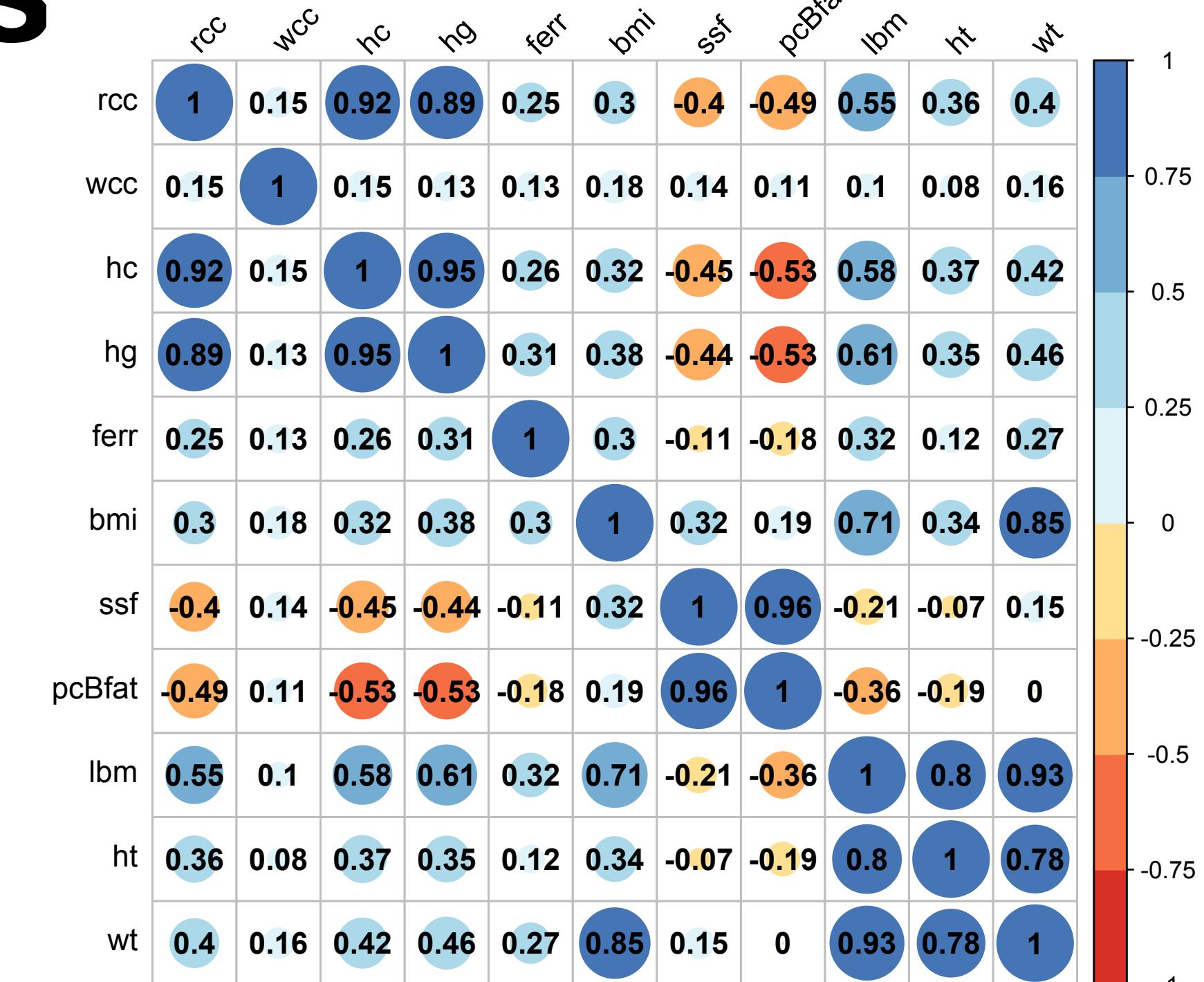
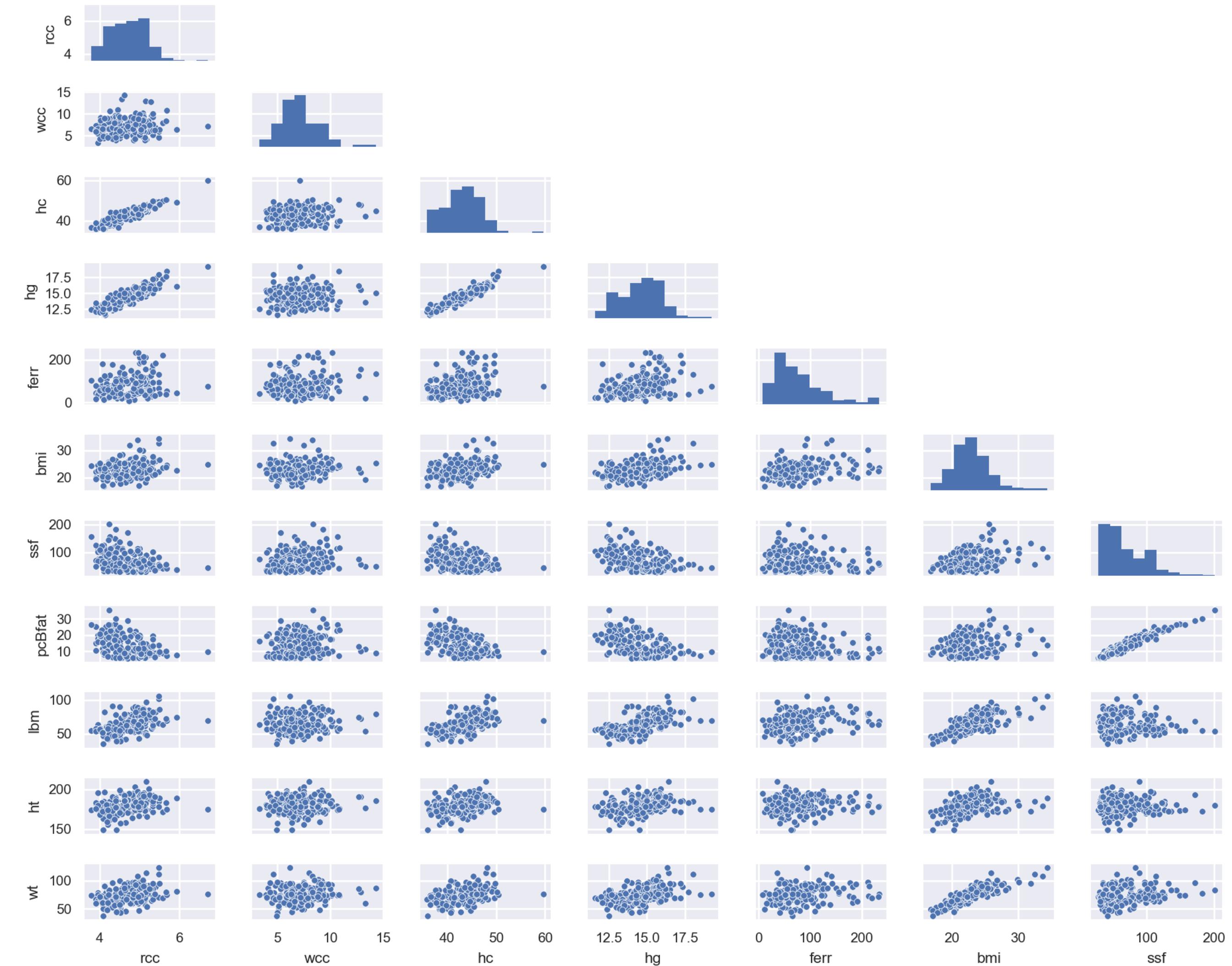
Applications supplémentaires

Matrice de corrélation



Applications supplémentaires

Pairplots



Analyse en composantes principales

Illustration

- Note: LDA (Linear Discriminant Analysis) autre algorithme de réduction linéaire
- Voir Notebook Jupyter

t-distributed Stochastic Neighbor Embedding (t-SNE)

Principes et objectifs

- Méthode de réduction de dimensionnalité non linéaire
- Permet d'identifier des patterns pertinents
- Préserve les structures locales
- Principales utilisations
 - Visualisation de données
 - Aide à estimer la possibilité d'utilisation du machine learning sur des données
- Note : UMap est un algorithme équivalent à t-SNE, aussi utilisé en data viz

t-distributed Stochastic Neighbor Embedding (t-SNE)

Intuition de fonctionnement

- t-SNE estime la distribution de probabilité de voisinage autour de chaque point
- Cherche à minimiser les différences entre les distributions de deux points pour les répartir dans un espace 2D
- Le paramètre principale de l'algorithme est la *perplexité* qui équivaut au nombre de voisins considérés (souvent entre 5 et 50) par l'algorithme
 - Une perplexité basse signifie qu'on se focalise sur les voisinages locaux
 - Une perplexité haute signifie qu'on se focalise sur les voisinages plus larges