

Classification non supervisée - Clustering

Dpt informatique - IUT Nancy-Charlemagne

24 mai 2024



Plan

1 Classification non supervisée

2 Algorithme KMeans

3 Algorithme DBSCAN

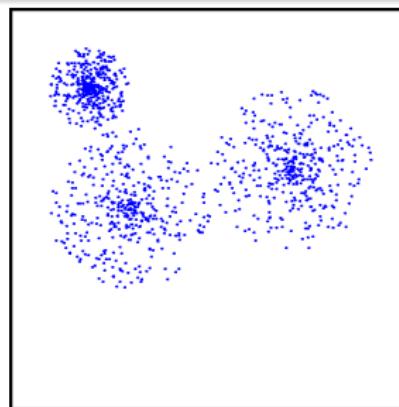
4 Algorithme HAC

5 Choix et validations

Classification non supervisée

Classification non supervisée :

- **Classification** : attribuer des étiquettes à des données
- **Non supervisé** : pas d'étiquettes à reconnaître



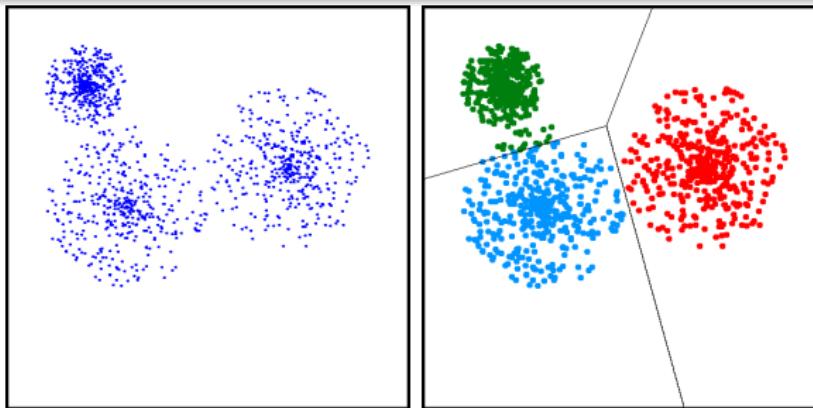
Idée

- Regrouper les données par groupes (clusters) → **Clustering**

Clustering / Regroupement

Point de départ

- **Données** $D = \{d_i\}_{i \in [1:n]}$ et **distance** $dist(d_1, d_2)$
- un nombre de groupes n_g

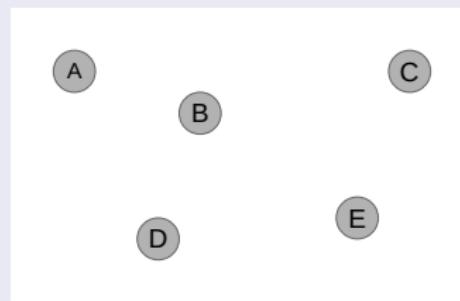


Objectif

- Regrouper en n_g groupes les données qui se ressemblent
- Attribuer à chaque donnée un numéro de groupe entre 1 et n_g

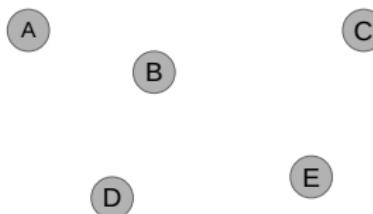
Exemple de regroupements

Données ABCDE en deux groupes

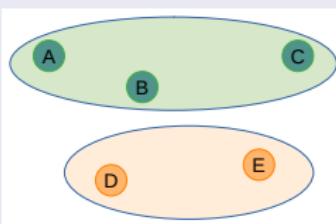


Exemple de regroupements

Données ABCDE en deux groupes

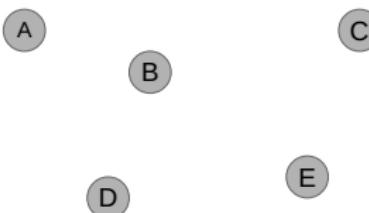


[ABC]+[DE]

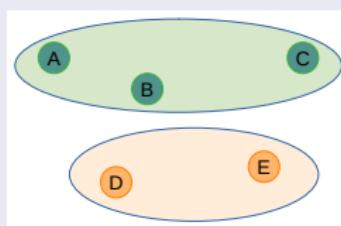


Exemple de regroupements

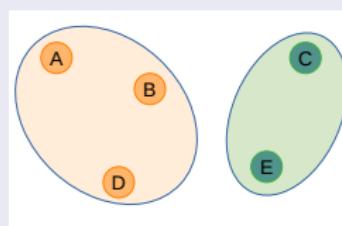
Données ABCDE en deux groupes



[ABC]+[DE]

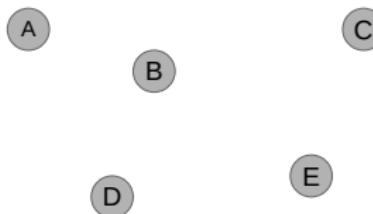


[ABD]+[CE]

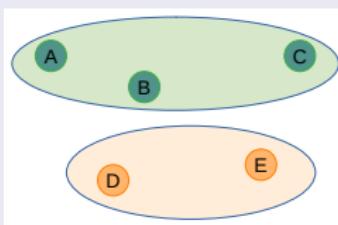


Exemple de regroupements

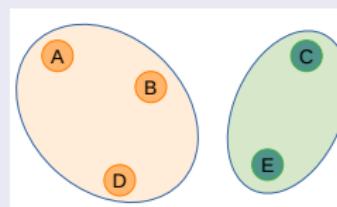
Données ABCDE en deux groupes



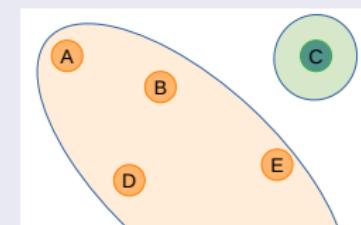
[ABC]+[DE]



[ABD]+[CE]



[ABDE]+[C]



Mesure pour évaluer un regroupement

Variabilité intra-classe

- pour chaque groupe
 - la somme des distances entre les points du groupe au centre
- Minimiser coût du regroupement

$$Cout = \sum_{g \in Groupes} [\sum_{d_i \in g} dist(d_i, \mu_g)^2]$$

[ABC]+[DE]

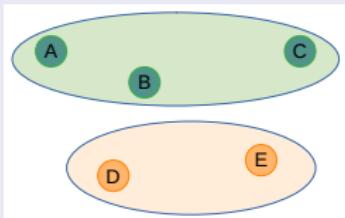
Mesure pour évaluer un regroupement

Variabilité intra-classe

- pour chaque groupe
 - la somme des distances entre les points du groupe au centre
- Minimiser coût du regroupement

$$Cout = \sum_{g \in Groupes} \left[\sum_{d_i \in g} dist(d_i, \mu_g)^2 \right]$$

[ABC]+[DE]



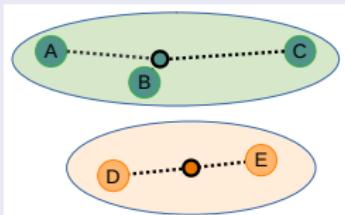
Mesure pour évaluer un regroupement

Variabilité intra-classe

- pour chaque groupe
 - la somme des distances entre les points du groupe au centre
- Minimiser coût du regroupement

$$Cout = \sum_{g \in Groupes} \left[\sum_{d_i \in g} dist(d_i, \mu_g)^2 \right]$$

[ABC]+[DE]



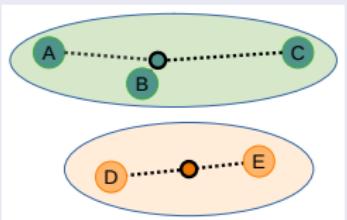
Mesure pour évaluer un regroupement

Variabilité intra-classe

- pour chaque groupe
 - la somme des distances entre les points du groupe au centre
- Minimiser coût du regroupement

$$Cout = \sum_{g \in Groupes} [\sum_{d_i \in g} dist(d_i, \mu_g)^2]$$

[ABC]+[DE]



$$A\mu_1^2 + B\mu_1^2 + C\mu_1^2 + D\mu_2^2 + E\mu_2^2$$

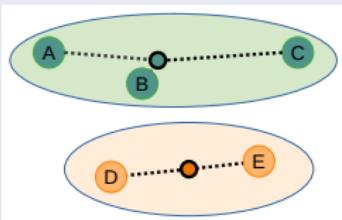
Mesure pour évaluer un regroupement

Variabilité intra-classe

- pour chaque groupe
 - la somme des distances entre les points du groupe au centre
- Minimiser coût du regroupement

$$Cout = \sum_{g \in Groupes} [\sum_{d_i \in g} dist(d_i, \mu_g)^2]$$

[ABC]+[DE]



$$A\mu_1^2 + B\mu_1^2 + C\mu_1^2 + D\mu_2^2 + E\mu_2^2$$

[ABD]+[CE]

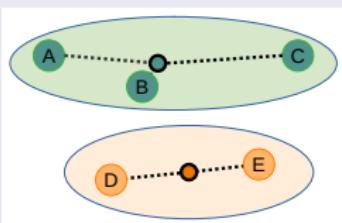
Mesure pour évaluer un regroupement

Variabilité intra-classe

- pour chaque groupe
 - la somme des distances entre les points du groupe au centre
- Minimiser coût du regroupement

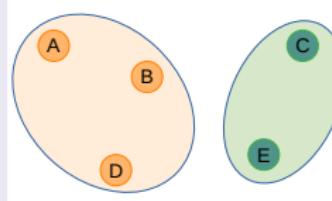
$$Cout = \sum_{g \in Groupes} [\sum_{d_i \in g} dist(d_i, \mu_g)^2]$$

[ABC]+[DE]



$$A\mu_1^2 + B\mu_1^2 + C\mu_1^2 + D\mu_2^2 + E\mu_2^2$$

[ABD]+[CE]



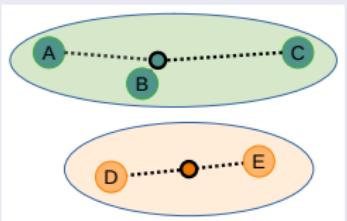
Mesure pour évaluer un regroupement

Variabilité intra-classe

- pour chaque groupe
 - la somme des distances entre les points du groupe au centre
- Minimiser coût du regroupement

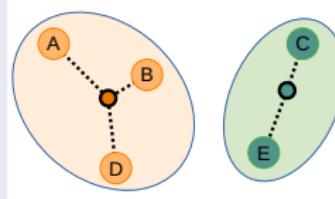
$$Cout = \sum_{g \in Groupes} [\sum_{d_i \in g} dist(d_i, \mu_g)^2]$$

[ABC]+[DE]



$$A\mu_1^2 + B\mu_1^2 + C\mu_1^2 + D\mu_2^2 + E\mu_2^2$$

[ABD]+[CE]



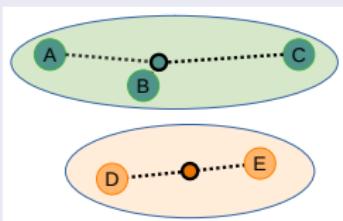
Mesure pour évaluer un regroupement

Variabilité intra-classe

- pour chaque groupe
 - la somme des distances entre les points du groupe au centre
- Minimiser coût du regroupement

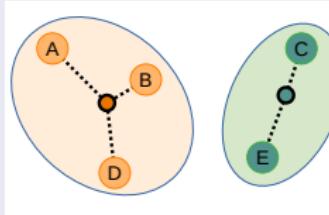
$$Cout = \sum_{g \in Groupes} [\sum_{d_i \in g} dist(d_i, \mu_g)^2]$$

[ABC]+[DE]



$$A\mu_1^2 + B\mu_1^2 + C\mu_1^2 + D\mu_2^2 + E\mu_2^2$$

[ABD]+[CE]



$$C\mu_1^2 + E\mu_1^2 + A\mu_2^2 + B\mu_2^2 + D\mu_2^2$$

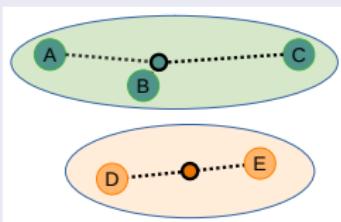
Mesure pour évaluer un regroupement

Variabilité intra-classe

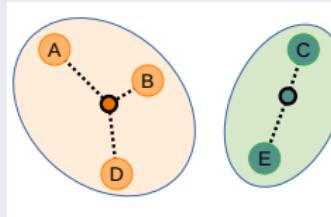
- pour chaque groupe
 - la somme des distances entre les points du groupe au centre
- Minimiser coût du regroupement

$$Cout = \sum_{g \in Groupes} [\sum_{d_i \in g} dist(d_i, \mu_g)^2]$$

[ABC]+[DE]



[ABD]+[CE]



$$A\mu_1^2 + B\mu_1^2 + C\mu_1^2 + D\mu_2^2 + E\mu_2^2$$

[ABDE]+[C]

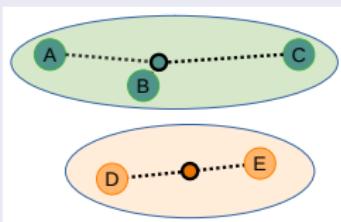
Mesure pour évaluer un regroupement

Variabilité intra-classe

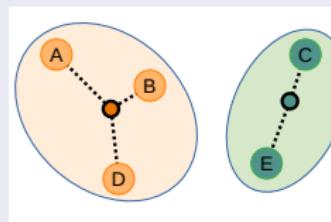
- pour chaque groupe
 - la somme des distances entre les points du groupe au centre
- Minimiser coût du regroupement

$$Cout = \sum_{g \in Groupes} [\sum_{d_i \in g} dist(d_i, \mu_g)^2]$$

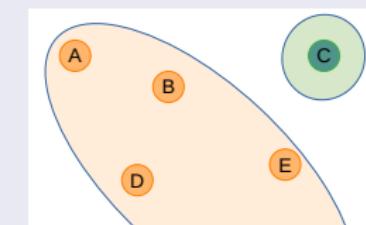
[ABC]+[DE]



[ABD]+[CE]



[ABDE]+[C]



$$A\mu_1^2 + B\mu_1^2 + C\mu_1^2 + D\mu_2^2 + E\mu_2^2$$

$$C\mu_1^2 + E\mu_1^2 + A\mu_2^2 + B\mu_2^2 + D\mu_2^2$$

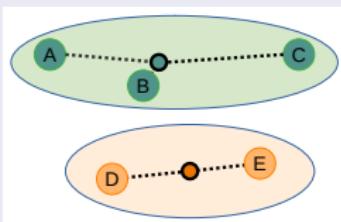
Mesure pour évaluer un regroupement

Variabilité intra-classe

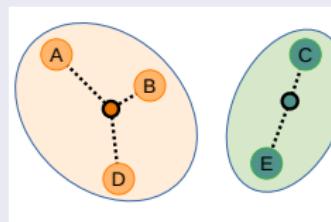
- pour chaque groupe
 - la somme des distances entre les points du groupe au centre
- Minimiser coût du regroupement

$$Cout = \sum_{g \in Groupes} [\sum_{d_i \in g} dist(d_i, \mu_g)^2]$$

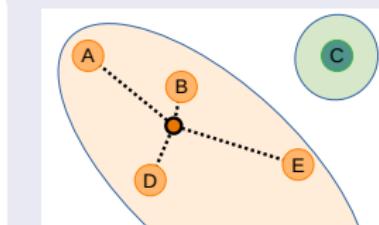
[ABC]+[DE]



[ABD]+[CE]



[ABDE]+[C]



$$A\mu_1^2 + B\mu_1^2 + C\mu_1^2 + D\mu_2^2 + E\mu_2^2$$

$$C\mu_1^2 + E\mu_1^2 + A\mu_2^2 + B\mu_2^2 + D\mu_2^2$$

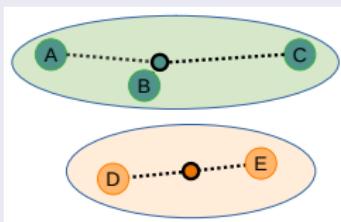
Mesure pour évaluer un regroupement

Variabilité intra-classe

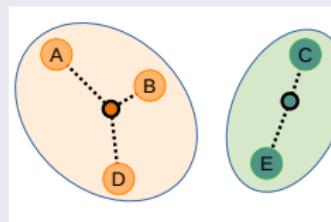
- pour chaque groupe
 - la somme des distances entre les points du groupe au centre
- Minimiser coût du regroupement

$$Cout = \sum_{g \in Groupes} [\sum_{d_i \in g} dist(d_i, \mu_g)^2]$$

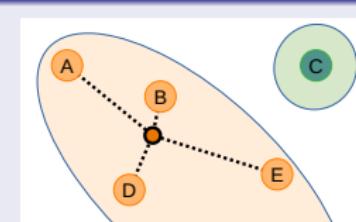
[ABC]+[DE]



[ABD]+[CE]



[ABDE]+[C]



$$A\mu_1^2 + B\mu_1^2 + C\mu_1^2 + D\mu_2^2 + E\mu_2^2$$

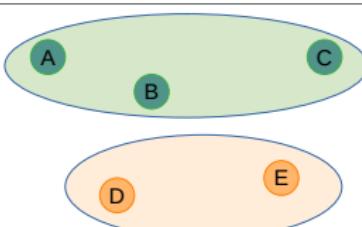
$$C\mu_1^2 + E\mu_1^2 + A\mu_2^2 + B\mu_2^2 + D\mu_2^2$$

$$0 + A\mu_1^2 + B\mu_1^2 + D\mu_2^2 + E\mu_2^2$$

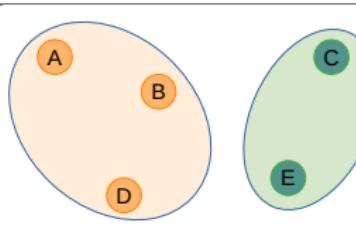
Algorithme basique

Idée

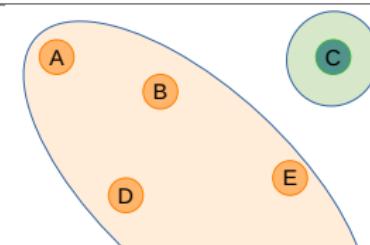
- Enumérer tous les regroupements et choisir le meilleur
- **Représentation** - Regroupement = vecteur d'affectations



ABCDE & [00011]
= [ABC]+[DE]



ABCDE & [11010]
= [ABD]+[CE]



ABCDE & [11011]
= [ABDE]+[C]

Dénombrement

Choix d'un regroupement

- pour chaque donnée, choisir son groupe parmi n_g
- nombre d'affectations : $n_g^{|D|}$
- diviser par permutations : $n_g!$
 - ① ABCDE + [00011] → [ABC] + [DE]
 - ② ABCDE + [11100] → [DE] + [ABC]

Dénombrement

Choix d'un regroupement

- pour chaque donnée, choisir son groupe parmi n_g
- nombre d'affectations : $n_g^{|D|}$
- diviser par permutations : $n_g!$
 - ① ABCDE + [00011] → [ABC] + [DE]
 - ② ABCDE + [11100] → [DE] + [ABC]

Exemples : 1 regroupement = 1 nanoseconde

Dénombrement

Choix d'un regroupement

- pour chaque donnée, choisir son groupe parmi n_g
- nombre d'affectations : $n_g^{|D|}$
- diviser par permutations : $n_g!$
 - ① ABCDE + [00011] → [ABC] + [DE]
 - ② ABCDE + [11100] → [DE] + [ABC]

Exemples : 1 regroupement = 1 nanoseconde

- 2 groupes et 10 données
 - $2^{10} = 1024 \sim 10^{-6}$ s

Dénombrement

Choix d'un regroupement

- pour chaque donnée, choisir son groupe parmi n_g
- nombre d'affectations : $n_g^{|D|}$
- diviser par permutations : $n_g!$
 - ① ABCDE + [00011] → [ABC] + [DE]
 - ② ABCDE + [11100] → [DE] + [ABC]

Exemples : 1 regroupement = 1 nanoseconde

- 2 groupes et 10 données
 - $\rightarrow 2^{10} = 1024 \sim 10^{-6}$ s
- 3 groupes et 10 données
 - $\rightarrow 3^{10} = 59049 \sim 0.05$ s

Dénombrement

Choix d'un regroupement

- pour chaque donnée, choisir son groupe parmi n_g
- nombre d'affectations : $n_g^{|D|}$
- diviser par permutations : $n_g!$
 - ① ABCDE + [00011] → [ABC] + [DE]
 - ② ABCDE + [11100] → [DE] + [ABC]

Exemples : 1 regroupement = 1 nanoseconde

- 2 groupes et 10 données
 - $\rightarrow 2^{10} = 1024 \sim 10^{-6}$ s
- 3 groupes et 10 données
 - $\rightarrow 3^{10} = 59049 \sim 0.05$ s
- 3 groupes et 50 données
 - $\rightarrow 3^{50} = 7.10^{23} \sim 7.10^{14}$ s = 23.10^6 ans

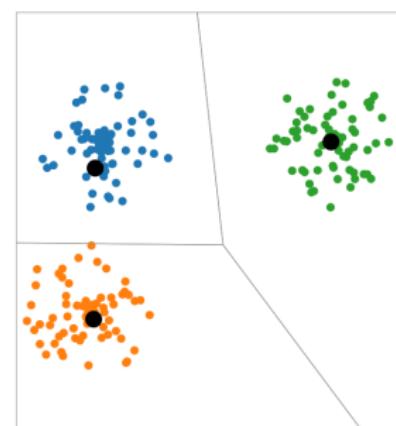
Algorithmes à base de prototypes

Principes

- Représente les groupes de données à l'aide de représentants (par ex. les barycentres)
- Exemples d'algorithmes : K-Means, Fuzzy C-Means

Propriétés

- Avantages : Faible complexité $O(N)$
- Inconvénients : nécessite de choisir le nombre de clusters et de conjecturer leur forme.



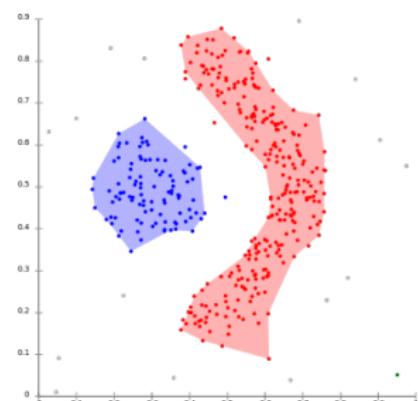
Algorithmes basés sur la densité

Principes

- Inspiré de la vision humaine : les groupes sont définis comme des espaces à haute densité et sont séparés par des zones à faible densité.
- Exemples d'algorithmes : DBSCAN, OPTICS

Propriétés

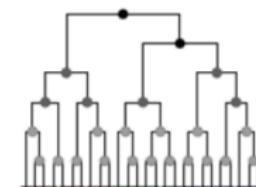
- Avantages : Ne présume pas de la forme des groupes ou de leur nombre, détecte les valeurs aberrantes.
- Inconvénients : Haute complexité $O(N^2)$, difficile à paramétrier.



Algorithmes hiérarchiques

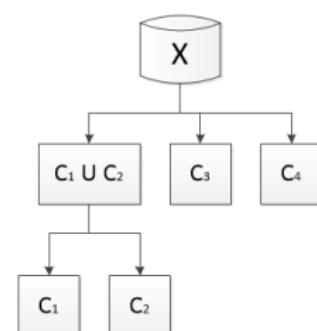
Principes

- L'idée est de construire un arbre de similarités entre données, les groupes correspondent à différentes branches de l'arbre.
- Exemples d'algorithmes : CAH, CURE, CLINK



Propriétés

- Avantages : Mise en évidence des structures hiérarchiques des groupes, visualisation complète des groupes.
- Inconvénients : Complexité élevée $O(N^2 \log N)$ à $O(N^3)$, le choix de l'endroit où couper l'arbre est difficile.



Plan

1 Classification non supervisée

2 Algorithme KMeans

3 Algorithme DBSCAN

4 Algorithme HAC

5 Choix et validations

KMeans - Notion de centroïdes

Centroïde

- Centroïde = valeur censée représenter un groupe
- Calcul : barycentre des données du groupe (\sim moyenne)

En 2D, centroïde d'un groupe G

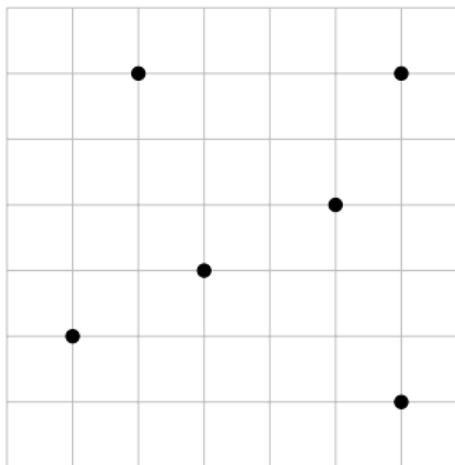
$$\bullet \quad x_c = \frac{\sum_{d \in G} x_d}{|G|}$$

$$\bullet \quad y_c = \frac{\sum_{d \in G} y_d}{|G|}$$

KMeans - Barycentre

Exemple

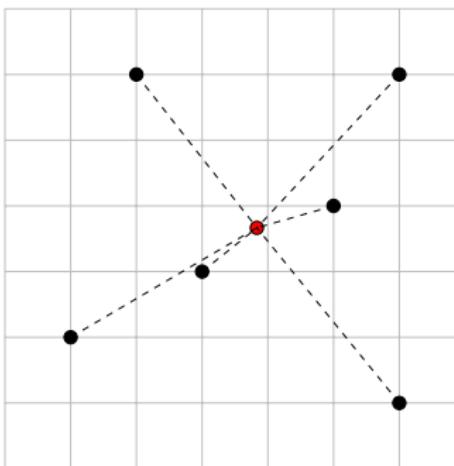
- A(1,2); B(5,4); C(2,6); D(6,6); E(3,3); F(6,1)
- Barycentre → (3.8333,3.666)



KMeans - Barycentre

Exemple

- A(1,2); B(5,4); C(2,6); D(6,6); E(3,3); F(6,1)
- Barycentre \rightarrow (3.8333,3.666)



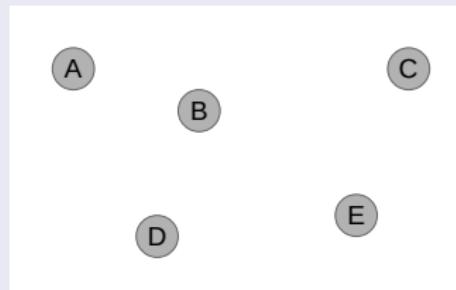
KMeans - Principe de l'algorithme (1)

Principe général

- ➊ Initialiser Centroïdes
- ➋ Répéter
 - ➌ Construire groupes à partir des centroïdes
 - ➍ Mettre à jour centroïdes en fonction des groupes

1. Centroïdes → Groupes

- Pour chaque donnée, affecter groupe du centre de cluster le + proche.



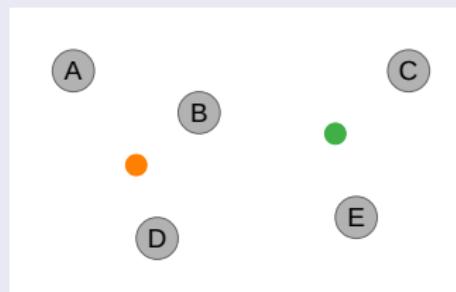
KMeans - Principe de l'algorithme (1)

Principe général

- ➊ Initialiser Centroïdes
- ➋ Répéter
 - ➌ Construire groupes à partir des centroïdes
 - ➍ Mettre à jour centroïdes en fonction des groupes

1. Centroïdes → Groupes

- Pour chaque donnée, affecter groupe du centre de cluster le + proche.



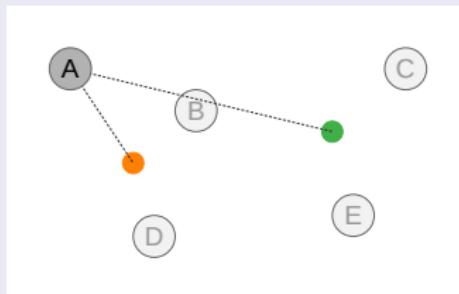
KMeans - Principe de l'algorithme (1)

Principe général

- ① Initialiser Centroïdes
- ② Répéter
 - ① Construire groupes à partir des centroïdes
 - ② Mettre à jour centroïdes en fonction des groupes

1. Centroïdes → Groupes

- Pour chaque donnée, affecter groupe du centre de données le + proche.



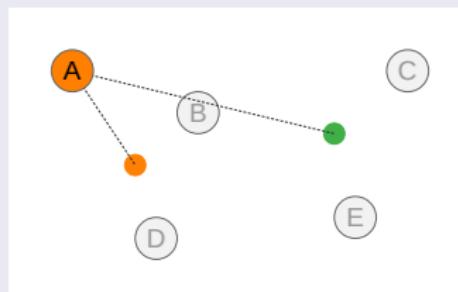
KMeans - Principe de l'algorithme (1)

Principe général

- ➊ Initialiser Centroïdes
- ➋ Répéter
 - ➌ Construire groupes à partir des centroïdes
 - ➍ Mettre à jour centroïdes en fonction des groupes

1. Centroïdes → Groupes

- Pour chaque donnée, affecter groupe du centre de cluster le + proche.



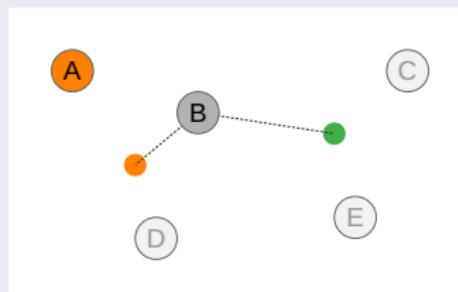
KMeans - Principe de l'algorithme (1)

Principe général

- ➊ Initialiser Centroïdes
- ➋ Répéter
 - ➌ Construire groupes à partir des centroïdes
 - ➍ Mettre à jour centroïdes en fonction des groupes

1. Centroïdes → Groupes

- Pour chaque donnée, affecter groupe du centre de cluster le + proche.



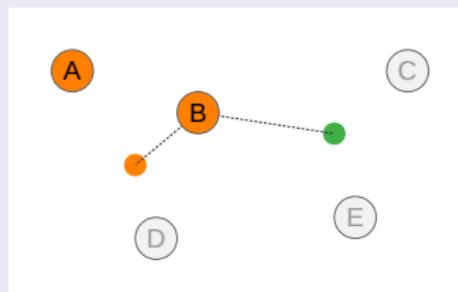
KMeans - Principe de l'algorithme (1)

Principe général

- ➊ Initialiser Centroïdes
- ➋ Répéter
 - ➌ Construire groupes à partir des centroïdes
 - ➍ Mettre à jour centroïdes en fonction des groupes

1. Centroïdes → Groupes

- Pour chaque donnée, affecter groupe du centre de données le + proche.



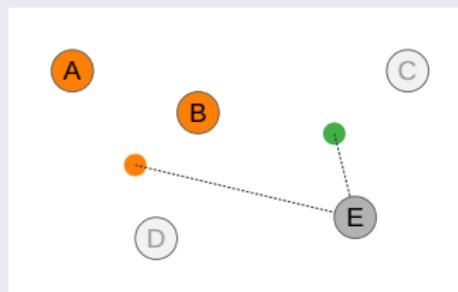
KMeans - Principe de l'algorithme (1)

Principe général

- ➊ Initialiser Centroïdes
- ➋ Répéter
 - ➌ Construire groupes à partir des centroïdes
 - ➍ Mettre à jour centroïdes en fonction des groupes

1. Centroïdes → Groupes

- Pour chaque donnée, affecter groupe du centre de cluster le + proche.



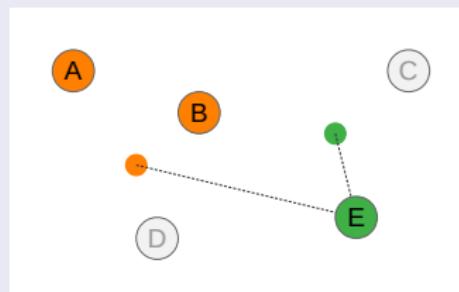
KMeans - Principe de l'algorithme (1)

Principe général

- ➊ Initialiser Centroïdes
- ➋ Répéter
 - ➌ Construire groupes à partir des centroïdes
 - ➍ Mettre à jour centroïdes en fonction des groupes

1. Centroïdes → Groupes

- Pour chaque donnée, affecter groupe du centre de cluster le + proche.



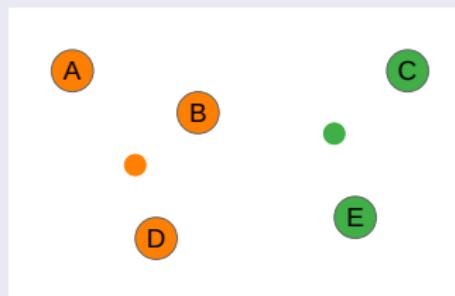
KMeans - Principe de l'algorithme (1)

Principe général

- ➊ Initialiser Centroïdes
- ➋ Répéter
 - ➌ Construire groupes à partir des centroïdes
 - ➍ Mettre à jour centroïdes en fonction des groupes

1. Centroïdes → Groupes

- Pour chaque donnée, affecter groupe du centre de cluster le + proche.



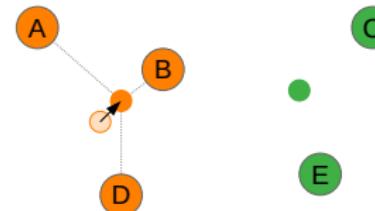
KMeans - Principe de l'algorithme (2)

Principe général

- ① Initialiser Centroïdes
- ② Répéter
 - ① Construire groupes à partir des centroïdes
 - ② Mettre à jour centroïdes en fonction des groupes

2. Groupes → Centroïdes

- Pour chaque groupe, centroïde = barycentre(pointsGroupe).



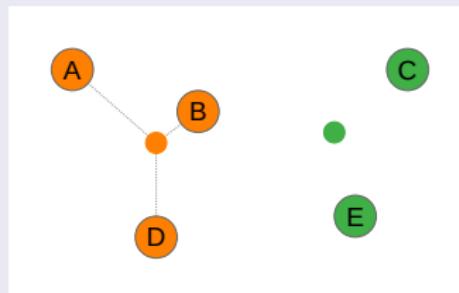
KMeans - Principe de l'algorithme (2)

Principe général

- ① Initialiser Centroïdes
- ② Répéter
 - ① Construire groupes à partir des centroïdes
 - ② Mettre à jour centroïdes en fonction des groupes

2. Groupes → Centroïdes

- Pour chaque groupe, centroïde = barycentre(pointsGroupe).



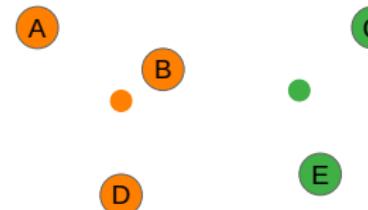
KMeans - Principe de l'algorithme (2)

Principe général

- ① Initialiser Centroïdes
- ② Répéter
 - ① Construire groupes à partir des centroïdes
 - ② Mettre à jour centroïdes en fonction des groupes

2. Groupes → Centroïdes

- Pour chaque groupe, centroïde = barycentre(pointsGroupe).



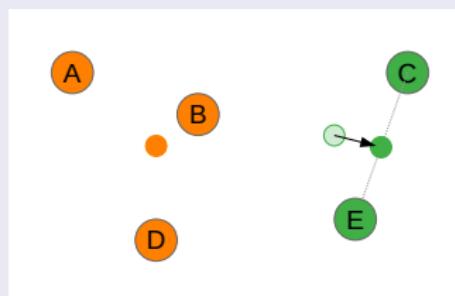
KMeans - Principe de l'algorithme (2)

Principe général

- ① Initialiser Centroïdes
- ② Répéter
 - ① Construire groupes à partir des centroïdes
 - ② Mettre à jour centroïdes en fonction des groupes

2. Groupes → Centroïdes

- Pour chaque groupe, centroïde = barycentre(pointsGroupe).



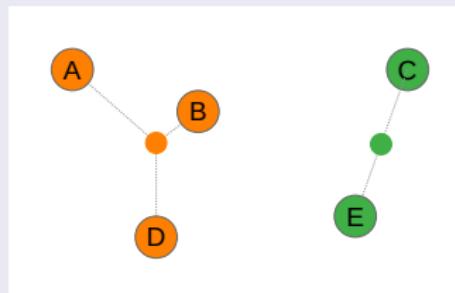
KMeans - Principe de l'algorithme (2)

Principe général

- ➊ Initialiser Centroïdes
- ➋ Répéter
 - ➊ Construire groupes à partir des centroïdes
 - ➋ Mettre à jour centroïdes en fonction des groupes

2. Groupes → Centroïdes

- Pour chaque groupe, centroïde = barycentre(pointsGroupe).

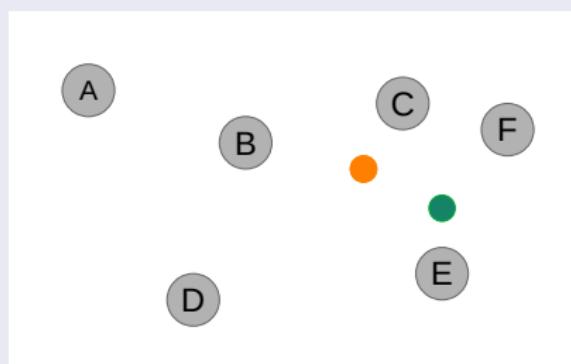


KMeans - Changement de groupe

Principe général

- 1 Construire groupes à partir des centroïdes
- 2 Mettre à jour centroïdes en fonction des groupes

Exemple - changement de groupe



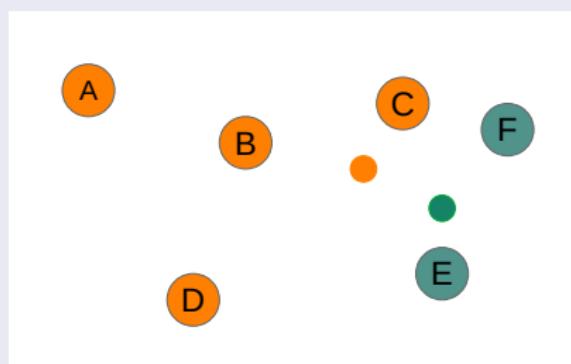
Centroïdes → Groupes

KMeans - Changement de groupe

Principe général

- 1 Construire groupes à partir des centroïdes
- 2 Mettre à jour centroïdes en fonction des groupes

Exemple - changement de groupe



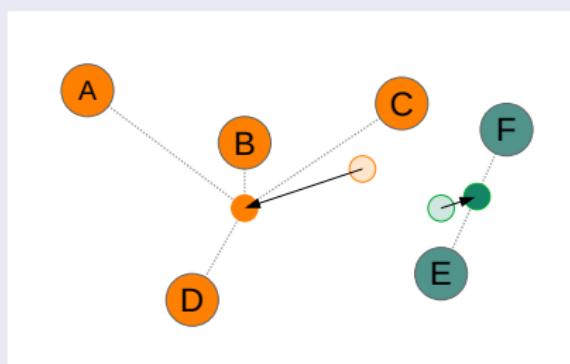
Centroïdes → Groupes

KMeans - Changement de groupe

Principe général

- 1 Construire groupes à partir des centroïdes
- 2 Mettre à jour centroïdes en fonction des groupes

Exemple - changement de groupe



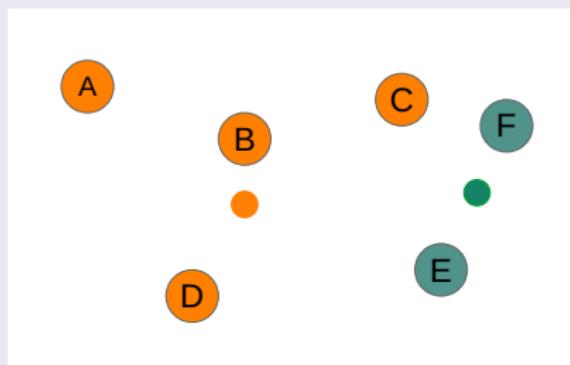
Mise à jour des Centroïdes

KMeans - Changement de groupe

Principe général

- ➊ Construire groupes à partir des centroïdes
- ➋ Mettre à jour centroïdes en fonction des groupes

Exemple - changement de groupe



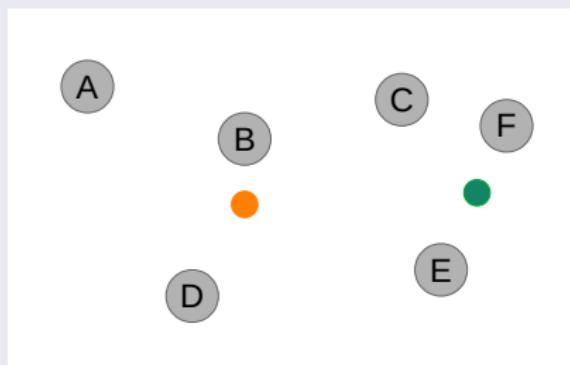
Nouveaux Centroïdes

KMeans - Changement de groupe

Principe général

- ➊ Construire groupes à partir des centroïdes
- ➋ Mettre à jour centroïdes en fonction des groupes

Exemple - changement de groupe



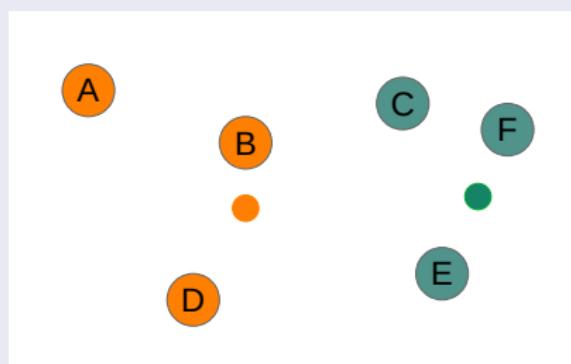
Itération suivante

KMeans - Changement de groupe

Principe général

- 1 Construire groupes à partir des centroïdes
- 2 Mettre à jour centroïdes en fonction des groupes

Exemple - changement de groupe



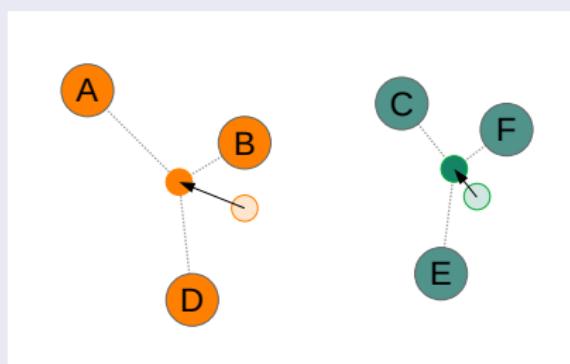
Centroïdes → Groupes

KMeans - Changement de groupe

Principe général

- 1 Construire groupes à partir des centroïdes
- 2 Mettre à jour centroïdes en fonction des groupes

Exemple - changement de groupe



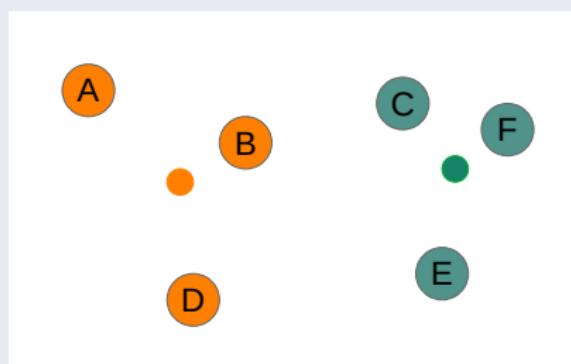
Mise à jour des Centroïdes

KMeans - Changement de groupe

Principe général

- 1 Construire groupes à partir des centroïdes
- 2 Mettre à jour centroïdes en fonction des groupes

Exemple - changement de groupe



Conclusion : status de C a changé

Algorithme détaillé

Algorithme 1 : K-Means

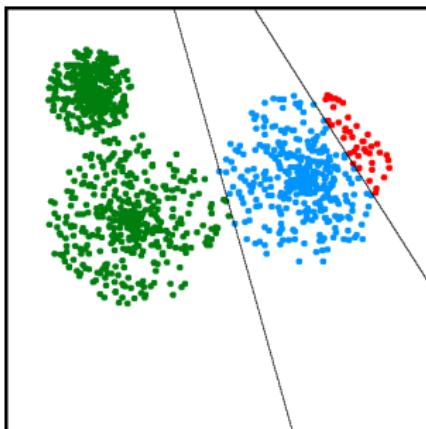
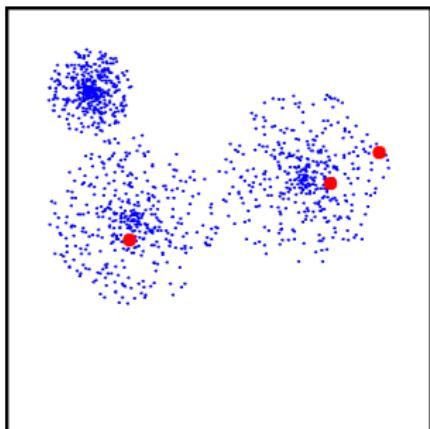
Entrées : $n_g \geq 0$ nombre de groupes

Entrées : $D = \{d_i\}_i$ données

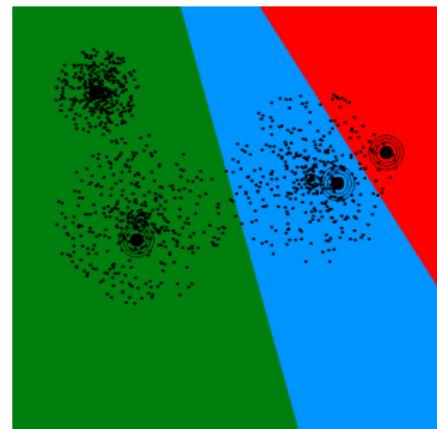
Résultat : Centroïdes mis à jour

```
/* Initialisation des centroïdes */  
1 pour  $i \in [0, n_g]$  faire  
2    $c_i \leftarrow \text{random}(D)$   
  
/* Boucle principale */  
3 tant que ( $\text{non}(\text{fini})$ ) faire  
4   /* Initialisation Groupes */  
5   pour  $i \in [0, n_g]$  faire  
6      $G_i \leftarrow \emptyset$   
7   /* Construction des Groupes */  
8   pour  $d \in D$  faire  
9      $k \leftarrow \text{indiceCentroidePlusProche}(d, \{c_i\}_i)$   
10     $G_k \leftarrow G_k \cup d$   
11  /* Mise à jour des centroïdes */  
12  pour  $i \in [0, n_g]$  faire  
13     $c_i \leftarrow \text{barycentre}(G_i)$ 
```

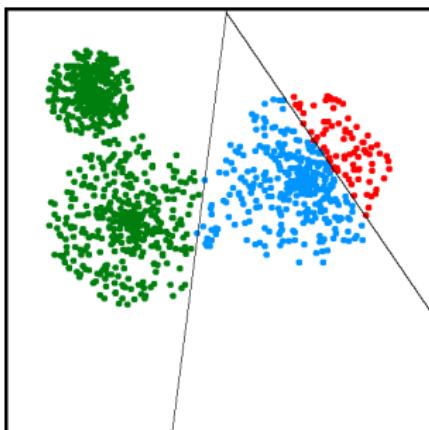
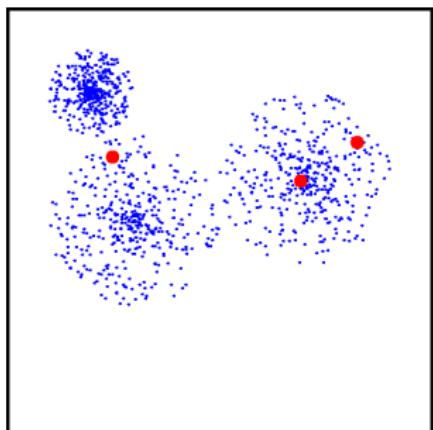
Exemple



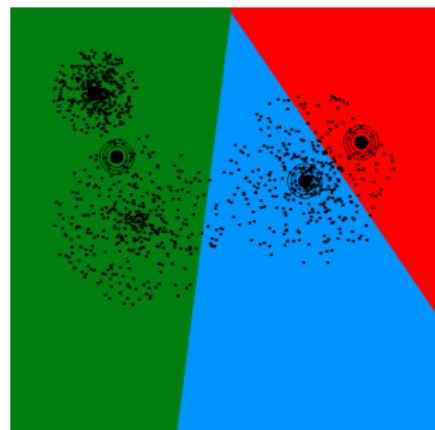
Itération 0



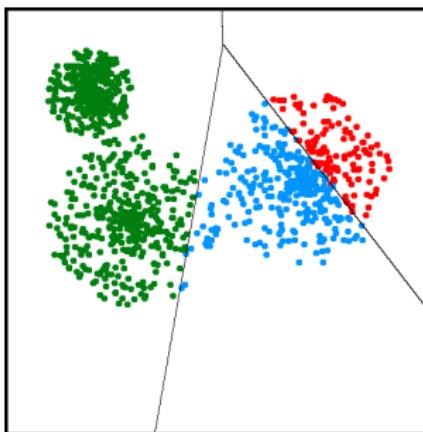
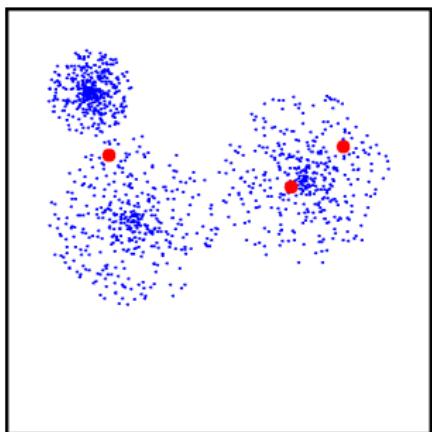
Exemple



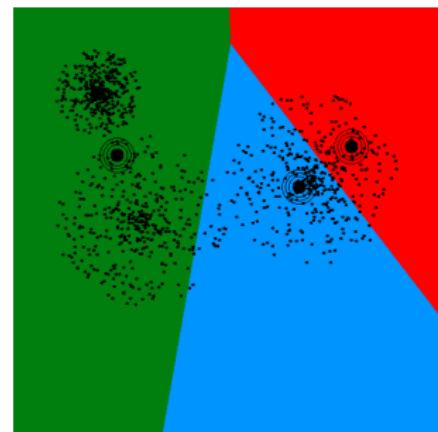
Itération 1



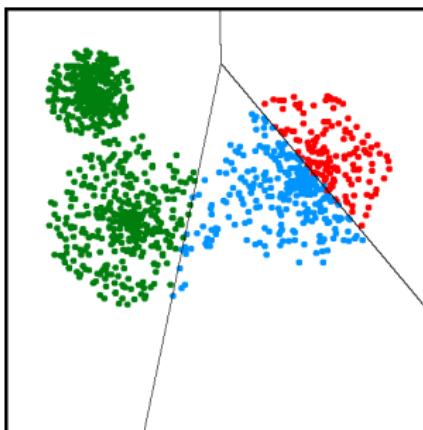
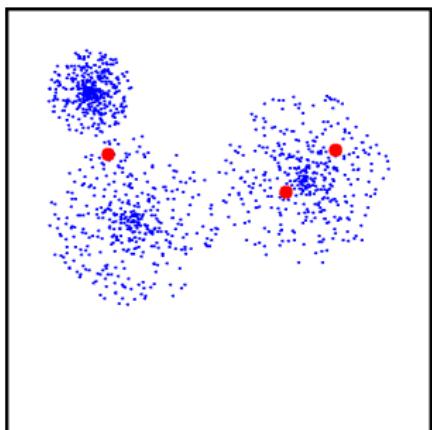
Exemple



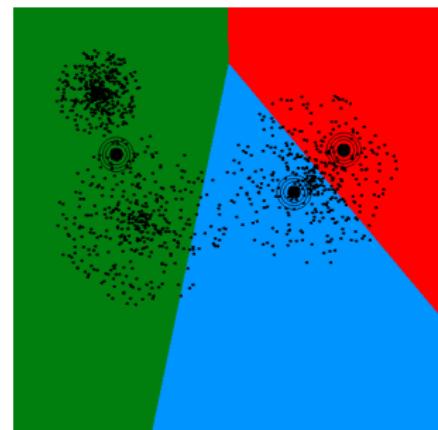
Itération 2



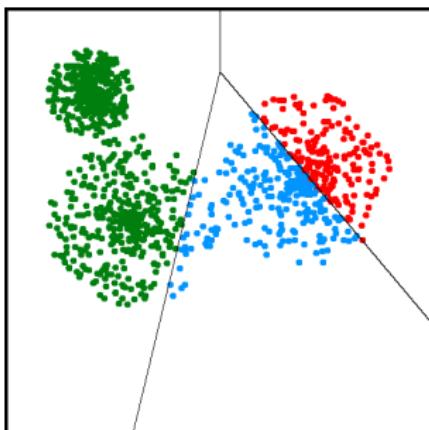
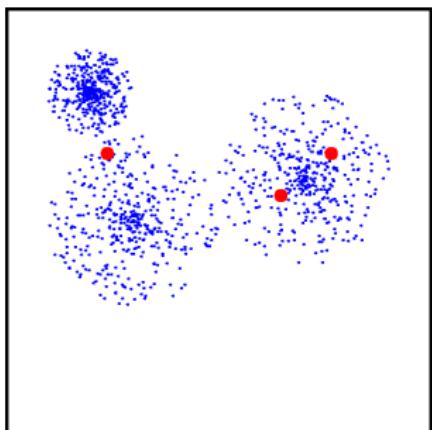
Exemple



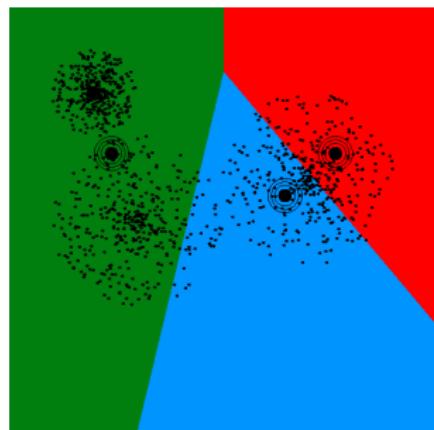
Itération 3



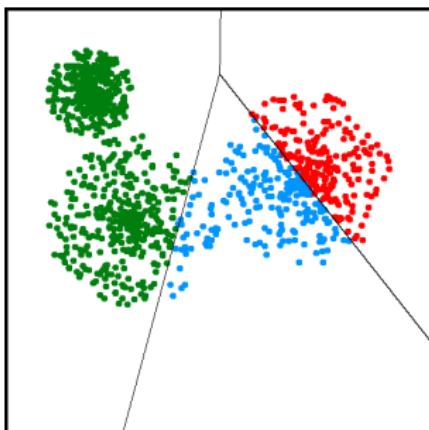
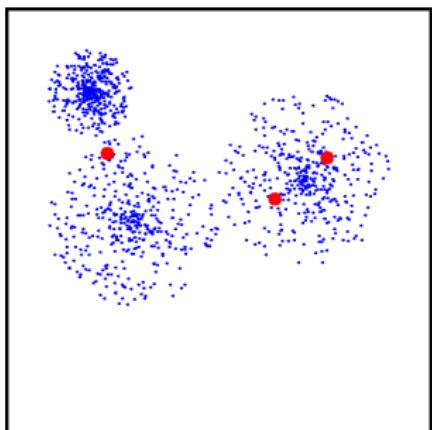
Exemple



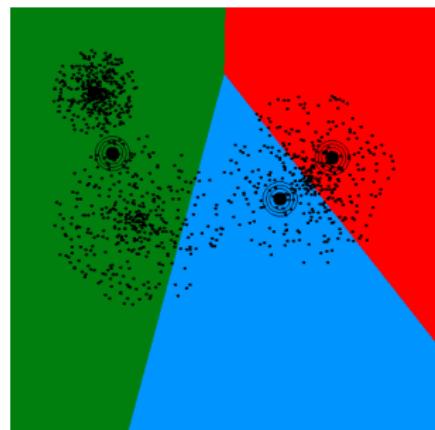
Itération 4



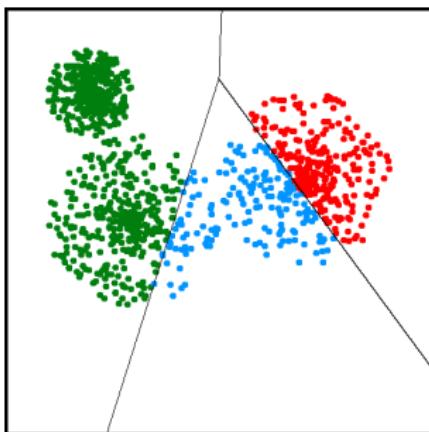
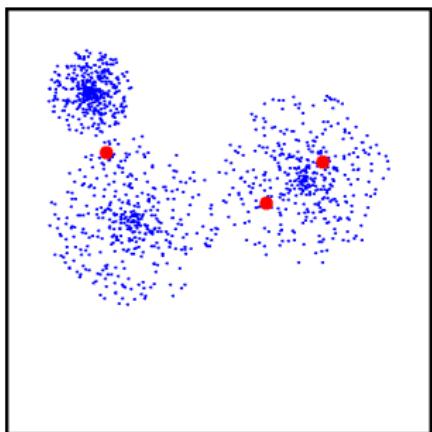
Exemple



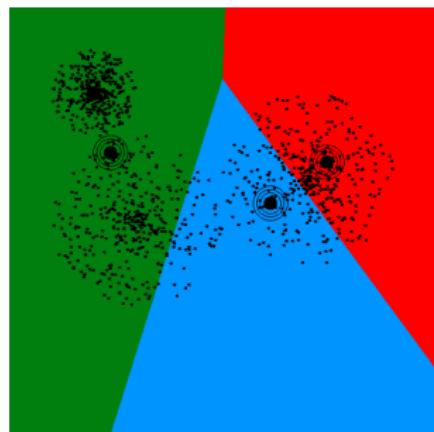
Itération 5



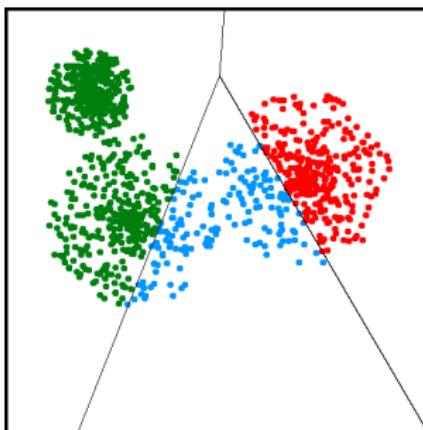
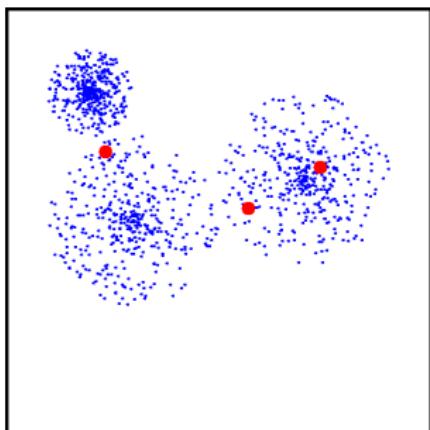
Exemple



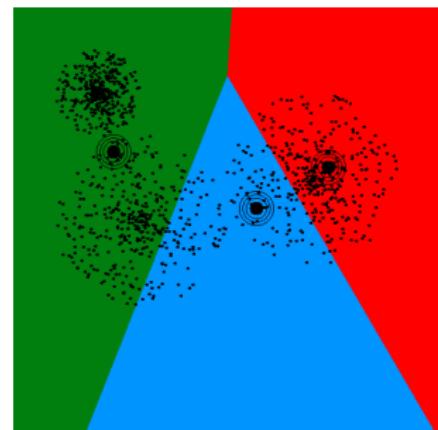
Itération 6



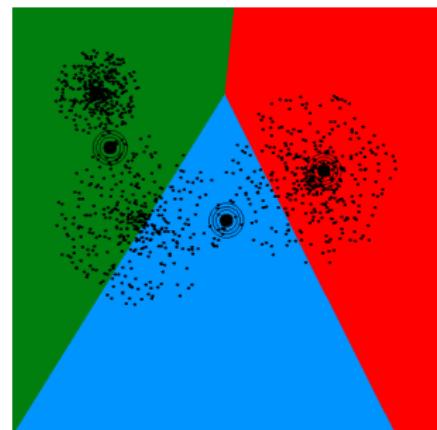
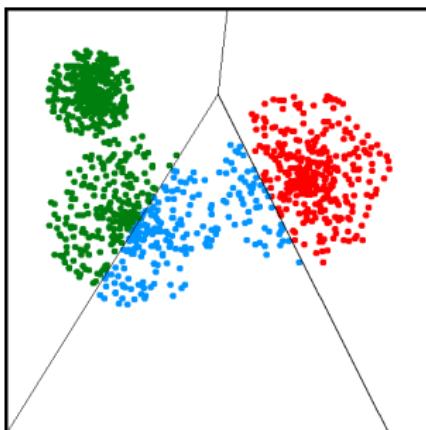
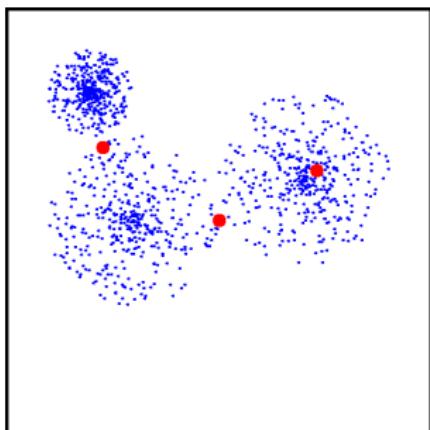
Exemple



Itération 7

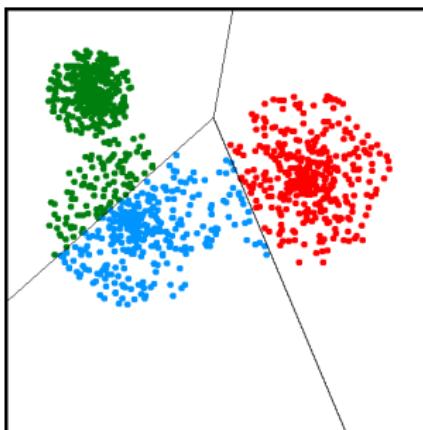
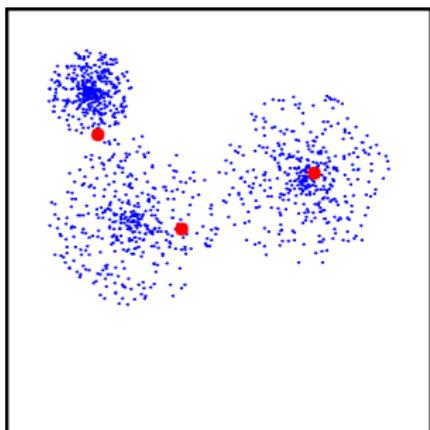


Exemple

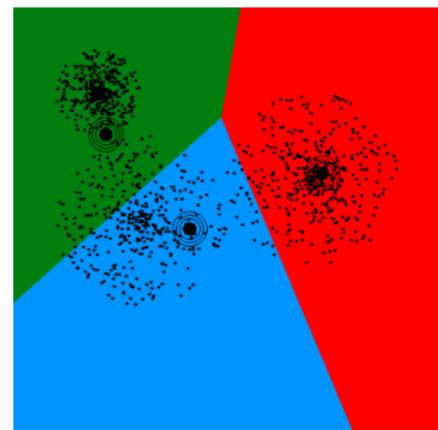


Itération 8

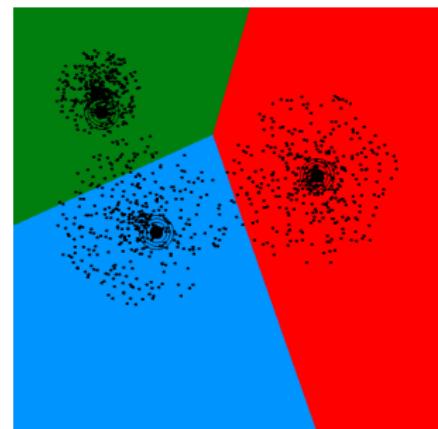
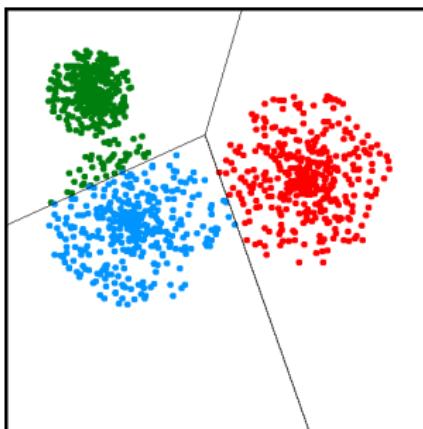
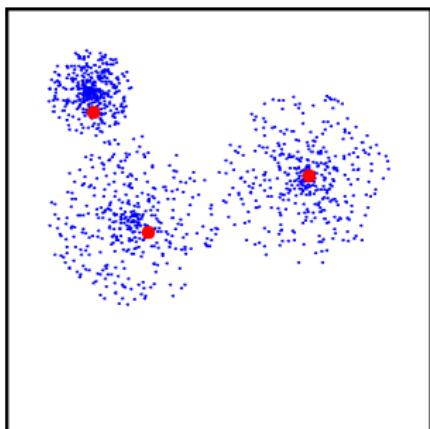
Exemple



Itération 9

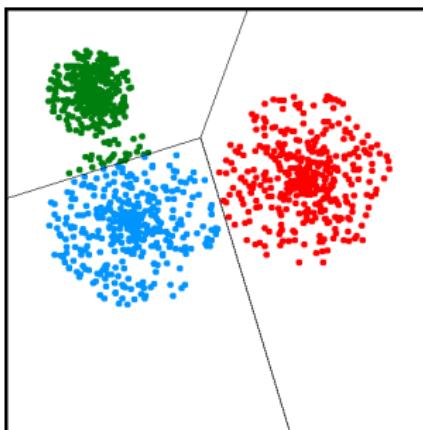
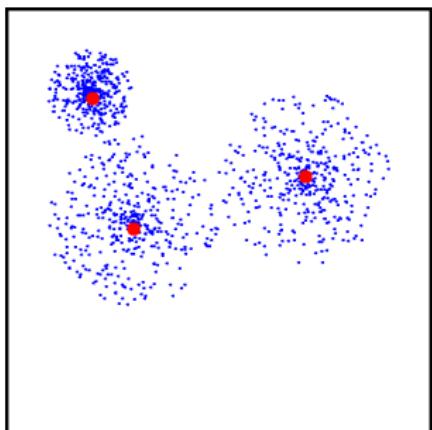


Exemple

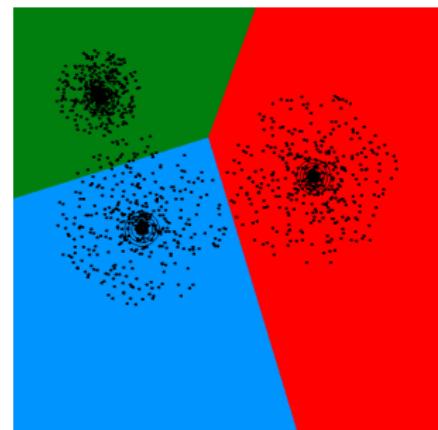


Itération 10

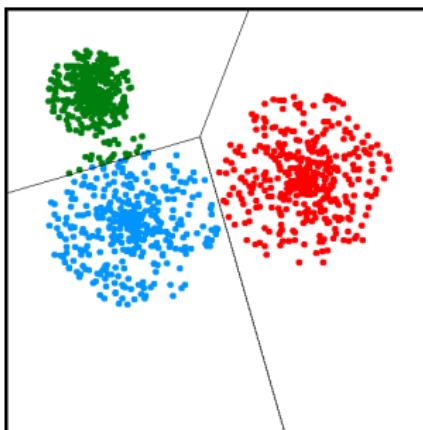
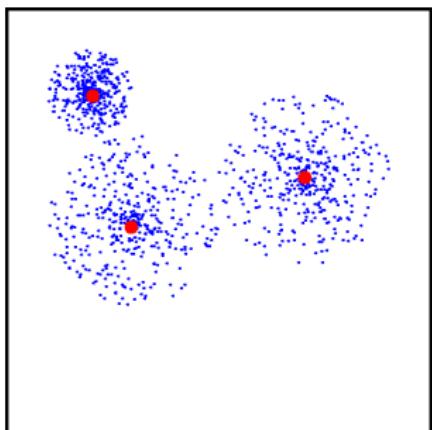
Exemple



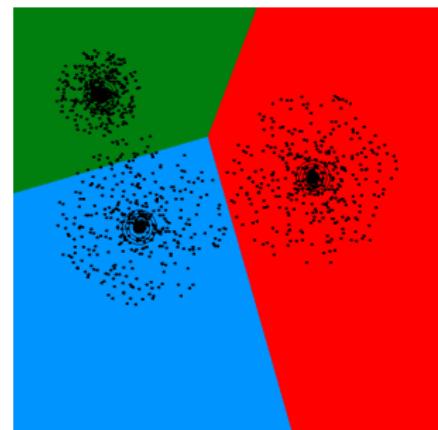
Itération 11



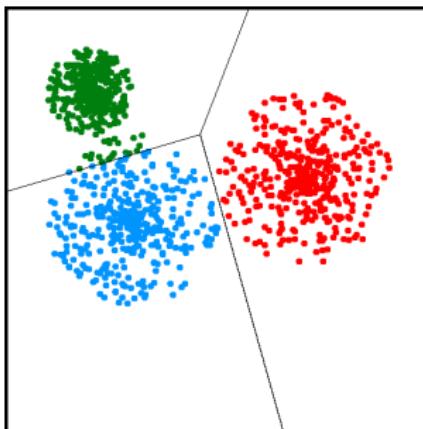
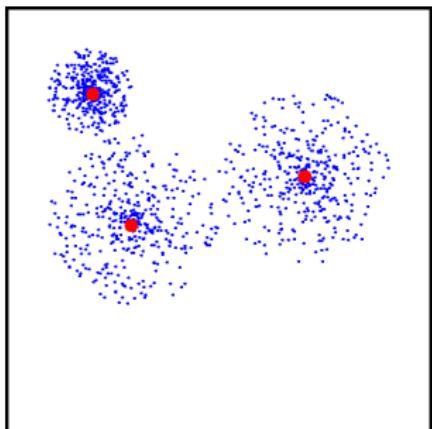
Exemple



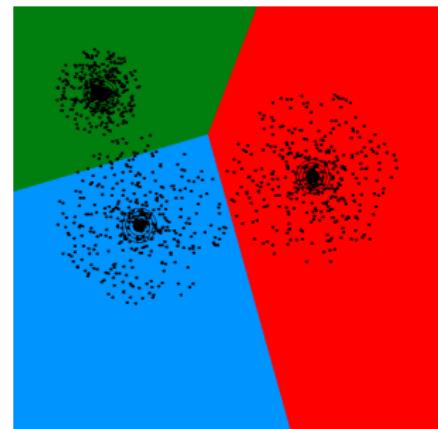
Itération 12



Exemple



Itération 13



Avantages / Inconvénients

Avantages

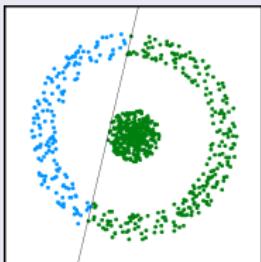
- Facile à mettre en œuvre.
- Donne des résultats rapides : complexité $O(N)$.

Inconvénients

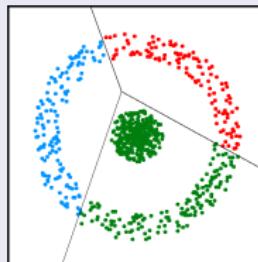
- Dépend du nombre de centroïdes.
- Dépend de la condition initiale (minima locaux).
- N'est pas directement en lien avec la fonction optimisée.

Avantages / Inconvénients

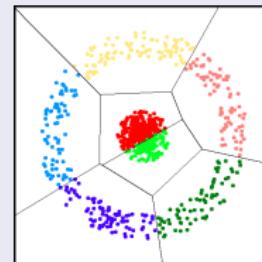
Résultats dépendent du nombre de centroïdes



2 centroïdes



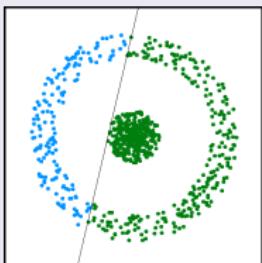
3 centroïdes



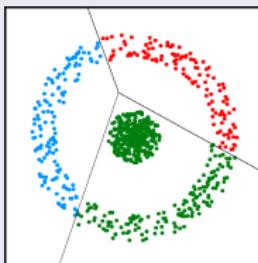
7 centroïdes

Avantages / Inconvénients

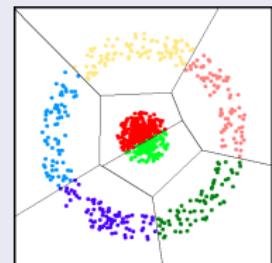
Résultats dépendent du nombre de centroïdes



2 centroïdes

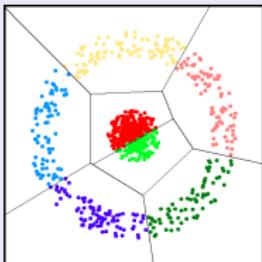


3 centroïdes

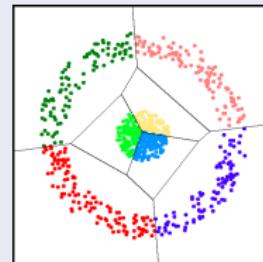


7 centroïdes

Résultats dépendent des conditions initiales (7 centroïdes)



Init 1.



Init 2.



Plan

1 Classification non supervisée

2 Algorithme KMeans

3 Algorithme DBSCAN

4 Algorithme HAC

5 Choix et validations

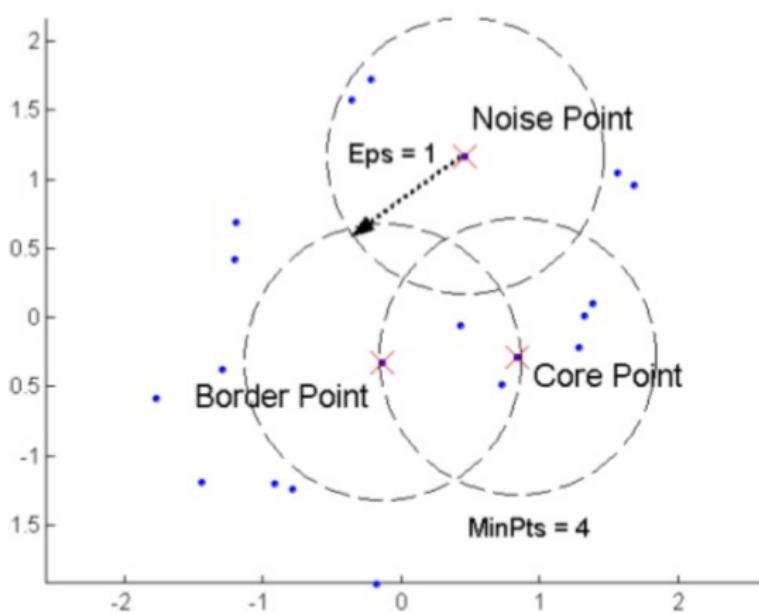
DBSCAN : Introduction

DBSCAN est un algorithme basé sur la densité

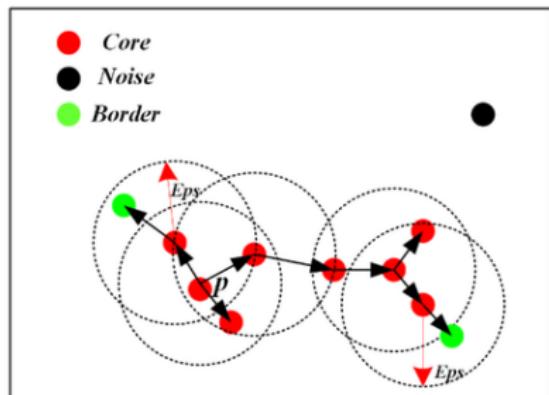
DBSCAN signifie « Density-Based Spatial Clustering of Applications with Noise ».

- La densité est basée sur le nombre de données dans le rayon de voisinage de taille « **Eps** » d'un élément donné.
- Un point est un **core point** s'il a plus de données dans son rayon de voisinage qu'un nombre spécifié de points « **MinPts** ».
- Un point dans le rayon d'un **core points** appartient au même cluster.
- Un **border point** a moins de points dans son rayon **Eps** que le nombre **MinPts** mais il est dans le rayon d'un **core point**.
- Un **noise point** a moins de points dans son rayon **Eps** que le nombre **MinPts** et n'est pas à portée d'un **core points**.

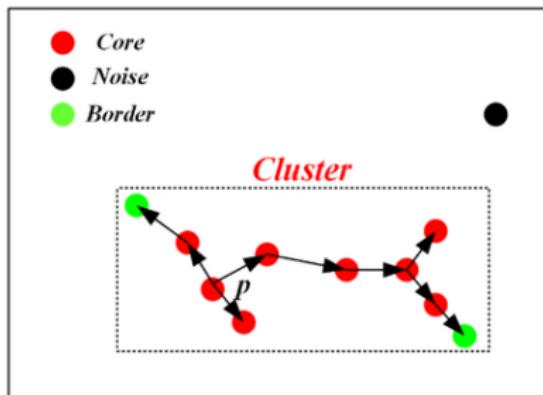
DBSCAN : Principes



DBSCAN : Principes



(a)



(b)

Algorithme détaillé

Algorithme 2 : DBSCAN

```

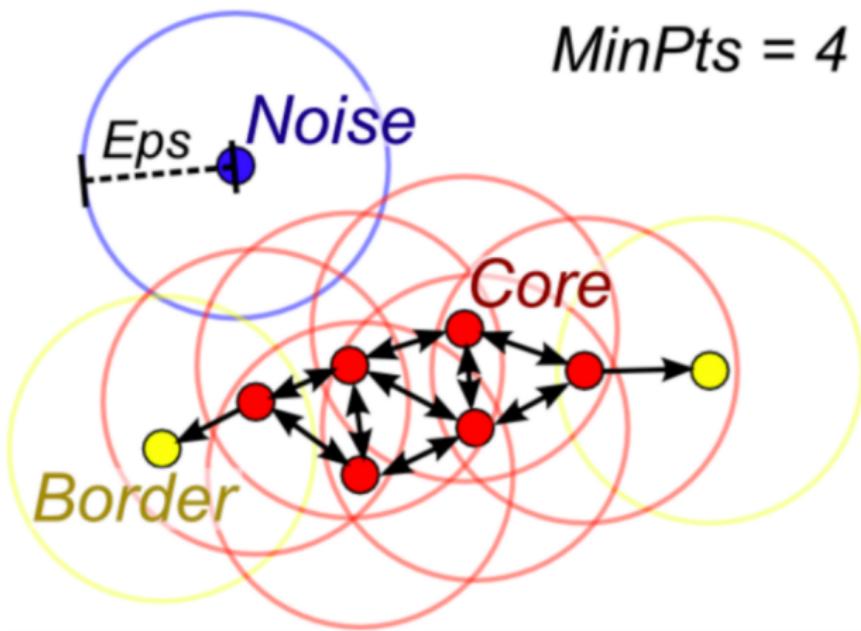
1 Function DBSCAN(X, Eps, MinPts)
2   C=0
3   pour chaque  $x_n \in X$  faire
4     si  $x_n$  n'a pas été traité alors marque  $x_n$  comme traité
5      $V_n = \text{regionQuery}(x_n, Eps)$ 
6     si taille( $V_n$ )  $\geq$  MinPts alors
7        $C \leftarrow C + 1$ 
8       expandCluster( $x_n, V_n, C, Eps, MinPts$ )
9     sinon
10      marque  $x_n$  comme Noise Point

11 Function expandCluster( $x_n, V_n, C, Eps, MinPts$ )
12   ajoute  $x_n$  au cluster C
13   pour chaque  $x_i \in V_n$  faire
14     si  $x_i$  n'a pas été traité alors
15       marque  $x_i$  comme traité
16        $V_i \leftarrow \text{regionQuery}(x_i, Eps)$ 
17       si taille( $V_i$ )  $>$  MinPts alors  $V_n = V_n + V_i$ 
18     si  $x_i$  n'appartient à aucun cluster alors ajoute  $x_i$  au cluster C

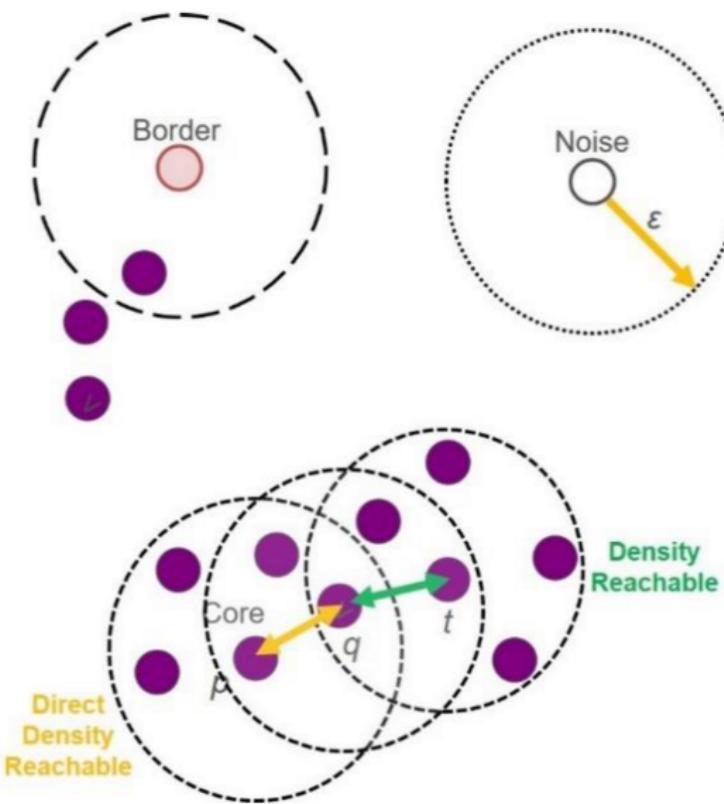
19 Function regionQuery( $x_n, Eps$ )
20   pour chaque  $x_i \in X$  faire
21     si  $i \neq n$  et  $d(x_n, x_i) \leq Eps$  alors ajoute  $\{x_i\}$  à la liste V
22   retourner V

```

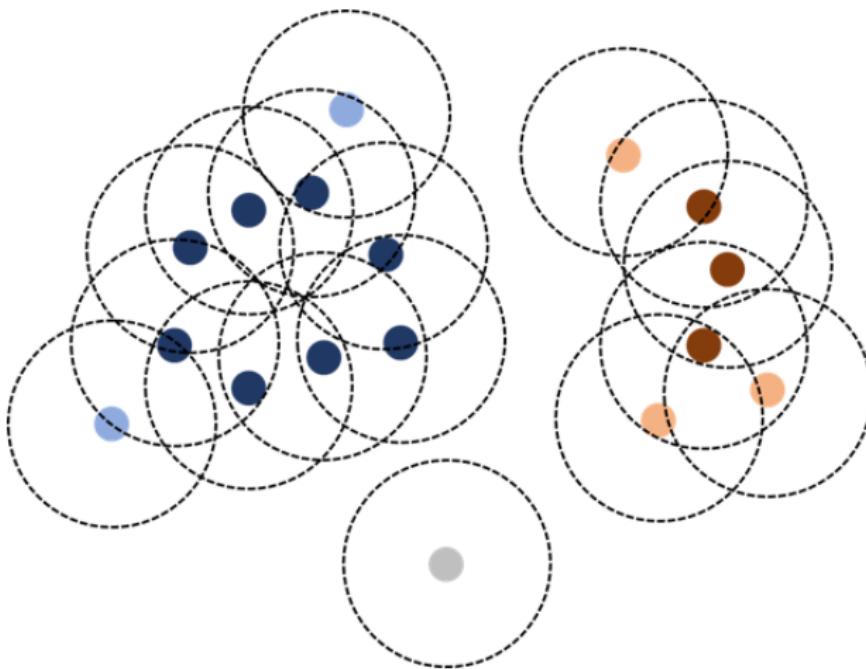
DBSCAN : Exemples



DBSCAN : Exemples



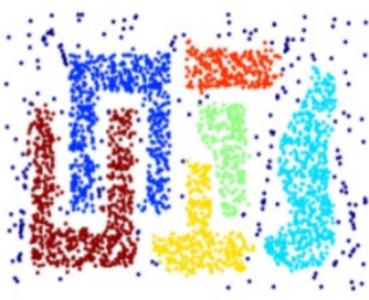
DBSCAN : Exemples



Avantages / Inconvénients



Original Points

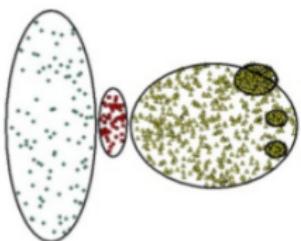


Clusters

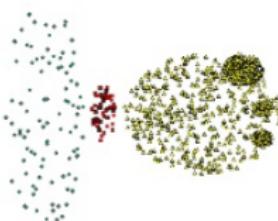
Avantages

- Résistant au bruit.
- Peut détecter des clusters de formes et de tailles différentes.

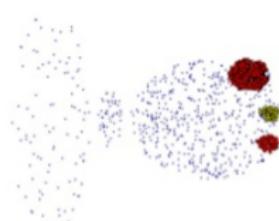
Avantages / Inconvénients



Original Points



(MinPts=4, Eps=9.92)



(MinPts=4, Eps=9.75).

Inconvénients

- Difficile à paramétriser.
- Lent : complexité $O(N^2)$.
- Ne gère pas bien les densités variables d'un cluster à l'autre.

Plan

1 Classification non supervisée

2 Algorithme KMeans

3 Algorithme DBSCAN

4 Algorithme HAC

5 Choix et validations

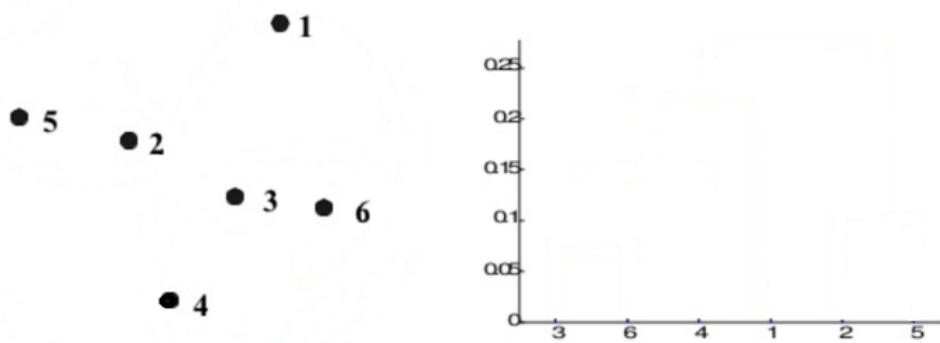
HAC : Introduction

Principe

HAC signifie « Hierarchical Agglomerative Clustering ».

- L'idée est de regrouper à chaque étape les deux groupes de données les plus proches.
- Au départ, chaque donnée est seul dans son groupe.
- Au cours du processus, un groupe peut être composé d'une ou plusieurs données.
- L'algorithme renvoie un arbre (dendrogramme) contenant l'historique des différentes agrégations.
- La taille des branches de l'arbre est proportionnel aux distances entre groupes.
- Les clusters sont finalement obtenus en « coupant » l'arbre à un niveau à définir.

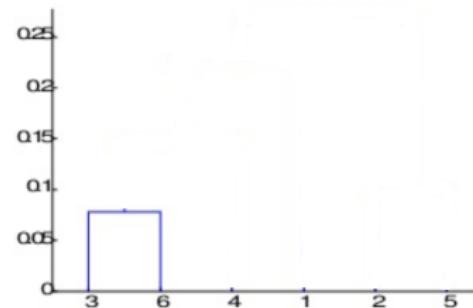
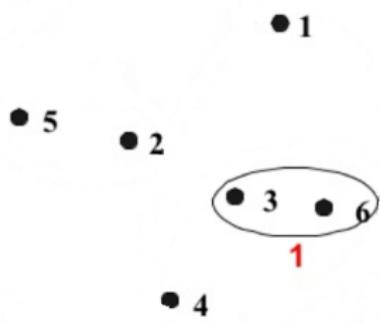
HCA : Example



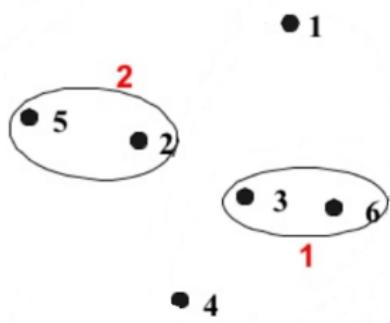
Nested Clusters

Dendrogram

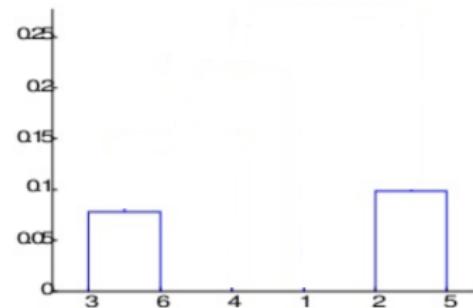
HCA : Example



HCA : Example

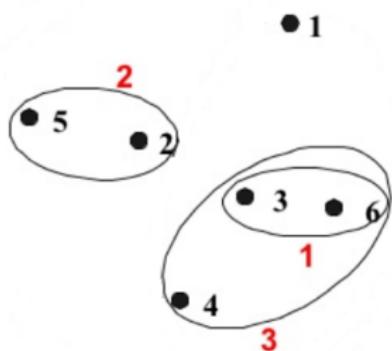


Nested Clusters

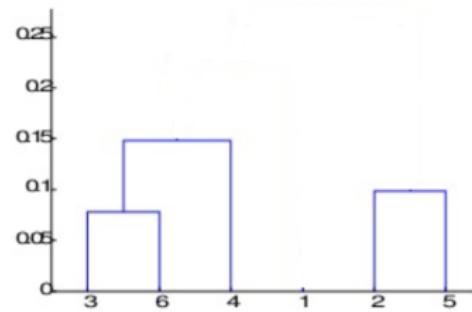


Dendrogram

HCA : Example

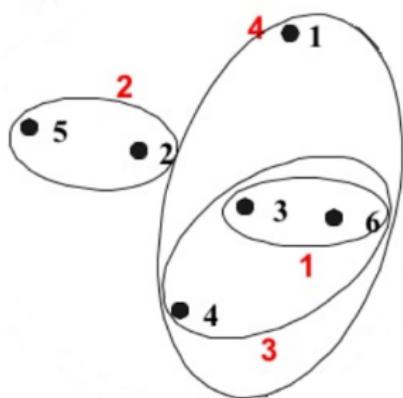


Nested Clusters

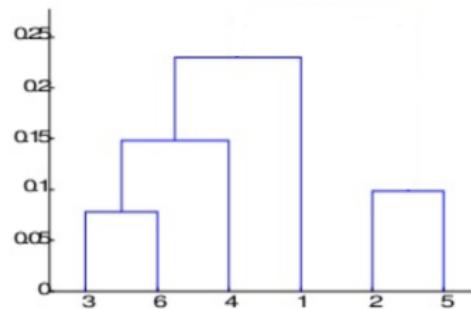


Dendrogram

HCA : Example

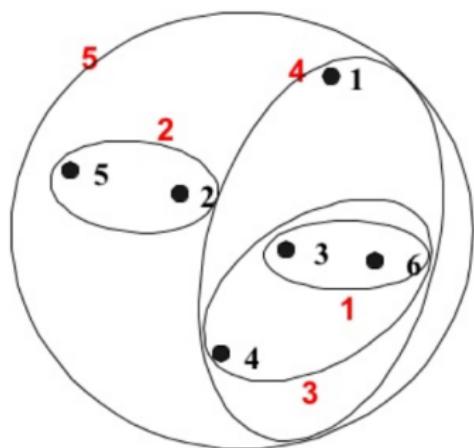


Nested Clusters

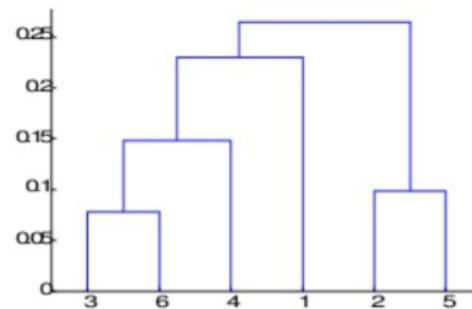


Dendrogram

HCA : Example



Nested Clusters



Dendrogram

Algorithme détaillé

Algorithme 3 : HAC

Entrées : D une matrice de distances $n \times n$ entre les données

Entrées : $linkage(C_1, C_2)$ une fonction de distance entre groupes de données

Résultat : Une hiérarchie de regroupements (un dendrogramme)

```

/* Initialisation */
```

- 1 Initialiser L avec n clusters, chacun contenant un point de donnée.

```

/* Boucle principale */
```

- 2 tant que $\text{taille}(L) > 1$ faire


```

/* Fusion de la paire la plus proche */
```

- 3 Trouver les paires (C_1, C_2) de L tel que $linkage(C_1, C_2)$ soit minimal
- 4 Fusionner C_1 et C_2 dans un nouveau cluster C

```

/* Mise à jour de la liste des clusters et de leurs distances */
```

- 5 Enlever C_1 et C_2 de L
- 6 pour chaque cluster $C' \in L$ faire


```

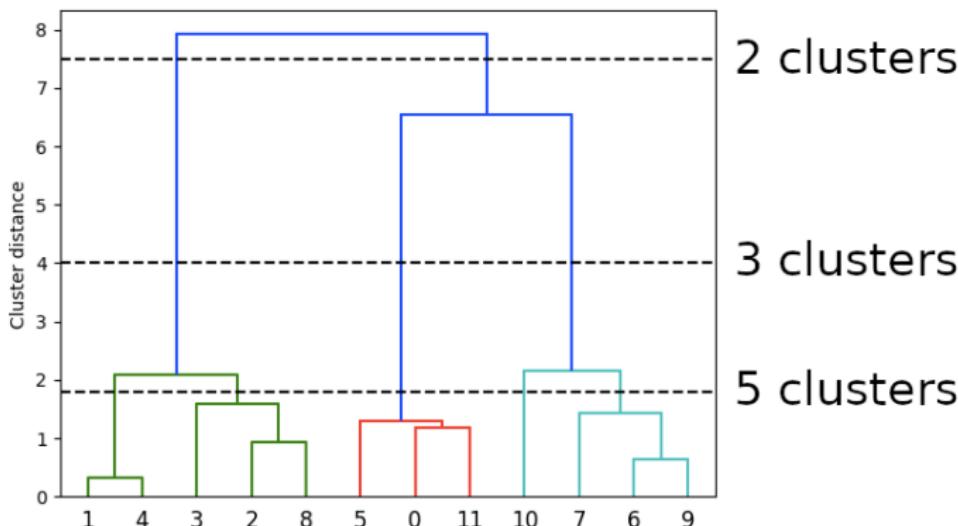
d ← linkage(C, C')
```

- 7 Mettre à jour la matrice de distance D avec la distance d entre C et C'
- 8 Enlever les distances concernant C_1 et C_2 de D
- 9 Ajouter C à L

```

11 retourner le dendrogramme à couper
```

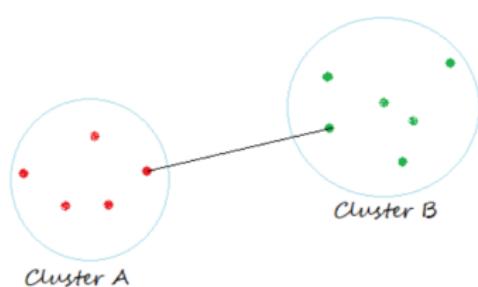
HAC : Coupe du dendrogramme



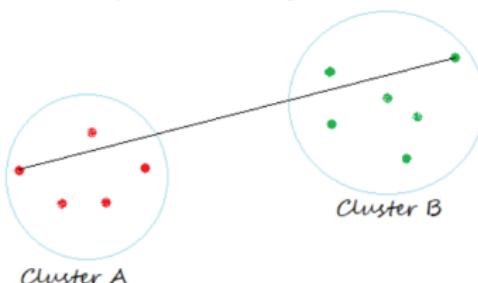
Le choix de la coupe dépend du nombre de clusters voulu

HCA : fonctions de distance (linkage)

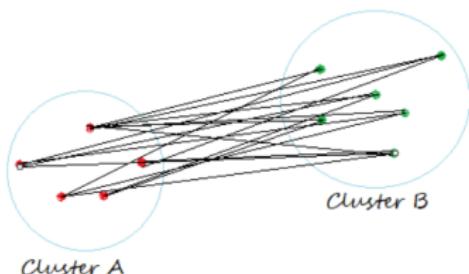
Single Linkage



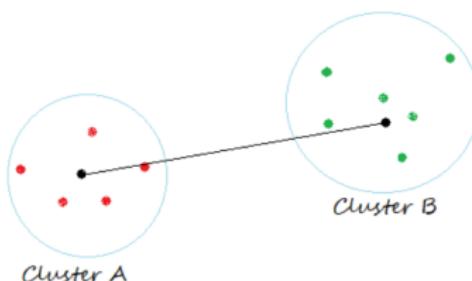
Complete Linkage



Average Linkage



Centroid Linkage



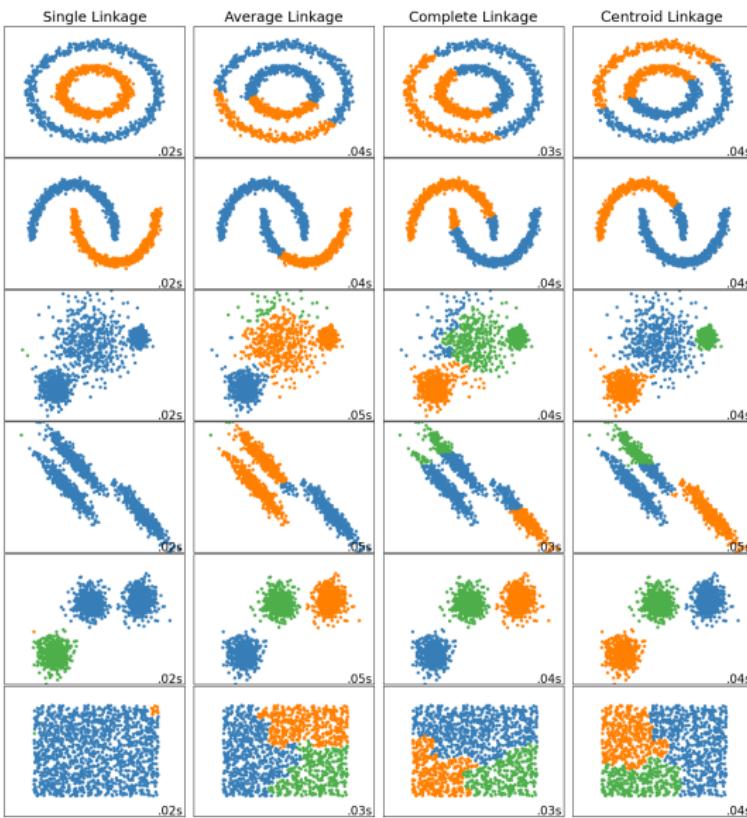
HCA : fonctions de distance (linkage)

Nom	Formule	Description
Single-linkage	$D_s(c_1, c_2) = \min_{x \in c_1, y \in c_2} d(x, y)$	Distance entre les deux éléments les plus proches
Complete-linkage	$D_c(c_1, c_2) = \max_{x \in c_1, y \in c_2} d(x, y)$	Distance entre les deux éléments les plus éloignés
Average-linkage	$D_a(c_1, c_2) = \frac{1}{ c_1 c_2 } \sum_{x \in c_1} \sum_{y \in c_2} d(x, y)$	Distance moyenne entre les paires d'éléments
Centroid-linkage	$D_\mu(c_1, c_2) = \mu_1 - \mu_2 $	Distance entre les centroïdes

HCA : fonctions de distance (linkage)

Nom	Avantages	Défauts
<u>Single-linkage</u>	Peut détecter des clusters de toute forme	Est sensible au bruit
Complete-linkage	Résistant au bruit	Détecte des clusters elliptiques, casse les gros clusters
Average-linkage	Résistant au bruit	Détecte des clusters elliptiques, lent
<u>Centroid-linkage</u>	Résistant au bruit	Détecte des clusters elliptiques, rapide

HCA : comparaisons de linkages



Avantages / Inconvénients

Avantages

- Résultats de bonnes qualité avec le bon choix de linkage.
- Visualisation sous forme de dendrogramme.
- Possibilité de tester différents nombres de cluster en une seule exécution.

Inconvénients

- Très lent : complexité $O(N^2 \log N)$ à $O(N^3)$.
- Ne peut pas à la fois être résistant au bruit et détecter toute formes de cluster.

Plan

1 Classification non supervisée

2 Algorithme KMeans

3 Algorithme DBSCAN

4 Algorithme HAC

5 Choix et validations

Choix de l'algorithme de clustering

Connaissances sur les clusters

Le choix du bon algorithme de clustering dépend de plusieurs facteurs et nécessite des connaissances préalables :

- Existe-t-il des clusters ? A-t-on une idée de leur nombre ?
- Quelle est la forme des clusters et leur degré de séparation ?

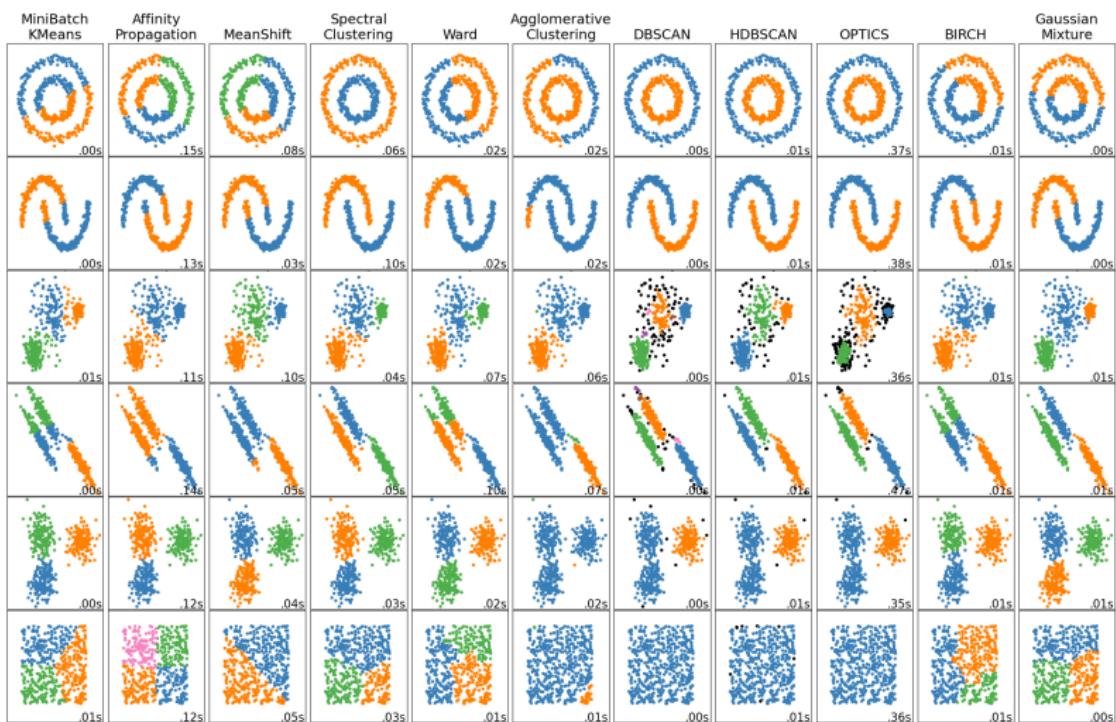
Connaissances sur les données

La complexité de l'algorithme dépend des données :

- Complexité en fonction du nombre de données à traiter.
- Complexité en fonction du nombre d'attributs des données.

Le choix d'un algorithme est souvent un compromis entre la qualité du résultat et la complexité de calcul.

Exemples de résultats avec différents algorithmes



Choix de la mesure de similarité

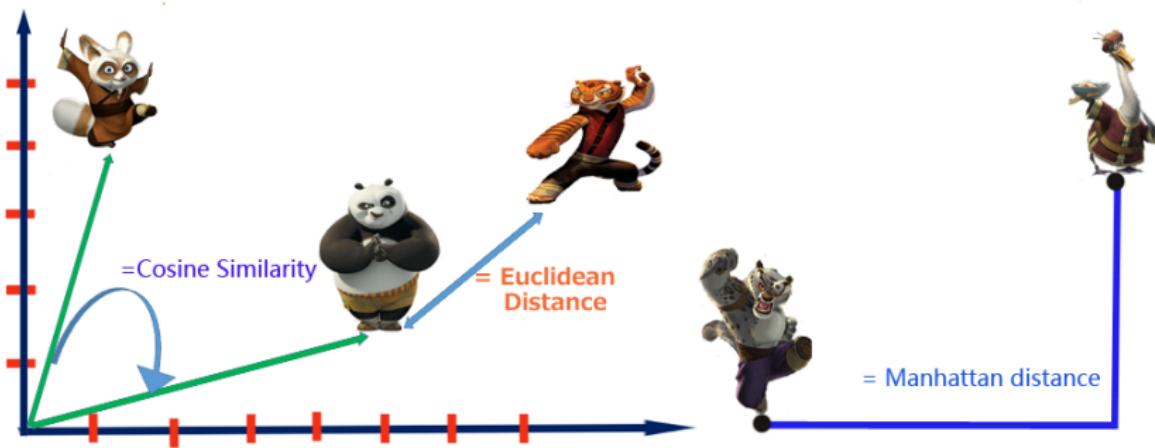
Quel que soit le type d'algorithme de clustering, ils reposent tous sur une mesure de similarité. Le choix d'une bonne mesure de similarité est donc d'une importance capitale.

- Il détermine le type de clusters considérés comme intéressants.
- Il modifie le résultat du clustering.

Remarques

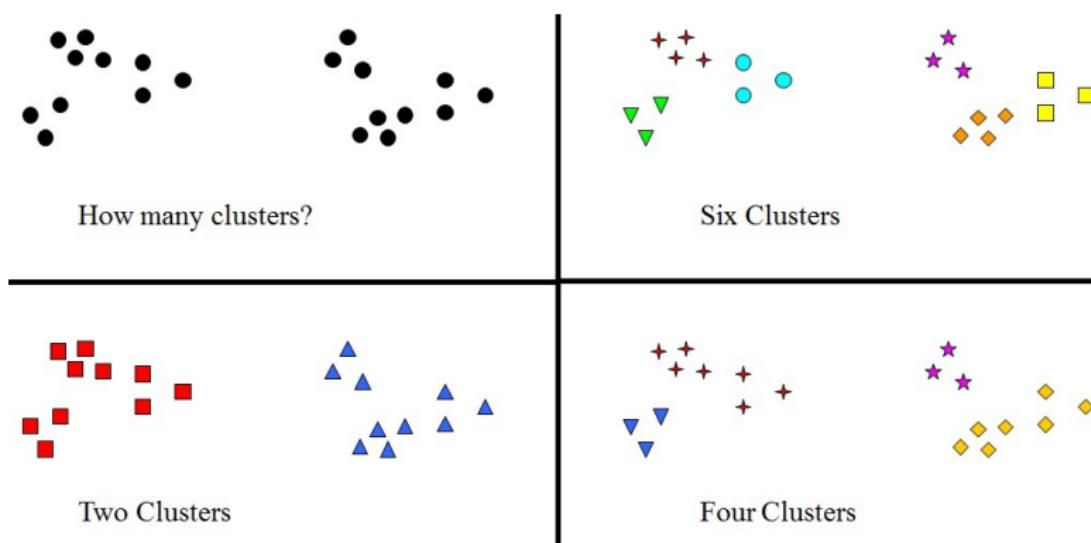
- Le choix d'une mesure de similarité, ou la construction d'une mesure de similarité, nécessite souvent une bonne connaissance des données à analyser, ou au moins du domaine dont elles sont issues.
- Le choix d'une mesure de similarité introduit déjà un biais dans une tâche qui est censée être exploratoire.

Exemples de mesures de similarité



Choix des paramètres

Le choix des paramètres de l'algorithme est un élément primordial pour faire un clustering. Notamment, la détermination du nombre exact de clusters est l'un des choix les plus importants pour de nombreux algorithmes.



Valider les résultats du clustering

La validation des résultats du clustering est difficile :

- Il n'existe pas de définition correcte d'un « bon cluster » ou de ce à quoi il doit ressembler.
- Il n'y a pas de « vérité de base » en apprentissage non supervisé pour vérifier la qualité d'un clustering.
- Les clusters peuvent être de différentes tailles et formes.
- Le nombre de clusters à trouver est généralement inconnu, et il arrive qu'il n'y en ait aucun.

Identification manuelle des clusters

Alors qu'il est généralement évident de repérer visuellement les clusters dans des données 2D ou 3D, il est presque impossible de le faire lorsqu'il y a plus de dimensions !

Indices de qualité internes

Les **indices internes** sont des critères qui peuvent être utilisés pour évaluer la qualité d'un clustering.

- Les indices internes peuvent être subjectifs car ils favorisent certaines formes de clusters et peuvent être biaisés en faveur d'un nombre inférieur de clusters.
- Ils évaluent généralement la compacité des clusters et si ils sont bien séparées les un des autres.
- Ils sont souvent utilisés pour comparer différents algorithmes ou différents paramètres d'un algorithme.

Remarque

Les indices internes sont dits « internes », par opposition aux indices « externes » qui comparent un résultat à une vérité de base externe.

Exemple d'indice interne : Davies-Bouldin

Soit S_i la dispersion moyenne d'un cluster c_i autour de son barycentre μ_i :

$$S_i = \frac{1}{|c_i|} \sum_{x \in c_i} \|x - \mu_i\|_2$$

Soit $M_{i,j} = \|\mu_i - \mu_j\|_2$ la distance moyenne entre deux barycentres de clusters c_i et c_j .

L'indice de Davies-Bouldin pour K clusters

$$DB = \frac{1}{K} \sum_{i=1}^K \max_{j \neq i} \frac{S_i + S_j}{M_{i,j}}$$

Exemple d'indice interne : Davies-Bouldin

L'indice de Davies-Bouldin a les propriétés suivantes :

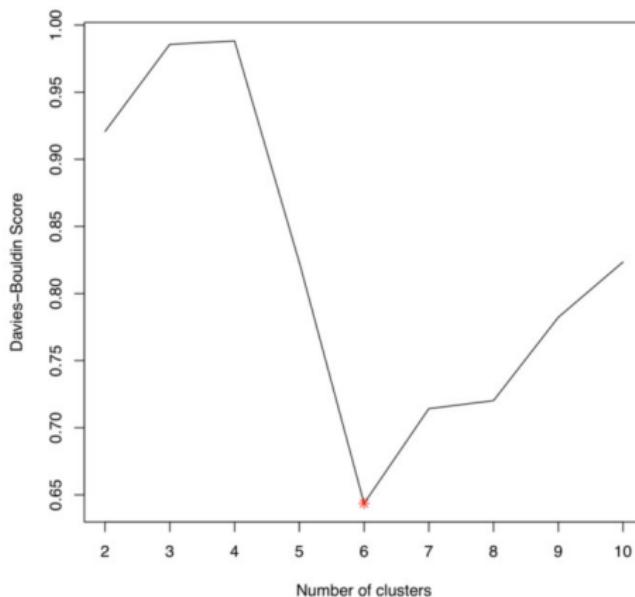
- Une valeur DB plus faible signifie un meilleur clustering.
- Il favorise les clusters sphériques.
- Il n'est pas normalisé : il a tendance à donner des valeurs plus faibles lorsqu'on cherche moins de clusters.

Remarque

L'indice de Davies-Bouldin est généralement l'indice interne privilégié pour évaluer les résultats de l'algorithme K-Means.

Indice interne : exemple d'utilisation

Choix du nombre de clusters K pour K-Means :



Le nombre optimal de clusters est 6 dans cet exemple.

Résumé

Étape d'une analyse de données par clustering

- ➊ Définir la notion de similarité et choisir une fonction de distance.
- ➋ Choisir un algorithme adapté.
- ➌ Choisir les bons paramètres pour l'algorithme.
- ➍ Évaluer les résultats du clustering.
- ➎ Comparer éventuellement les résultats de différentes approches.