



Rapport TP Apprentissage Non-Supervisé

MEJRI Hazem
BOUKEZZATA Imad El Dine
5 SDBD B1

Table des matières

<u>Introduction</u>	2
<u>1 Clustering K-Means</u>	2
<u>1.1 Principe</u>	2
<u>1.2 Interet</u>	4
<u>1.3 Metrique d'évaluation</u>	4
<u>1.3.1 Silhouette_score</u>	4
<u>1.3.2 Davies Bouldin</u>	5
<u>1.3.3 Calinski Harabasz</u>	5
<u>1.4 Points forts/ Points faibles</u>	6
<u>1.4.1 Diamond 9</u>	6
<u>1.4.2 Donut 2</u>	7
<u>1.5 Mini-Batch</u>	8
<u>2 Clustering Agglomératif</u>	9
<u>2.1 Dendrogramme</u>	9
<u>2.2 Linkage</u>	10
<u>2.2.1 Single</u>	11
<u>2.2.2 Complete</u>	12
<u>2.2.3 Average</u>	13
<u>2.2.4 Ward</u>	13
<u>3 Clustering DBSCAN</u>	15
<u>3.1 Principe</u>	15
<u>3.2 Choix des paramètres : epsilon et min_pts</u>	15
<u>3.2.1 Min_pts</u>	15
<u>3.2.2 Epsilon</u>	16
<u>3.3 Points forts/ Points faibles</u>	17
<u>4 Clustering HDBSCAN</u>	18
<u>4.1 Principe</u>	18
<u>4.2 Comparaison de performance DBSCAN/HDBSCAN</u>	18
<u>5 Analyse Comparative</u>	20
<u>Conclusion</u>	22

Introduction:

Ce rapport a pour but de rendre compte de nos travaux en TP d'apprentissage non supervisé. Il est composé de quatre chapitres, un pour chaque méthode de clustering étudiée et un chapitre comparatif de ces méthodes.

Dans ce projet, nous présenterons les résultats obtenus pour chacune des méthodes appliquées aux ensembles de données choisis, en analysant leurs performances à l'aide de divers critères d'évaluation tels que les indices de Davies-Bouldin, Calinski-Harabasz, et Silhouette. Enfin, une analyse comparative nous permettra de mettre en avant les forces et faiblesses de chaque approche, ainsi que leur pertinence selon les spécificités des données.

1. Clustering k-means:

1.1 Principe:

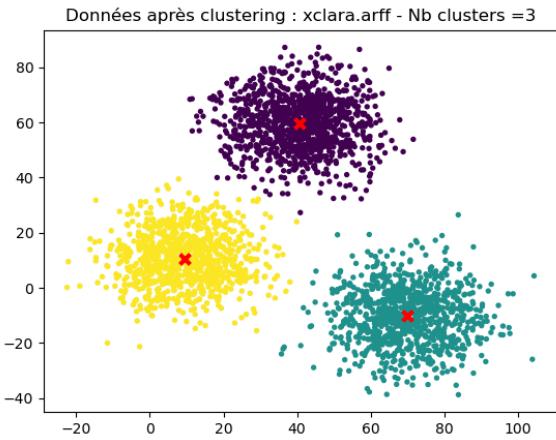
Cet algorithme commence par la mise en place d'autant de centres de gravité que de nombre de clusters que l'on souhaite obtenir. Chaque centre de gravité (pas forcément un élément du jeu) est affecté à un cluster.

A chaque boucle , on commencera par affecter chaque point au centre de gravité le plus proche avant de réévaluer les centre de gravité de chacun des clusters créés. Cette boucle se termine à la **condition d'arrêt** imposée (Nombre d'itérations K max, centre de gravités stables)

L'affectation d'un point à un cluster le plus proche se fait par un calcul de distance euclidienne.

1.2 Interet: (xclara)

L'algorithme permet une facilité d'interprétation couplée à sa simplicité d'implémentation et son efficacité. Il est particulièrement fonctionnel sur des banques contenant un grand nombre de points ou des banques contenant des clusters sphériques



Test avec la dataset **sizes5**:

On veut déterminer pour quelle valeur de k la méthode k-means est optimale, on fait donc varier la valeur de k pour le dataset size 5.arff

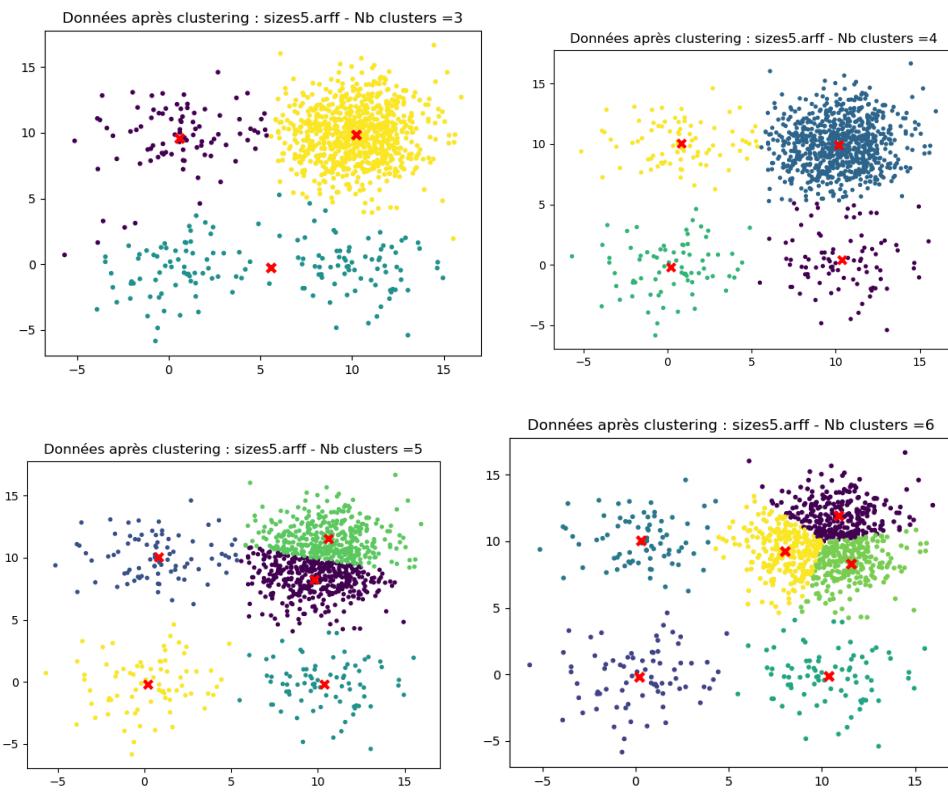


Figure 1: Evolution du clustering en fonction de k

Optimisation de k par rapport au gain en inertie (condition d'arrêt: evaluation métrique)

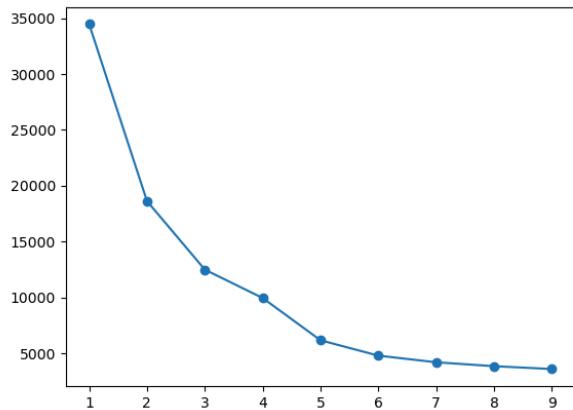


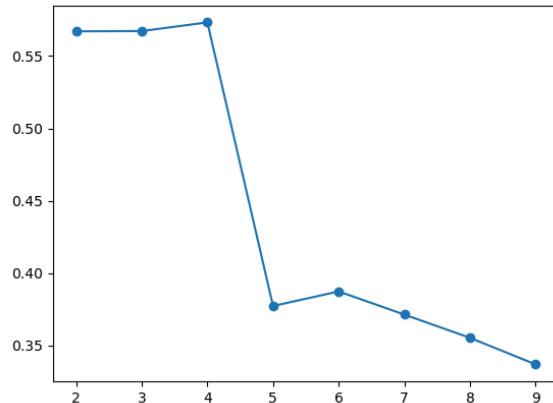
Figure 2: Courbe de l'inertie en fonction de k

Nous avons utilisé l'indice d'inertie, une mesure courante pour évaluer la qualité d'un clustering. Elle quantifie la dispersion des points à l'intérieur de chaque cluster. La valeur optimale se situe au point de "cassure" de la courbe, ici pour ($k = 5$). Cette métrique est utile pour déterminer la valeur optimale d'un paramètre de modèle. Cependant, pour d'autres jeux de données, cette cassure peut être moins marquée, ce qui donnerait un intervalle de valeurs optimales plutôt qu'une unique valeur.

1.3 Les métriques d'évaluation

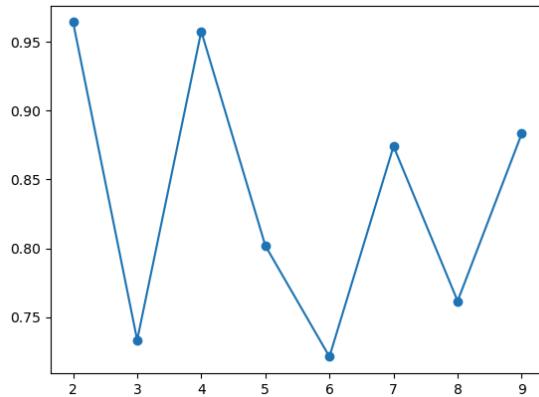
1.3.1 Silhouette_score()

Compris entre [-1, 1], ce coefficient combine 2 mesures : la distance moyenne entre chaque élément et le reste des éléments de son cluster ainsi que la distance moyenne minimale de l'élément aux éléments des autres clusters



1.3.2 Davies Bouldin (min)

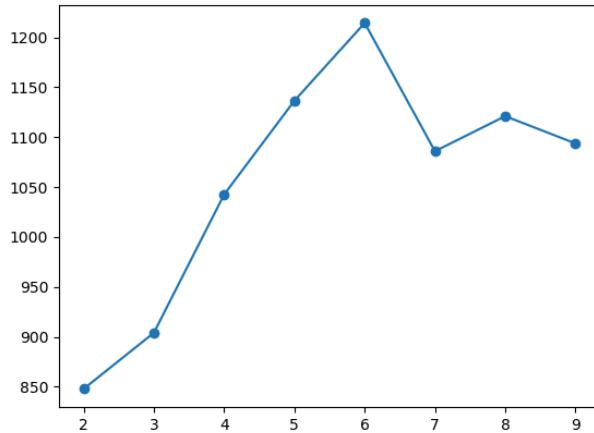
Compris entre $[0, +\infty]$, ce coefficient s'appuie sur la moyenne des distances entre chaque élément et le centre de gravité de son cluster ainsi que sur la distance entre les centres des clusters



1.3.3 Evaluation Calinski-Harabasz (max)

Compris entre $[0, +\infty]$, ce coefficient correspond au rapport entre la somme de la dispersion entre éléments du même cluster et la somme de la dispersion des éléments de clusters différents. Ce coefficient est particulièrement sensible à la taille du jeu de données.

On souhaite ici maximiser au maximum ce coefficient



En utilisant ces métriques, on souhaite automatiser notre initialisation du nombre de cluster en créant une boucle augmentant progressivement le nombre de cluster en stoppant les itérations lorsque la métrique d'évaluation perd en performance maximale (minimale pour DB)

1.4 Point faible/ Point fort

1.4.1 Diamond9 (+)

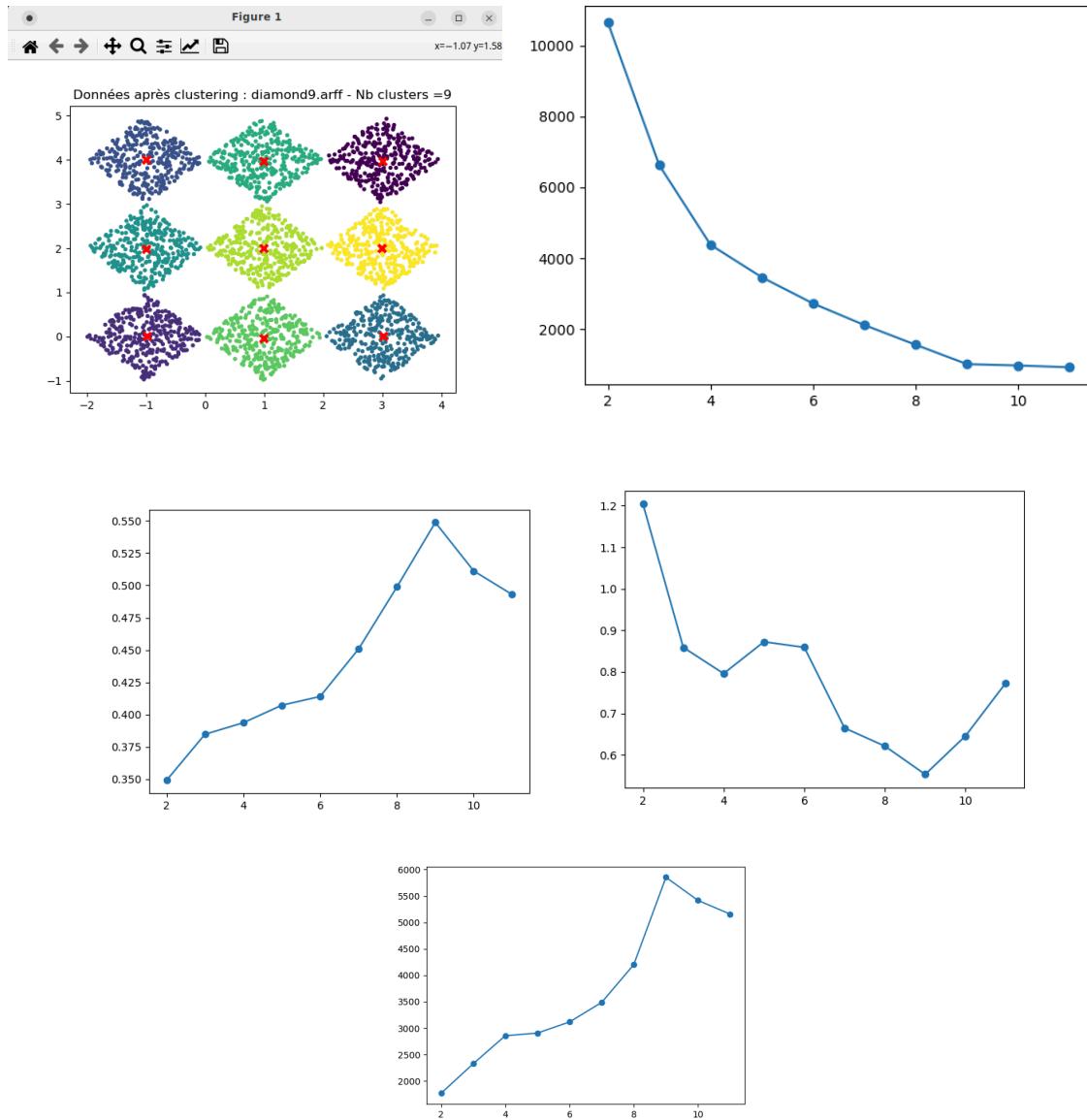


Figure 3 : Courbe d'inertie (1) et de l'indice de Calinski-Harabasz (2) l'indice de Davies_Bouldin (3) l'indice de Silhouette_Score (4) en fonction de k

Ici, visuellement, on peut voir que le nombre optimal de cluster est 9.

—> A l'aide des courbes **l'inertie , silhouette, Davies bouldin** de l'indice de **Calinski-Harabasz** en fonction de k, on voit très clairement que la valeur optimale de k est neuf.

1.4.2 Donut2 (-)

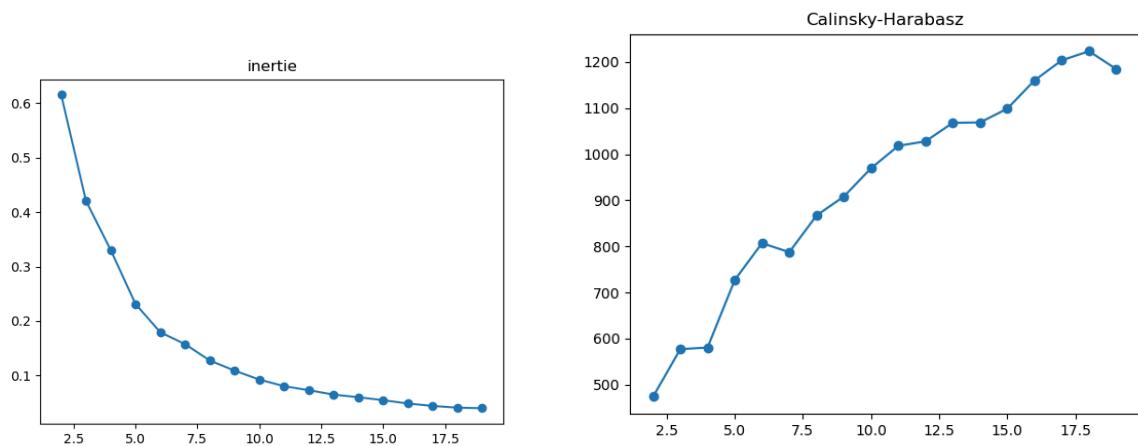
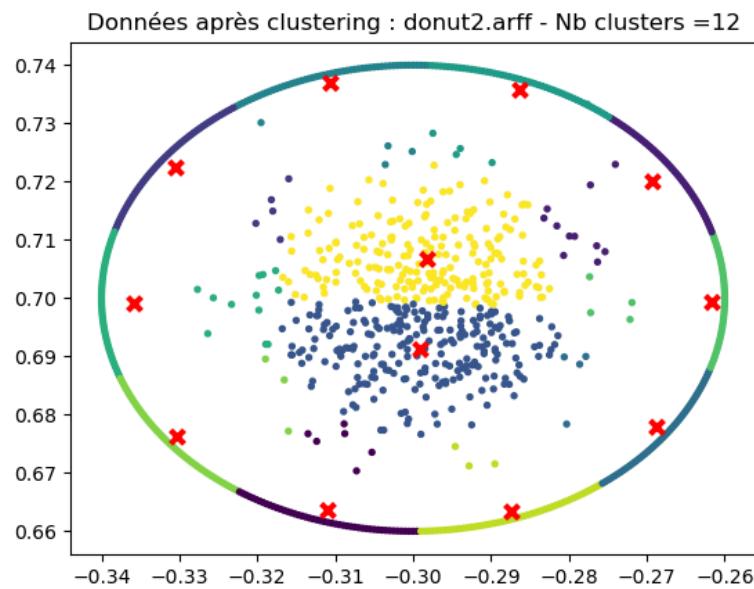


Figure 4: Courbe d'inertie (1) et de l'indice de Calinski-Harabasz(2) en fonction de k

L'algorithme ne considère pas les points en cercle comme un seul est même cluster pour la simple et bonne raison que le calcul de distance est fait à partir d'un point moyen. Cela a comme impact que les points proches entre eux mais sur de grandes distances peuvent être considérés comme appartenant à des clusters différents.

1.5. Mini-Batch

Test sur Xclara.arff en variant le nombre de Mini-Batch avec K =3 (optimal)
Résultat:

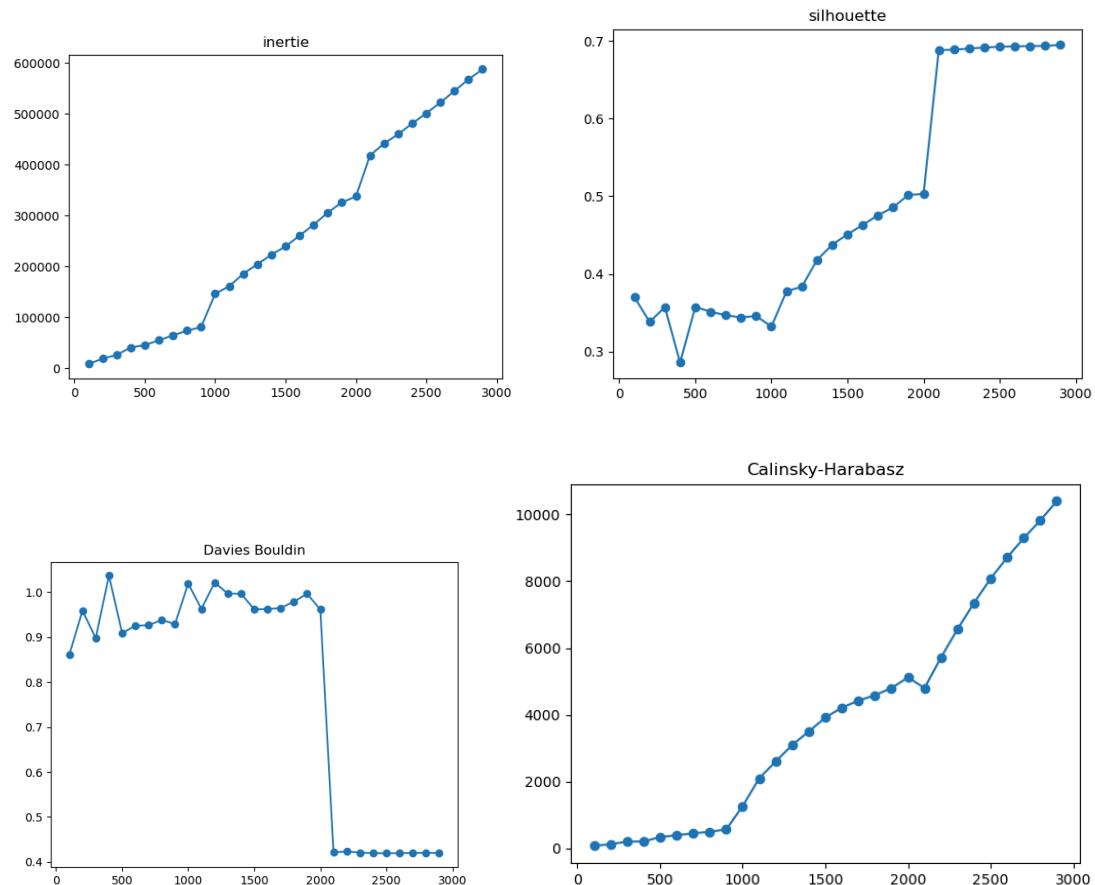


Figure 5: Courbe de l'évolution des indices en fonction de la Batch size

En regardant les courbes des indices des évaluations métriques, on peut voir l'impact de la batch-size sur le clustering.

Il faut cependant déterminer une bonne valeur de batch size avant de lancer le clustering.

2. Clustering Agglomératif

2.1 Dendrogramme:

Le dendrogramme est une représentation graphique utilisée pour visualiser les résultats d'un clustering hiérarchique agglomératif. Chaque nœud du dendrogramme correspond à un cluster, et la hauteur à laquelle deux clusters sont fusionnés reflète leur niveau de similarité.

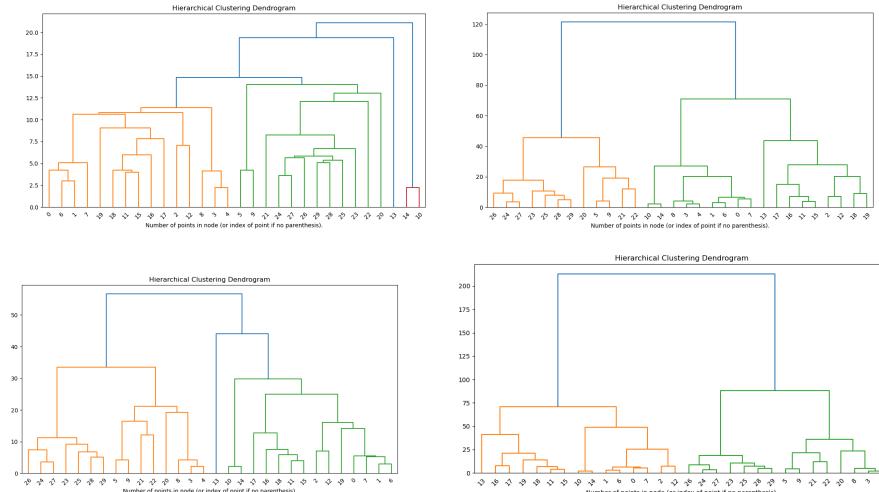


Figure 6 : Algorithme Clustering Agglomératif en fonction des différents type de linkage

Single (1) Complete (2) Average (3) Ward(4)

	xclara	aml28	size5	cluto-t8-8k
Single	59.15 mss	1.18ms	16.01ms	561.42ms
Complete	121.31ms	1.68ms	48.42ms	1786.15ms
Average	127.74 ms	2.8ms	23.41ms	1931.52ms
Ward	129.24 ms	1.73ms	31.33ms	2049.96ms

Tableau 1 : Temps d'exécution pour la méthode de clustering agglomératif

On observe que la méthode de linkage **single** présente le temps d'exécution le plus court, quelle que soit la taille du dataset. La taille du dataset influence bien entendu le temps d'exécution, affectant l'ordre de grandeur : ainsi, le dataset le plus petit, **aml28**, affiche le temps d'exécution le plus rapide, tandis que le dataset le plus volumineux, **cluto-t8-8k**, nécessite un temps beaucoup plus long, avoisinant presque la seconde.

2.2.Linkage

Le but est de fusionner petit à petit de plus en plus de clusters, jusqu'à ce que finalement tous les points soient dans un seul cluster pour cela la fusion se fait selon plusieurs méthodes de linkage:

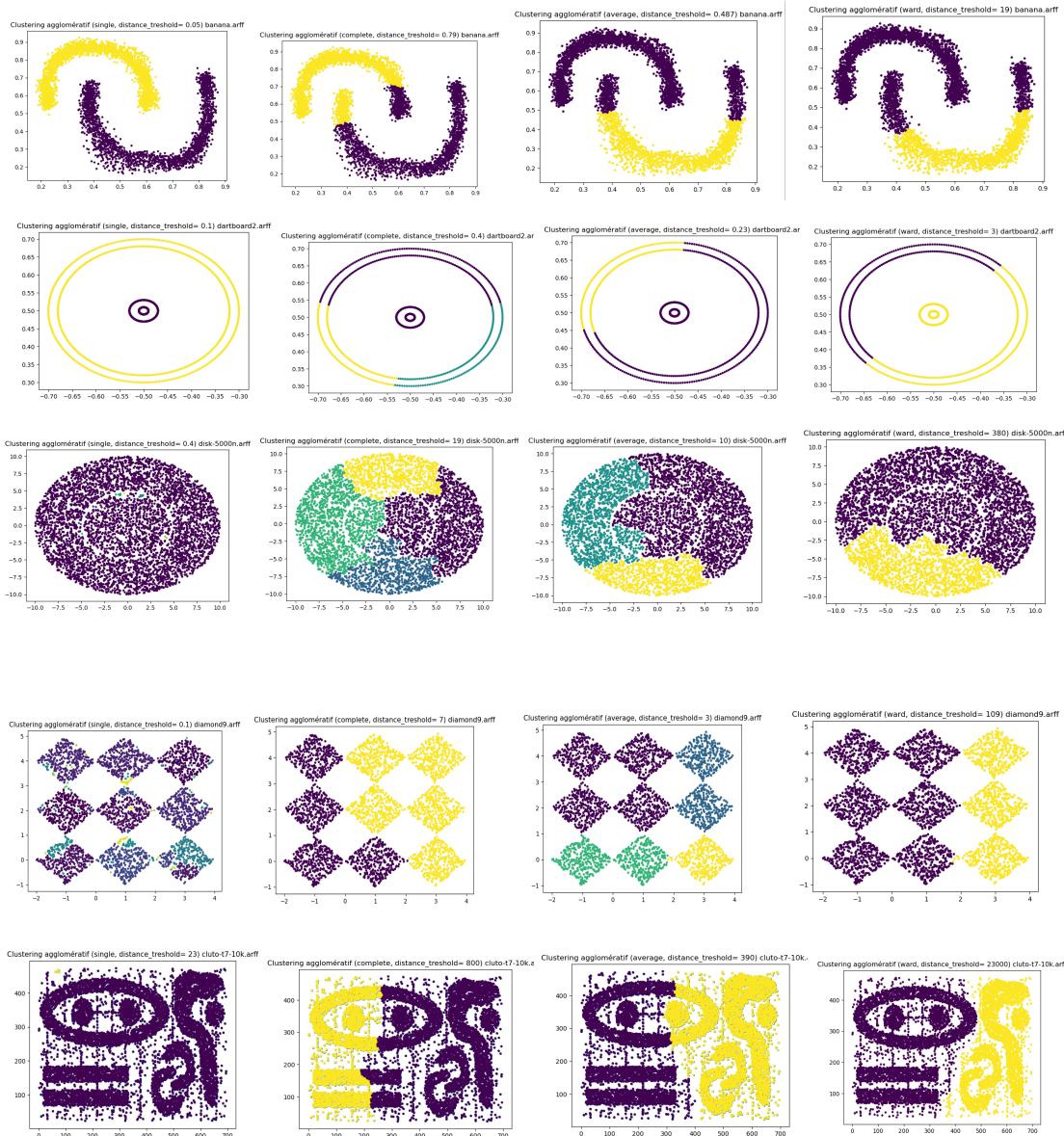


Figure 7 : Linkage Single (1) Complete (2) Average (3) Ward(4)

2.2.1 Single

Le linkage Single utilise le minimum de la distance euclidienne entre deux points de chaque clusters, cette méthode est utile lorsque les points sont relativement petit et bien séparés

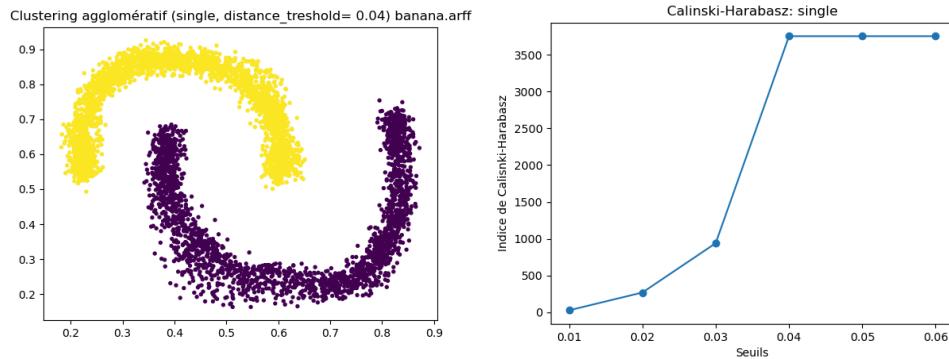


Figure 8: Courbe de l'indice de Calinski-Harabasz en fonction du seuil sur Banana.arff

Ce dataset contient des données proches entre elles et permet de mettre en valeur le linkage Single

La deuxième courbe montre que entre [0.04,0.06] le seuil de distance atteint son maximum d'indice de Calinski-Harabasz

→ Cette méthode a tendance à favoriser des clusters allongés et chaînés car elle ne regarde que la paire de points la plus proche entre deux clusters.

+ Bruit

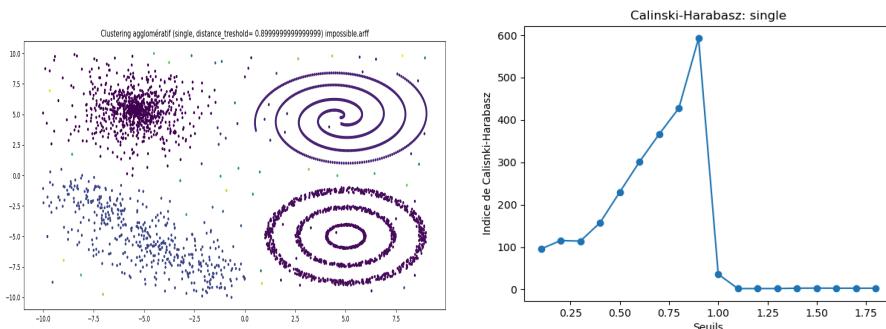


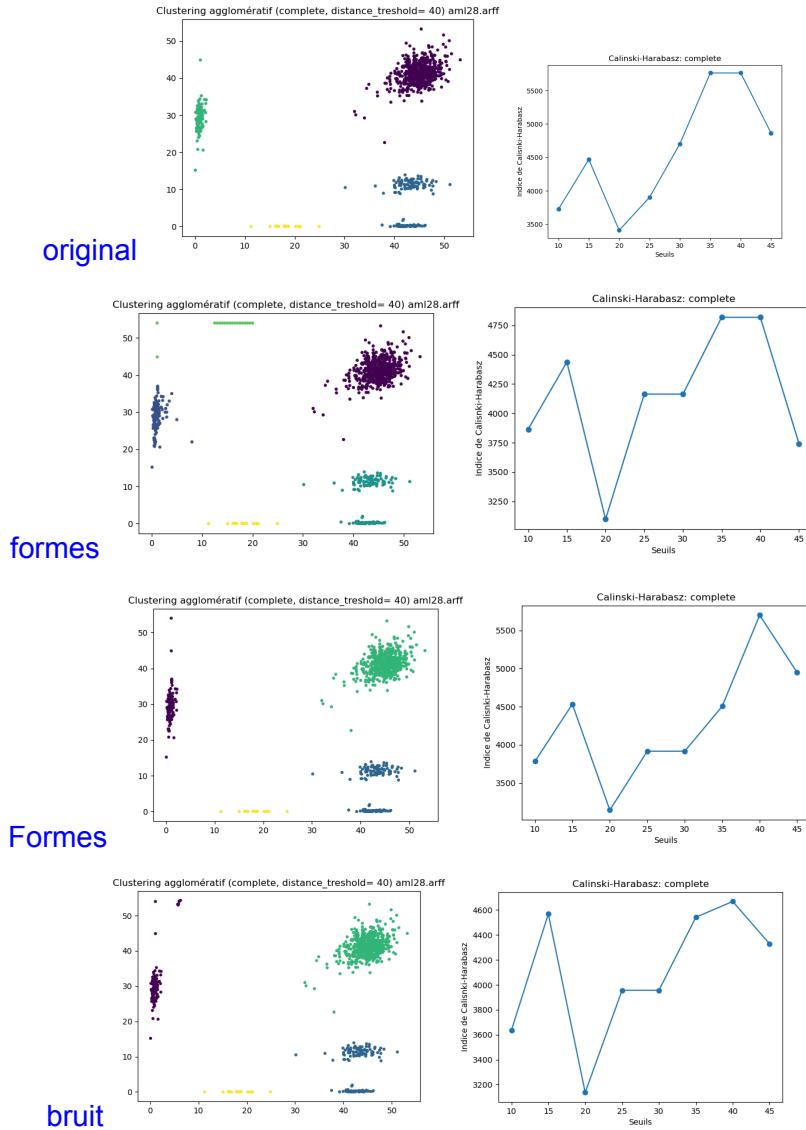
Figure 8 : Courbe de l'indice de Calinski Harabasz en fonction du seuil sur impossible.arff

Néanmoins, en présence de bruit, le modèle atteint son point maximum pour ensuite se détériorer et se stabiliser à une valeur inférieure à son maximum

→ Un point isolé peut être relié à un cluster, simplement parce qu'il est proche d'un seul point du cluster, même si sa densité est faible ou qu'il est éloigné du reste des points du cluster

2.2.2 Complete

Le lien complet relie les points de distance euclidienne maximale, utile aux clusters dense et bien séparés



> Tendance à former des clusters **compacts et sphériques** et moins sensible au bruit que Single linkage

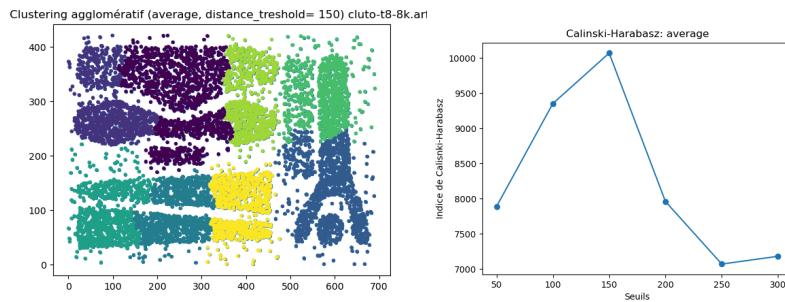
>**Sensibilité aux clusters de petite taille** : Cette méthode peut parfois sur-diviser des clusters denses en plusieurs petits clusters (2)

>**Sous-estime les structures complexes** (2) (3)

2.2.3 Average

Cette méthode fusionne les clusters en utilisant la distance moyenne entre tous les points des 2 clusters

Utile aux clusters denses mais faible face aux points bruits

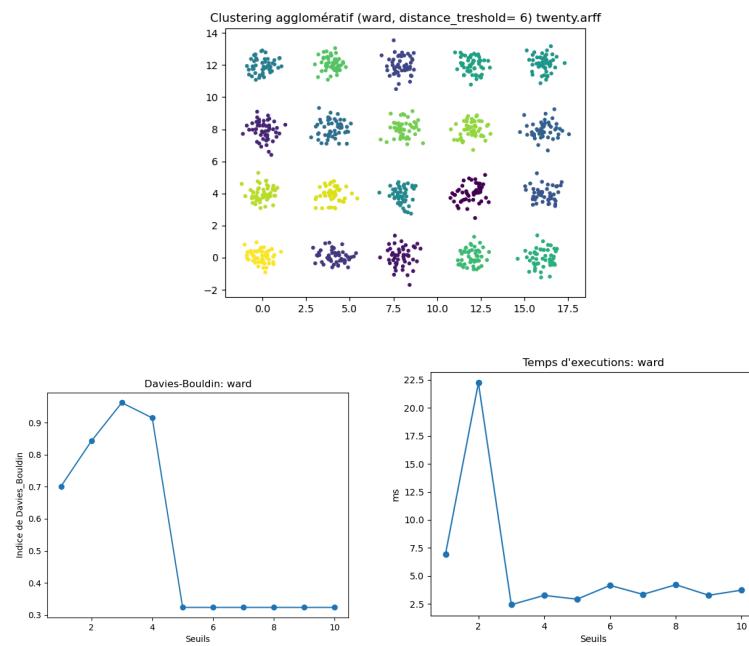


->+Cette méthode privilégié un équilibre entre la compacité des clusters et leur capacité à être légèrement allongés

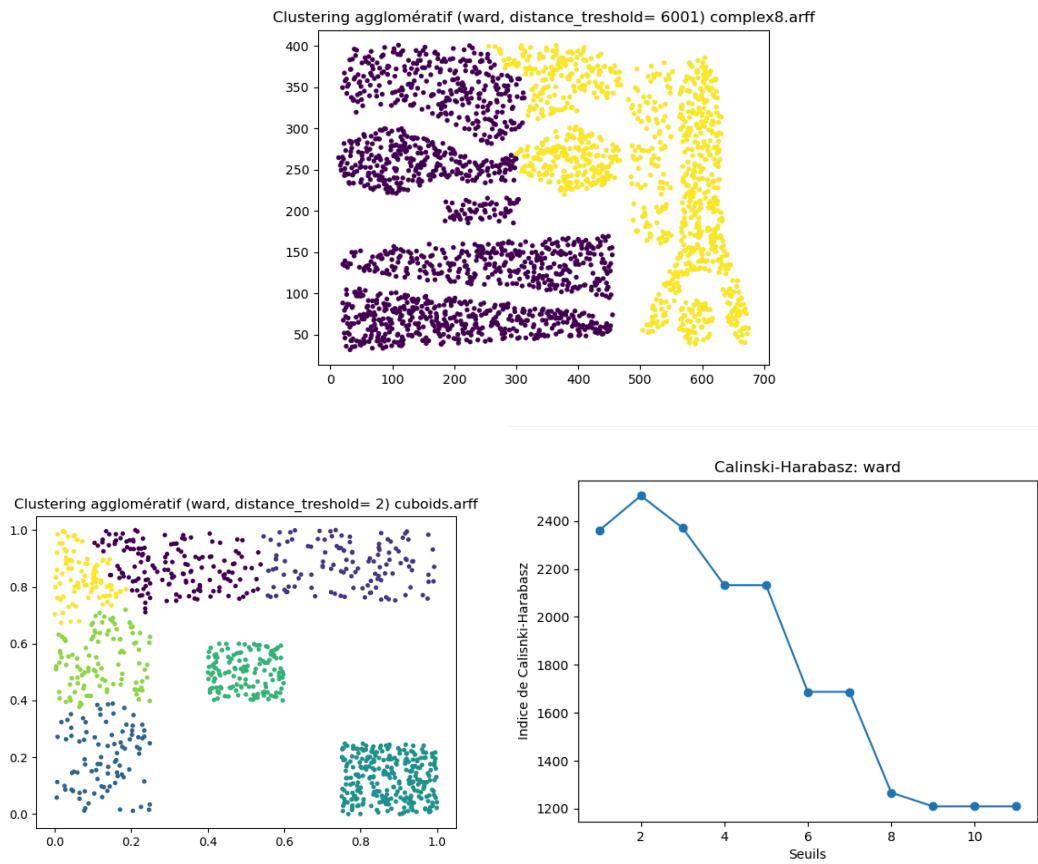
-Si les clusters ont des densités très différentes, cette méthode peut mal les séparer, car elle fait une moyenne des distances

2.2.4 Ward

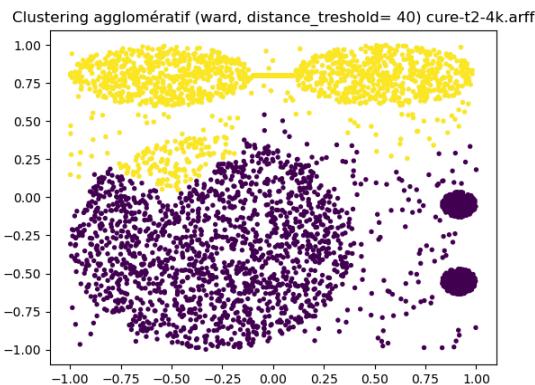
Cette méthode minimise la somme des carrés des distances entre les points d'un même cluster



-> Elle favorise fortement la formation de **clusters compacts et sphériques** avec une faible variance interne Les clusters formés sont **très compacts et homogènes**



-> 1. Moins performant pour des clusters de formes allongées ou complexes.



-> 2. Le bruit peut perturber la construction des clusters comme Ward se base sur la variance interne des clusters, ce qui peut être fortement influencé par des points anormaux

3. Clustering DBSCAN

3.1 Principe

L'algorithme DBSCAN repose sur deux paramètres essentiels : la distance **Épsilone (ϵ)** et le nombre minimum de points **min_samples**. Ces paramètres déterminent la sensibilité de l'algorithme à la densité des données, ce qui permet d'identifier des clusters aux formes variées. En examinant les voisins de chaque point de manière itérative, DBSCAN attribue les points à des clusters ou les qualifie de bruit, créant ainsi une segmentation robuste et flexible des données.

Afin de composer un cluster l'algorithme se compose de 2 parties

1. “**core point**”: En premier temps on définit ,pour un point, une zone locale de diamètre **Épsilone (ϵ)**, cette zone doit contenir le nombre minimale de voisin **min_pts**
2. On définit un cluster en commençant par un point “**core point**” , si un voisin sous le diamètre **Épsilone (ϵ)** est un “**core point**” il rejoint son cluster , sinon il devient lui-même un “**core point**”

Un voisin devenu “**core point**” rejoint le cluster sans que ses voisins deviennent un Les points non **core point**” sont considérés comme bruit

3.2 Choix des paramètres avec les méthodes d'évaluation

3.2.1 Min pts:

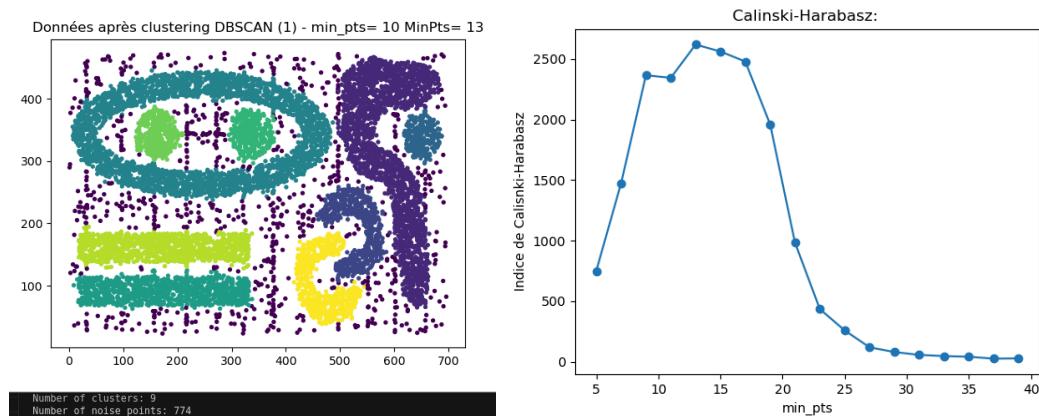
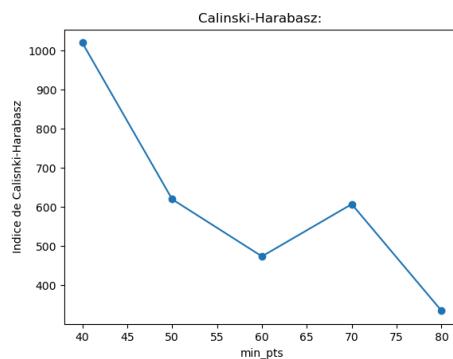
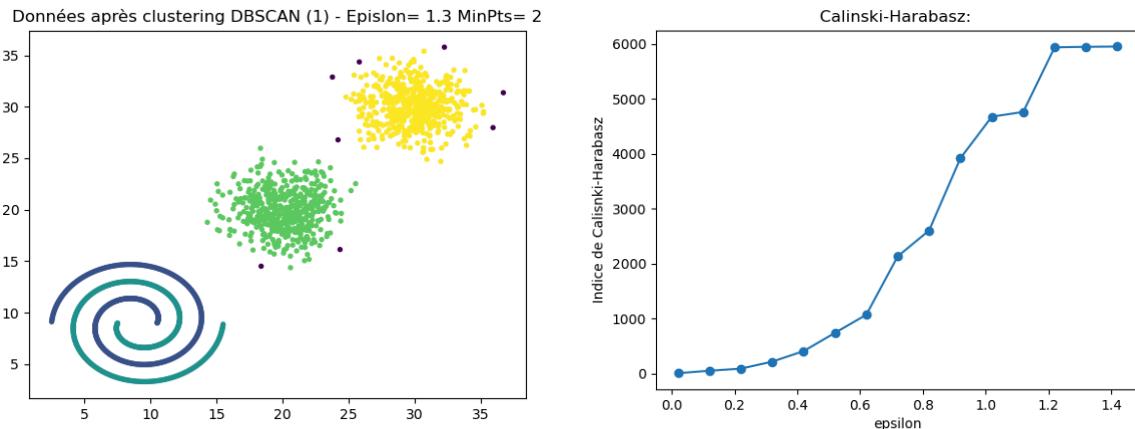


Figure 9: Courbe de l'indice de Calinski-Harabasz en fonction de min_pts

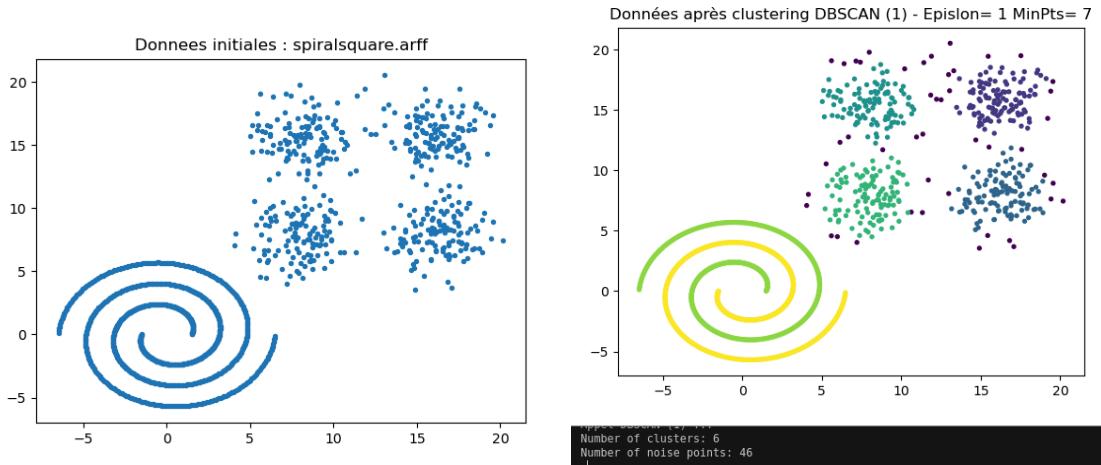
3.2.2 Epsilon:



Une valeur de epsilon trop faible peut fragmenter les clusters, tandis qu'une valeur trop élevée peut les fusionner, rendant l'identification des groupes moins précise.

Dans la dernière courbe, on observe qu'avec un epsilon fixé, l'augmentation de min_pts, le nombre minimum de points requis pour qu'un groupe soit considéré comme un cluster, dégrade l'indice de qualité du clustering. Cela s'explique par le fait qu'un nombre élevé de min_pts impose une densité de points plus stricte, ce qui exclut certains points des clusters, rendant la structure globale moins cohérente et détériorant ainsi les performances du modèle

3.3 Points forts / Points faibles



+Identification du bruit: Contrairement à certaines méthodes qui pourraient assimiler les valeurs aberrantes à des clusters, DBSCAN les classe comme du bruit.

-Sensibilité aux paramètres Min-pts et Epsilon: Les performances de l'algorithme dépendent fortement du réglage de ces paramètres,

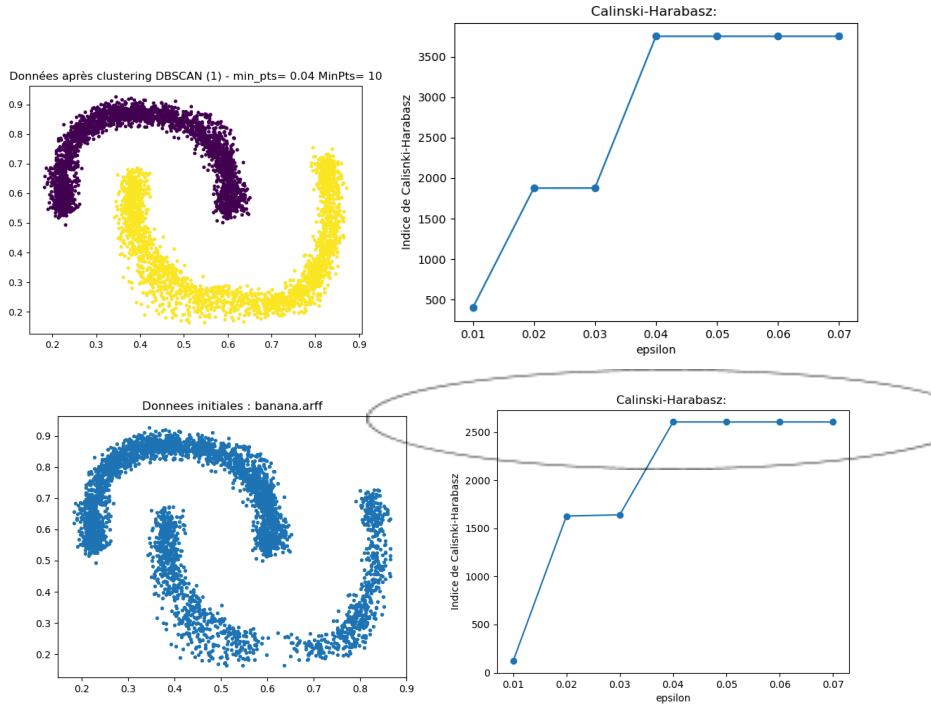


Figure 10: Dégradation du max l'indice de Calinski-Harabasz

-Densité: Si les clusters ont des densités différentes , DBSCAN ne peut les gérer correctement

4. Clustering HDBSCAN

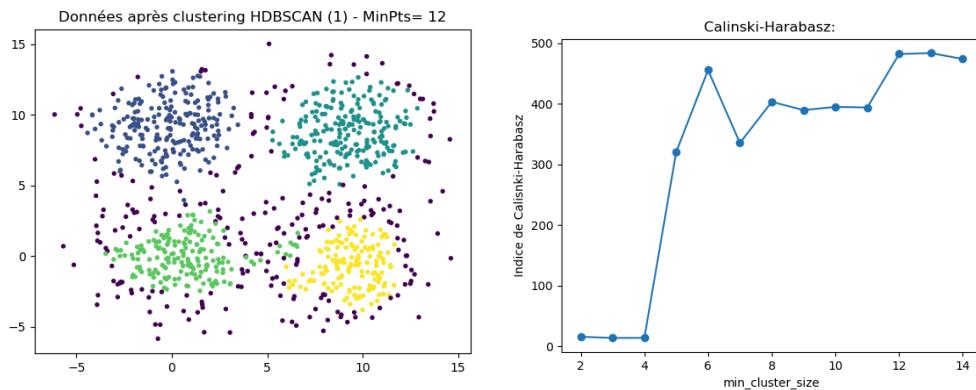
4.1 Principe

L'algorithme HDBSCAN avec un seul paramètre `min_cluster_size` est basé sur l'algorithme précédent DBSCAN et l'algorithme hiérarchique.

Afin de composer un cluster, l'algorithme HDBSCAN se compose de plusieurs parties:

1. **Distance mutualisée des k plus proches voisins:** Pour chaque point on définit la **distance core** qui est la distance jusqu'au k-ième plus proche voisin. Elle définit la densité locale autour du point. Ensuite la **distance mutualisée** entre deux points est calculée comme étant le maximum entre leurs **distances cores** et la distance qui les sépare
2. **Hiérarchie de densité:** A partir de ces distances, HDBSCAN construit un arbre couvrant minimal qui connecte tous les points avec les distances les plus courtes
3. **Clusters:** En réduisant la densité, l'algorithme identifie les clusters. Les points qui ne font pas partie des clusters **stables** sont considérés comme du **bruit**

4.2 Comparaison



	Indice de Davies-Bouldin	Indice de Calinski-Harabasz	Indice de Silhouette-Score	Temps d'exécution (en ms)	Jeu de Données
Dbscan 6eps , 8 min pts	1.76	1750	0.44	1750ms	xclara
Hdbscan 6	1.57	6412	0.65	3702.18ms	

Dbscan 1 eps 20 min	1.25	514.57	0.42	599.71ms	sizes3
Hdbscan 10	1.56	461.86	0.48	447.11ms	
Dbscan	0.31	125	0.51	2033.09ms	long2
Hdbscan 7	1.99	119	0.29	2562.36ms	
Dbscan 1.2 eps pour 8	1.49	381.46	0.47	1006.2ms	sizes4
Hdbscan 8	1.43	433.62	0.49	2602.85ms	

Tableau 2 : Comparaison de DBSCAN et HDBSCAN sur différents jeu de données

Par rapport à DBSCAN,

Plus robuste à la variation de paramètres

Gère bien les densités variables

Conclusion

- **DBSCAN** est simple à implémenter et fonctionne bien sur des clusters de densité uniforme, mais il est limité lorsque les données ont des structures de densité variable.
- **HDBSCAN** est plus flexible et performant pour gérer des clusters de densités variables, mais il est plus complexe à paramétrier et peut être plus **lent**.

5. Analyse Comparative

L'objectif de cette partie est de l'analyse de différents jeu de données en comparant à chaque fois la performance de chaque algorithme de clustering de point de vue temps et indices des métriques d'évaluation

2d-4c	Indice de Davies-Bouldin	Indice de Calinski-Harabasz	Indice de Silhouette-Score	Temps d'execution (en ms)
K-Means	0.16	43805.32	0.86	80.66
Agg (Single)	0.16	36113.11	0.86	11.42
Agg (Complete)	0.16	36113.11	0.86	28.1
Agg (Average)	0.16	36113.11	0.86	19.9
Agg (Ward)	0.16	36113.11	0.86	24.37
DBSCAN ($\epsilon = 5$)	0.16	36113.11	0.86	36.6
HDBSCAN	0.16	36113.11	0.86	32.07

Tableau 3 : Tableau comparatif des résultats de chaque méthode pour le dataset 2d-4c

triangle1	Indice de Davies-Bouldin	Indice de Calinski-Harabasz	Indice de Silhouette-Score	Temps d'execution (en ms)
K-Means	0.35	6547.67	0.78	5.03
Agg (Single)	0.37	6185.76	0.78	99.96
Agg (Complete)	0.37	4642.17	0.73	36.86
Agg (Average)	0.35	6315.15	0.77	103.04
Agg (Ward)	0.37	5185.76	0.78	104.32
DBSCAN	0.37	6185.76	0.78	23.74
HDBSCAN	0.37	6185.76	0.78	100

Tableau 4 : Tableau comparatif des résultats de chaque méthode pour le dataset 2d-4c

cluto-t5-8k	Indice de Davies-Bouldin	Indice de Calinski-Harabasz	Indice de Silhouette-Score	Temps d'execution (en ms)
K-Means	0.6	45380.15	0.55	103.26
Agg (Single)	1.38	2.58	-0.59	535.02
Agg (Complete)	0.7	30396.77	0.43	1954.04
Agg (Average)	0.6	43922.48	0.54	2031.77
Agg (Ward)	0.6	44217.81	0.54	1927.28
DBSCAN ($\epsilon = 7$)	1.7	914.28	-0.36	442.01
HDBSCAN	2.00	12163.48	0.48	547.92

Tableau 5 : Tableau comparatif des résultats de chaque méthode pour le dataset cluto-t5-8k

En nous basant sur les tableaux comparatifs, voici une analyse de chaque algorithme en fonction des indicateurs d'évaluation et du temps d'exécution :

- **K-means** : Le K-means se distingue par sa rapidité d'exécution, bien qu'il soit limité aux clusters sphériques. Il obtient de bons scores en termes de Silhouette et de Calinski-Harabasz dans des cas de clusters bien séparés. Cependant, ses performances chutent dès que les clusters prennent des formes complexes ou présentent des densités variées.
- **Clustering agglomératif** : Les différentes méthodes de linkage (Single, Complete, Average, Ward) dans le clustering agglomératif permettent de s'adapter à des configurations variées, mais au prix d'un temps d'exécution plus élevé, en particulier pour des ensembles de données de grande taille. Le linkage « Ward » offre une bonne homogénéité des clusters, bien qu'il soit plus sensible aux données bruitées par rapport aux autres méthodes de linkage.
- **DBSCAN** : Cet algorithme est efficace pour détecter des clusters de formes irrégulières et gère bien le bruit. Les performances de DBSCAN, toutefois, dépendent de manière significative du choix des paramètres epsilon et du nombre minimal de points (min_pts), ce qui peut s'avérer délicat à régler.
- **HDBSCAN** : Reprenant le principe de DBSCAN avec une approche hiérarchique, HDBSCAN se révèle plus robuste pour des données à densité variable et offre une flexibilité accrue dans la gestion du bruit. Cependant, il est généralement plus lent et plus complexe à paramétrier que DBSCAN, nécessitant une fine compréhension des caractéristiques des données pour une utilisation optimale.

Conclusion

En conclusion, cette étude comparative des méthodes de clustering met en lumière les spécificités et les performances variées de chaque algorithme en fonction des caractéristiques des données. Le choix de l'algorithme de clustering optimal dépend de la structure des données, de la forme des clusters, et de la densité.

K-means, simple et efficace, est adapté aux clusters sphériques, tandis que le clustering agglomératif offre une grande flexibilité en termes de clusters, bien que souvent plus coûteux en temps de calcul. DBSCAN et HDBSCAN s'avèrent particulièrement robustes pour les clusters de densité variable, mais nécessitent une attention particulière aux paramètres pour garantir de bons résultats.