RAPPORT DE TP DE CLUSTERING DE TEO GENEAU

ANALYSE ET TRAITEMENT DE DONNÉES – BIG DATA

21 janvier 2022

Informations relatives au document

INFORMATIONS GÉNÉRALES

Auteur(s)	Teo Geneau
Lieu	INSA Toulouse
Version	V3.0
Référence	EA1

HISTORIQUE DES MODIFICATIONS

Version	Date	Rédigé par	Modifications
V3.0	23/01/2022	Teo Geneau	Version Finale : ajouts de chapitres et corrections

DESTINATAIRES

Nom	Fonction / Entité
Marie-José Huguet	Professeur / INSA

https://github.com/TeoGeneau/Big-Data-ISS

TABLE DES MATIERES

TABLE DES MATIERES

INTRODUCTION

Lors de ces TP d'analyse et traitement de données, nous nous sommes familiarisés avec différentes méthodes de clustering :

- Le clustering k-Means
- Le clustering agglomératif
- Le clustering DBSCAN

Le premier objectif est de tester ces différentes méthodes de clustering sur des jeux de données différents afin d'étudier l'impact des paramètres et sur quel type de données ils peuvent s'appliquer, et cela, afin de dégager les points forts et les points faibles de chaque méthode.

Dans un second temps, il s'agira de tester ces méthodes sur de nouveaux jeux de données afin de réaliser une analyse comparative des méthodes sur ces nouvelles données.

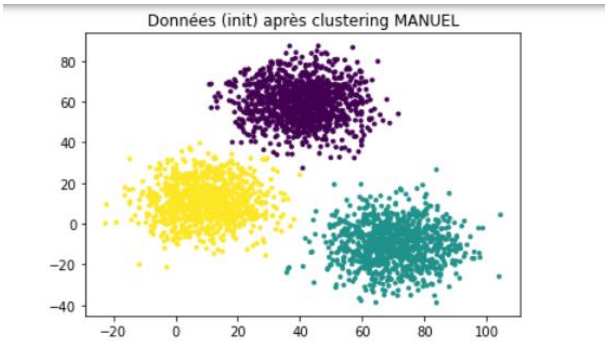
Méthodes de clustering

Clustering k-Means

La méthode K-Means, aussi appelée méthode des centres mobiles, est apparue dans les années 50/60, il s'agit d'une méthode très populaire due à sa simplicité et sa rapidité de calcul. L'algorithme est simple et applique une stratégie gloutonne se reposant sur un nombre k de cluster à fixer, puis on affecte à chaque donné le centre le plus proche, on recalcule des nouveaux centres et on réitère jusqu'à atteindre une certaine stabilité ou bien lorsqu'on atteint un nombre défini d'itérations. Ainsi, on peut discriminer des groupes dans un jeu de données.

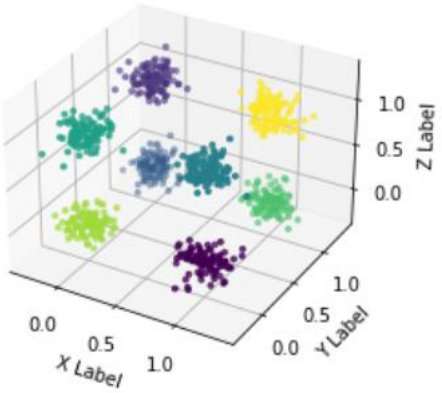
Afin d'étudier cette méthode, plusieurs jeux de données pertinents ont été sélectionnés.

On applique la méthode pour un nombre fixé de clusters sur différents jeux de données en 2D et 3D :




```
nb clusters = 3 , nb iter = 3 , runtime =  23.0 ms
Inertie :  611605.8806933893
Coefficient de silhouette :  0.6945587736089913
```

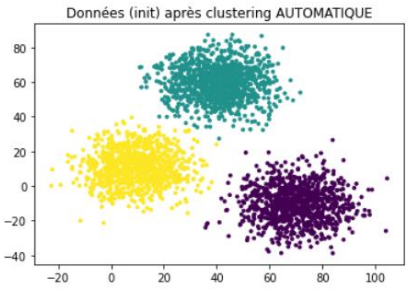
```
Affichage des données en 3D avec
```



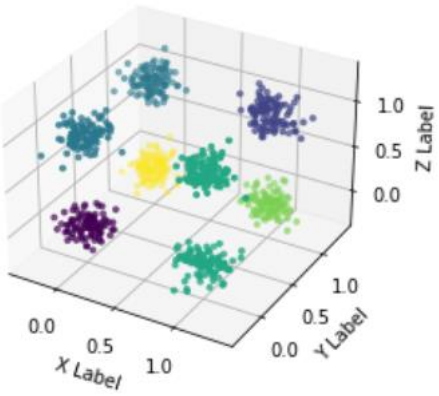
Lorsqu'on déterminer manuellement le nombre de clusters, on obtient de bons résultats pour les jeux de données xclara.arff en 2D et hypercube.arff en 3D avec respectivement k=3 et k=8.

Nous allons maintenant chercher à déterminer le nombre de cluster automatiquement et mesurer différentes métriques pour évaluer l'algorithme.

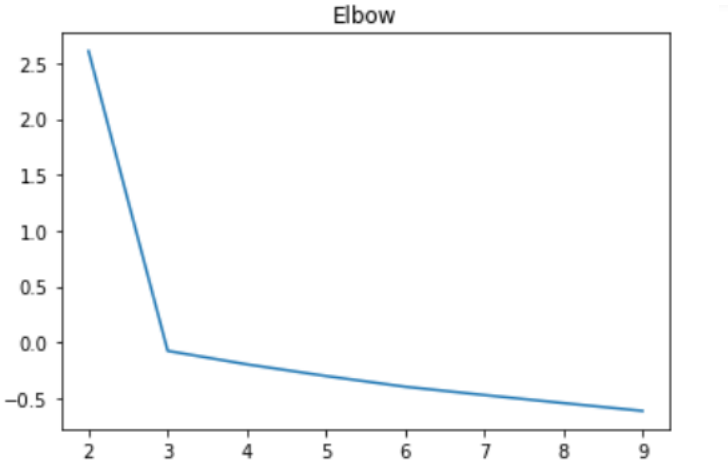
On peut alors observer que si pour le jeu de données xclara notre algorithme a bien fonctionné en retrouvant le nombre exact de cluster, on peut voir qu'il a trouvé k=6 au lieu de k8 pour le jeu hypercube.
L'algorithme n'arrive pas à dissocier les différents clusters.
De plus, le temps de calcul ne semble pas optimisé pour notre algorithme puisqu'il est passé de 40ms à 600ms entre le paramétrage de k manuel et automatique.

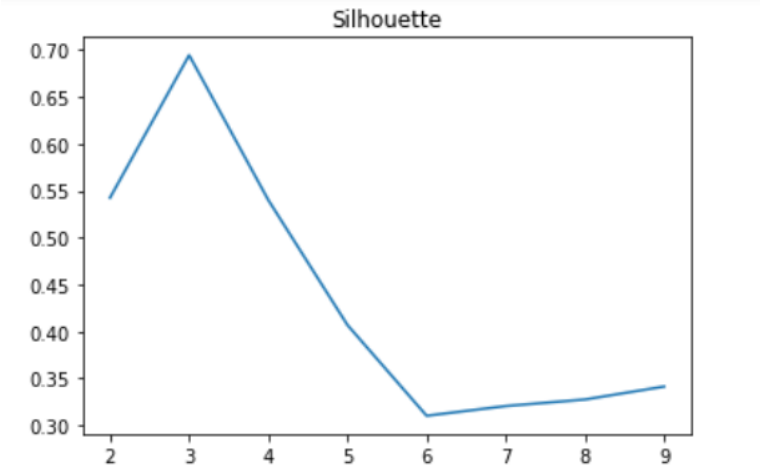


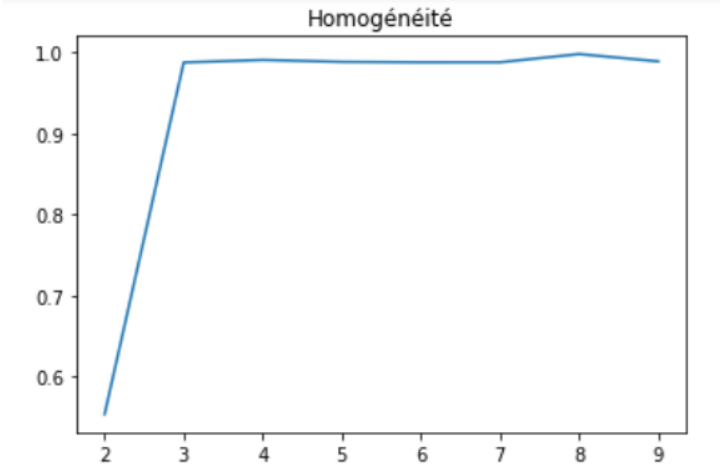

```
nb clusters = 3 , nb iter = 2 , runtime =  2715.35 ms
```

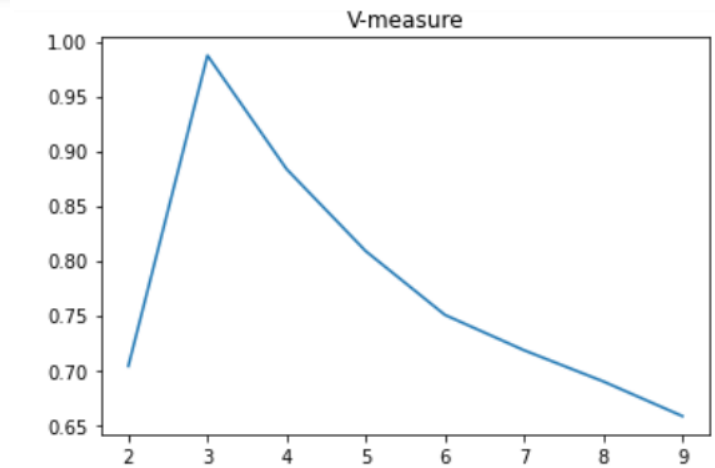


On évalue la méthode en utilisant plusieurs métriques : valeur d'inertie (elbow), la valeur du coefficient de silhouette, l'homogénéité et le V-measure. Pour le jeu de données xclara on peut observer que ces quatre métriques ont toute la même valeur soit 3 tandis que pour le jeu de données hypercube on obtient : inertie = 3, silhouette = 8, v_measure =9 et homogénéité = 2 donnant une moyenne arrondi à 6 clusters contre 8.



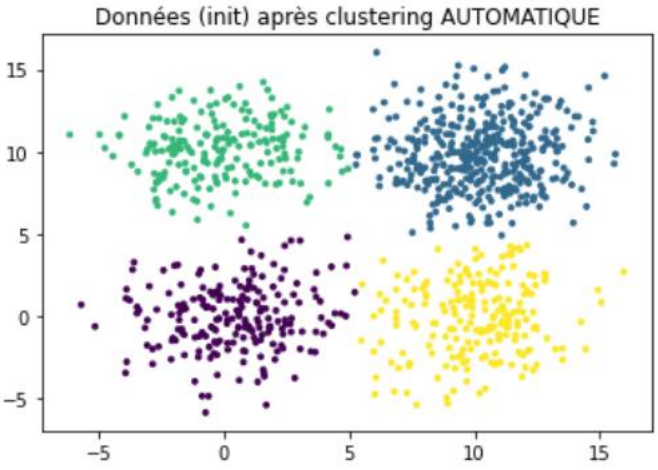




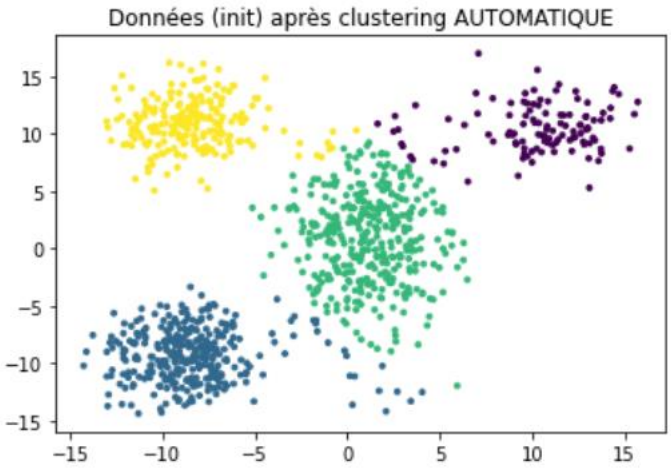


Regardons les résultats pour d'autres jeux de données :

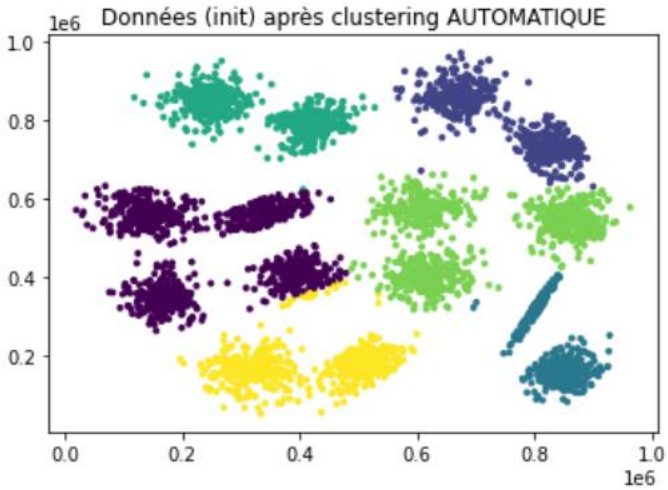
Pour sizes1.arff : fonctionne bien



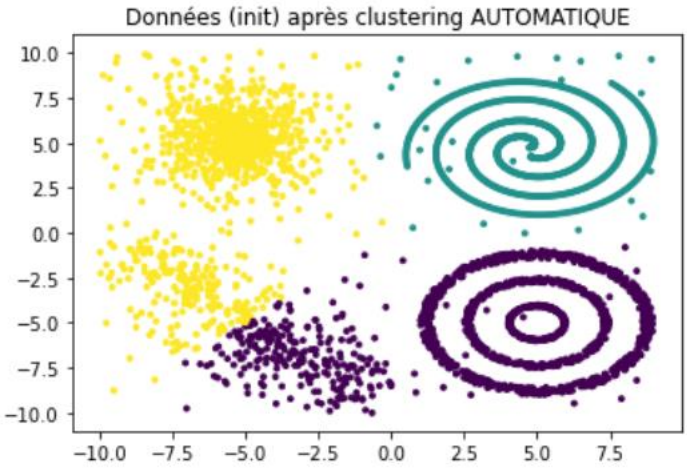
Pour triangle2.arff : fonctionne bien



Pour s-set1.arff : fonctionne mal



Pour impossible.arff : fonctionne mal



Points faibles

Comme on peut le voir sur les jeux de données triangle2, s-set1 et impossible, certaines données sont attribuées aux mauvais clusters, cela est notamment due à la sensibilité de la méthode k-Means à la densité des points, à leur forme à leur taille, ce qui signifie qu'une donnée peut être attribué au mauvais cluster si la densité du cluster est plus forte pour le mauvais cluster par exemple.

De plus, on peut observer sur les figures des jeux impossible et s-set1 que l'algorithme n'arrive pas à classifier correctement les clusters, en effet, c'est une méthode basée sur la distance entre les points, or, la distance en y et en x n'est pas assez élevée pour dissocier correctement les clusters et l'algorithme semble bloqué dans un optimum local. D'où la nécessité de travailler avec des jeux de données comportant des clusters de mêmes tailles, avec une géométrie plate (l'algorithme fonctionne moins bien pour les données en 3 dimensions) et un nombre limité de clusters.

D'autres points faibles existent, comme la nécessité de fixer le nombre de cluster au départ, or cela a un fort impact sur le résultat final. La phase d'initialisation est importante.

Points forts

Cette méthode offre certains avantages, premièrement elle est simple à utiliser, il n'y a qu'un seul paramètre à gérer (k) et le temps de calcul est plutôt faible.

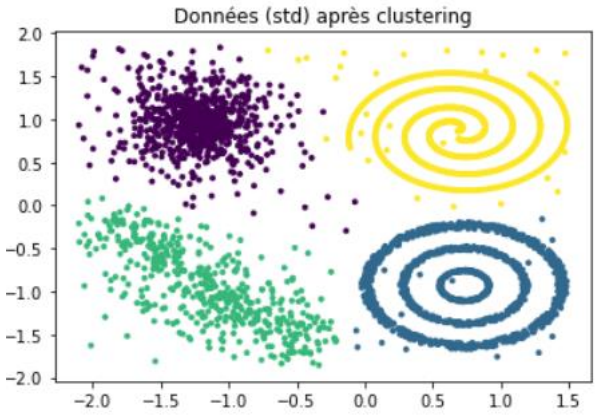
Clustering agglomératif

Dans cette partie, nous allons évaluer une méthode hiérarchique ascendante, c'est-à-dire le clustering agglomératif. La méthode consiste à considérer chaque point au départ comme un cluster et puis peu à peu de fusionner les observations les plus proches par similarité, on itère jusqu'à obtenir 1 seul cluster.
On représente les résultats avec un dendrogramme.
Le nombre de cluster initial est nécessaire pour initialiser l'algorithme mais aussi le choix de la méthode pour déterminer les similarités, nous en avons testé quatre différents :

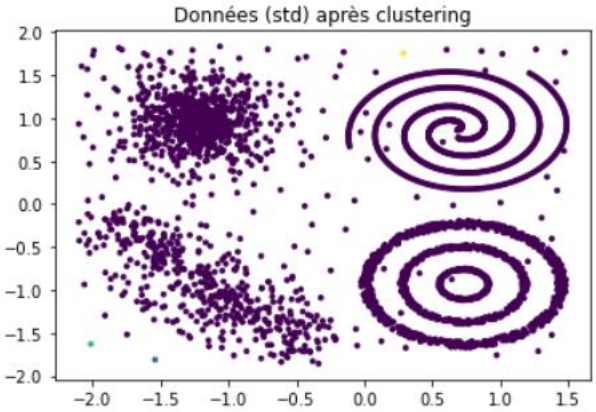
- Single linkage : consistant à minimiser la distance à celle des deux points les plus proches deux à deux.
- Complete linkage : correspondant à la distance maximale entre deux paires de clusters/
- Ward : l'augmentation de la variance intra-cluster est minimale.
- Average linkage : la distance moyenne entre deux points correspond à la moyenne des distances de tous les points.

On teste tous ces paramètres sur les jeux de données xclara.arff et impossible.arff :

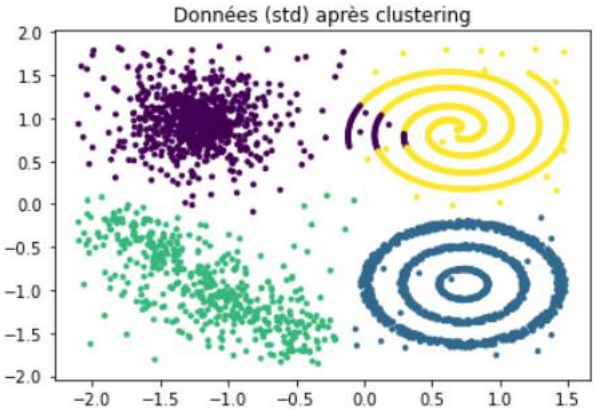
Pour impossible.arff : méthode Average



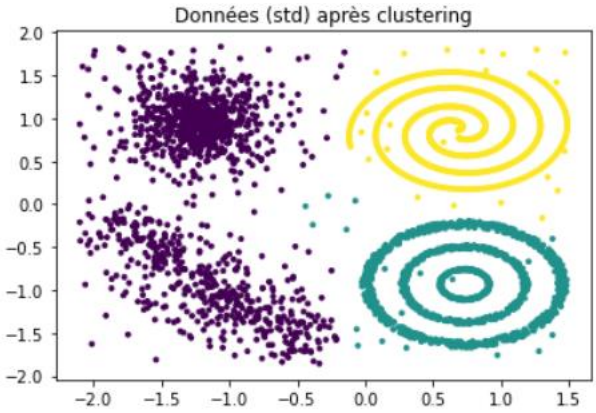
Pour impossible.arff : méthode Single



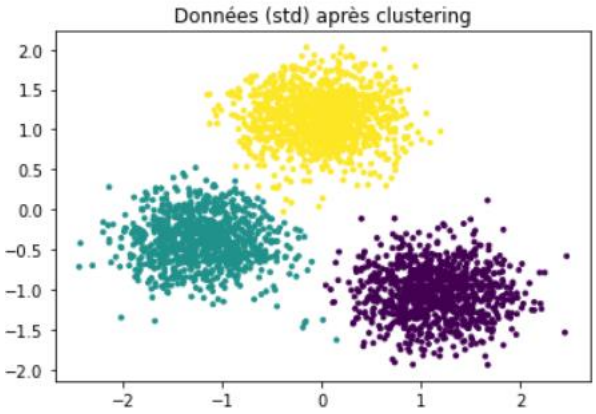
Pour impossible.arff : méthode Complete



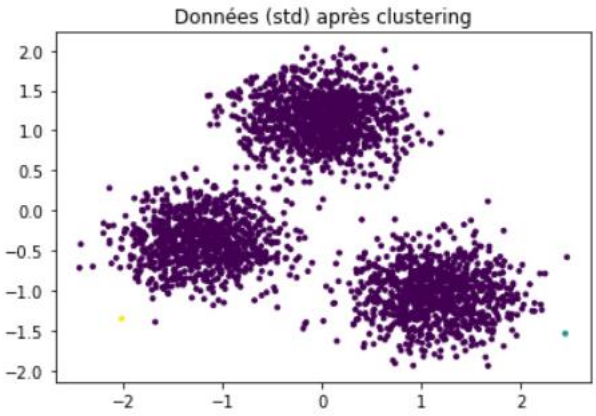
Pour impossible.arff : méthode Ward



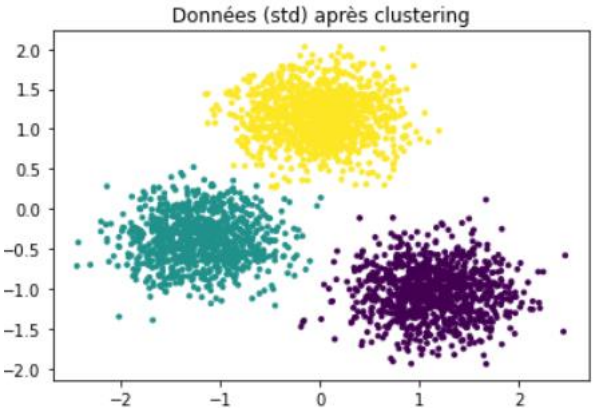
Pour xclara.arff : méthode Average



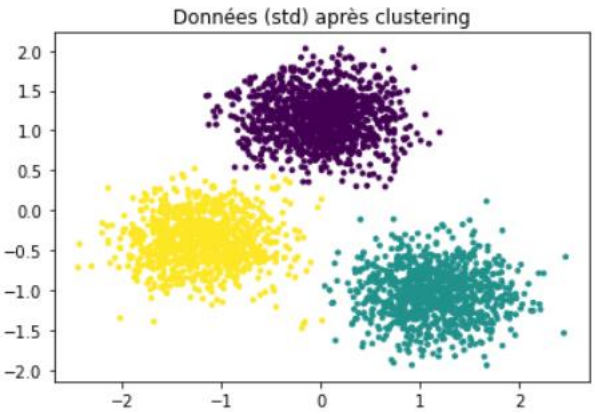
Pour xclara.arff : méthode Single



Pour xclara.arff : méthode Complete



Pour xclara.arff : méthode Ward



Si pour le jeu xclara, 3 méthodes sur 4 sont plutôt concluantes, pour le jeu impossible, seule la méthode Average a permis de déterminer le bon nombre de clusters, il s'agit de la meilleure méthode pour ce jeu précis. Comme on peut le voir, la meilleure méthode à appliquer varie selon le jeu de données.

Pour évaluer cette méthode on a utilisé les métriques suivantes : complexité, homogénéité, silhouette, v-measure, et l'indice de Davies-Bouldin.

En prenant le jeu de données impossible.arff, pour le linkage average, on obtient :

- Complexité = 3
- Homogénéité = 2
- Silhouette = 4
- V-measure = 5
- Davies-Bouldin = 4

Soit, une moyenne de 4 clusters. Si on le compare à la méthode précédente, on observe des temps d'exécution plus élevé (3000ms contre 2000ms), surement dû au calcul de similarité qui augmente le coût de calcul.

Points faibles

La complexité du calcul augmente sensiblement, on passe ici à une échelle plus difficile. L'algorithme est sensible aux différentes anomalies car elle est toujours basée sur un calcul de distance. La lecture du dendrogramme peut être compliquée pour des jeux avec beaucoup de données et de clusters.

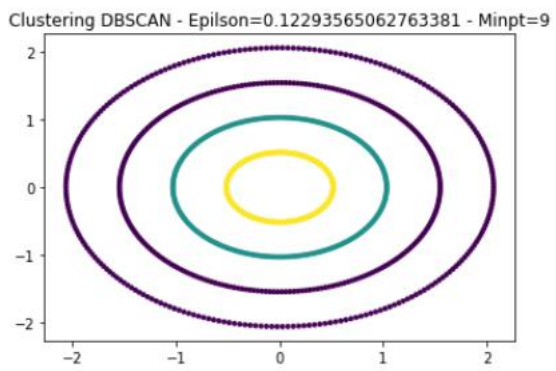
Points forts

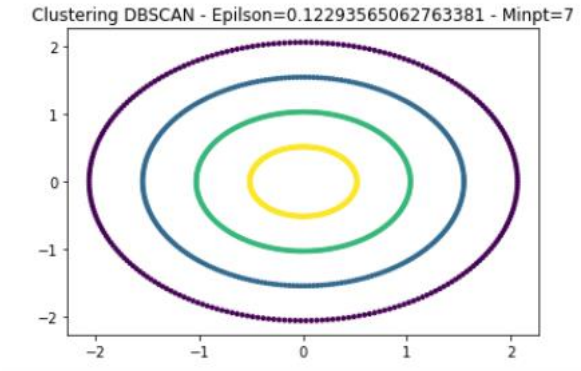
Utilisable avec un nombre de clusters plus élevé que la méthode K-Means. Par ailleurs, il n'a pas besoin d'avoir un nombre fixe de clusters, ce qui en fait une méthode plutôt flexible, ce dernier est à établir en fonction du dendrogramme.

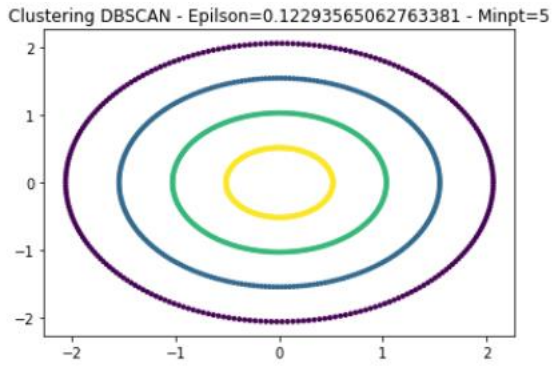
Clustering DBSCAN

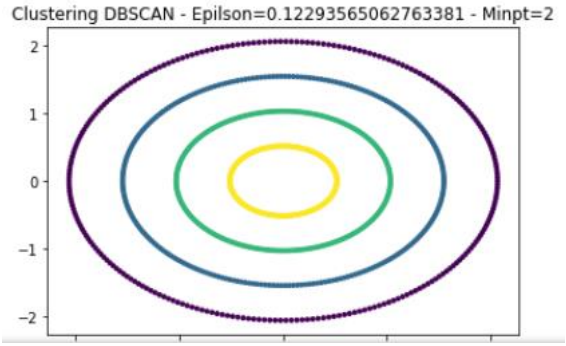
L'objectif de cette méthode est de pouvoir obtenir des clusters avec des formes non convexes. Il fonctionne alors par voisinage (le nombre de point compris dans un rayon donné) : tous les points atteignables dans un rayon sont considérés comme appartenant au même cluster. Elle se repose donc sur la distance entre les deux points les plus proches. Deux paramètres sont utilisés pour optimiser l'algorithme : epsilon (la distance entre deux points) et min-samples (le nombre de points dans un même cluster).

On applique l'algorithme à un jeu de données représentant une forme non convexe comme dartboard1.arff :





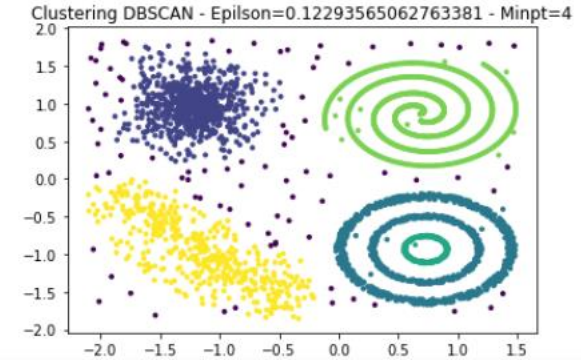




On peut remarquer que le résultat dépend des paramètres utilisés, pour trouver le meilleur résultat il est alors nécessaire de fixer l'un et de faire varier l'autre et inversement jusqu'à obtenir les paramètres optimisant le clustering.

De plus, en prenant le jeu de données impossible, on peut observer que l'algorithme a déterminé 5 clusters pour ces paramètres précis, cependant, certains points ne sont associés à aucun cluster, ils sont considérés comme du bruit ici. Les points correspondants sont en violet dans l'image ci-dessous.

```
Estimated number of clusters: 5
Estimated number of noise points: 86
```



Points faibles

Les paramètres ne sont pas évidents à déterminer et le fait de devoir trouver le bon nombre de voisins ainsi que la taille du voisinage complexifie l'implémentation de l'algorithme. En effet, si on utilise une valeur d'epsilon trop faible, on aura beaucoup d'anomalies, alors que si on met une valeur trop grande on aura, au contraire, une discrimination de clusters trop faible et certains clusters ne seront pas dissociés.
De plus, l'algorithme est sensible aux variations de densité des données.

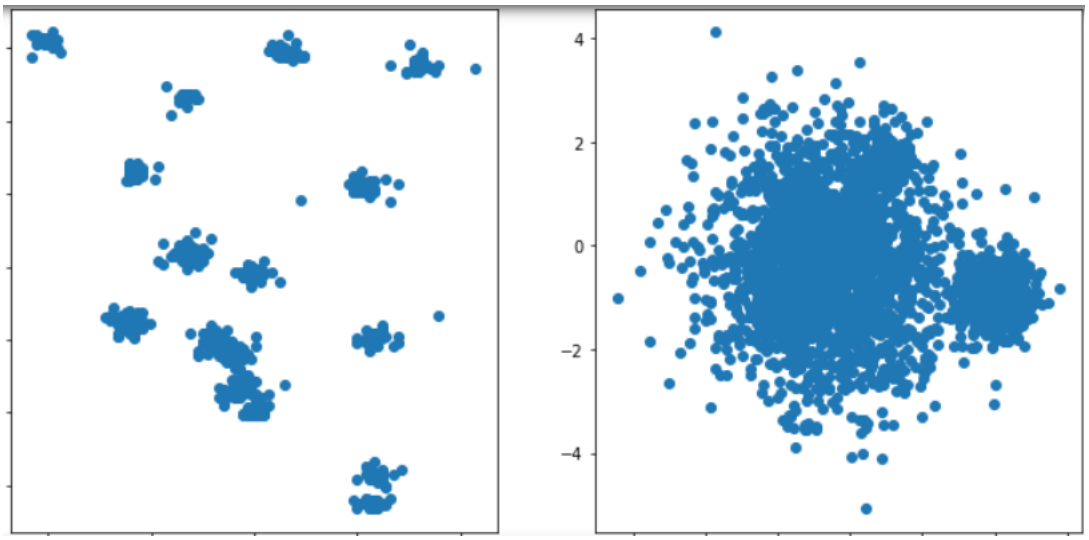
Points forts

L'un des principaux intérêts par rapport aux autres méthodes est qu'il n'y a pas besoin de fixer le nombre de cluster k. De plus, cette méthode offre l'avantage de pouvoir déterminer des clusters non convexes. Comme nous l'avons vu avec l'exemple du jeu impossible.arff, il permet aussi d'éliminer le bruit, ce qui le rend plus robuste aux anomalies que les autres méthodes précédentes.

Analyse comparative sur les nouvelles données fournies

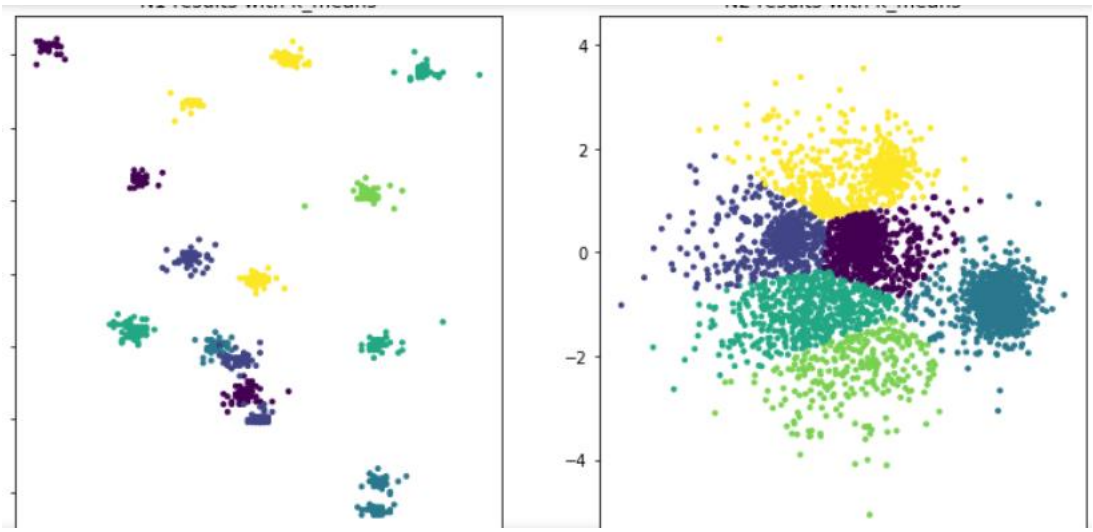
Il s'agit d'appliquer les 3 méthodes étudiées précédemment à un nouveau jeu de données afin de réaliser une synthèse. Pour cela, nous allons utiliser les jeux de données d64 (à gauche) et n2 (à droite).

Observation des deux jeux de données :



Méthode k-Means :

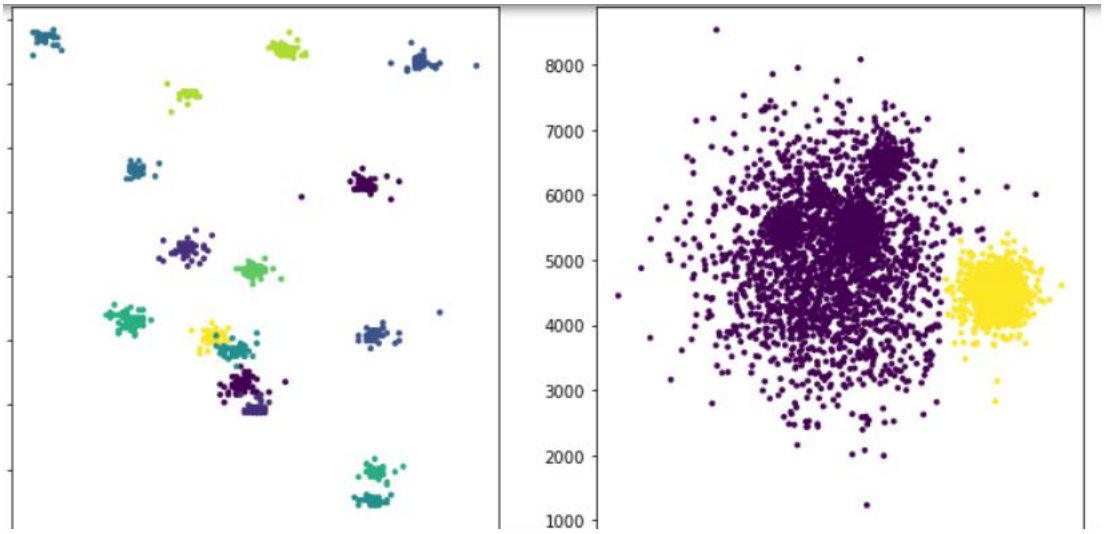
La méthode k-Means n'est pas la plus appropriée pour dissocier les clusters, surtout pour les données du jeu d64.



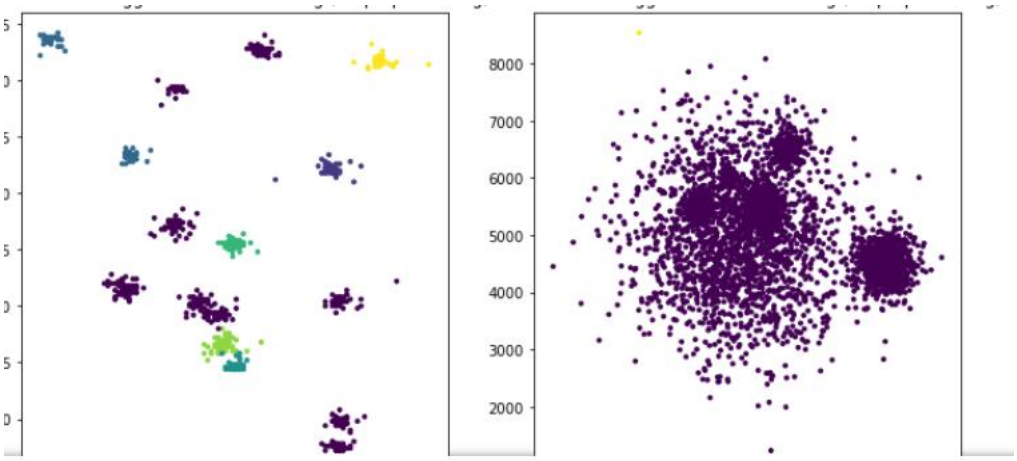
Méthode clustering agglomératif :

On peut observer que pour le jeu n2, la méthode agglomérative, utilisant le linkage ward et average permet de séparer les 2 clusters. Le linkage ward est le plus approprié pour le jeu d64, cependant, il ne permet pas d'obtenir le bon nombre de clusters.

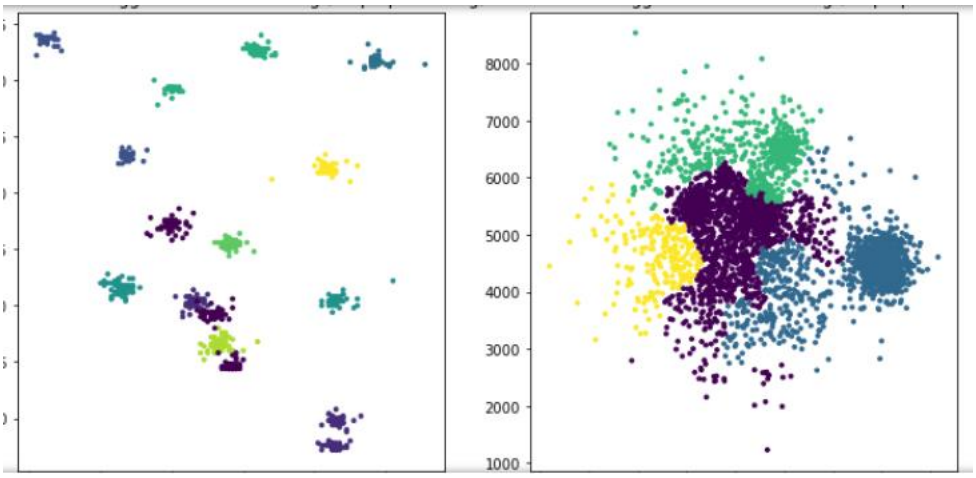
Ward



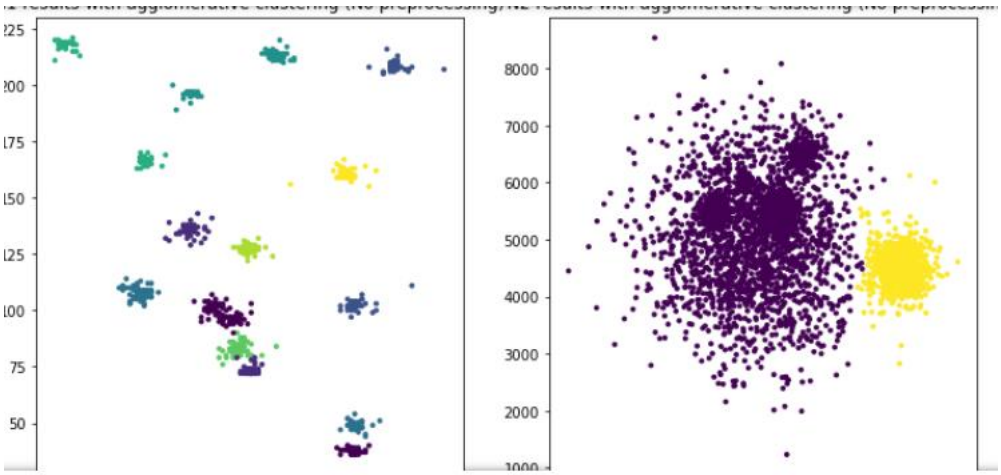
Single



Complete

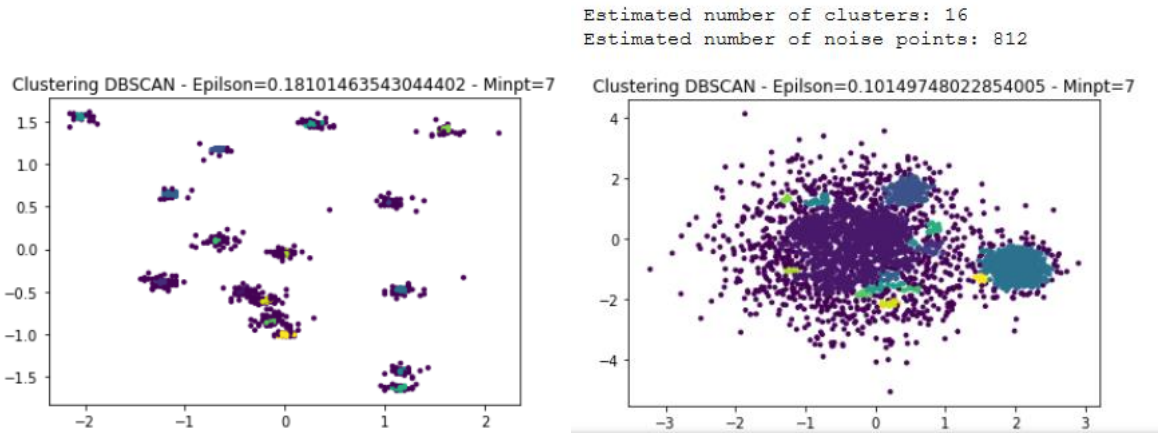


Average



Méthode DBSCAN :

La méthode DBSCAN ne semble pas approprié pour aucun des jeux de données, même si les résultats sont meilleurs pour le jeu n2 que pour le jeu d64. Cela peut s'expliquer car aucun de ces jeux n'est vraiment de forme non convexe.



CONCLUSION

Au cours de ces différents Travaux Pratiques, j'ai pu explorer différentes méthodes de clustering, appréhender leurs avantages et désavantages afin d'être en mesure de déterminer quelles méthodes seraient la plus approprié selon le contexte du jeu de données.

Ayant été seul pour réaliser ces TP, ce ne fut pas évident de terminer complétement tous les objectifs, cependant j'ai réussi à utiliser la librairie scikit-learn au travers de ce projet.

Maîtrisant déjà les parties de collecte, transport, stockage et visualisation des données, ce projet m'a donné envie d'aller plus loin afin de maîtriser la partie analyse et traitement des données, je compte ainsi exploiter les différentes méthodes vues sur d'ancien et futurs projets ainsi que tester de nouvelles méthodes qu'offre la librairie scikit-learn.