21 février 2018

EXERCICES DE CLASSIFICATION NON-SUPERVISEE

1. INTRODUCTION

Le but de l'exercice est d'implémenter les algorithmes de K-moyennes et de classification hiérarchique ascendante. Ces algorithmes seront testés sur 2 jeux de données.

2. ALGORITHME DE K-MOYENNES

Le code a été réalisé en Python en utilisant les librairies *numpy* pour les matrices et *matplotlib* pour le tracé des graphes.

A. Prérequis

On définit une fonction loader pour charger un fichier de données, et une fonction *distanceEucl* qui calcule la distance euclidienne entre 2 vecteurs de dimension quelconque.

B. Implémentation

Les étapes de la première implémentation rapide sont :

- Choix de k centre de classe au hasard dans la liste de vecteurs
- Répartition des autres points dans la classe la plus proche
- Calcul des nouveaux centres des classes
- Répétition des 2 étapes ci-dessus jusqu'à ce que le centre soit stable

Les résultats dépendent alors des premiers centres pris au hasard. On implémente les calculs d'inerties inter-groupe et intra-groupe. On implémente ensuite 2 fonctions qui réalisent n classifications et qui sélectionnent la meilleure au sens de l'inertie inter-groupe ou intra-groupe.

C. Visualisation et analyse des résultats

On implémente une fonction qui affiche les différentes classes dans des couleurs différentes et qui affiche les centres en rouge.

a) Jeu de données 1

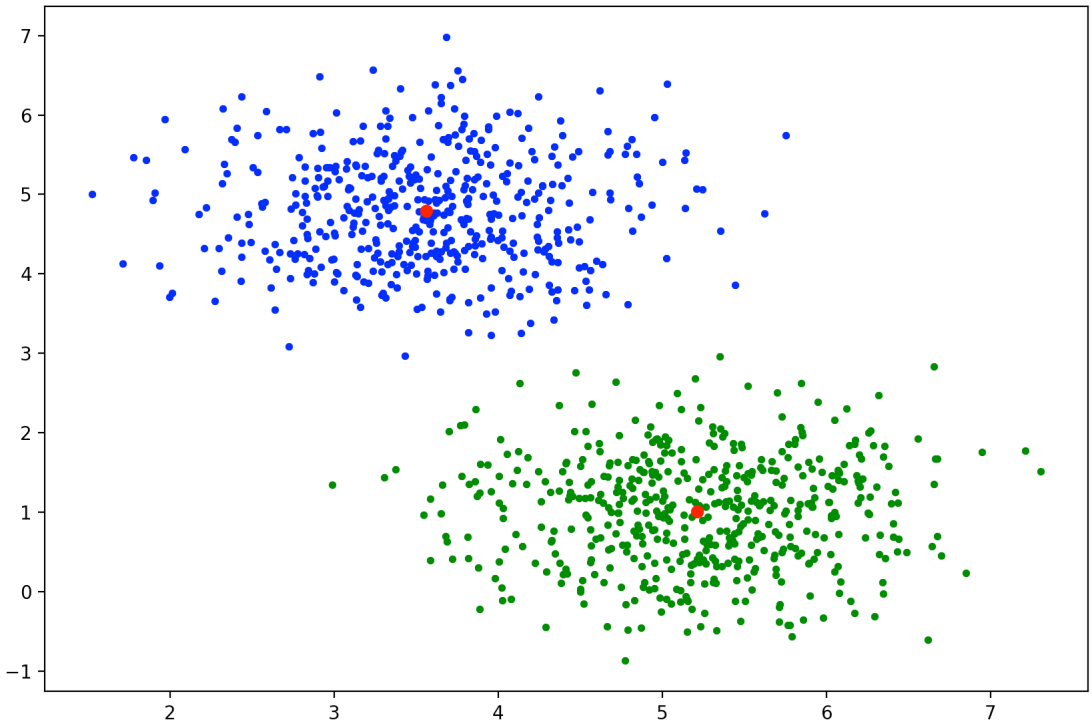

Figure 1 : Résultat d'une classification K-Means sur le jeu de données 1 avec k =2

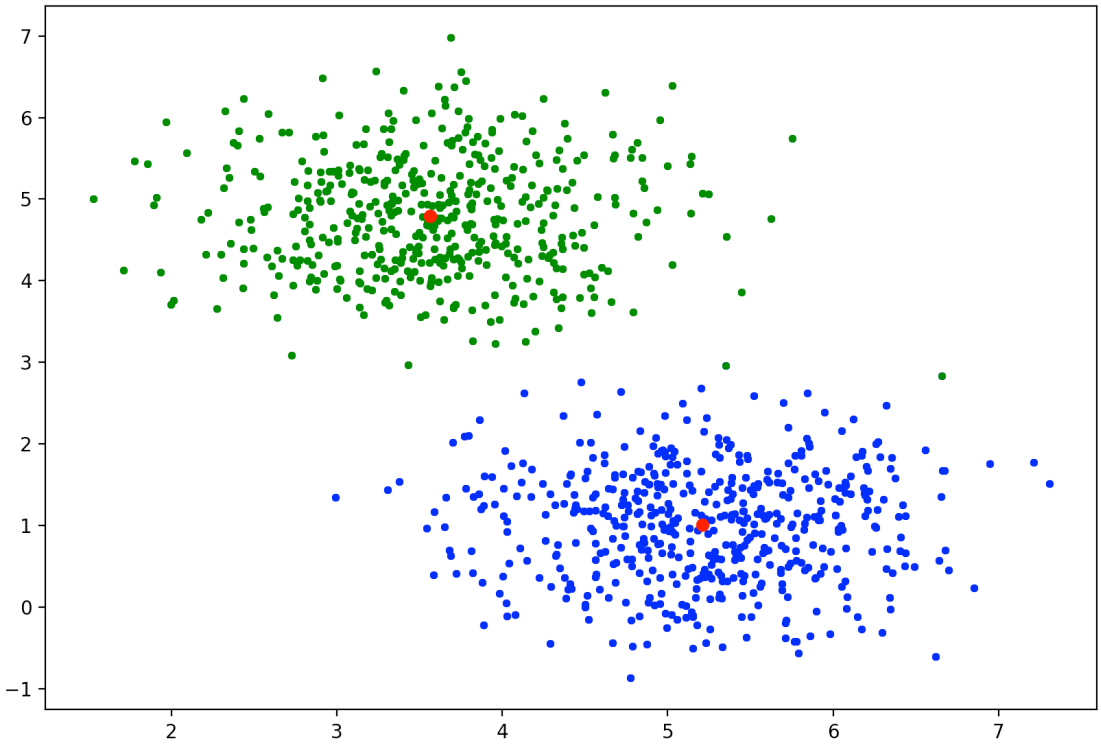

Figure 2 : Résultat d'une autre classification K-Means sur le jeu de données 1 avec k =2

Le jeu de données 1 contient 2 grandes classes distinctes : les résultats d'une classification K-Means sont très réguliers. On peut tout de même remarquer que 2 points ((5.35,2.9) et (6.8,2.85)) ne se retrouvent pas dans la même classe.

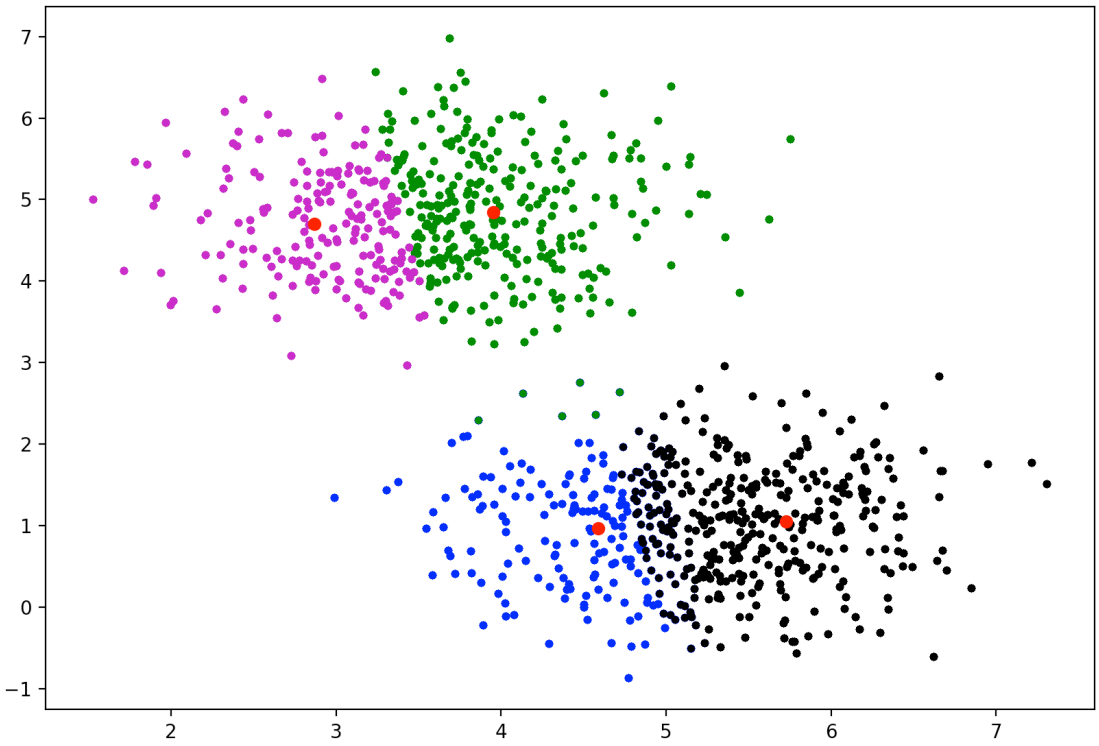

Figure 3 : Résultat d'une classification K-Means sur le jeu de données 1 avec k =4

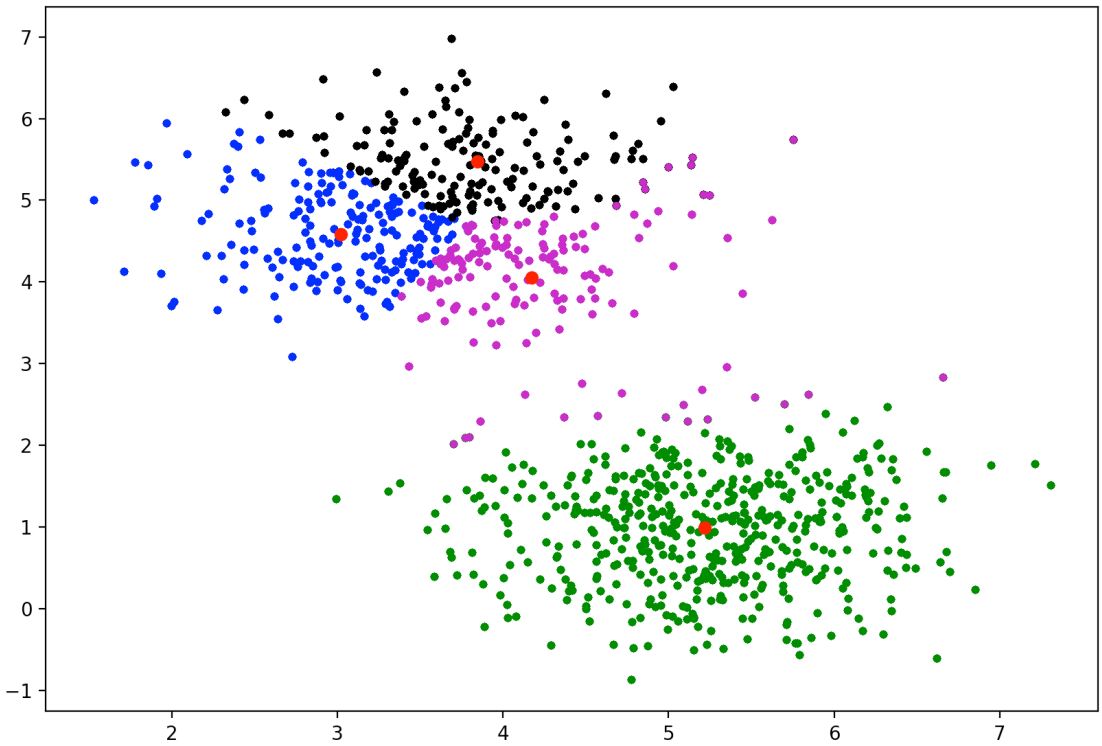

Figure 4 : Résultat d'une autre classification K-Means sur le jeu de données 1 avec k =4

On observe mieux les irrégularités avec k=4 puisqu'il n'y a pas 4 groupes distincts évidents dans le jeu de données 1.

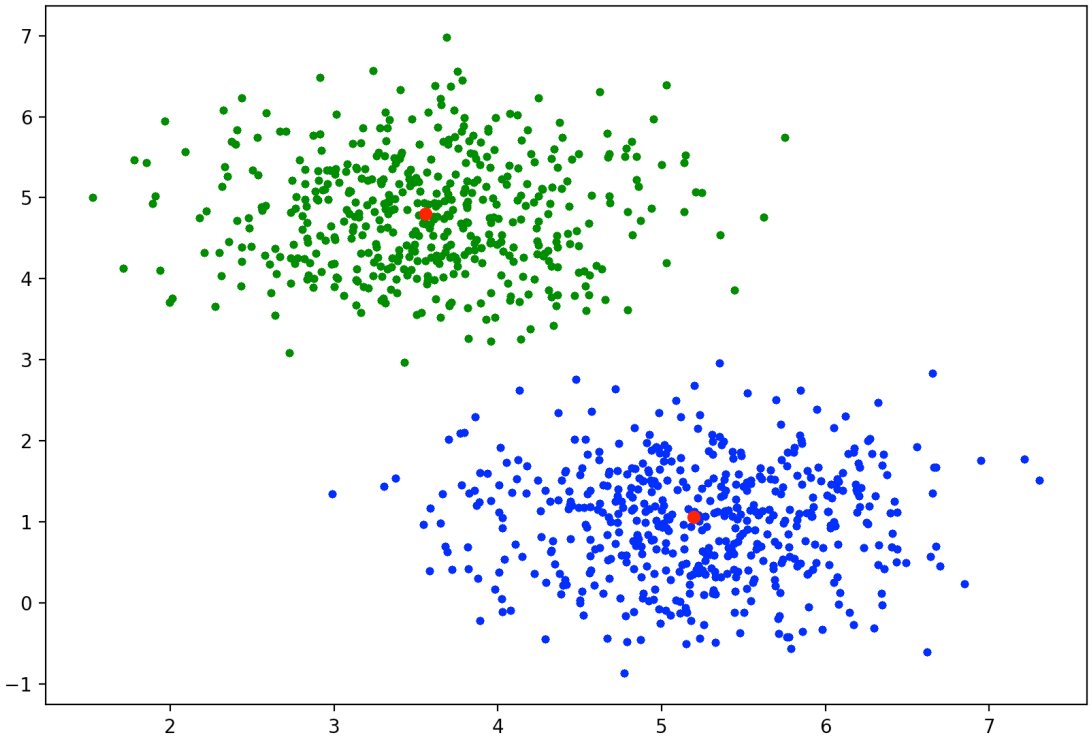

Figure 5 : Résultat d'une classification K-Means sur le jeu de données 1 avec k =2, sélectionnée parmi 8 à l'aide du critère d'inertie intra-groupes

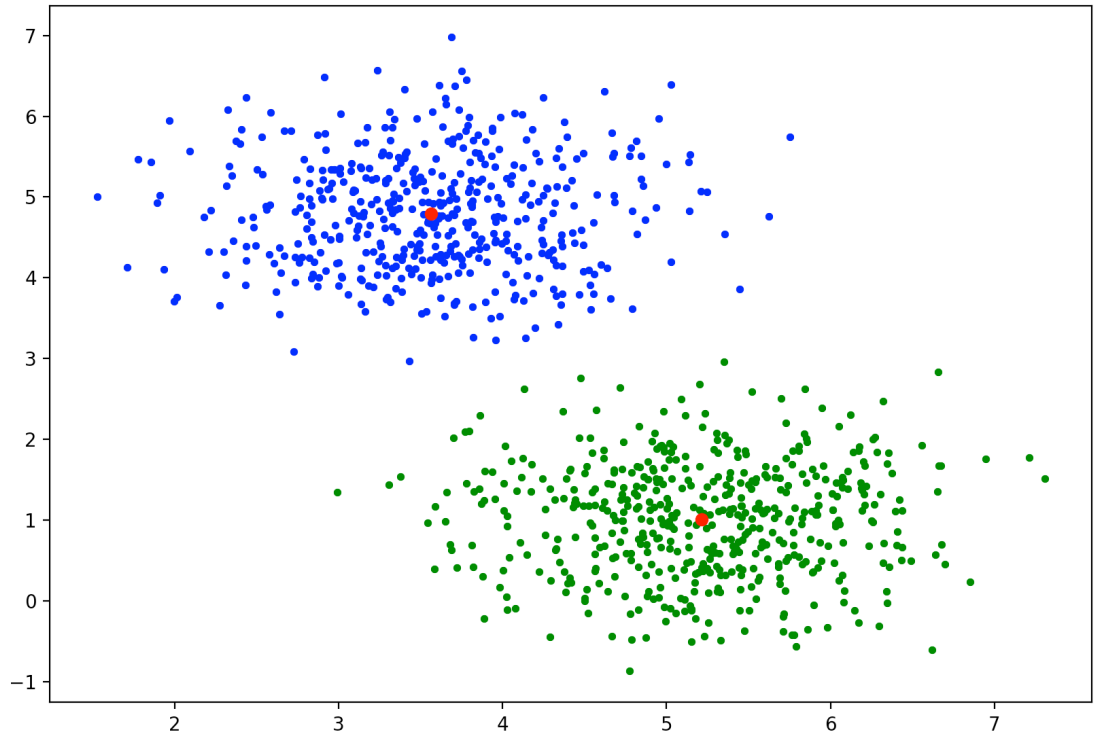

Figure 6 : Résultat d'une classification K-Means sur le jeu de données 1 avec k =2, sélectionnée parmi 8 à l'aide du critère d'inertie inter-groupes

Les sélections à l'aide de critères inter-groupes et intra-groupes donnent les mêmes résultats. L'algorithme fonctionnait déjà bien sans les critères, les groupes font sensiblement la même taille, et ont une répartition interne dan l'espace semblable. Il semble donc logique que les 2 critères donnent le même résultat.

b) Jeu de données 2

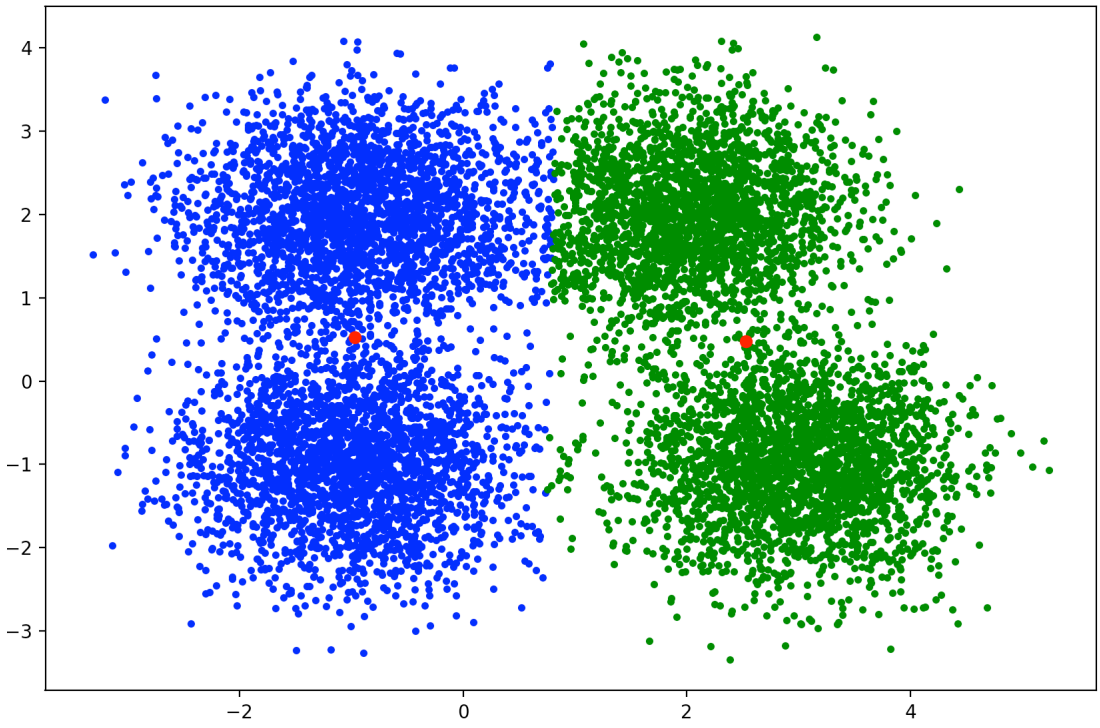

Figure 7 : Résultat d'une classification K-Means sur le jeu de données 2 avec k =2

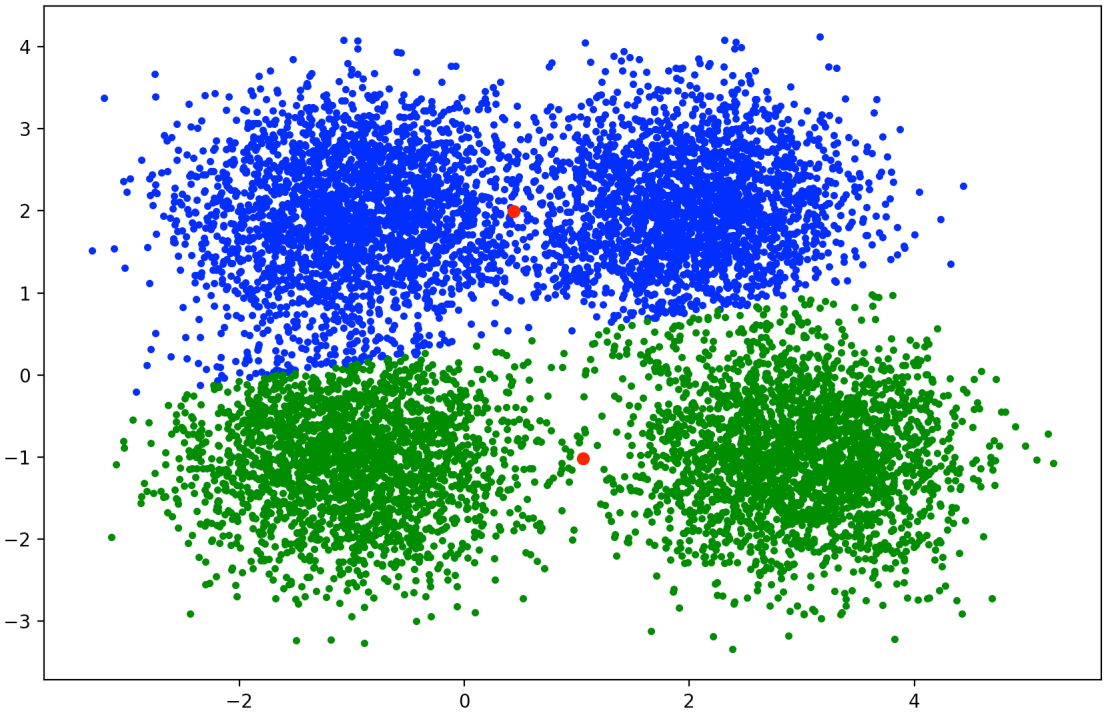

Figure 8 : Résultat d'une classification K-Means sur le jeu de données 2 avec k =2

Le jeu de données 2 contient 4 classes distinctes. Avec k=2, on remarque que l'algorithme tend à rassembler 2 classes.

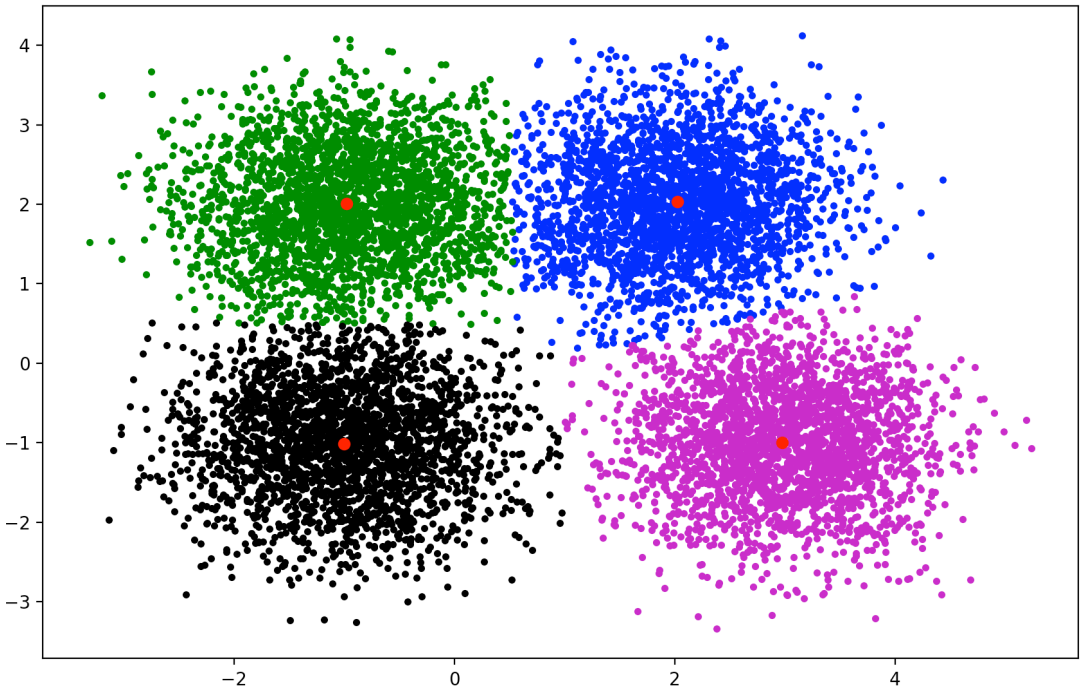

Figure 9 : Résultat d'une classification K-Means sur le jeu de données 2 avec k =4

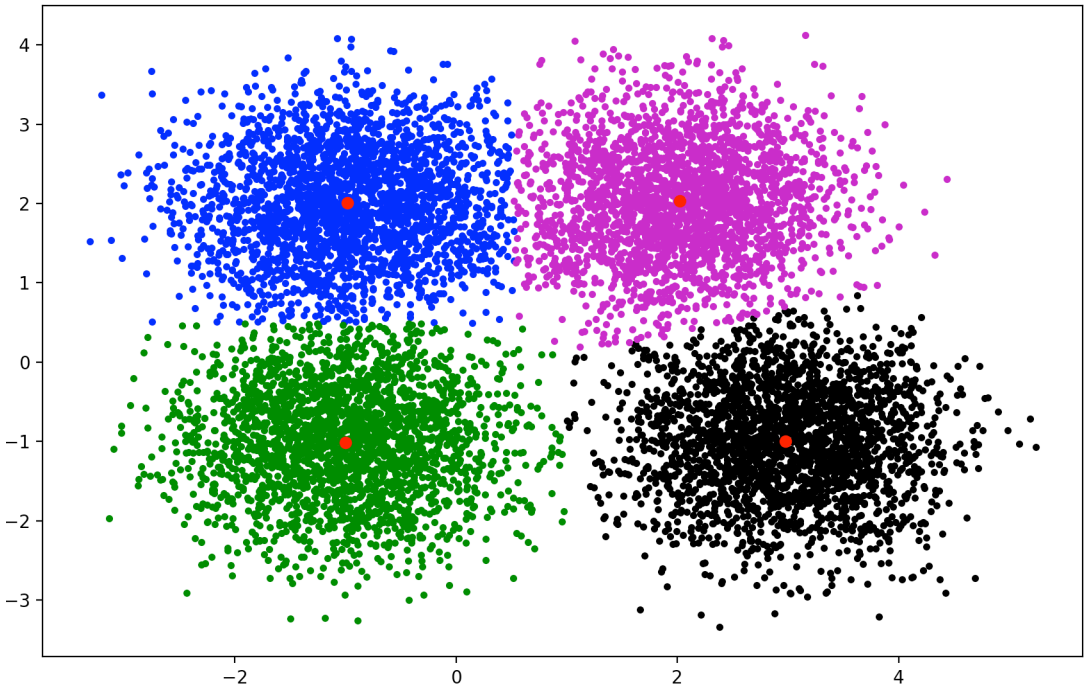

Figure 10 : Résultat d'une autre classification K-Means sur le jeu de données 2 avec k =4

On obtient de bons résultats intéressants avec k=4. Il y a peu de variation entre les itérations.

3. ALGORITHME DE CLASSIFICATION HIERARCHIQUE ASCENDANTE

A. Prérequis

On peut utiliser différents critères d'agrégation dans l'algorithme de classification hiérarchique. On implémente donc les fonctions de critère de saut minimal, saut maximal, et distance moyenne.

B. Implémentation

Une première implémentation naïve, mais nécessaire pour utiliser les critères de saut minimal et saut maximal, consiste à calculer la matrice de dissimilarité à chaque itération de l'algorithme et à parcourir la liste des éléments d'une classe à chaque calcul de critères d'agrégation. La taille de la matrice de dissimilarité diminue au cours de l'algorithme car on fusionne des classes. Néanmoins, cette approche naïve reporte la complexité dans le calcul du critère d'agrégation. Le calcul du critères d'agrégation a alors une complexité de l*m avec l et m les tailles des 2 classes. On peut majorer très grossièrement cette complexité par $((n-k+2)/2)^2$. On retient surtout que le temps de calcul du critère d'agrégation n'est pas constant. La complexité de l'algorithme global est alors en $O(0.5n^4)$.

On implémente ensuite une fonction rapide qui utilise la méthode de Ward : on retient les barycentres des classes et on travaille avec ces barycentres. Le temps de calcul du critères d'agrégation devient donc constant. Le temps de fusion de deux classes est égal à sa longueur, et le nombre de fusion peut être majoré par n-k. La complexité des fusions peut donc être majoré par n^2. La complexité de l'algorithme global est donc en $O(n^3)$. Noter que cette algorithme ne peut pas être utilisé avec les critères de saut minimal et de saut maximal, qui nécessitent de conserver tous les points.

C. Mesure des performances

Une complexité en n^3 étant tout de même importante, et Python n'étant pas très performant en tant que langage interprété, on mesure les temps d'exécution :

Algorithme	Temps d'exécution
Naïf Distance Moyenne	63 minutes
Méthode de Ward (distance Moyenne)	22 minutes
Ward optimisé Numba	7 minutes

Il y a bien un gain important de l'algorithme naïf à la méthode de Ward. Numba, une librairie d'optimisation qui compile à la volée le langage Python en C, a été utilisée pour gagner en temps d'exécution. Noter que cette librairie a été appliquée à tous les autres algorithmes ensuite.

D. Analyse des résultats

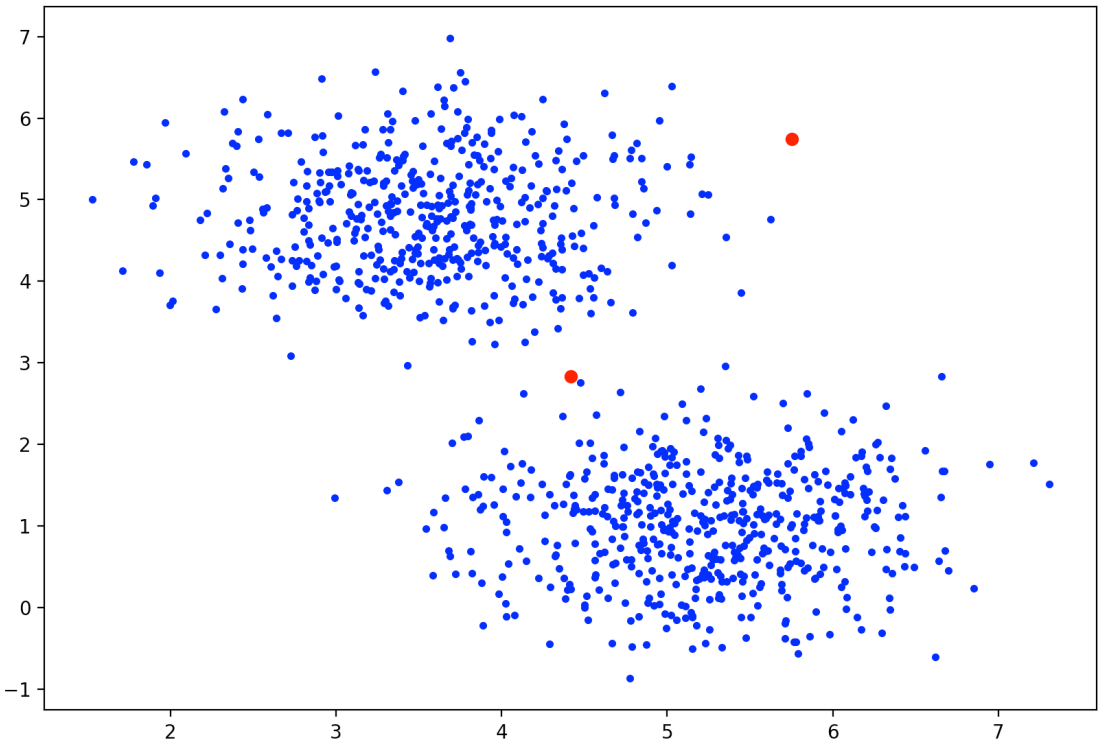

Figure 11 : Résultat d'une CHA sur le jeu de données 2 avec le critère de saut minimal

Un point très isolé est loin de tous les points, mêmes ceux de sa classe apparente. Le saut d'une classe à l'autre était donc plus petit que le saut du point à sa classe. Par le critère du saut minimal, les 2 classes ont donc été fusionnées et ce point est resté isolé.

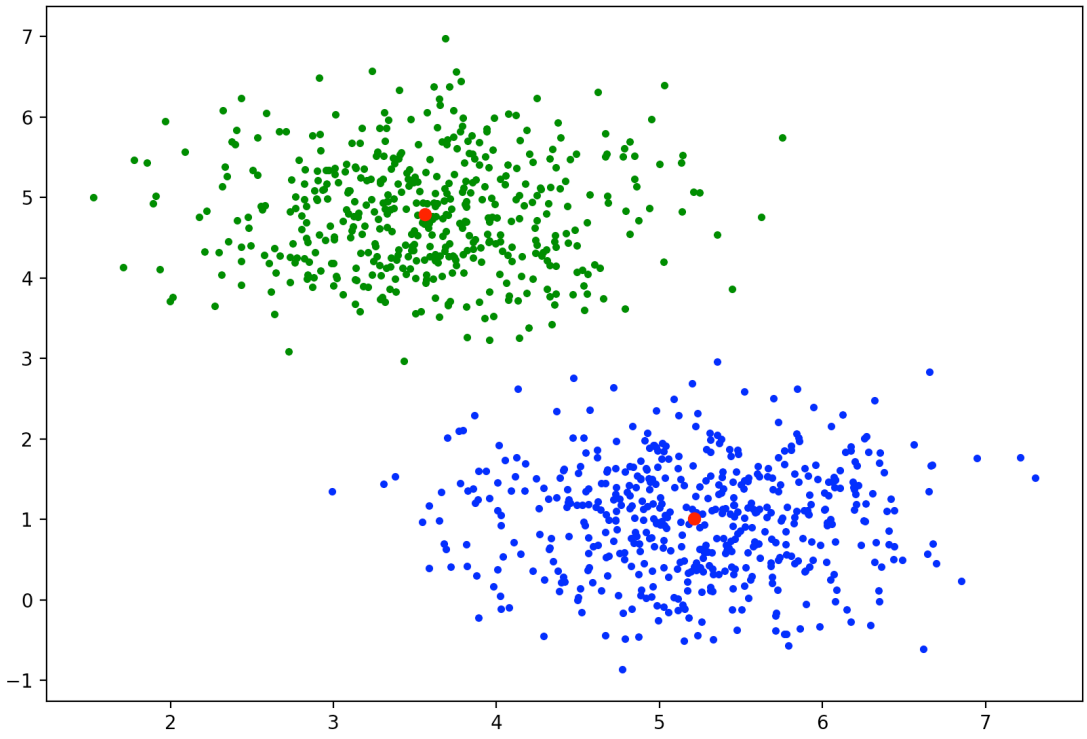

Figure 12 : Résultat d'une CHA sur le jeu de données 2 avec le critère de saut maximal

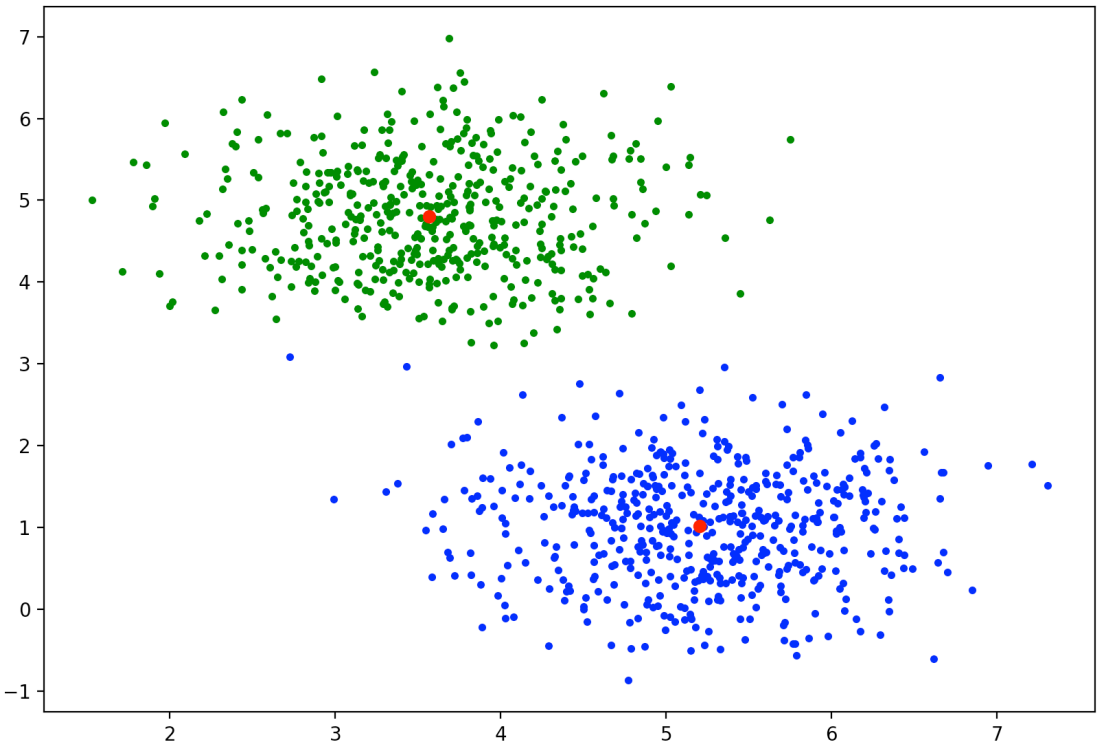

Figure 13 : Résultat d'une CHA sur le jeu de données 2 avec le critère de distance moyenne

Les critères du saut minimal et de la distance moyenne donnent des résultats très similaires.

Noter que l'algorithme de classification hiérarchique ascendante donne toujours le même résultat à chaque exécution par définition des distances.

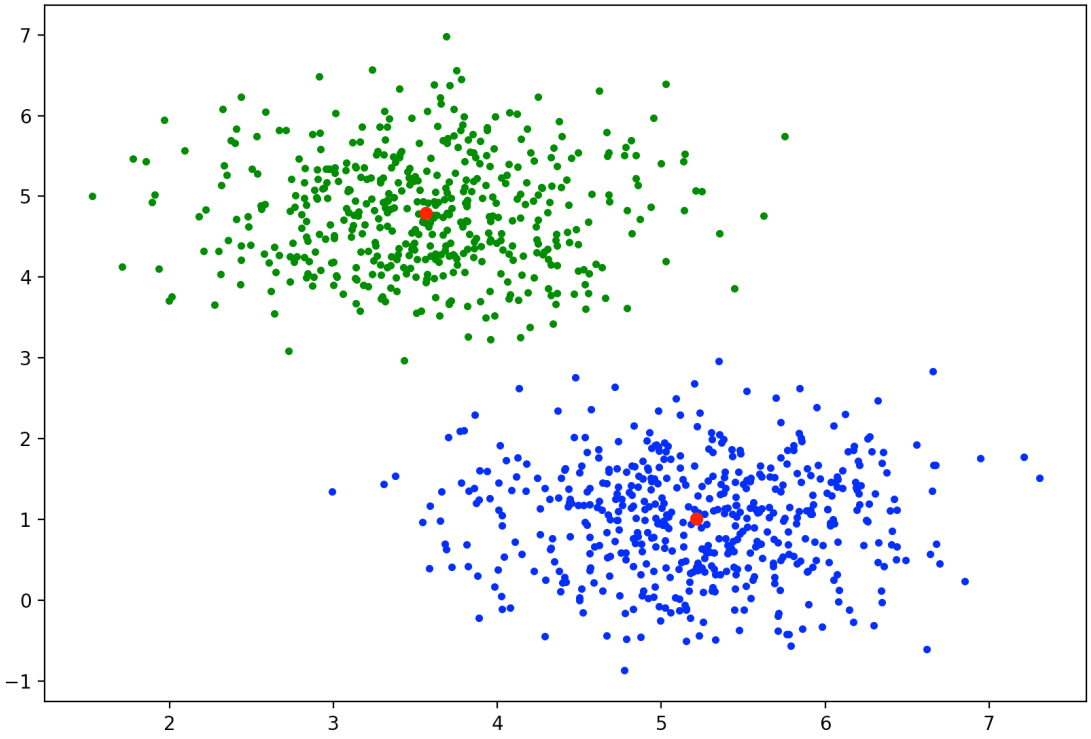

Figure 14 : Résultat d'une CHA sur le jeu de données 2 avec la méthode de Ward

La méthode de Ward donne un résultat très légèrement différent de la méthode naïve qui utilise les distances moyennes. Je pense que cela est dû à des arrondis réalisés lors des calculs des barycentres dans la méthode de Ward.

Les algorithmes n'ont pas été testés sur le jeu de données 2, les temps d'exécution étaient trop longs.

4. CONCLUSION

L'algorithme K-means est plutôt rapide, mais ses résultats dépendent des points de départ tirés au sort. De plus, l'algorithme n'aide pas trop à choisir le nombre de classes optimal. L'algorithme de classification hiérarchique est lui beaucoup plus lent mais a toujours le même résultat, seul le critère d'agrégation est à bien choisir. De plus, il peut aider à choisir k en traçant un dendrogramme.