Rapport Projet - Intelligence Artificielle et Apprentissage

Gabriel Lucchini et Mohand Oukhemanou

31 Mai 2023

Clustering de Données Réelles, Comparaison, Évaluation et Explications

Introduction

Dans le cadre de l'EC Intelligence Artificielle et Apprentissage, nous sommes amenés à implémenter différents algorithmes de clustering et à comparer les résultats obtenus afin de répondre à une hypothèse.

Nous avons donc choisit une base de données comportant des pingouins appartenant à différentes espèces et vivant sur différentes îles. On possède des données sur la longueur et la profondeur de leur bec, leur masse corporelle et la longueur de leur nageoire.
Le but de notre étude est de déterminer si l'espèce ou l'habitat a une influence sur les caractéristiques physiques de l'animal.
Pour cela nous avons décidé d'implémenter l'algorithme K-Means, l'algorithme GMM (ou Gaussian Mixture Model), l'algorithme Mean-Shift, l'algorithme DBSCAN et l'algorithme CAH.
Dans ce but, nous utilisons les bibliothèque "sklearn" et "scipy", et pour l'affichage des résultats nous utilisons les bibliothèques "matplotlib" et "seaborn".

Dans la suite du rapport, nous présenterons puis analyserons les résultats obtenus au travers des différents algorithmes afin d'en tirer des conclusions quant à nos hypothèses.

Algorithmes

Algorithme K-Means

Principe

L'algorithme K-Means (ou K-Moyenne) est un algorithme de clustering de données, utilisé en apprentissage non supervisé.
Il permet de séparer notre jeu de données en groupes homogènes et repose sur plusieurs étapes.
La première étape consiste à initialiser les centres de gravité qui serviront de premières références pour le partitionnement. Ils sont choisis aléatoirement parmi les points du jeu de données.
La seconde étape consiste à affecter les points de nos données au groupe qui leur correspond, pour cela on calcule leur distance aux différents centres initialisés précédemment et on les affecte au groupe du centre dont ils sont le plus proche.

La troisième étape consiste au calcul de nouveau centre pour chaque groupe, pour cela on fait la moyenne des points dans chaque groupe et on réaffecte les points au groupe du nouveau centre dont ils sont le plus proche.
Enfin, pour obtenir un bon résultat, la dernière étape consiste en la répétition des deuxième et troisième étapes jusqu'à ce que les points ne changent plus de groupe.

Résultats

Nous avons, dans un premier temps, déterminé le nombre de clusters idéal via la méthode du coude. Les résultats ont montré que le nombre de cluster idéal est aux alentours de 3.
Puis dans un second temps nous avons regardé la qualité du clustering pour k allant de 2 à 10, on a pu voir alors que pour 3 clusters, le silhouette score est de 0.45 (soit le deuxième plus qualitatif).
Ainsi, on a déterminé que le résultat le plus intéressant sera avec 3 clusters.

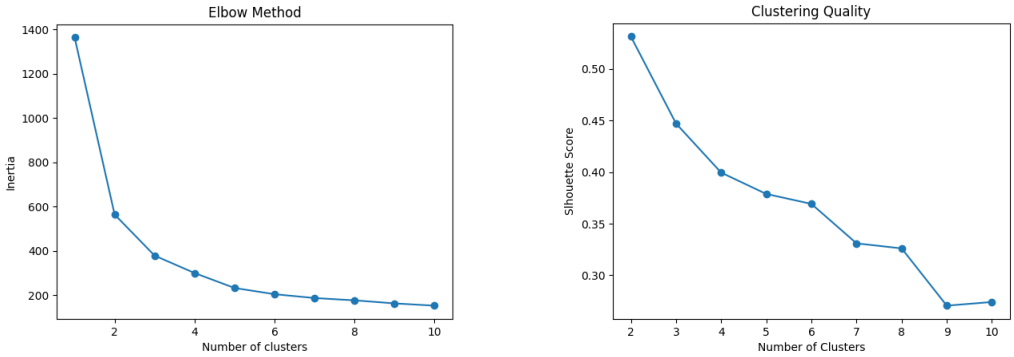

FIGURE 1 – Nombre de Clusters Idéal & Qualité des Clusters.

Ainsi, avec l'algorithme K-Means pour 3 clusters on obtient le partitionnement suivant :

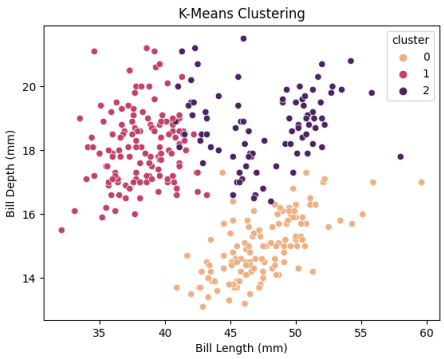

FIGURE 2 – Clustering des Données avec K-Means.

Algorithme GMM

Principe

L'algorithme GMM, pour Gaussian Mixture Model (ou Modèle de Mélange Gaussien) est un algorithme de clustering de données, utilisé en apprentissage non supervisé et basé sur un modèle statistique s'exprimant selon une fonction de densité. Il permet de regrouper les données en clusters homogènes sur la base de distributions gaussiennes.
Les modèles supposent qu'il y a un certain nombre de distributions gaussiennes, et que chacune d'entre elles représente un cluster. On regroupe alors les données qui appartiennent à la même distribution.
Pour cela, l'algorithme calcule une moyenne et une variance pour chaque cluster, puis il calcule la probabilité que la donnée appartient ou non au cluster.

Résultats

L'étude du silhouette score montre un résultat de 0.45 pour 3 clusters (soit le deuxième plus qualitatif), ainsi le résultat le plus intéressant sera donc aussi avec 3 clusters.

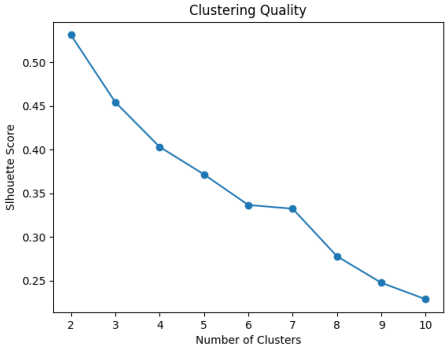

FIGURE 3 – Qualité des Clusters.

Ainsi, avec l'algorithme GMM pour 3 clusters on obtient le partitionnement suivant :

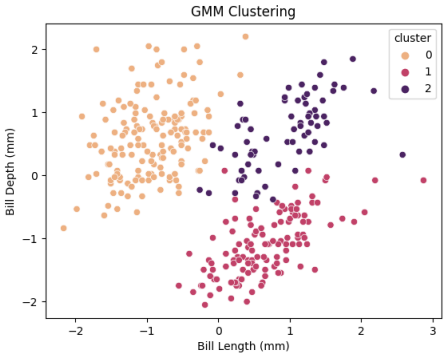

FIGURE 4 – Clustering des Données avec GMM.

Algorithme Mean-Shift

Principe

L'algorithme Mean-Shift est un algorithme de clustering de données, utilisé en apprentissage non supervisé. Il permet d'identifier des clusters dans un ensemble de données sur la base de la densité. Il est très utile pour des données dont les clusters ne sont pas bien délimités.
Pour cela, il va décaler chaque point de nos données vers la densité la plus élevée de la distribution de nos données dans un certain rayon.
L'algorithme effectue itérativement ces décalages jusqu'à ce que les données convergent vers un maximum local de la fonction de densité. Ces maximums locaux représentant nos clusters.
L'avantage est que contrairement à d'autres algorithmes, il n'est pas nécessaire de choisir le nombre de clusters au préalable, celui-ci est déterminé par l'algorithme par rapport aux données.
Ainsi, dans une première étape on initialise les points de données en tant que centre de gravité de cluster.
Puis jusqu'à ce que la convergence ou un nombre maximum d'itérations soit atteint, on calcule la moyenne de tous les points dans un rayon centré sur le point de la donnée, ensuite on déplace ce point vers la moyenne.
On prend alors comme centre de gravité du cluster, les points qui n'ont pas bougé après la convergence.

Résultats

L'algorithme ayant déterminé que notre clustering devait se faire avec 3 clusters, confirme nos observations précédentes. De plus, on obtient un silhouette score de 0.5 (se rapprochant des 0.45 vus précédemment). Ainsi, avec l'algorithme Mean-Shift pour 3 clusters on obtient le partitionnement suivant :

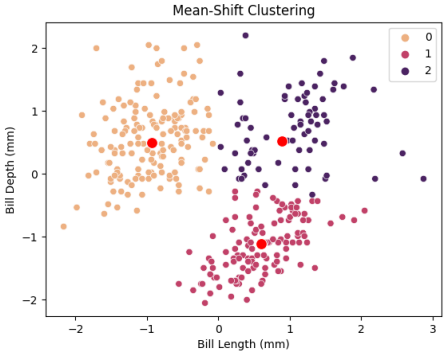

FIGURE 5 – Clustering des Données avec Mean-Shift.

Algorithme DBSCAN

Principe

L'algorithme DBSCAN, pour Density-Based Spatial Clustering of Applications with Noise est un algorithme de clustering de données, basé sur la densité estimée des clusters pour diviser les données en k groupes homogènes et compacts. L'algorithme fonctionne en utilisant l'estimation de la densité locale.
Pour cela, on va déterminer, pour chaque donnée, le nombre de points à distance epsilon de celle-ci, que l'on appelle epsilon-voisinage. Puis si une donnée compte de nombreux voisins, alors elle est considérée comme données à haute densité.
Ainsi toutes celles au voisinage d'une donnée à haute densité appartiennent au même cluster. Et toutes celles qui n'ont pas de voisins ou qui ne sont pas une donnée à haute densité, sont considérées comme une anomalie (ou bruit).
L'algorithme DBSCAN fonctionne avec deux paramètres.
La valeur epsilon, qui est la distance qui spécifie un voisinage, deux points sont considérés comme voisins si la distance qui les sépare est inférieure ou égale à epsilon.
La valeur minPts : le nombre de points minimum définissant un cluster.

Résultats

Comme avec Mean-Shift, le nombre de cluster est choisit automatiquement. Mais l'on a une étape en plus de recherche des plus proches voisins qui consiste à trouver pour un point, dans un ensemble d'autres points, quel sont les k plus proches. On obtient alors la courbe suivante :

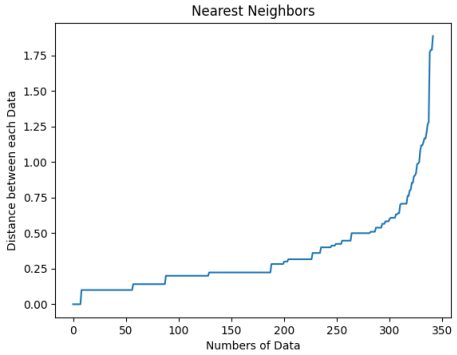

FIGURE 6 – Les K Plus Proche Voisins.

Ainsi, avec l'algorithme DBSCAN pour 3 clusters on obtient le partitionnement suivant :

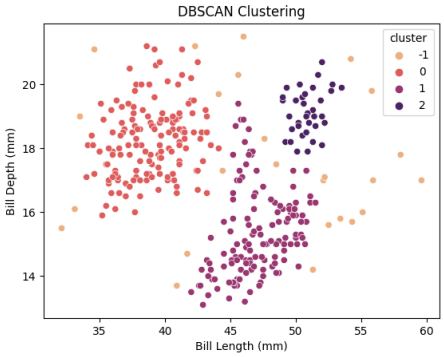

FIGURE 7 – Clustering des Données avec DBSCAN.

Algorithme CAH

Principe

L'algorithme CAH pour Classification Ascendante Hiérarchique est un algorithme de clustering de données, utilisé notamment en classification automatique.
On part de la situation initiale où chaque donnée forme une classe (on a alors n classes), on va ensuite chercher à réduire ce nombre et former des clusters de plus en plus grands. Pour cela, à chaque étape, on fusionne deux classes qui sont les plus proches, c'est-à-dire celles dont la dissimilarité entre elles est minimale. Cette valeur de dissimilarité, aussi appelée indice d'agrégation, aura une valeur faible pour la première itération mais elle va croître à chaque nouvelle itération.

Résultats

Avec l'algorithme CAH, on obtient les dendrogrammes suivants :

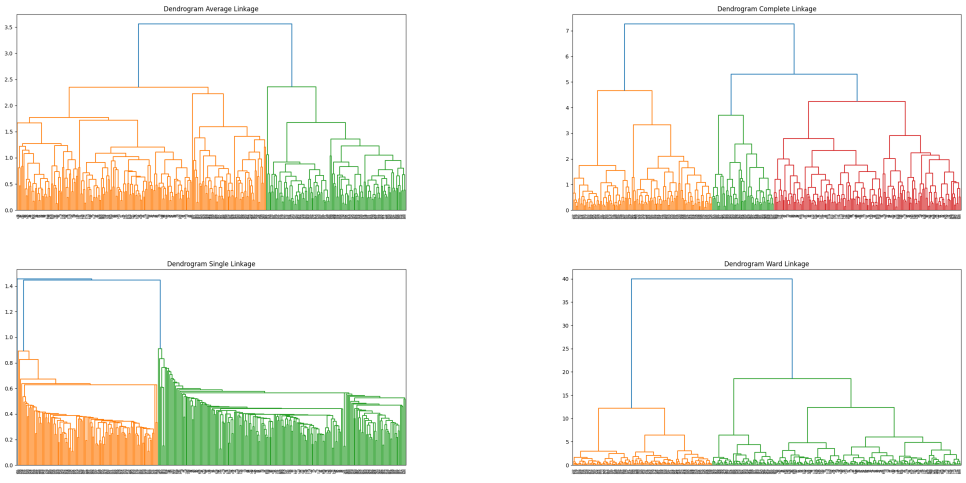

FIGURE 8 – Dendrogramme pour les méthodes "Average", "Complete", "Single" et "Ward".

Ainsi, avec l'algorithme CAH pour 3 clusters et avec la méthode de linkage "Ward", on obtient le partitionnement suivant :

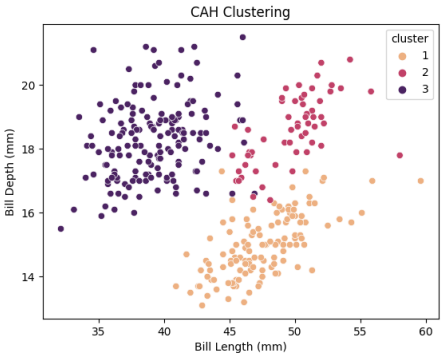

FIGURE 9 – Clustering des Données avec CAH.

Analyse des Résultats

Si l'on observe la dispersion des données en fonction des espèces et des îles, on obtient les résultats suivants :

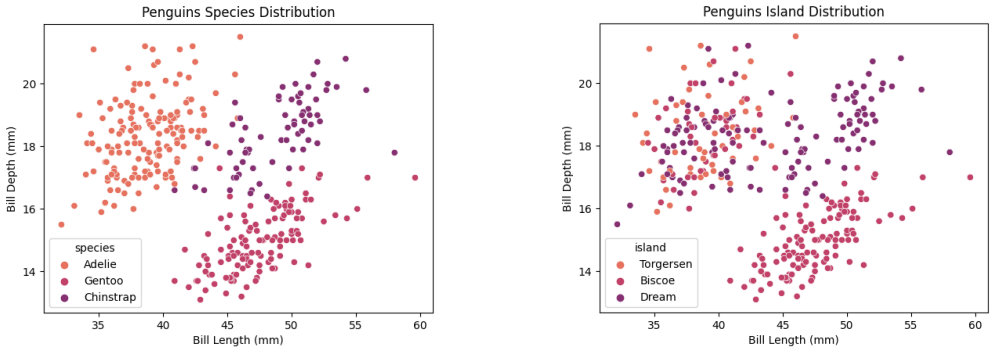

FIGURE 10 – Dispersion des données en fonction des espèces & des îles.

En comparant nos résultats précédents avec ces dispersions, on peut voir que nos clustering se rapprochent grandement de la dispersion en fonction des espèces (avec une petite exception pour le DBSCAN).
En effet, les clusters contiennent principalement (sauf quelques bruits) des pingouins appartenant à la même espèce (cela se vérifie également dans nos fichiers labels).

Conclusion

En conclusion, au travers de notre projet et de notre rapport, nous avons pu voir différents types de clustering (que ce soit ceux basés sur les centroïdes, ceux sur la densité, ceux sur la distribution ou encore les hiérarchiques), avec pour chacun des résultats de partitionnement similaires.
Ainsi quant à notre hypothèse de départ qui était "L'espèce influe sur les caractéristiques physiques des pingouins.", nous pouvons la confirmer, puisque nos résultats de clustering sont très similaires à la dispersion en fonction de l'espèce.
À contrario, nous pouvons infirmer la seconde hypothèse qui était "L'habitat influe sur les caractéristiques physiques des pingouins.", puisque qu'aucun de nos résultats nous permet d'aboutir à une telle conclusion.
Aussi, pour aller plus loin, nous pourrions essayer de voir s'il y a des différences entre les pingouins mâles et femelles appartenant à la même espèce.