

# Analyse de données pour guider les choix d'affectation de l'aide humanitaire internationale

Bahrman Louis - Bizeul Solal

## **I. Description du projet**

Ce projet en analyse de données multivariées a pour objectif d'identifier les pays qui ont le plus besoin d'aide humanitaire.

Pour cela, nous allons effectuer une classification non supervisée de 167 pays à partir de données socio-économiques et de santé. En identifiant les pays se trouvant dans une crise exceptionnelle, on pourra ainsi distribuer l'aide de la manière la plus juste.

Dans un premier temps, nous allons commencer par faire une analyse comparative de quatre méthodes de clustering: le K-Means, le Clustering Hiérarchique Ascendant (CHA), le modèle de mélange gaussien et le DBSCAN. Nous ferons cette analyse en testant ces algorithmes sur des données simulées.

Dans un second temps, nous utiliserons nos connaissances fraîchement acquises sur le jeu de données des pays. Nous ferons un examen des données, puis une préparation de celles-ci pour le clustering. Nous effectuerons ensuite le clustering, en testant différents paramètres pour chaque méthode. Nous en tirerons la liste des pays dans lesquels il est nécessaire d'intervenir.

Finalement, nous répéterons ce processus sur le même jeu de données, en lui faisant subir préalablement une ACP afin de réduire la dimension des données. Nous pourrons ensuite comparer ces deux listes afin d'en tirer des conclusions sur l'utilité de l'ACP dans ces conditions, ainsi que trouver la liste définitive des pays ayant le plus besoin d'aide.

## **II. Etude préalable : Comparaison des méthodes de clustering sur des données simulées**

## 1. Etude de la méthode DBSCAN

La méthode DBSCAN est particulièrement utile sur des clusters distincts, et denses, où certains points sont considérés comme du bruit. Elle s'exécute de la manière suivante. Étant donnés  $\epsilon$  et  $n$ , on parcourt les données. Pour chaque point, on observe son voisinage dans un rayon  $\epsilon$ . Si on trouve plus de  $n$  voisins à ce point, ce point est considéré comme un point central. On applique donc le même processus à ses voisins, ce qui augmente la taille du cluster de proche en proche. Les points ayant moins de  $n$  voisins dans un rayon  $\epsilon$ , sont considérés comme du bruit et n'appartiennent donc à aucun cluster.

## 2. Etude des classes et modules relatifs aux différentes méthodes

Avant d'expérimenter avec les quatre classes de clustering, il semble utile d'étudier de plus près les paramètres d'appel, attributs, fonctions et méthodes de chacune afin d'améliorer notre efficacité lors de leur utilisation.

### Méthode K-Means

La classe K-Means provient de `sklearn.cluster.KMeans`. Les paramètres les plus importants utilisés lors de l'initialisation de cette classe sont `n_clusters`, `init`, `n_init` et `algorithm`. `N_clusters` indique le nombre de clusters qui seront formés. Il faudra ajuster ce paramètre en fonction de nos besoins. `init` précise la méthode d'initialisation des centres des clusters. `N_init` indique le nombre de fois que l'algorithme sera exécuté avec des centres de clusters différents. `Algorithm` représente l'algorithme des K-Means à utiliser (classique ou Elkan). Les méthodes que nous utiliserons sont `fit` et `predict`. `Fit` calcule le clustering et `predict` prédit le cluster auquel appartient chaque point du tableau.

### Méthode CHA

La méthode du clustering ascendant est située dans le module `sklearn.cluster.hierarchy`. Pour effectuer ce clustering, il faut d'abord effectuer le calcul de la matrice des distances et en déduire le dendrogramme à l'aide de la fonction `linkage`. Cette fonction admet comme unique paramètre obligatoire les données, ou la matrice des distances triangulaire supérieure. On peut ensuite y spécifier la méthode (`method`) permettant de calculer la distance intercluster, ainsi que la distance à utiliser. Nous utiliserons ici la distance euclidienne, compatible avec toutes les méthodes. Afin de faciliter la lecture du dendrogramme, il est possible d'ordonner les feuilles de l'arbre, au détriment d'un temps de calcul plus important. Il faut ensuite effectuer l'affectation des points à leur cluster en appliquant la fonction `scipy.cluster.hierarchy.fcluster(z,t,criterion='distance')` où `z` est la matrice obtenue après le `linkage`. `T` représente l'inertie interclasse.

### Méthode Gaussienne

La classe de mélange gaussien provient de *sklearn.mixture.GaussianMixture*. Les paramètres les plus importants sont *n\_components* qui correspond au nombre de clusters voulu et *covariance\_type* qui régit le type de covariance à utiliser. Le type de covariance régit la direction et la longueur des axes des contours de densité de la distribution gaussienne. 'Full' laisse les composantes s'adapter et prendre n'importe quelle forme. 'Tied' les oblige à prendre la même forme, mais cette forme peut être n'importe laquelle. 'Diag' signifie que les axes des contours doivent être orientés selon les axes cartésiens. 'Spherical' est semblable à 'Diag', mais avec des contours sphériques. Le paramètre *n\_init*, et les méthodes *fit* et *predict* sont les mêmes que pour le K-Means.

#### Méthode DBSCAN

La fonction *dbscan*, du module *scipy.cluster.hierarchy* admet comme paramètres *t* représentant le rayon du voisinage, ainsi que *n*, le nombre de voisins nécessaires pour considérer un point comme étant central. Il est aussi possible de sélectionner la distance utilisée (par défaut une distance de minkowski avec  $p=2$ , c'est-à-dire une distance euclidienne).

### 3. Expérimentations

Le fichier Aggregation a 7 clusters d'après la référence, certains sont clairement séparés et d'autres sont reliés entre eux. Le fichier g2-2-20 a 2 clusters clairement séparés. Le fichier g2-20-100 a 1 seul gros cluster: il peut être intéressant de remarquer le nombre de clusters assigné à ces données par le CHA et le DBSCAN. Le fichier jain a 2 clusters de forme quadratique. Le fichier pathbased a 3 clusters: 2 de forme classique et 1 qui entoure les deux autres.

#### Méthode K-Means

Nous utiliserons les paramètres suivants. *n\_clusters* sera égal au nombre réel de clusters. Pour *init*, nous utiliserons le paramètre par défaut "k-means++" puisqu'il sélectionne ces centres de façon à optimiser la vitesse de convergence. Nous prendrons *n\_init* = 50 car cela permet aux résultats de converger sans que l'exécution ne prenne trop de temps pour la taille de données avec laquelle on travaille. D'après l'article de recherche "Using the Triangle Inequality to Accelerate k-Means", l'algorithme Elkan accélère beaucoup le calcul des clusters du K-Means, surtout pour des données à beaucoup de dimensions et avec un nombre de clusters important. Ce n'est pas notre cas dans cette étude, mais nous utiliserons tout de même Elkan puisqu'il n'a pas d'inconvénients.

Avec ces paramètres, nous obtenons les résultats suivants.

Aggregation : Indice de Rand = 0.838 (ici, l'algorithme n'isole pas les deux petits clusters à gauche. Par contre, pour  $k=6$ ,  $\text{rand} = 0.989$  car il isole ces deux petits clusters ensemble)

G-2-20 : clustering visuellement parfait, Coefficient de Silhouette = 0.75

G-2-100 : difficile d'estimer puisqu'il n'y a qu'un seul cluster

Jain : Indice de Rand = 1

Pathbased : Indice de Rand = 0.632 (ici, l'algorithme n'isole pas le cluster entourant)

### Méthode CHA

La méthode cha nécessite de tracer le dendrogramme avant de pouvoir calculer le paramètre  $t$ . Nous utiliserons pour cela la méthode de Ward par défaut.

Avec ces paramètres, nous obtenons les résultats suivants :

Aggregation : La taille et la densité des clusters étant variable, la méthode Average permet d'obtenir de meilleurs résultats. En choisissant alors  $t=0.9$  on obtient un indice de Rand de 0.9

G-2-20 : Optimal pour  $t=30$  donnant un clustering visuellement parfait et un coefficient de silhouette égal à 0.75

G-2-100 : En choisissant une grande valeur pour  $t$ , quelle que soit la méthode utilisée, on obtient un unique cluster.

Jain : Les clusters étant bien séparés mais pas de la forme d'une boule, on ne peut pas utiliser une méthode basée sur les centres des clusters. La méthode 'single' correspondant à la distance minimale intercluster est alors optimale. On choisit  $t=0.316$ , ce qui laisse un cluster contenant un unique point, pouvant être interprété comme du bruit.

Pathbased : Quelle que soit la méthode utilisée, on n'obtient pas les 3 clusters attendus. Le résultat donné ici est obtenu avec la méthode de Ward pour  $t=20$ .

### Méthode gaussienne

Nous utiliserons les paramètres suivants.  $n\_components$  sera égal au nombre réel de clusters. Nous garderons  $n\_init = 50$ . Nous choisissons 'full' comme  $covariance\_type$  car il laisse le plus de liberté aux axes de contour de densité.

Avec ces paramètres, nous obtenons les résultats suivants.

Aggregation : Indice de Rand = 1

G-2-20 : clustering visuellement parfait, Coefficient de Silhouette = 0.75

G-2-100 : à nouveau, difficile à évaluer

Jain : Indice de Rand = 1

Pathbased : Indice de Rand = 1

### Méthode DBSCAN

Nous choisissons par défaut  $n=5$ .

Aggregation : En choisissant  $\epsilon=0.1235$ , on obtient un indice de Rand de 0.975. Certains points entre les clusters sont considérés comme du bruit.

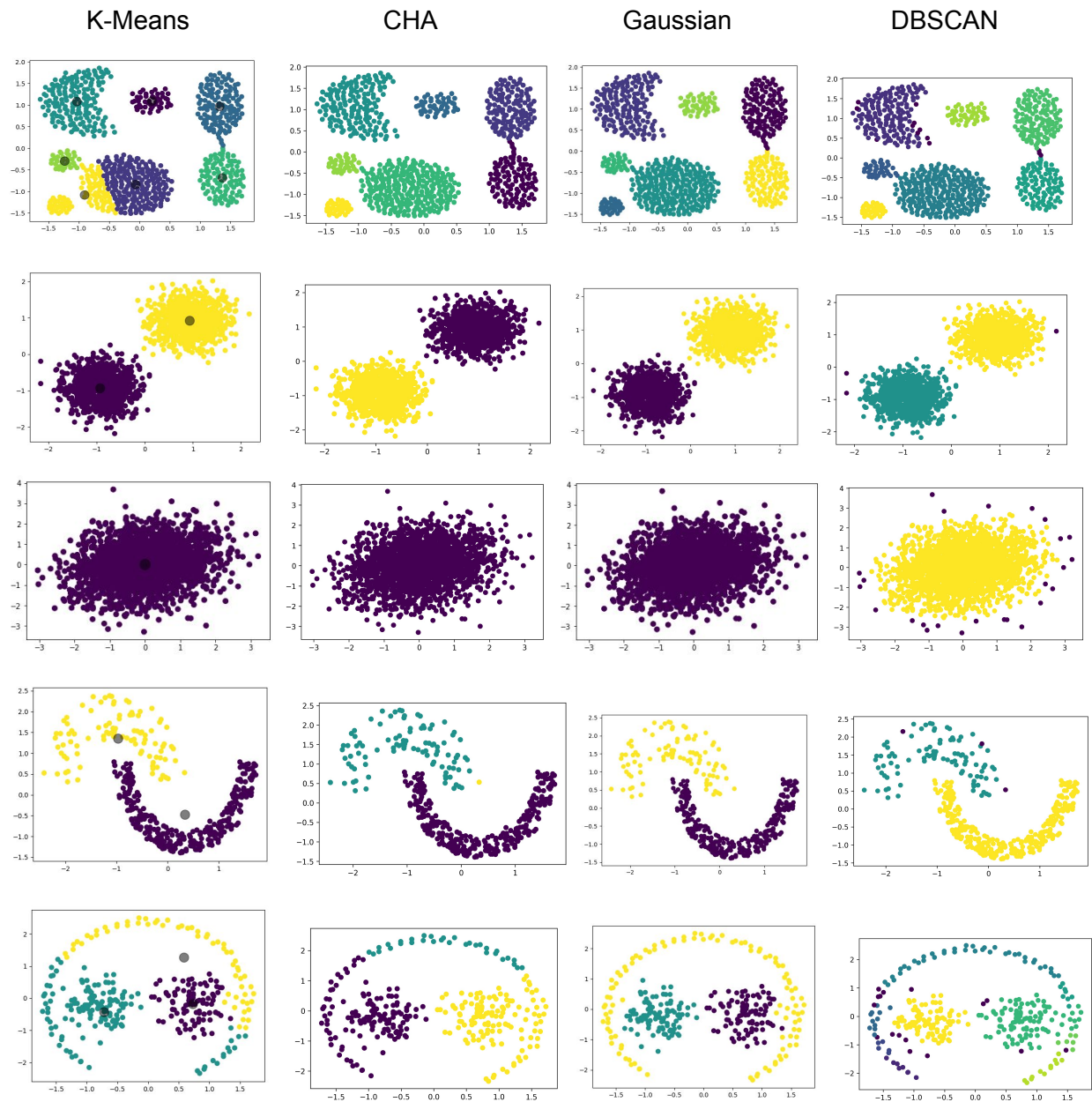
G-2-20 : On choisit  $\epsilon=0.292$ , on obtient un coefficient de silhouette de 0.71. 3 points sont considérés comme du bruit.

G-2-100 : Le DBSCAN se distingue ici par sa capacité à supprimer les points bruités.

Jain : Malgré 3 points bruités, le DBSCAN distingue les clusters.

Pathbased : Quel que soit  $\epsilon$ , les performances sont faibles, les clusters n'étant pas bien séparés et de densités variables. La représentation ci-dessous a été faite avec  $\epsilon=0.263$

## Représentations des différents clusters et méthodes



La méthode des K-Means fonctionne grâce au principe des centres des clusters et d'un rayon autour de ces centres : elle fonctionne donc mieux lorsque les clusters sont sphériques, ou au moins lorsque les clusters ne s'entrelacent pas.

Le CHA est adapté à des formes de clusters plus variées (comme pour le jeu de données Jain), car on peut adapter la métrique utilisée pour calculer la distance intercluster. Néanmoins, le CHA n'est pas adapté pour distinguer des clusters entrelacés.

La méthode gaussienne semble fonctionner correctement peu importe la forme des clusters. Cette méthode sera donc à privilégier lors de l'analyse des données réelles.

La méthode DBSCAN est particulièrement efficace pour des données bruitées, comportant des points à l'écart de tout cluster. Le clustering s'effectuant de proche en proche, les clusters peuvent avoir des formes variées, mais doivent être densément peuplés et être séparés d'une grande distance les uns des autres.

### **III. Classement des principaux pays du monde en fonction de leur développement**

#### **1. Examen des données**

Ce jeu de données a une taille de 10 x 167. La première colonne contient les noms des pays, et les 9 suivantes des statistiques socio-économiques les concernant. On peut remarquer que ces données ne sont pas homogènes. La première colonne contient des données de type qualitatif (object), tandis que les suivantes contiennent des données chiffrées (int ou float).

En utilisant la fonction `isna()`, nous pouvons remarquer qu'il manque en effet certaines données. Il manque la donnée 'GDP' pour la Norvège et l'Italie, ainsi que la donnée 'total\_fertility' pour la France et le Niger.

En construisant des histogrammes de chaque variable, nous avons pu repérer des valeurs aberrantes. La donnée 'life\_expectation' du Bangladesh est de 0, ce qui est impossible. De plus, l'inflation du Nigeria est de 104 %, ce qui semble très haut. En effet, en faisant des recherches, nous avons trouvé qu'entre 1960 et 2020, l'inflation du Nigeria n'avait jamais dépassé 72 %. Le jeu de données est censé provenir de 2010 environ, lorsque l'inflation du Nigeria tournait autour de 13 %. Nous avons donc considéré cette donnée comme aberrante. Finalement, nous avons trouvé 3 données aberrantes dans la colonne 'GDP'. Le Royaume-Uni, l'Australie, et les Etats-Unis ont 1 000 000 comme valeur de GDP, ce qui représente presque 10

fois plus que le plus grand suivant (le Luxembourg). En vérifiant le GDP de ces pays sur d'autres sites d'information, nous avons pu confirmer que ces valeurs étaient aberrantes.

## 2. Préparation des données

Tout d'abord, nous devons résoudre le problème des données manquantes et aberrantes. Au total, nous avons 9 données qui sont manquantes ou aberrantes. Il ne semble donc pas judicieux de supprimer les variables associées puisque le pourcentage de données erronées est très petit. Nous pourrions attribuer une valeur conforme à la distribution de la variable. Mais au vu du faible nombre de données erronées et du fait que ces données sont facilement accessibles grâce à d'autres sources d'information sur Internet, nous avons choisi cette option-ci.

Les données quantitatives ne sont pas toutes du même ordre de grandeur : il faudra donc centrer et réduire les données grâce à StandardScaler.

## 3. Recherche de corrélations

En observant la matrice de corrélation, nous pouvons observer que les corrélations les plus fortes sont entre `life_expectation` et `child_mortality`, `total_fertility` et `child_mortality`, `total_fertility` et `life_expectation`, `imports` et `exports`, et `GDP` et `income`. Nous pouvons remarquer que ces corrélations forment trois axes qui représentent différents critères. Le trio `life_expectation`, `child_mortality` et `total_fertility` forme l'axe de la santé, le duo `imports-exports` indique l'interaction du pays avec les autres, et l'axe `GDP-income` représente le niveau de richesse moyen des habitants.

D'après la scatter matrix, nous pouvons bien remarquer que la relation entre les couples mentionnés plus haut est bien linéaire. Nous pouvons aussi remarquer d'autres relations entre les variables. Les paires de variables `income-child mortality`, `income-life expectancy` et `GDP-child mortality` suivent une courbe en forme de  $1/x^2$ . Ces variables sont donc bien reliées. Par exemple, on peut remarquer les pays ayant un GDP très bas ont un taux de mortalité infantile très haut, mais le reste des pays ont un taux de mortalité infantile très bas.

Nous pourrions regarder si nous retrouvons ces corrélations dans les vecteurs propres lors de l'ACP.

## 4. Clustering des données

## A. Méthode et métrique utilisées

Nous effectuons le clustering sur les pays afin de les répartir en plusieurs classes selon leurs caractéristiques. Ainsi, nous espérons obtenir un cluster correspondant aux pays ayant besoin d'une aide humanitaire.

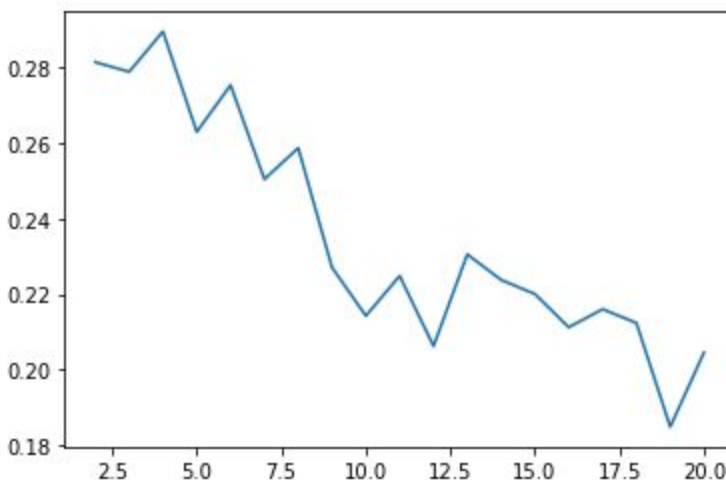
Nous avons choisi le coefficient de Silhouette pour évaluer la qualité des partitions. En effet, la plupart des métriques nécessitent le vrai partitionnement pour estimer la qualité du clustering. Nous ne pouvons donc pas utiliser ceux-là. Il nous reste le coefficient de Silhouette, l'indice de Calinski-Harabasz, et le score de Davies-Bouldin. Le score de Davies-Bouldin est une version moins coûteuse du coefficient de Silhouette, mais qui n'est pas compris entre -1 et 1. Au vu de la petite taille du jeu de données, nous n'avons pas besoin de diminuer le coût des calculs de score, donc nous n'utiliserons pas le score de Davies-Bouldin. L'indice de Calinski-Harabasz est rapide à calculer, ce qui n'est pas crucial ici au vu de la taille du jeu de données, et est efficace pour estimer le clustering lorsque les clusters sont denses et bien écartés, ce qui n'est pas le cas ici. De plus, il peut prendre n'importe quelle valeur, et pas seulement des valeurs entre -1 et 1. Nous avons donc choisi de garder uniquement le coefficient de Silhouette comme unique métrique pour estimer la meilleure valeur de k.

## B. Clustering

### Méthode K-Means

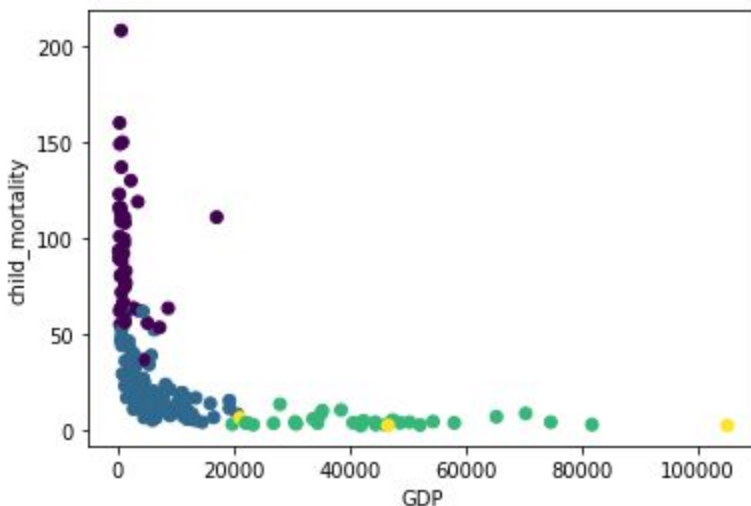
Les paramètres utilisés sont les mêmes que dans la partie II, pour les mêmes raisons que précisées dans cette partie. Nous choisissons donc `init = 'k-means++'`, `n_init = 50`, et `algorithm = 'elkan'`. Nous essayons de chercher la meilleure valeur de k, donc nous allons faire une boucle sur le paramètre `n_clusters`.

Voici le graphique obtenu en affichant le coefficient de Silhouette en fonction de k :





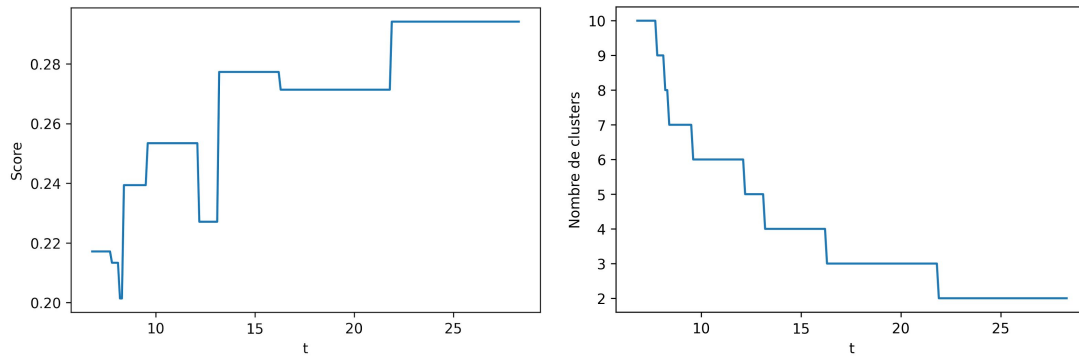
Nous nous sommes arrêtés à 20 clusters, car il était clair que le coefficient de Silhouette ne faisait que diminuer à partir de là. Nous pouvons remarquer que le coefficient de Silhouette est maximal pour des petites valeurs de  $k$  ( $k=4$  en particulier maximise la métrique). Nous prendrons donc cette valeur de  $k$  pour la suite. Le premier cluster contient les pays Luxembourg, Malte et Singapour. Ce sont des petits pays, riches et en bonne santé, avec des grandes valeurs d'import et d'export. Le deuxième cluster contient les pays d'Europe de l'Ouest et du Nord, ainsi que les pays d'Amérique du Nord, des pays riches du Moyen-Orient et de l'Asie de l'Est. Ce sont des grands pays riches en bonne santé. Le troisième cluster est le plus gros cluster, et contient la plupart des pays d'Amérique du Sud, d'Europe de l'Est, d'Asie du Sud et de l'Est et du Moyen-Orient. Comparés aux pays des autres clusters, ces pays sont dans la moyenne en termes de santé et de richesse socio-économique. Le dernier cluster est composé en grande partie de pays d'Afrique ainsi que quelques pays du Moyen-Orient et quelques îles. Ce cluster contient clairement les pays nécessitant le plus d'aide humanitaire. Nous pouvons visualiser ces clusters grâce au graphique suivant :



Le premier cluster correspond aux points jaunes, le deuxième aux points verts, le troisième aux points bleus et le dernier aux points violets. Nous pouvons voir que ce clustering sépare bien les pays ayant besoin d'aide humanitaire des autres. Le problème est que ce dernier cluster contient 46 pays, alors que nous ne cherchons que les 10 pays qui ont le plus besoin d'aide. Nous allons donc devoir restreindre cette liste.

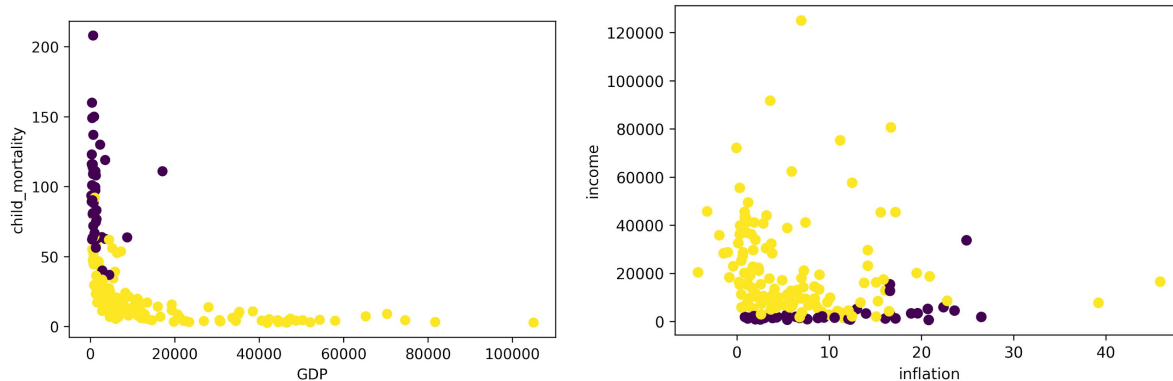
Méthode CHA :

Ne connaissant pas d'avance la forme des clusters que l'on cherche à obtenir, nous utilisons la méthode de Ward, qui garantit la plus lente augmentation de l'inertie intra-cluster. Afin de paramétrer  $t$ , qui nous donnera le nombre de clusters, on affiche deux graphes :



Sur le premier, on recherche  $t$  maximisant le silhouette-score. On trouve  $t=25$ . Ensuite, on se sert de cette valeur de  $t$  pour en déduire le nombre de clusters, soit 2.

Voici ci-dessous les deux projections de ce clustering :

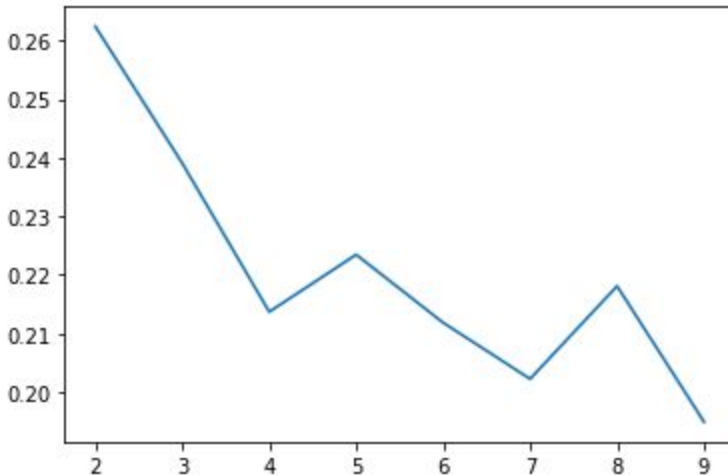


Le clustering est bien visible à la fois sur les données corrélées et décorrélées.

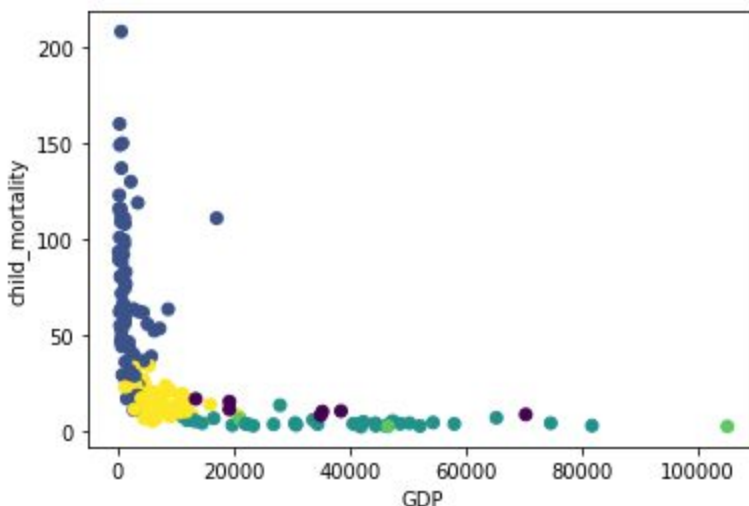
Néanmoins, le nombre de pays du cluster semblant avoir besoin d'aide humanitaire est très important, c'est pourquoi il nous faudra restreindre la sélection par la suite.

### Méthode gaussienne

Les paramètres utilisés sont les mêmes que dans la partie II, sauf le  $n_{init}$  que l'on augmente à 500. En effet, avec  $n_{init} = 50$ , d'une exécution à une autre, les valeurs optimales de  $k$  changeaient. Nous avons donc décidé d'augmenter le  $n_{init}$  afin de faire converger l'algorithme. Nous faisons une boucle sur le paramètre  $n_{components}$  afin de trouver la valeur optimale de  $k$ . Le graphique obtenu en affichant le coefficient de Silhouette est le suivant :



Nous pouvons voir que le maximum est atteint pour  $k = 2$ . Néanmoins, connaissant notre but final qui est la construction de la liste de pays, il ne serait pas judicieux de séparer l'ensemble des données en 2. En effet, avec  $k=2$ , nous trouvons une liste des pays ayant le plus besoin d'aide contenant 110 pays. Ceci n'est pas assez utile pour restreindre la recherche. Nous choisissons de prendre  $k = 5$ . En effet, ceci nous donnera des clusters plus pertinents tout en gardant le coefficient de Silhouette tout de même assez haut (0.262 comparé à 0.224). Comme pour le K-Means, nous avons un cluster formé par le Luxembourg, Singapour et Malte, un cluster de pays d'Europe et d'Amérique riches et en bonne santé, un cluster de pays d'Europe de l'Est et d'Amérique du Sud dans la moyenne, et un cluster de pays africains pauvres. Le cinquième cluster est composé de 6 pays riches du Moyen-Orient. Nous pouvons visualiser ces clusters grâce au graphique suivant :



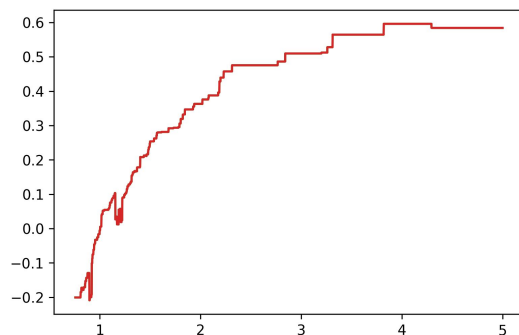
Le premier cluster est en vert clair, le deuxième est en vert foncé, le troisième en jaune, le quatrième en bleu et le dernier en violet. À nouveau, il est clair que les pays à aider se trouvent

dans le cluster bleu. Malheureusement, ce cluster contient 71 pays, ce qui reste beaucoup trop pour pouvoir affiner la sélection finale.

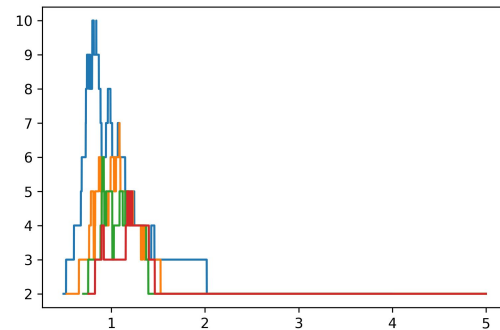
## Méthode DBSCAN

Afin de déterminer les paramètres pertinents du dbscan, on étudie 3 indicateurs : le coefficient de silhouette en fonction de  $\varepsilon$  et de  $n$ , le nombre de clusters obtenus par le partitionnement sont commun avec le cha, car ces deux algorithmes n'admettent pas le nombre de clusters comme paramètres. De plus, certains pays nécessitant de l'aide humanitaire peuvent être isolés, donc considérés comme du bruit par le dbscan, c'est pourquoi il faut minimiser le nombre de pays considérés comme du bruit par le dbscan.

Voici ci dessous les graphiques obtenus :







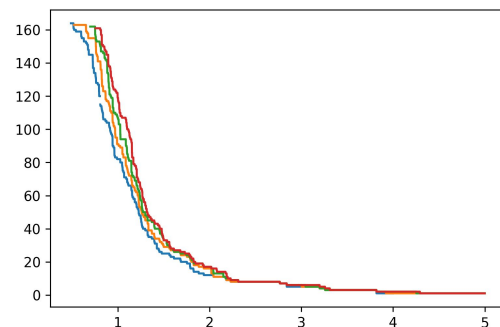
Score en fonction de  $\varepsilon$  (les courbes sont superposées)



Nombre de partitions (dont celle du bruit) en fonction de  $\varepsilon$

Légende :

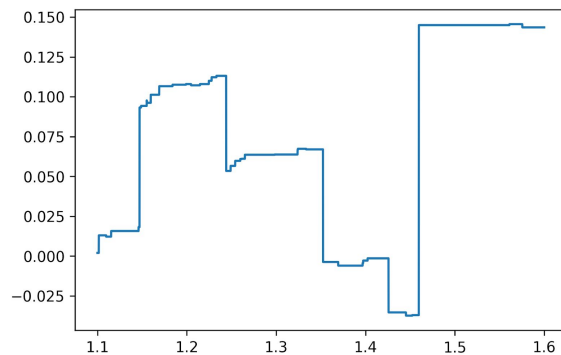
	$n=3$
	$n=4$
	$n=5$
	$n=6$



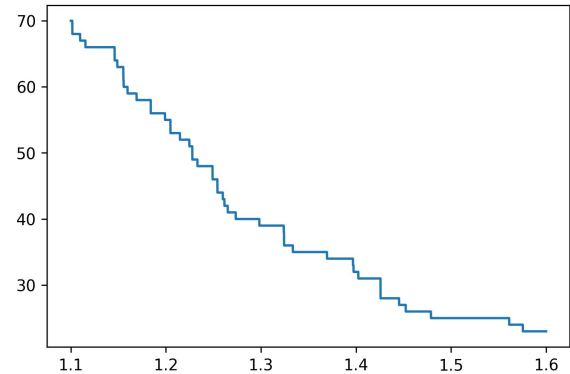
Nombre de pays considérés comme du bruit en fonction de  $\varepsilon$

L'examen de ces données montre que le paramètre  $n$  a peu d'importance sur le silhouette-score.  $\varepsilon=4$  semble être la valeur idéale, pour maximiser le score et minimiser la taille des données bruitées. Néanmoins, le nombre de clusters (en excluant les données bruitées) est

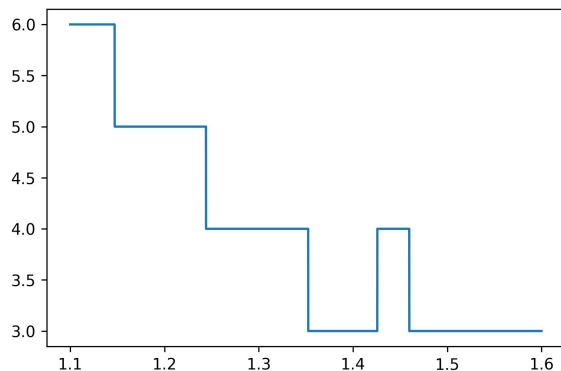
de 1, ce qui ne correspond pas à l'objectif de réduction des données sur lesquelles porter l'étude. Il semble alors plus cohérent de choisir  $\varepsilon=2$  et  $n=3$  pour obtenir le plus haut silhouette-score et la plus faible taille du cluster bruité. Néanmoins, lorsqu'on partitionne les données, on trouve des clusters de taille très inégale, puisque l'un des clusters comprend 152 pays, tandis que l'autre ne comprend que 3 pays. En restreignant la plage de données examinées, on trouve les graphiques suivants (pour  $n=3$ ) :



Score en fonction de  $\varepsilon$



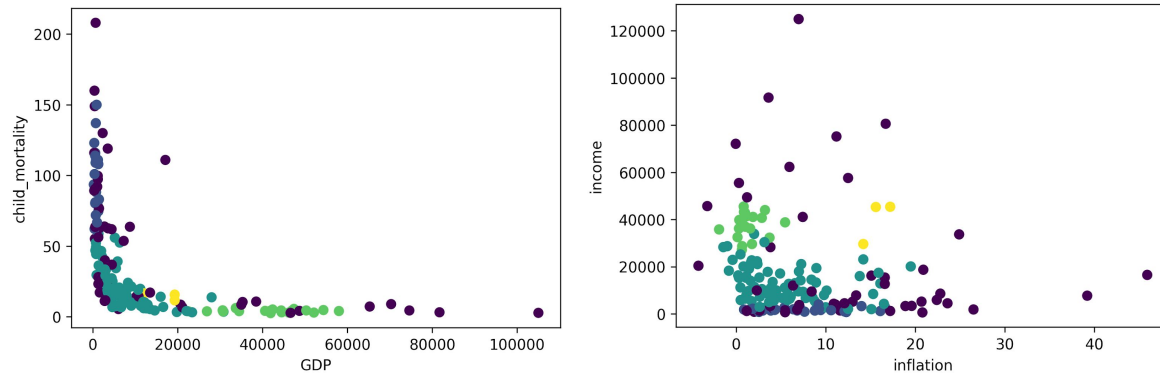
Nombre de pays considérés comme du bruit en fonction de  $\varepsilon$



Nombre de partitions (dont celle du bruit) en fonction de  $\varepsilon$

On choisit alors  $\varepsilon=1.24$ , et on obtient ainsi 4 clusters. Le nombre de données bruitées est alors de 48, ce qui est assez haut. Néanmoins, le cluster semblant correspondre aux pays ayant besoin d'aide est de 23, ce qui est plus précis que le cluster trouvé à l'aide des autres algorithmes, mais supérieur aux 10 pays demandés.

Voici ci-dessous le deux projections de ce clustering :



On remarque que beaucoup de données sont considérées comme bruitées (en violet foncé). Les données ignorées par le dbSCAN semblent occuper des points recouverts par des clusters, et on peut s'interroger sur la pertinence de l'utilisation de cet algorithme, qui risque d'ignorer des pays ayant besoin d'aide.

### C. Exploitation des clusters et détermination de la liste des pays

Une fois le clustering effectué, il nous faut déterminer le cluster des pays qui ont le plus besoin d'aide. Pour faire cela, nous pourrions procéder graphiquement et estimer ce cluster. Ceci est faisable lorsque le nombre de clusters est petit. Avec 3 ou 4 clusters, il est facile d'identifier le cluster des pays "riches", celui des pays "moyen" et celui des pays "pauvres". Plus le nombre de clusters augmente et plus il est difficile de l'identifier puisqu'il y aura moins de pays par cluster. Il sera alors nécessaire de le trouver par un autre moyen.

Nous avons identifié deux types de caractéristiques : les caractéristiques qui nous indiquent clairement si un pays a besoin d'aide, et celles qui sont incertaines. Par exemple, le taux de mortalité infantile appartient à la première catégorie, car plus ce taux est haut et plus le pays a besoin d'aide. Nous appellerons ces caractéristiques des caractéristiques déterminantes. La statistique d'import, par exemple, appartient à la deuxième catégorie, car les Etats-Unis, pays qui n'a absolument pas besoin d'aide, et le Nigeria, qui a beaucoup plus besoin d'aide, ont des taux d'imports très bas. Nous appellerons ces caractéristiques non-déterminantes.

En sachant cela, nous avons identifié les caractéristiques suivantes comme déterminantes : mortalité infantile, salaire, espérance de vie, fertilité totale et GDP. Un pays ayant besoin d'aide a souvent une mortalité infantile haute, une haute fertilité, une espérance de vie basse, un salaire bas et un GDP bas. Pour chacune de ces catégories, nous trouvons le cluster auquel appartient le pays avec le pire score dans cette catégorie. Par exemple, pour la mortalité infantile, nous trouvons le cluster auquel appartient Haïti, qui est le pays avec la plus haute mortalité infantile. Ceci est fait grâce à la fonction *clustersParCaracteristiques*, qui renvoie un dictionnaire comme ceci par exemple :

```
{'child_mortality': 2,
 'income': 2,
```

```
'life_expectation': 2,  
'total_fertility': 3,  
'GDP': 2}
```

Ceci signifie que le cluster 2 contient le pays avec la plus haute mortalité infantile, le pays avec le salaire le plus bas, le pays avec l'espérance de vie la plus basse et le pays avec le GDP le plus bas, tandis que le cluster 3 contient le pays avec la fertilité totale la plus haute. A partir de ces informations, nous pouvons en déduire que le cluster 2 est celui qui a le plus besoin d'aide. La fonction *clusterSatisfaisantLePlusDeCaracteristiques* nous donne cette information.

Connaissant le cluster qui a le plus besoin d'aide, nous pouvons maintenant créer un pays fictif. Ce pays fictif est le pays qui, d'après les caractéristiques données, aurait le plus besoin d'aide s'il existait. Pour lui assigner ses caractéristiques déterminantes, nous prenons la valeur extrême présente dans les données. Par exemple, le pays fictif a le taux de mortalité infantile de Haïti et le GDP du Burundi. Pour les caractéristiques non-déterminantes, nous prenons la moyenne des pays inclus dans le cluster qui a le plus besoin d'aide (cluster identifié à l'étape précédente). Ce pays fictif est donné par la fonction *paysFictif*.

Ce pays fictif nous sert à trouver la liste des 10 pays à aider à partir du cluster des pays qui ont le plus besoin d'aide. Nous faisons une recherche k-NN pour trouver les 10 plus proches voisins du pays fictif parmi les pays du cluster qui a besoin d'aide. La recherche k-NN se fait en brute force, et non à l'aide d'un Kd-tree ou un LSH, car la taille des données est telle que ce calcul ne prend pas beaucoup de temps. De plus, cela nous donne un résultat exact et non un résultat approché. Le code de la fonction de la recherche k-NN provient du cours de Mr.Tirilly.

Nous pouvons répéter ce processus avec chaque méthode, et obtenir quatre listes de 10 pays à aider. Notre liste finale correspond à l'intersection de ces quatre listes. Le DBSCAN est trop restrictif et limite trop l'intersection : nous avons donc choisi de ne pas l'inclure dans la liste finale.

K-Means: ['Haïti', 'Congo Dem. Rep.', 'Sierra Leone', 'Chad', 'Guinea', 'Nigeria', 'Zambia', 'Central African Republic', 'Angola', 'Malawi']

CHA: ['Chad', 'Haïti', 'Sierra Leone', 'Congo Dem. Rep.', 'Guinea', 'Malawi', 'Central African Republic', 'Nigeria', 'Zambia', 'Angola']

Gaussian: ['Haïti', 'Congo Dem. Rep.', 'Chad', 'Sierra Leone', 'Angola', 'Guinea', 'Zambia', 'Central African Republic', 'Malawi', 'Nigeria']

DBSCAN: ['Guinea', 'Zambia', 'Burkina Faso', 'Mozambique', 'Mali', 'Chad', 'Malawi', 'Afghanistan', 'Niger', 'Côte d'Ivoire']

Liste Finale: ['Angola', 'Central African Republic', 'Chad', 'Congo Dem. Rep.', 'Guinea', 'Haïti', 'Malawi', 'Nigeria', 'Sierra Leone', 'Zambia']

Ces pays ont principalement besoin d'aide dans les domaines déterminants. Il faudrait que l'aide humanitaire arrive à diminuer la mortalité infantile, augmenter le salaire moyen, augmenter l'espérance de vie, diminuer la fertilité et augmenter le GDP.

## 5. Clustering des données après réduction de dimension

### A. Clustering

Nous voulons diminuer le nombre de dimensions afin d'améliorer les performances du clustering. Nous utilisons la classe PCA de *sklearn.decomposition*. Le seul paramètre sur lequel nous allons influencer est le nombre d'axes à garder, `n_components`. Nous utiliserons la fonction *fit\_transform* afin de transformer les données.

Nous avons fait le choix de garder 3 axes selon la règle de Kaiser (les valeurs propres sont 4.17, 1.56 et 1.22).

Le premier axe est [-0.41596419, 0.28141702, 0.15695008, 0.16084958, 0.39806856, -0.20027536, 0.42523544, -0.40300361, 0.39429234].

Nous pouvons remarquer que les valeurs les plus grandes correspondent aux caractéristiques déterminantes que nous avons définies plus tôt. Cette information nous confirme que notre division en caractéristiques déterminantes et non-déterminantes était pertinente. De même, parmi ceux-là, nous voyons que le signe du coefficient de la mortalité infantile et de la fertilité est opposé au signe du salaire, de l'espérance de vie et du GDP. Ceci est logique, car un pays ayant besoin d'aide est un pays avec une haute mortalité, une haute fertilité, une espérance de vie basse, un salaire bas et un GDP bas.

Le deuxième axe est [0.18666955, 0.61447551, -0.24340834, 0.67406972, 0.02103113, -0.0109703, -0.21857273, 0.14896975, -0.05854941].

Ceci est l'axe correspondant principalement à la relation proportionnelle entre les imports et les exports.

Le troisième axe est [-0.13657735, 0.14454675, -0.65567254, -0.26409625, 0.21262093, 0.62307445, 0.1505581, -0.06803205, -0.00860434].

Ce dernier axe comprend la relation inversement proportionnel de health et inflation.

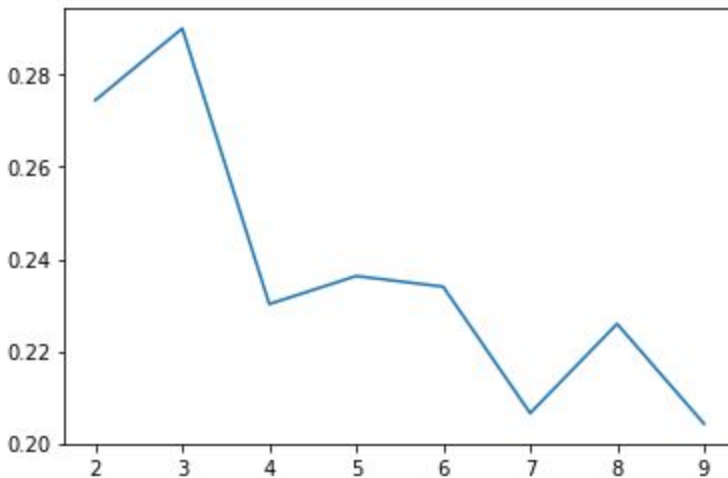
Nous pouvons remarquer que la combinaison des trois axes comprend les 9 variables étudiées.

Nous appliquons désormais le clustering à ces nouvelles données en utilisant les quatre méthodes.

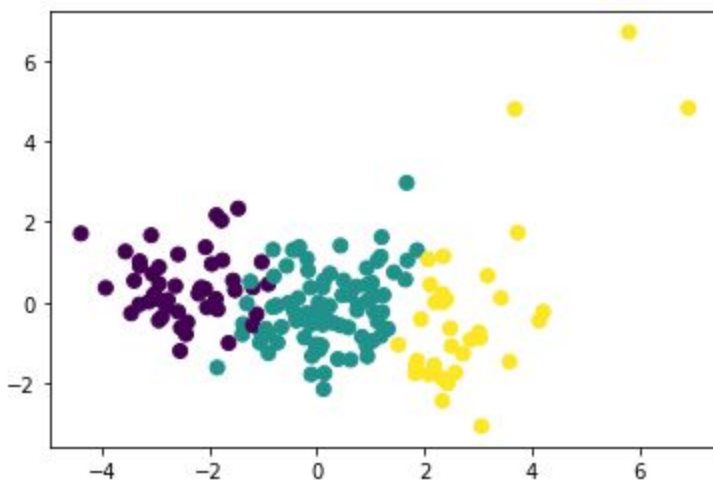
Méthode K-Means



Pour le nombre de clusters, nous avons choisi de prendre  $k = 3$  car cela maximisait clairement le coefficient de Silhouette comme montré sur le graphique ci-dessous :



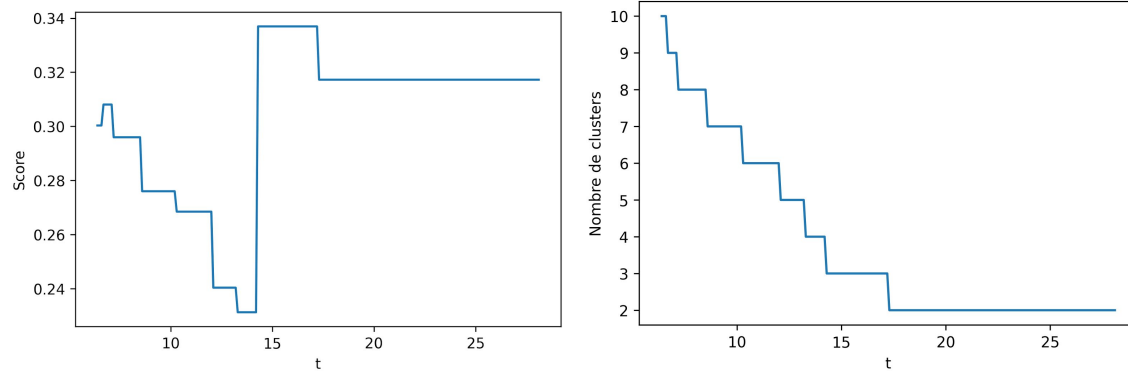
Le clustering ressemble à ceci en représentant les clusters sur les 2 premiers axes:



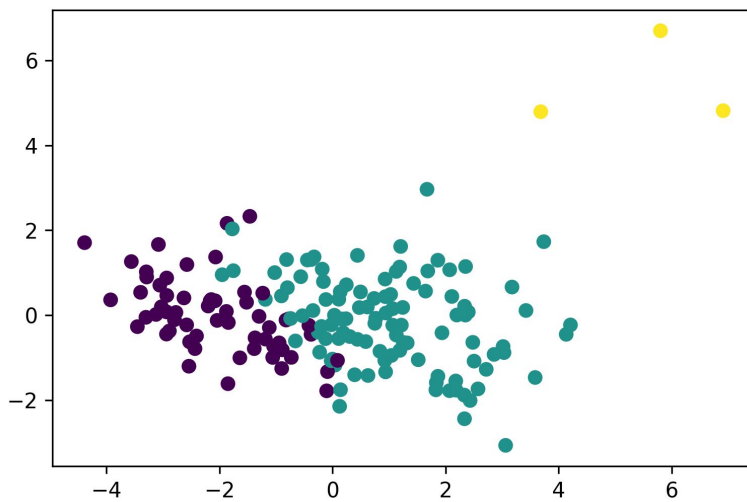
À nouveau, il est clair que le cluster contenant les pays qui ont le plus besoin d'aide est le cluster violet, qui contient des pays comme Haïti et le Mozambique. Ce cluster sera environ le même à chaque méthode, et il ne sera donc pas précisé à chaque fois que c'est celui-ci que nous utiliserons.

#### Méthode CHA

Comme avec les données initiales, on trace le score et le nombre de clusters en fonction de  $t$ . On choisit  $t=15$ .

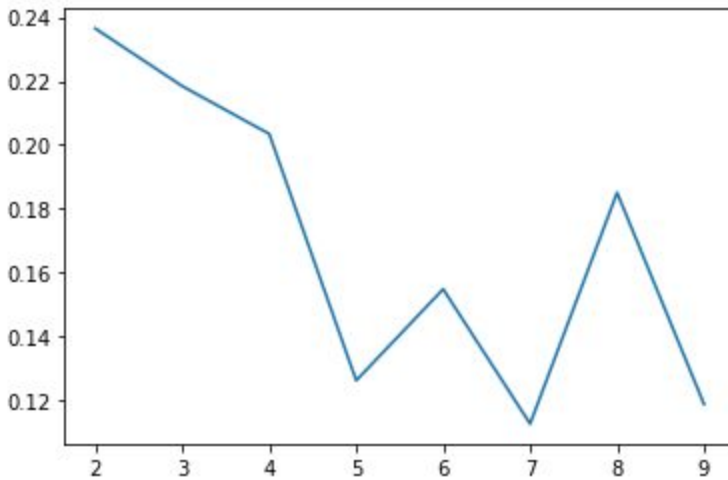


On obtient ainsi les clusters suivants :

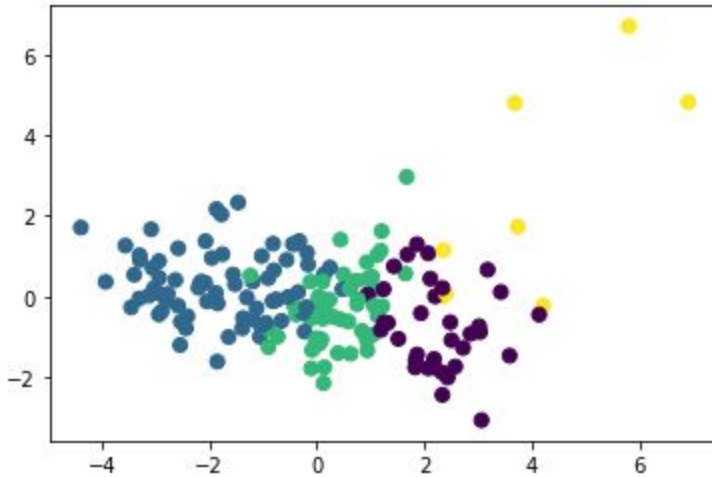


### Méthode gaussienne

Pour le nombre de clusters, nous avons choisi de prendre  $k = 4$ , car le coefficient de Silhouette diminuait rapidement après cette valeur comme montré sur le graphique ci-dessous :



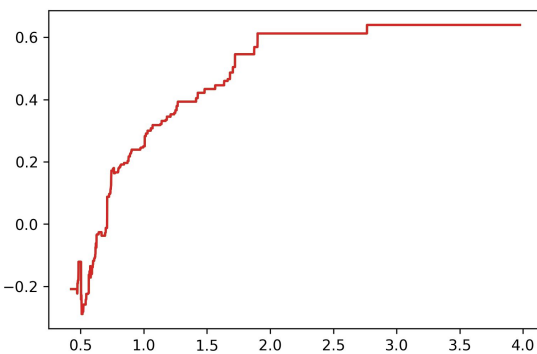
Les clusters obtenus sont représentés ci-dessous :



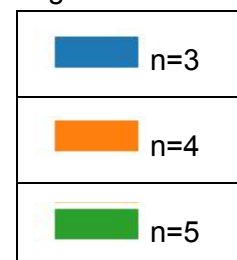
### Méthode DBSCAN

Les graphes de coefficient de silhouette, nombre de partitions et de pays bruités en fonction de  $\epsilon$  sont très similaires aux graphes obtenus avec les données complètes :

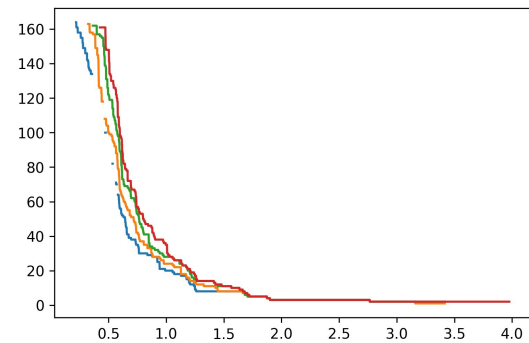
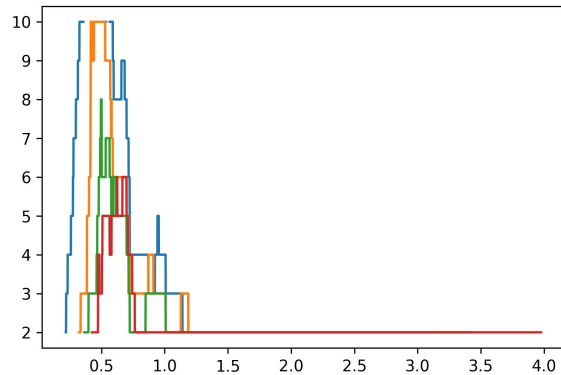
Score en fonction de  $\epsilon$  (les courbes sont superposées)



Légende :



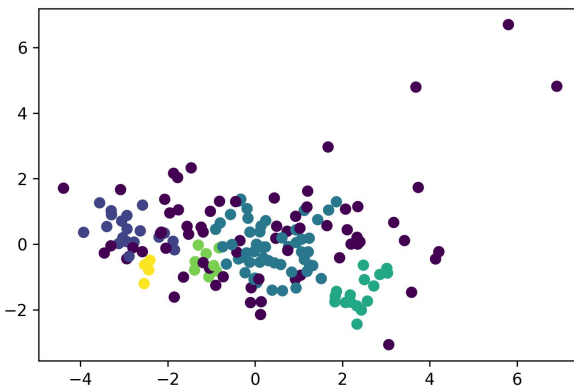
  $n=6$



Nombre de pays considérés comme du bruit en fonction de  $\varepsilon$

Nombre de partitions (dont celle du bruit) en fonction de  $\varepsilon$

Là encore, il faut examiner en détail les données pour déduire que  $\varepsilon=0.6$  et  $n=4$  sont les paramètres optimaux.



Le clustering ainsi obtenu est le suivant :

Les données bruitées sont ici très nombreuses

## B. Exploitation des clusters et détermination de la liste des pays

Nous allons suivre le même processus pour l'exploitation des clusters que celui de la partie sans ACP. La seule différence provient du calcul du cluster des pays qui ont le plus besoin d'aide. Pour trouver celui-ci, nous avons choisi d'utiliser le fait que le premier axe de l'ACP correspondait à nos caractéristiques déterminantes. Nous calculons donc le centre de chaque cluster, puis nous comparons la première coordonnée de chaque centre entre eux. Le cluster dont le centre a la plus faible première coordonnée est pris comme cluster contenant les pays ayant le plus besoin d'aide.

Après avoir trouvé ce cluster, nous créons notre pays fictif, et faisons une recherche k-NN pour trouver les 10 plus proches voisins de ce pays, comme dans la partie précédente.

Les résultats obtenus sont les suivants:

K-Means : ['Mali', 'Guinea', 'Chad', 'Angola', 'Nigeria', 'Central African Republic', 'Zambia', 'Congo Dem. Rep.', 'Haiti', 'Mozambique']

CHA : ['Congo Dem. Rep.', 'Chad', 'Angola', 'Central African Republic', 'Nigeria', 'Guinea', 'Zambia', 'Mali', 'Mozambique', 'Malawi']

Gaussian : ['Haiti', 'Central African Republic', 'Chad', 'Angola', 'Nigeria', 'Congo Dem. Rep.', 'Mozambique', 'Guinea', 'Mali', 'Zambia']

DBSCAN : ['Central African Republic', 'Chad', 'Congo Dem. Rep.', 'Mali', 'Guinea', 'Niger', 'Mozambique', 'Zambia', 'Malawi', 'Burkina Faso']

Liste finale: ['Mali', 'Guinea', 'Chad', 'Central African Republic', 'Zambia', 'Congo Dem. Rep.', 'Mozambique']

Par rapport à la liste finale obtenue sans ACP, nous pouvons remarquer que l'intersection a renvoyée moins de pays (7 uniquement). Ceci est dû au fait que le DBSCAN a retourné des pays partiellement différents des autres listes. La République Centrafricaine, le Chad, la République Démocratique du Congo, la Guinée, et la Zambie sont communs aux deux listes. Cette liste semble logique puisque ces pays sont en effet des pays en grand besoin d'aide humanitaire.

Les domaines d'action sont les mêmes que ceux préconisés dans la partie précédente.