Exercice 6
Algorithmes de Clustering

Distances

Distance de Manhattan :	$d(A, B) = \sum_{i=1}^{n} \lvert A_i - B_i \rvert$
Distance euclidienne :	$d(A, B) = \sqrt{\sum_{i=1}^{n} (A_i - B_i)^2}$
Distance de Minkowski :	$d(A, B) = (\sum_{i=1}^{n} \lvert A_i - B_i \rvert^p)^{1/p}$
Distance de cosine :	$d(A, B) = 1 - \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} {A_i}^2} \sqrt{\sum_{i=1}^{n} {B_i}^2}}$
Distance de Hamming :	$d(A, B) = \#\{ i : A_i \neq B_i \}$

Algorithme

Algorithme : k-means

Entrée : *D* : Dataset ; *k* : le nombre de cluster à former ;
Sortie : D: Dataset étiqueté **;**
Début
  Choisir aléatoirement k instances comme centroïdes.
  Répéter
    Calculer la distance entre chaque instances D[i] et les k centroïdes ;
    Affecter chaque instances D[i] au groupe dont il est le plus proche de son centre;
    Calculer le nouveau centre de chaque cluster et modifier le centroïde ;
  Jusqu'à D[i] identique à D[i-1] ;
  Retourner D ;
Fin.

Evaluation

Le coefficient (ou score) de silhouette se définit d'abord sur un point i dont le groupe est $k = C(i)$. Il se base sur la distance moyenne du point à son groupe $a(i) = \frac{1}{|I_k|-1} \sum_{j \in I_k, j \neq i} d(x^i, x^j)$ et la distance moyenne du point à son groupe voisin $b(i) = \min_{k' \neq k} \frac{1}{|I_{k'}|} \sum_{i' \in I_{k'}} d(x^i, x^{i'})$. Le coefficient de silhouette du point i s'écrit alors :

$$\boldsymbol{s_{sil}(i) = \frac{b(i) - a(i)}{max(a(i), b(i))}}$$

On peut le moyenner groupe par groupe pour comparer leurs homogénéités : ceux avec le coefficient de silhouette les plus forts sont les plus homogènes. Sur l'ensemble de données, il aura pour expression :

$$\boldsymbol{S_{sil} = \frac{1}{K} \sum_{k=1}^{K} \frac{1}{|I_k|} \sum_{i' \in I_{k'}} s_{sil}(i)}$$

BELKADI Widad Hassina (G1-G2): belkadi.wh.usthb@gmail.com
HOUACINE Naila Aziza (G3-G4) : n.a.houacine@gmail.com

Questions :

1- Écrire une fonction Python permettant de calculer la distance entre deux instances du dataset (implémenter les distances applicables parmi ceux lister).

2- Écrire une fonction python permettant de calculer le centroïde d'un ensemble d'instances.

3- Écrire une fonction python permettant de trouver le cluster dont une instance donnée est la plus proche.

4- Implémenter en python l'algorithme k-means.

5- Déduire les clusters formés (sans considérer la classe comme un attribut).

6- Evaluer vos résultats en utilisant le coefficient de silhouette.