Master : IIBDCC 1

Workshop: K-means and PCA for segmentation

Nom : FAKHIR Badr

* **Importation des librairies de Python nécessaires.**

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.datasets import make\_blobs

from sklearn.cluster import KMeans

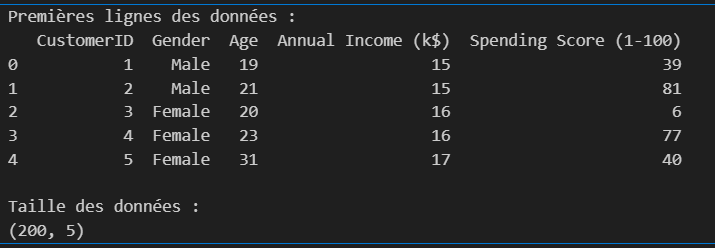
from sklearn.metrics import silhouette\_score, calinski\_harabasz\_score

from sklearn.preprocessing import StandardScaler

from sklearn.decomposition import PCA

* **Chargement des données ou bien simuler aléatoirement les données de la dimension supérieur à 4.**
* file\_path = "Mall\_Customers.csv"
* df = pd.read\_csv(file\_path)

* **Visualisez et donner la taille des données**
* print(df.head())
* print(df.shape)



* **Implémentez l’algorithme K-moyenne avec les stratégies d’initialisation des centres suivantes :  
  a. Aléatoire  
  b. K-means++**
* X = df[['Age', 'Annual Income (k$)', 'Spending Score (1-100)']]
* scaler = StandardScaler()
* X\_scaled = scaler.fit\_transform(X)
* kmeans\_random = KMeans(n\_clusters=5, init='random', random\_state=0)
* kmeans\_random.fit(X\_scaled)
* centroids\_random = kmeans\_random.cluster\_centers\_
* kmeans\_data = KMeans(n\_clusters=5, init='k-means++', random\_state=0)
* kmeans\_data.fit(X\_scaled)
* centroids\_data = kmeans\_data.cluster\_centers\_
* **Implémentez les méthodes de validation de Clustering.**

silhouette\_random = silhouette\_score(X\_scaled, kmeans\_random.labels\_)

silhouette\_data = silhouette\_score(X\_scaled, kmeans\_data.labels\_)

print(f"Score de silhouette (initialisation aléatoire): {silhouette\_random}")

print(f"Score de silhouette (initialisation à partir des données): {silhouette\_data}")



* **Interprétez les résultats obtenus.**
* Le score de silhouette obtenu (~0.42) indique une séparation modérée des clusters. Cela suggère que les données pourraient avoir des zones de chevauchement entre les clusters
* l'indice de Calinski-Harabasz est de 125.10 pour les deux initialisations, ce qui est relativement bas
* **Quel est le meilleur modèle de Clustering (meilleurs paramètres).**

K=4 avec la méthode K-means++

* **Peut on représenter les données avec les poids des centres obtenus ?**
* calinski\_random = calinski\_harabasz\_score(X\_scaled, kmeans\_random.labels\_)
* calinski\_data = calinski\_harabasz\_score(X\_scaled, kmeans\_data.labels\_)
* print(f"Indice de Calinski-Harabasz (initialisation aléatoire): {calinski\_random}")
* print(f"Indice de Calinski-Harabasz (initialisation à partir des données): {calinski\_data}")



* **Donner la nouvelle matrice des observations.**
* pca = PCA(n\_components=2)
* X\_pca = pca.fit\_transform(X\_scaled)
* print("Nouvelle matrice des observations après PCA :")
* print(X\_pca[:5])

Une image contenant texte, Police, capture d’écran

Description générée automatiquement

* **Afficher les valeurs propres et les vecteurs propres associes aux axes principaux.**
* centroids\_pca = pca.transform(centroids\_data)
* print("Valeurs propres des axes principaux :")
* print(pca.explained\_variance\_)
* print("Vecteurs propres (composantes principales) :")
* print(pca.components\_)



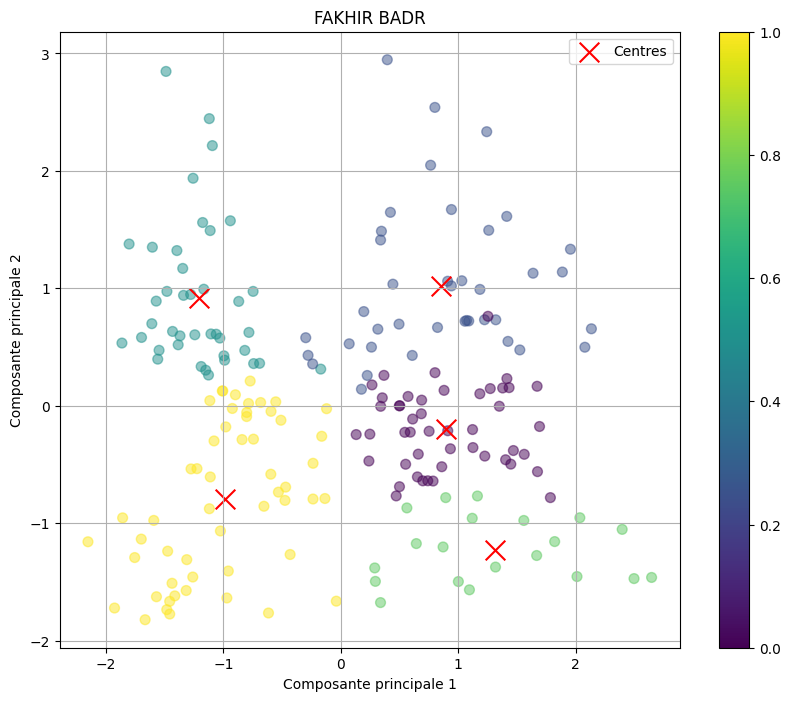
* **Donner l’inertie de chaque axe.**
* print("Inertie de chaque axe :")
* print(pca.explained\_variance\_ratio\_)



* **Vérifier que la somme des inerties de chaque axe égal la dimension de la base de données.**
* print("Somme des inerties des axes :")
* print(np.sum(pca.explained\_variance\_ratio\_))



* **Représenter les données ainsi que les centres obtenus par l’algorithme du k-means sur les deux axes principaux.**
* plt.figure(figsize=(10, 8))
* plt.scatter(X\_pca[:, 0], X\_pca[:, 1], c=kmeans\_data.labels\_, cmap='viridis', s=50, alpha=0.5)
* plt.scatter(centroids\_pca[:, 0], centroids\_pca[:, 1], marker='x', s=200, c='red', label='Centres')
* plt.title('Clusters obtenus par K-means (PCA réduit à 2 dimensions)')
* plt.xlabel('Composante principale 1')
* plt.ylabel('Composante principale 2')
* plt.colorbar()
* plt.legend()
* plt.grid(True)
* plt.show()



* **Interpréter les résultats obtenus**  
    
  Une bonne séparation entre les clusters dans l'espace des composantes principales suggère une bonne performance du clustering