Master 1 SD

Nom : NIANG
Prénom : Souna

Analyse et visualisation de données

Séance de TP 5
Analyse en Composantes Principales (ACP)

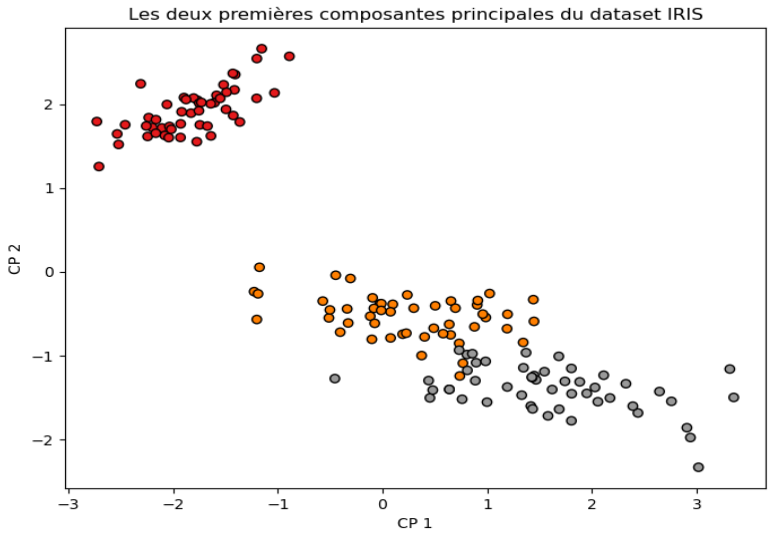
1- ACP

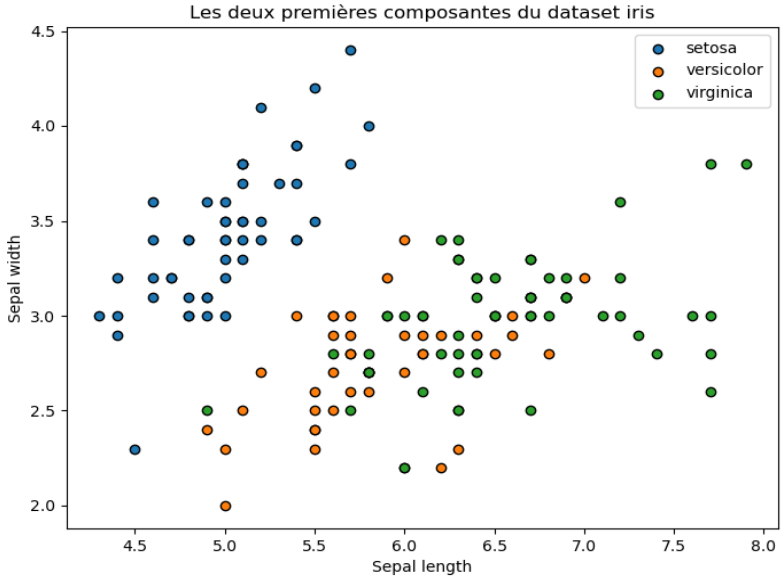
a. *Programmer une ACP en vous conformant au modèle proposé*

```
def myACP(X):
    n = X.shape[1]
    m = X.shape[0]
    moy = np.sum(X,0)/m # axe de la matrice selon lequel on somme
    np.reshape(moy,(n,1))
    # données centrées
    XC = X-moy.T

    # covariance
    S = X.T@X/n

    # calcule des valeurs propres et vecteurs propres
    # vecteurs propres de norme 1 rangés en colonnes

    Valp, Vectp = np.linalg.eig(S)
    # il faut ordonner dans l'ordre des valeurs propres décroissantes
    Valp,Vectp = TriVP(Valp,Vectp)
    # on projette sur les deux premiers axes principaux
    Projection = XC @ Vectp[:,:2]

    return Projection
```

b. *Visualiser les deux premières composantes principales du dataset IRIS, et comparer la représentation obtenue avec le nuage de points formé des deux premières composantes.*

Réponse



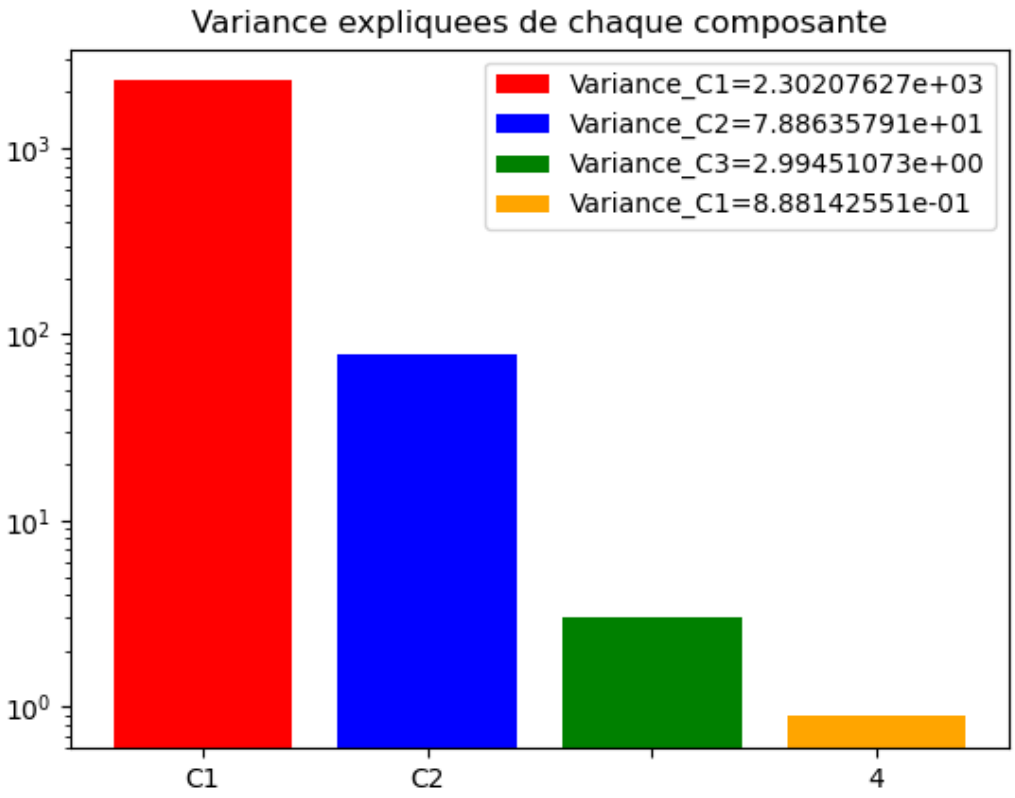


On voit que les deux composantes principales donnée par la méthode **myACP** représentent bien les deux premières composantes du dataset iris. On a trouvé les mêmes distributions pour les trois classes du dataset iris donc on garder l'information.

c. *Déterminer les variances expliquées des quatre composantes et les représenter graphiquement.*

Réponse

Les variances les variances expliquées *des* quatre composantes repensent respectivement les valeurs propres associées aux vecteurs propres dont on a projeté les données.



Les vraies valeurs des variances expliquées que l'algorithme a trouvé sont représentées en haut à droite. Pour pouvoir représenter les valeurs des deux dernières composantes qui sont trop petites j'ai pris une échelle logarithmique.

On voit la première composante est associe à une valeur propre très élevé ce qui explique sa grande variance expliquée par rapport aux autres composantes . C'est normal parce que l'algorithme recherche la composante qui résume les données avec une plus grande variance.

d. *Déterminer les directions propres des quatre composantes.*

Réponse

Les directions propres correspondent aux vecteurs propres de la matrice de covariance. Les données vont êtres projeté sur ces directions.
La première direction principale sera orientée dans la direction de la plus grande des données.

```
array([[ 0.75110816,  0.2841749 ,  0.50215472,  0.32081425],
       [ 0.38008617,  0.5467445 , -0.67524332, -0.31725607],
       [ 0.51300886, -0.70866455, -0.05916621, -0.48074507],
       [ 0.16790754, -0.34367081, -0.53701625,  0.75187165]])
```

Voici les 4 vecteurs propres qui correspondent aux directions

2- ACP à noyaux

a. Programmer l'ACP à noyau en complétant le code de la méthode ci-dessous, où la méthode Kernel (fournie avec le code) permet de calculer la matrice de Gram pour différents noyaux (linéaire, rbf, polynomial), et la méthode TriVP permet de trier les vecteurs propres dans l'ordre décroissant des modules des valeurs propres.

b. En décommentant le lignes correspondantes du programme principal, tester votre code avec un noyau linéaire et le comparer à l'ACP simple et à l'ACP à noyau linéaire de scikitlearn.

c. Visualiser les valeurs propres de l'ACP à noyau linéaire rangées dans l'ordre décroissant des modules. Expliquer ce résultat.

d. Réaliser une ACP à noyau Gaussien RBF en choisissant la valeur par défaut du paramètre Gamma. Comparer avec Scikitlearn.

e. Que pouvez-vous conclure quant à l'intérêt d'un ACP à noyau ?