Science des données

TP 1 – Iris

Taillieu Victor
21 154 689

Vaio Luca
21 154 698

1er octobre 2021

Table des matières

Introduction

Les concepts et techniques permettant de comprendre, visualiser et séparer des données sont multiples. À travers ce TP, nous allons explorer des données en combinant ces différents outils afin d'acquérir une bonne capacité d'analyse et une maîtrise des techniques utilisées. L'objectif étant de montrer le mieux possible la séparation entre plusieurs classes d'un jeu de données.

Celui qui est utilisé pour ce TP est Iris : https://archive.ics.uci.edu/ml/datasets/iris. Il contient 150 observations réparties en trois classes (`setosa`, `versicolor` et `virginica`) de 50 observations chacune. Les attributs décrivant ces observations sont au nombre de quatre : `sepal_length`, `sepal_width`, `petal_length` et `petal_width`. Les classes d'Iris sont relativement bien séparées dans l'espace à quatre dimensions. Nous devons donc montrer que ces classes sont en effet bien distinctes dans un espace plus réduit de une à deux dimensions.

Pour commencer, nous tenterons de présenter cette séparation en se basant sur des fonctions de séparation et des mesures de distance. Enfin, nous utiliserons une approche plus visuelle à travers des figures de nuages de points ou d'histogrammes avec ou sans transformation des données initiales.

Méthode I

Calcul de distances

Objectifs de cette méthode

Avec cette première méthode, nous cherchons à montrer la séparation entre les différentes classes des données d'Iris grâce à des fonctions de séparation. Celles-ci permettent d'obtenir des mesures quantitatives indiquant l'écart entre les observations.

Ici, nous nous concentrerons sur des mesures de distance et de qualité de classe que sont la distance euclidienne, la distance de Mahalanobis, la distance intra-classe (cohésion) et la distance inter-classe (séparation).

Notre démarche pour cette méthode est la suivante :

- Implémenter les différentes fonctions de distance
- Calculer les distances intra et inter-classe en utilisant les distances euclidienne et de Mahalanobis pour chaque paire de classe
- Tester différentes combinaisons de variables pour les calculs de distance
- Employer une autre méthode pour calculer la matrice de covariance pour la distance de Mahalanobis

Partie 1

Distances spatiales

1.1 Cohésion et séparation

Cohésion La distance intra-classe (ou cohésion) représente la distance maximale entre un objet quelconque d'une classe `A` et le centre de cette classe. Autrement dit, c'est le plus grand écart entre la moyenne de la classe et un point de celle-ci.

Séparation La distance inter-classe (ou séparation) est quant à elle définie comme étant la distance minimale entre un objet quelconque d'une classe `A` et le centre de la classe `B`. Il s'agit donc du plus petit écart entre la moyenne de la classe `B` et un point de la classe `A`.

Par déduction, si la distance intra-classe est inférieure à la distance inter-classe, alors on peut conclure que les classes en question sont bien séparées. En effet, cela signifie qu'il n'y a pas de point de la classe `A` plus proche du centre de la classe `B` que n'importe quel point de la classe `B`.

1.2 Mesures de distance

1.2.1 Distance euclidienne

La distance euclidienne entre deux vecteurs u et v est caractérisée par le carré de la différence de leurs composantes, comme le montre la formule suivante :

$$\text{dist}(u,v) = \sqrt{\sum_{k=1}^{d}(u_k - v_k)^2} \tag{1.1}$$

C'est en fait un cas particulier de la distance de Minkowski, qui est la suivante :

$$\text{dist}(u,v) = \sqrt[L]{\sum_{k=1}^{d}|u_k - v_k|^L} \tag{1.2}$$

1.2.2 Distance de Mahalanobis

La distance de Mahalanobis est quand à elle caractérisée par la formule suivante :

$$\text{dist}(u,v) = \sqrt{(u-v)^T \cdot \Sigma^{-1} \cdot (u-v)} \tag{1.3}$$

Où Σ est la matrice de variance-covariance de l'ensemble de données X :

$$\Sigma = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \mu)\cdot(x_i - \mu)^T \tag{1.4}$$

La principale différence entre ces deux mesures de distance est que la distance de Mahalanobis tient compte de la forme de dispersion des données. Par exemple elle détecte les formes allongées de type elliptique alors que la distance euclidienne se contente des formes de type sphérique.

Partie 2

Résultats

2.1 Distance euclidienne

La formule de la distance euclidienne est assez simple et son calcul est direct. Elle prend en paramètres deux observations (sous forme de vecteurs) puis retourne un nombre réel représentant la distance séparant ces deux objets.

Il y a différentes façons d'appliquer la distance euclidienne. En effet, il est possible de ne sélectionner qu'un sous-ensemble des attributs des objets. Dans notre cas, il y a quatre attributs (`sepal_length`, `sepal_width`, `petal_length` et `petal_width`). Il est donc possible de choisir n'importe quelle combinaison de ces variables.

Pour chaque paire de classe, calculons donc la distance intra-classe de la première puis la distance inter-classe entre les deux classes. Ce calcul est effectué dans les deux sens car la distance inter-classe est directionnelle (de la classe 2 à la classe 1). Le calcul de ces distances en utilisant la formule d'Euclide avec les quatre variables donne les résultats suivants :

Classe 1	Classe 2	Intra-classe	Inter-classe	Séparées
setosa	versicolor	1.25	1.99	oui
setosa	virginica	1.25	3.50	oui
versicolor	setosa	1.55	2.86	oui
versicolor	virginica	1.55	0.76	non
virginica	setosa	2.07	4.34	oui
virginica	versicolor	2.07	0.65	non

TABLE 2.1 – Distances selon la formule euclidienne.

D'après le tableau ci-dessus, les distances inter-classe sont supérieures aux distances intra-classe pour toutes les paires contenant `setosa`. Ainsi, on peut en conclure que la classe `setosa` est bien séparée des classes `versicolor` et `virginica`. Cependant, nous ne pouvons pas affirmer que les classes `versicolor` et `virginica` sont séparées en se basant sur ces résultats.

Nous pouvons aller un peu plus loin en testant les différentes combinaisons possibles de variables comme énoncé plus haut. Il n'en résulte toutefois pas de meilleurs résultats, ils sont en fait soit similaires, soit moins bons. Nous pouvons néanmoins relever un point important : toutes les combinaisons contenant la variable `petal_length` permettent de séparer la classe `setosa` des deux autres. Cet attribut est donc primordial dans la distinction de cette classe. Voici le tableau des résultats obtenus en utilisant uniquement `petal_length` dans le calcul de la mesure de distance :

Classe 1	**Classe 2**	**Intra-classe**	**Inter-classe**	**Séparées**
setosa	versicolor	0.46	1.54	oui
setosa	virginica	0.46	3.04	oui
versicolor	setosa	1.26	2.36	oui
versicolor	virginica	1.26	0.24	non
virginica	setosa	1.35	3.65	oui
virginica	versicolor	1.35	0.45	non

TABLE 2.2 – Distances euclidiennes avec `petal_length` seulement.

2.2 Distance de Mahalanobis

2.2.1 Covariance globale

La distance de Mahalanobis permet un peu plus de flexibilité dans son calcul. En plus de devoir fournir les deux objets pour lesquels la distance les séparant doit être mesurée, on doit fournir une matrice de covariance. Ainsi, nous disposons d'un certain degré de liberté dans l'élaboration de cette matrice.

Cette mesure de distance devrait être plus intéressante, pourtant après une première tentative, les résultats ne sont pas concluants. En utilisant l'ensemble des données pour calculer la matrice de covariance, toutes les classes se retrouvent non séparées selon les critères des distances intra et inter-classe vus plus haut.

La méthode n'est en fait pas la bonne. Pour que la mesure de distance soit cohérente, la matrice de covariance doit être calculée sur la classe à partir de laquelle la mesure est effectuée.

2.2.2 Covariance par classe

L'idée est donc, par exemple pour le calcul de la distance intra-classe de `setosa` et inter-classe entre `setosa` et `versicolor`, de calculer la matrice de covariance sur les données de `setosa`.

En suivant cette méthode, voici les résultats obtenus :

Classe 1	**Classe 2**	**Intra-classe**	**Inter-classe**	**Séparées**
setosa	versicolor	3.51	11.65	oui
setosa	virginica	3.51	20.94	oui
versicolor	setosa	3.53	7.31	oui
versicolor	virginica	3.53	2.32	non
virginica	setosa	3.70	11.11	oui
virginica	versicolor	3.70	1.66	non

TABLE 2.3 – Distances de Mahalanobis, covariances par classe.

Nous avons donc réussi à séparer la classe `setosa` par rapport aux autres. Ensuite, comme pour la distance euclidienne, nous avons testé différentes combinaisons de variables et les résultats sont similaires, les classes séparées de la même manière.

On constate également que la distance de Mahalanobis permet d'amplifier la distance inter-classe vis-à-vis de la distance intra-classe. Cela renforce l'aspect sépararé des classes par rapport à la distance euclidienne.

Toutefois, malgré les différentes tentatives, les deux autres classes que sont `virginica` et `versicolor` n'ont pas pu être séparées en utilisant des fonctions de séparation.

Méthode II

Visualisation des données

Objectifs de cette méthode

Avec cette deuxième méthode, nous cherchons à visualiser la séparation entre les différentes classes des données d'Iris par des figures d'histogrammes (pour une variable) ou de nuages de points (pour deux variables). Les différents graphiques permettront d'observer les groupements de classe et de voir ou non leur séparation.

Cette méthode est donc plutôt visuelle tandis que la première était quantitative. Nous travaillerons avec les variables issues du jeu de données original Iris ainsi que des transformations de ces dernière en utilisant l'analyse en composantes principales (ACP).

Notre démarche pour cette méthode a été la suivante :

- Visualiser la distribution des classes par paire selon chaque variable
- Visualiser la distribution des classes par paire selon chaque composante principale
- Visualiser les nuages de points pour toutes combinaisons de deux variables
- Visualiser les nuages de points pour toutes combinaisons de deux composantes principales

Partie 1

Visualisation univariée

1.1 Données initiales

Pour cette première visualisation univariée, nous avons produit des histogrammes représentant la distribution de chaque variable et pour chaque paire de classes, ce qui donne 4 variables $\times$ 3 combinaisons de paires de classes $=$ 12 visualisations. Nous avons choisi d'afficher ici, pour chaque paire, l'histogramme de la variable montrant le mieux la séparation entre les deux classes sélectionnées, en l'occurrence `petal_length`. Lorsque les distributions se chevauchent, les barres sont affichées l'une par-dessus l'autre.

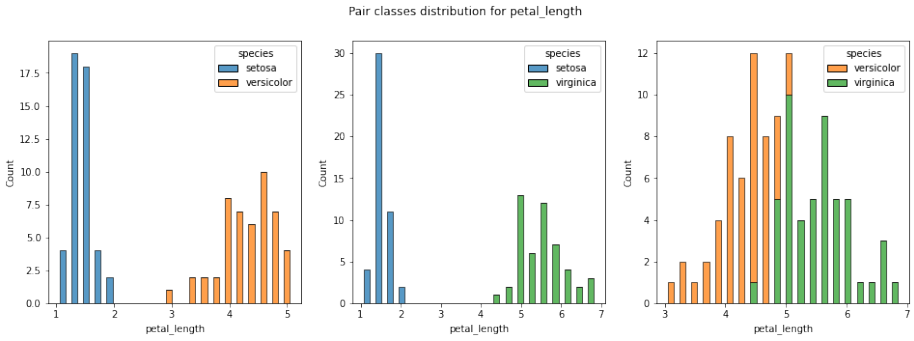

FIGURE 1.1 – Distribution des classes par paires pour `petal_length`

On constate donc, comme attendu, que la classe `setosa` (en bleu) est clairement séparée des autres classes. Par contre, les classes `versicolor` et `virginica` se chevauchent sur un ensemble assez large de données.

1.2 Données avec transformation ACP

Nous avons ensuite réalisé la même démarche en appliquant cette fois-ci une analyse en composantes principales (ACP). Ainsi, nous obtenons quatre nouvelles variables, calculées à partir des quatre attributs du jeu de données original.

1.2.1 Données initiales

À chaque composante principale est associée une valeur propre, qui représente le niveau d'explication de la variance des données par la composante correspondante. Commençons par travailler sur nos données brutes, voyons quelles sont les valeurs propres pour notre ACP.

λ_1	λ_2	λ_3	λ_4
4.23	0.24	0.08	0.02

La première valeur propre semble bien plus élevée que les autres, et en effet en voici la distribution :

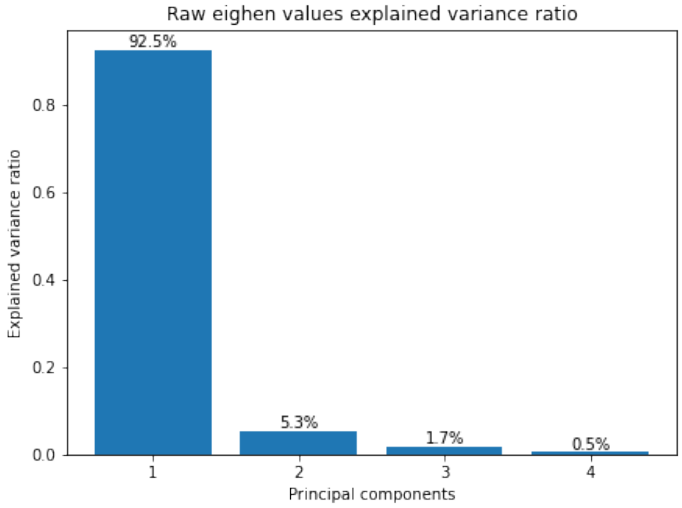

FIGURE 1.2 – Distribution des valeurs propres, données initiales

De fait, c'est bien évidemment la composante principale #1 qui permet le mieux de distinguer les paires de classes pour chaque combinaison (voir page suivante).

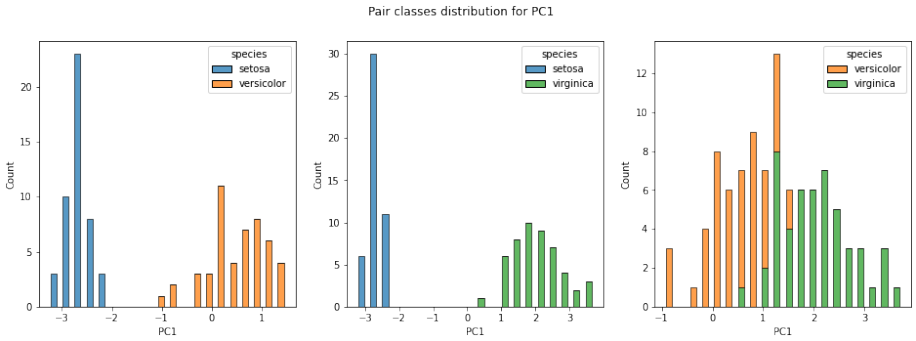

FIGURE 1.3 – Distribution des classes selon la composante principale 1

On constate à nouveau sur ce graphe que les classes `versicolor` et `virginica` se chevauchent beaucoup. L'ACP n'a donc pas permis de séparer ces deux classes.

1.2.2 Données normalisées

En procédant à l'ACP avec les données normalisées, la première valeur propre est moins prédominante, expliquant 73% de la variance seulement.

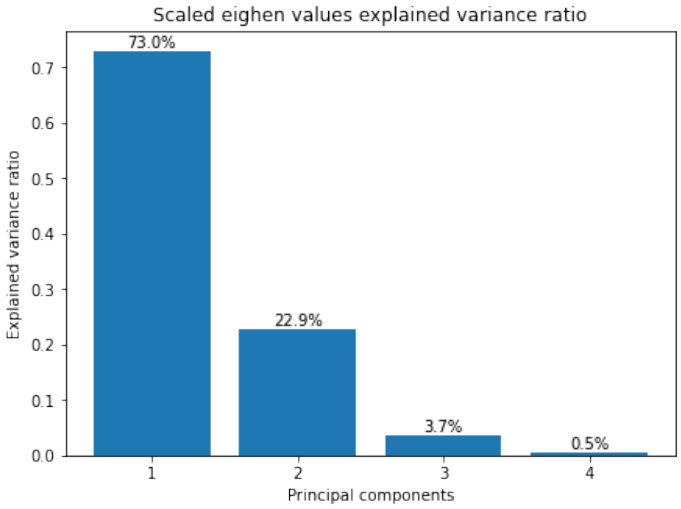

FIGURE 1.4 – Distribution des valeurs propres, données normalisées

Cependant, la normalisation des données ne nous a pas permis non plus de clairement séparer les trois classes, peu importe les composantes choisies pour la projection. C'est pourquoi nous avons ensuite tenté d'appliquer l'analyse en composantes principales autrement.

1.2.3 ACP sur chaque classe

Dans cette section, plutôt que de déterminer les composantes sur l'ensemble du jeu de données, nous allons le faire sur chacune des trois classes `setosa`, `versicolor` et `virginica`. Ensuite seulement nous projetterons l'ensemble des données sur ces composantes.

Le but de cette manipulation est de déterminer les vecteurs expliquant au mieux la variance d'une classe de données bien précise, ce qui semble finalement plus pertinent.

Nous avons donc effectué l'analyse sur toutes les classes, à la fois avec les données brutes et les données normalisées. Cela ne nous a néanmoins pas permis de séparer les classes `versicolor` et `virginica`.

Le meilleur résultat sur une dimension est obtenu sur `setosa`, avec les données normalisées, projetées sur la troisième composante.

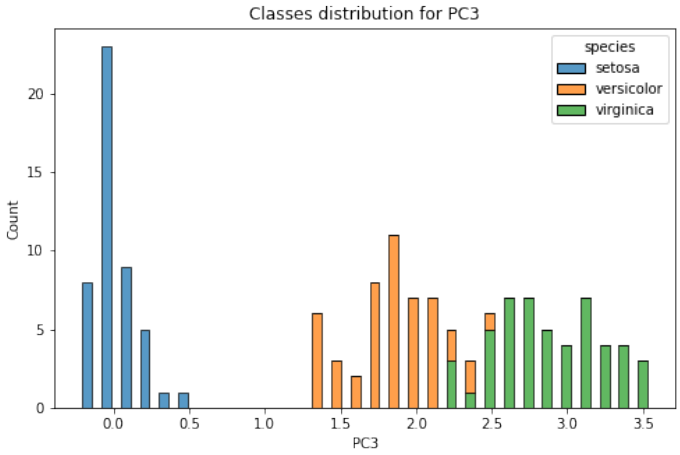

FIGURE 1.5 – ACP sur `setosa`, données normalisées

Partie 2

Visualisation bivariée

2.1 Données initiales

Visualisons maintenant les classes selon deux variables sur des nuages de points. De la même façon, nous avons un total de 12 graphiques à visualiser. La meilleure façon de comparer la distribution des données selon chaque variable est de générer une grille de paires de variables, comme suit :

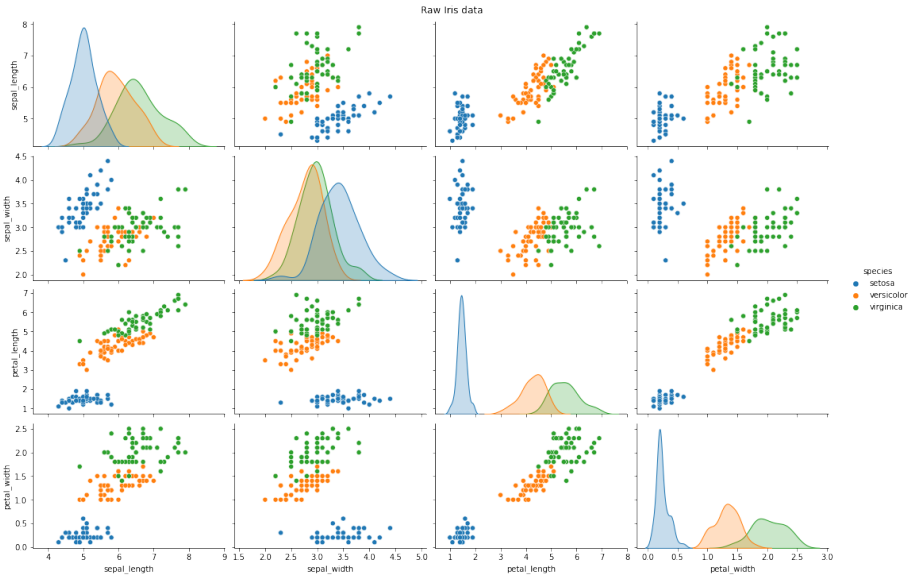

FIGURE 2.1 – Grille des données initiales selon chaque paire de variable

La meilleure visualisation est celle représentant les classes selon la longueur et la largeur de leurs pétales.

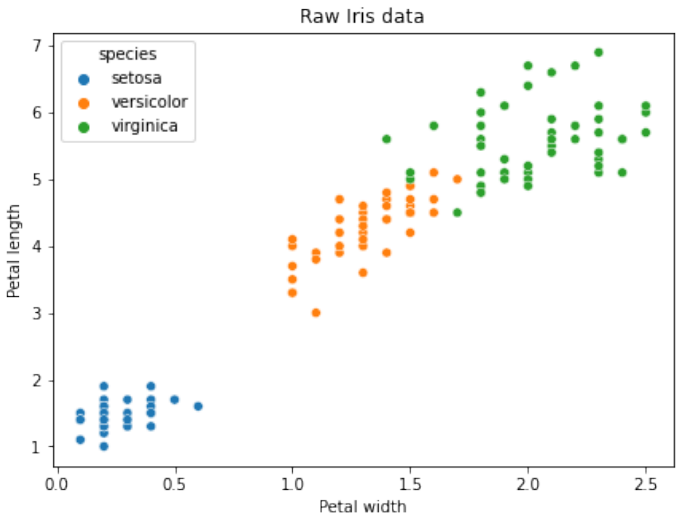

FIGURE 2.2 – Données initiales selon `petal_width` et `petal_length`

Comme précédemment, on voit clairement la classe `setosa` se détacher des deux autres. Par contre, `virginica` et `versicolor` restent légèrement superposées.

2.2 Données avec transformation ACP

Pour finir, reprenons les transformations ACP effectuées précédemment afin d'afficher les résultats selon deux composantes cette fois. Après visualisation, nous constatons que la meilleure projection se fait sur les composantes 1 et 2. Les projections avec les composantes 3 ou 4 par contre sont inintéressantes.

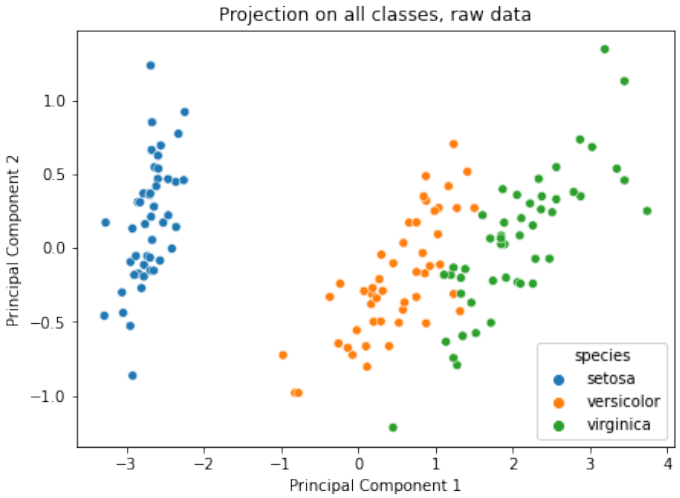

FIGURE 2.3 – Données transformées par ACP selon les composantes 1 et 2

Enfin, en appliquant l'ACP sur une seule classe, la meilleure visualisation semble être la projection sur les composantes 3 et 4, calculées avec `virginica`. Les trois classes, cependant, ne sont pas entièrement séparées.

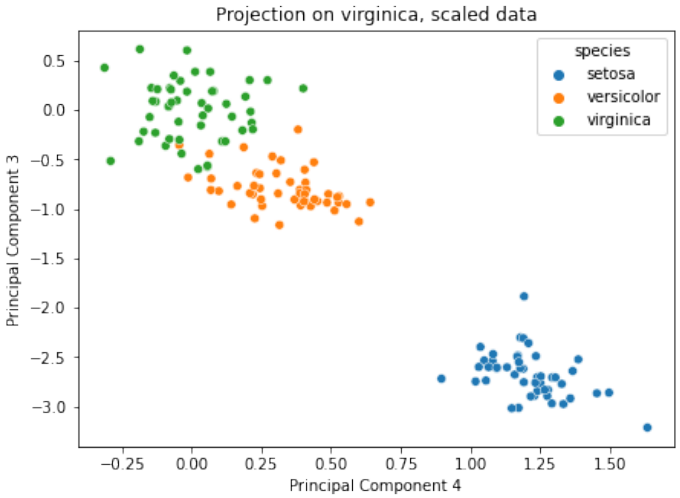

FIGURE 2.4 – ACP sur `virginica` selon les composantes 4 et 3

Conclusion

Pour conclure, nous avons vu deux méthodes permettant d'évaluer la séparation des données.

D'abord par le calcul de distances, et nous avons vu que la distance de Mahalanobis est une approche plus précise lorsque la distribution est elliptique.

Ensuite, par la visualisation, donc à une ou deux dimensions. La méthode ici est donc de projeter le jeu de données sur de nouveaux axes potentiellement plus propices à la compréhension de sa structure.

Pour finir, les différentes visualisations ne nous ont pas permis de clairement séparer les trois classes, mais ce TP nous a permis d'appréhender différentes options de transformation des données qui semblent tout de même efficaces.