

VDD TP

| Nom / Prenom | Numéro Etudiant |
|---------------|-----------------|
| Sheikh Rakib | 11502605 |
| Daudin Louise | 11606555 |

TP1 Python VDD

Sur le terminal taper les commandes suivant pour installer scikit

```
pip install -U scikit-learn scipy matplotlib  
-- Pour python 3  
pip3 install -U scikit-learn scipy matplotlib
```

Ensuite faire dans le terminal python :

```
from sklearn import *  
import numpy  
import matplotlib.pyplot  
  
iris = datasets.load_iris()  
  
len(iris.data)  
  
print(iris.target)  
print(iris.feature_names)  
print(iris.target_names)
```

Print the number of data, name of variables, name of classes :

```
len(iris.data)  
print(iris.feature_names)  
print(iris.target_names)
```

TP-1 version fr

Importez les librairies numpy et preprocessing

```
import numpy  
import sklearn.preprocessing
```

Creer la matrice X suivante :

Référence : <https://www.programiz.com/python-programming/matrix>

```
X = numpy.matrix('1 -1 2; 2 0 0; 0, 1, -1')  
X.mean()  
X.var()
```

```
0.4444444444444444  
1.1358024691358024
```

```
x_scaled = preprocessing.scale(X)  
print(x_scaled)
```

```
[[ 0.          -1.22474487  1.33630621]  
 [ 1.22474487  0.          -0.26726124]  
 [-1.22474487  1.22474487 -1.06904497]]
```

On constate que la fonction scale ajoute des valeurs décimale au hasard.

4. Calcule de la moyenne et de la variance de la matrice X Normalisé :

```
X_scaled.mean()  
X_scaled.var()
```

```
4.9343245538895844e-17  
1.0
```

Lorsque on a normalisé la matrice, ça permet de remettre la variance à 1.

C Normalisation et reduction de dimensions

1. Créez la matrice de données X2 suivante :

```
X2 = numpy.matrix('1 -1 2; 2 0 0; 0 1 -1')
```

2. Visualisez la matrice et calculez la moyenne sur les variables

```
print(X2)  
X2.mean()
```

```
print(X2)  
[[ 1 -1  2]  
 [ 2  0  0]  
 [ 0  1 -1]]  
  
X2.mean()  
0.4444444444444444
```

3. Normalisez les données dans l'intervalle [0,1], Visualiser les données normalisées et calcule de moyenne sur les variables.

```
min_max_scaler = preprocessing.MinMaxScaler()  
X2_min_max = min_max_scaler.fit_transform(X2)
```

```
X2_min_max  
array([[0.5      , 0.        , 1.        ],  
       [1.        , 0.5      , 0.33333333],  
       [0.        , 1.        , 0.        ]])
```

4. Charger les données IRIS

```
iris = datasets.load_iris()
```

5. Afficher les données, les noms des variables et le nom des classes

```
len(iris.data)  
  
print(iris.target)  
print(iris.feature_names)  
print(iris.target_names)
```

6. Visualisez les nuages de points en 2D avec des couleurs correspondant aux classes en utilisant toutes les combinaisons de variables

```
plt.figure()  
plt.scatter(iris.data[:,0],iris.data[:,1], c = iris.target)  
plt.scatter(iris.data[:,0],iris.data[:,2], c = iris.target)
```

Il y a 6 manières de faire le plot (0-1 | 0-2 | 0-3 | 1-2 | 1-3 | 2-3).

8. Analysez le manuel d'aide pour ces deux fonctions (pca et lda) et appliquez-les sur la base Iris. Il faudra utiliser `pca.fit(Iris).transform(Iris)` et sauvegardez les résultats dans IrisPCA pour la PCA et IrisLDA pour la LDA

```
from sklearn.decomposition import PCA  
from sklearn.lda import LDA  
import pandas as pd  
  
pca_iris = PCA(n_components =2)  
principalCompo_iris = pca_iris.fit_transform(iris.data)  
principal_iris_df = pd.DataFrame(data = principalCompo_iris, columns = ['PComponent 1','PC2'])
```

#ça marche pas D:

TP 2

Exercice 1

```
from sklearn.datasets.mldata import fetch_mldata  
  
mnist = fetch_mldata('MNIST Original', data_home = custom_data_home)
```

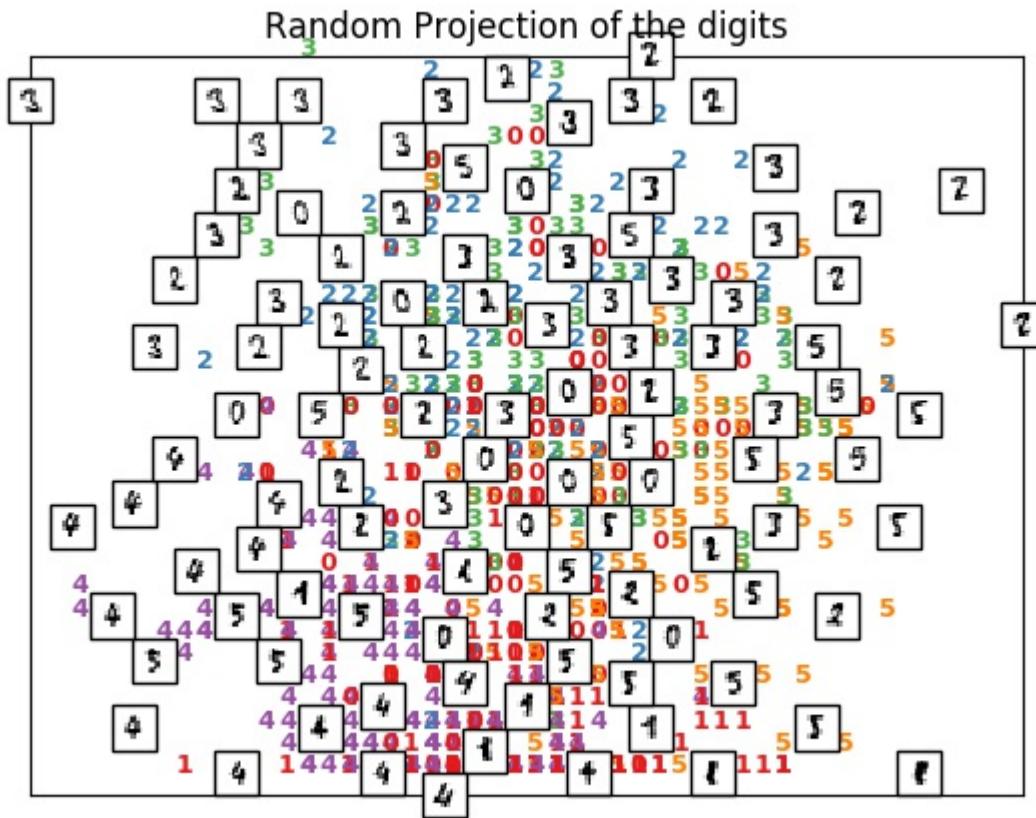
- figure 1

A selection from the 64-dimensional digits dataset

| | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 | 5 | 0 | 1 | 2 | 3 | 4 | 5 | 0 | 5 |
| 5 | 5 | 0 | 4 | 1 | 3 | 5 | 1 | 0 | 0 | 2 | 2 | 0 | 1 |
| 4 | 4 | 1 | 5 | 0 | 5 | 2 | 4 | 0 | 0 | 1 | 3 | 2 | 1 |
| 3 | 4 | 4 | 0 | 5 | 3 | 1 | 5 | 4 | 4 | 2 | 2 | 5 | 5 |
| 2 | 3 | 4 | 5 | 0 | 1 | 2 | 3 | 4 | 5 | 0 | 1 | 2 | 5 |
| 0 | 4 | 4 | 3 | 5 | 1 | 0 | 0 | 2 | 2 | 1 | 0 | 1 | 2 |
| 4 | 5 | 0 | 5 | 2 | 2 | 1 | 0 | 1 | 2 | 3 | 4 | 5 | 4 |
| 0 | 5 | 0 | 5 | 2 | 2 | 0 | 0 | 1 | 3 | 2 | 1 | 3 | 4 |
| 0 | 5 | 2 | 4 | 5 | 4 | 4 | 1 | 2 | 1 | 5 | 5 | 4 | 0 |
| 5 | 0 | 4 | 2 | 3 | 4 | 5 | 0 | 1 | 2 | 3 | 4 | 5 | 5 |
| 3 | 5 | 4 | 1 | 0 | 0 | 2 | 2 | 2 | 0 | 1 | 2 | 3 | 3 |
| 5 | 2 | 2 | 0 | 0 | 4 | 3 | 2 | 1 | 4 | 3 | 1 | 4 | 0 |
| 3 | 1 | 5 | 4 | 4 | 2 | 2 | 2 | 5 | 5 | 4 | 4 | 0 | 5 |
| 0 | 1 | 2 | 3 | 4 | 5 | 0 | 1 | 2 | 3 | 4 | 5 | 0 | 1 |
| 5 | 1 | 0 | 0 | 1 | 2 | 2 | 0 | 1 | 2 | 3 | 3 | 3 | 4 |
| 1 | 2 | 0 | 0 | 1 | 3 | 2 | 1 | 4 | 3 | 1 | 3 | 1 | 4 |
| 1 | 5 | 4 | 4 | 2 | 1 | 2 | 5 | 5 | 4 | 4 | 0 | 3 | 0 |
| 2 | 3 | 4 | 5 | 0 | 1 | 2 | 3 | 4 | 5 | 0 | 5 | 5 | 0 |
| 0 | 0 | 1 | 2 | 2 | 0 | 1 | 2 | 3 | 3 | 3 | 4 | 4 | 4 |
| 0 | 0 | 2 | 1 | 2 | 2 | 0 | 1 | 2 | 3 | 3 | 4 | 4 | 4 |
| 0 | 0 | 1 | 3 | 2 | 1 | 4 | 3 | 1 | 3 | 1 | 4 | 3 | 1 |
| 4 | 4 | 2 | 1 | 2 | 5 | 5 | 4 | 4 | 0 | 0 | 1 | 2 | 3 |
| 2 | 3 | 4 | 5 | 0 | 1 | 2 | 3 | 4 | 5 | 0 | 5 | 5 | 4 |
| 0 | 0 | 2 | 1 | 2 | 2 | 0 | 1 | 2 | 3 | 3 | 4 | 4 | 4 |
| 0 | 0 | 1 | 3 | 2 | 1 | 4 | 3 | 1 | 3 | 1 | 4 | 3 | 1 |
| 4 | 4 | 2 | 1 | 2 | 5 | 5 | 4 | 4 | 0 | 0 | 1 | 2 | 3 |

On a ici une représentation sous forme de matrice 20x20 des chiffres choisis aléatoirement dans une base de dimension 64.

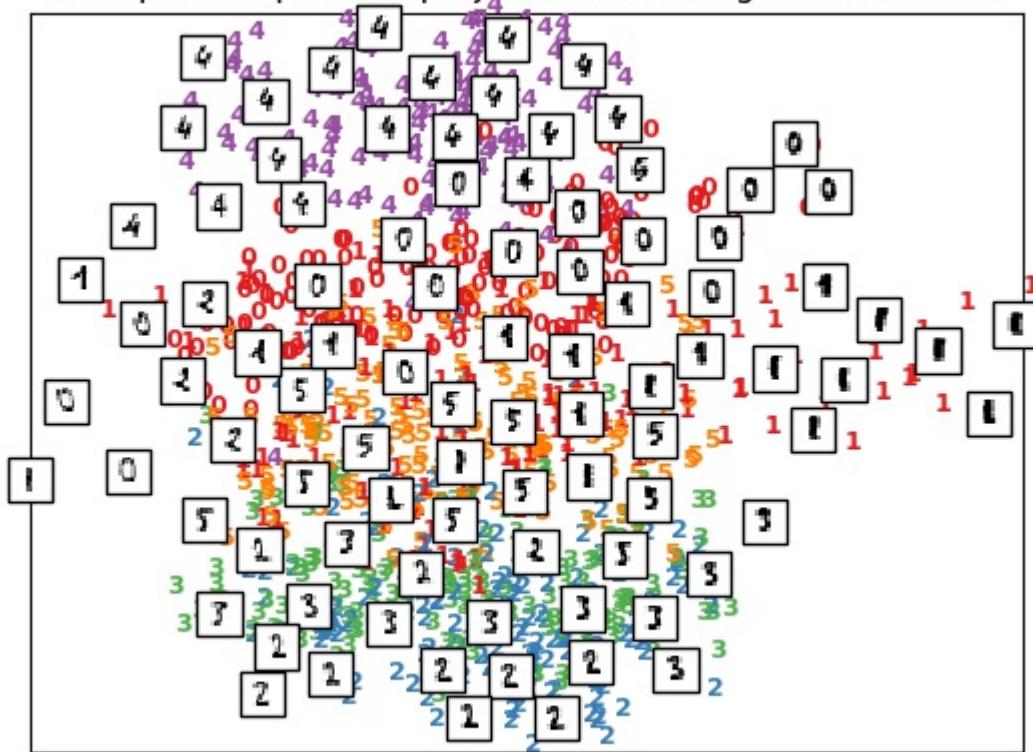
- figure 2



Nous avons ici une projection des points aléatoires des nombres.

- Figure 3

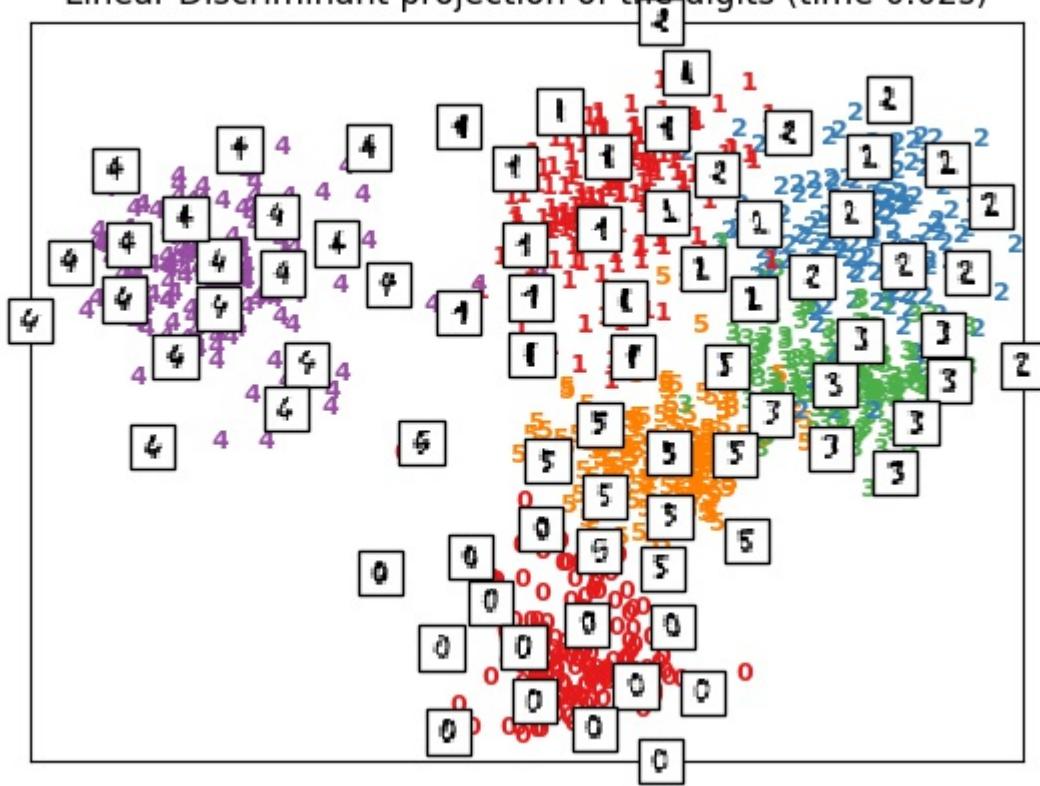
Principal Components projection of the digits (time 0.01s)



Nous avons ici un ensemble principal des projections des nombres. On remarque que les points sont un peu près réunis par bloc mais pas totalement.

- Figure 4

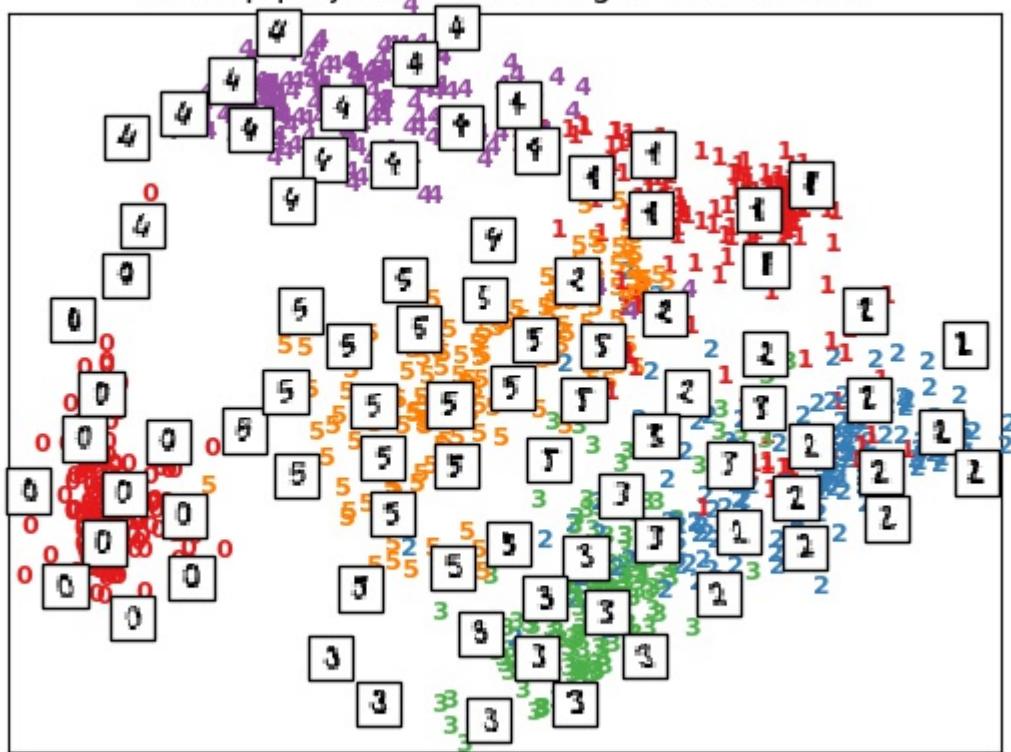
Linear Discriminant projection of the digits (time 0.02s)



On voit déjà qu'il y a une meilleure représentation des points des nombres, en effet, il sont regroupé par une projection linéaire discriminatoire.

- Figure 5

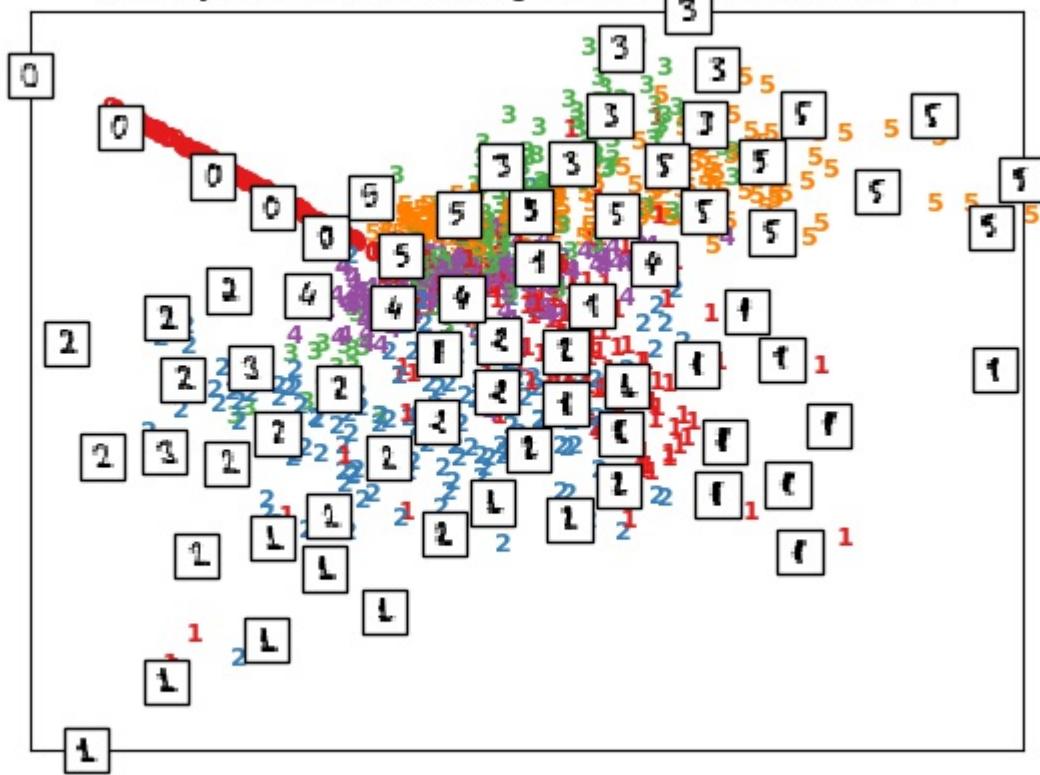
Isomap projection of the digits (time 1.55s)



La projection isomap à réussi à regrouper les points de nombres mais ce n'est pas la meilleure projection puisqu'il y a des petites quantités des nombres qui sont mélangées avec d'autres.

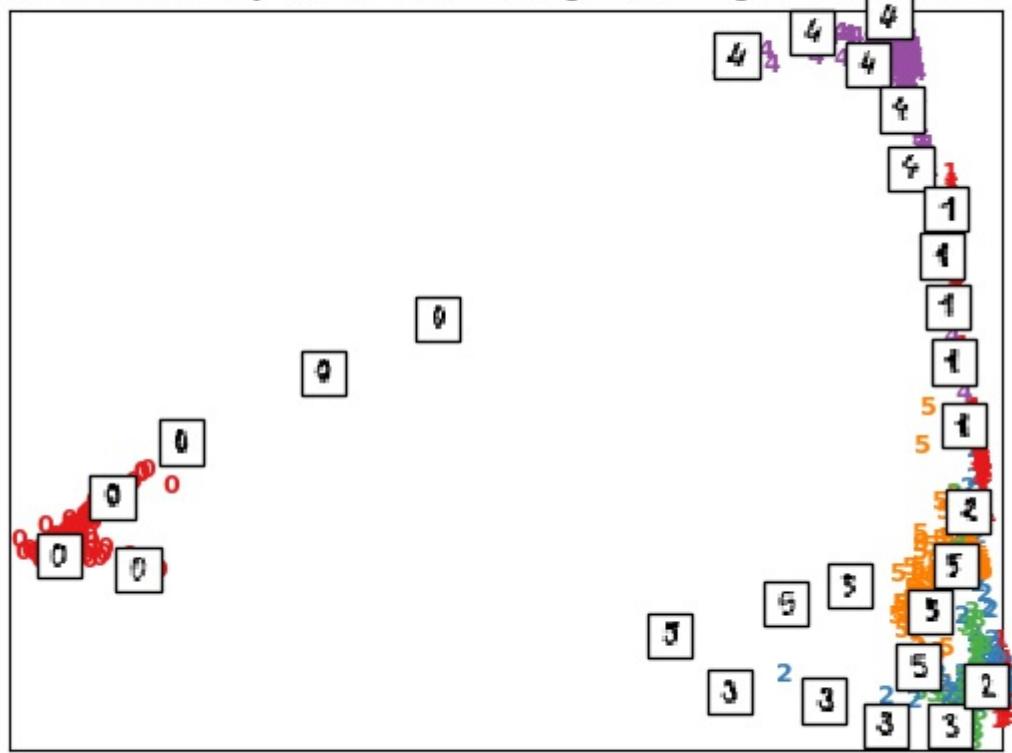
- Figure 6 (Intégration linéaire des nombres locaux)

Locally Linear Embedding of the digits (time 0.54s)

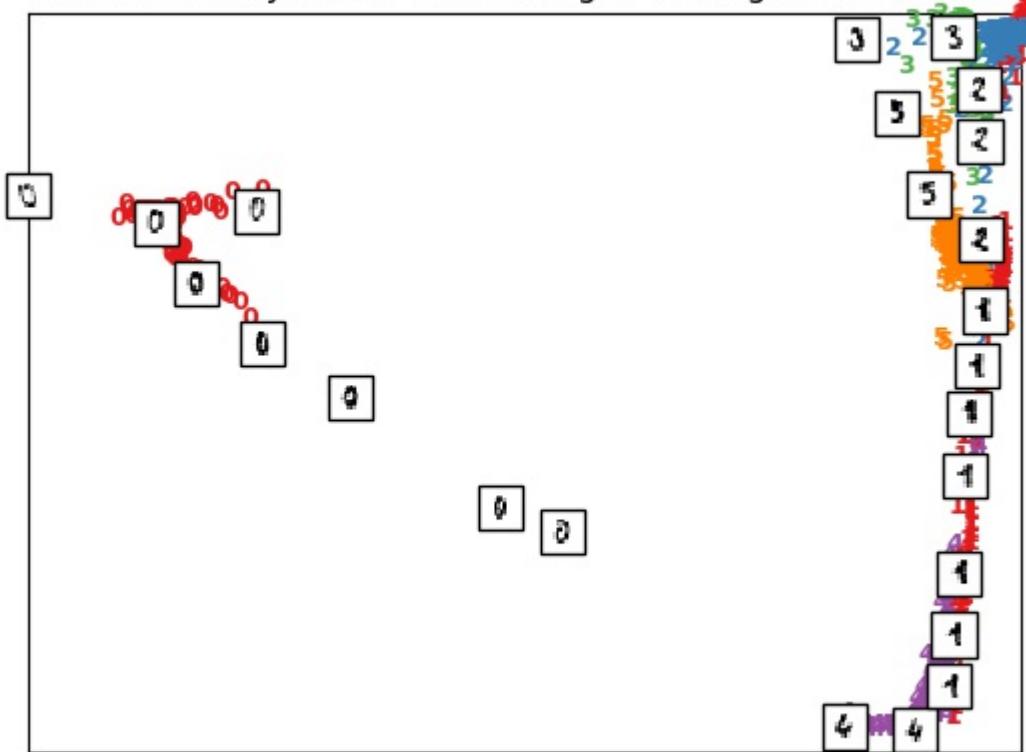


On a ici une intégration linéaire, mais

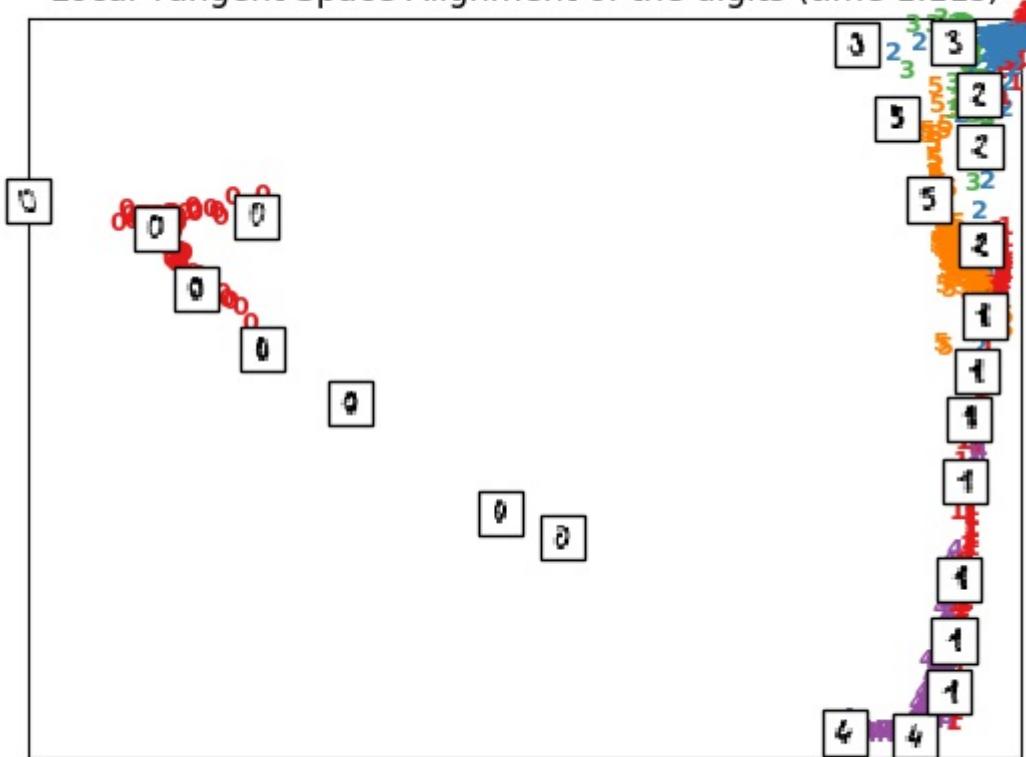
Modified Locally Linear Embedding of the digits (time 1.23s)



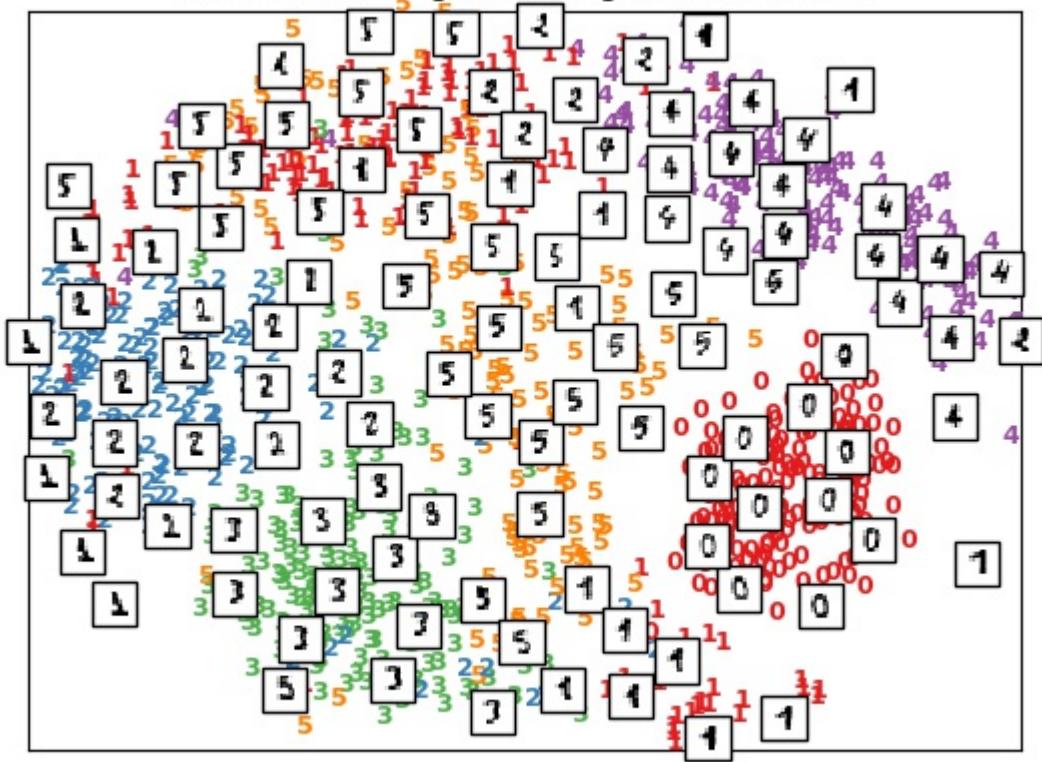
Hessian Locally Linear Embedding of the digits (time 1.45s)



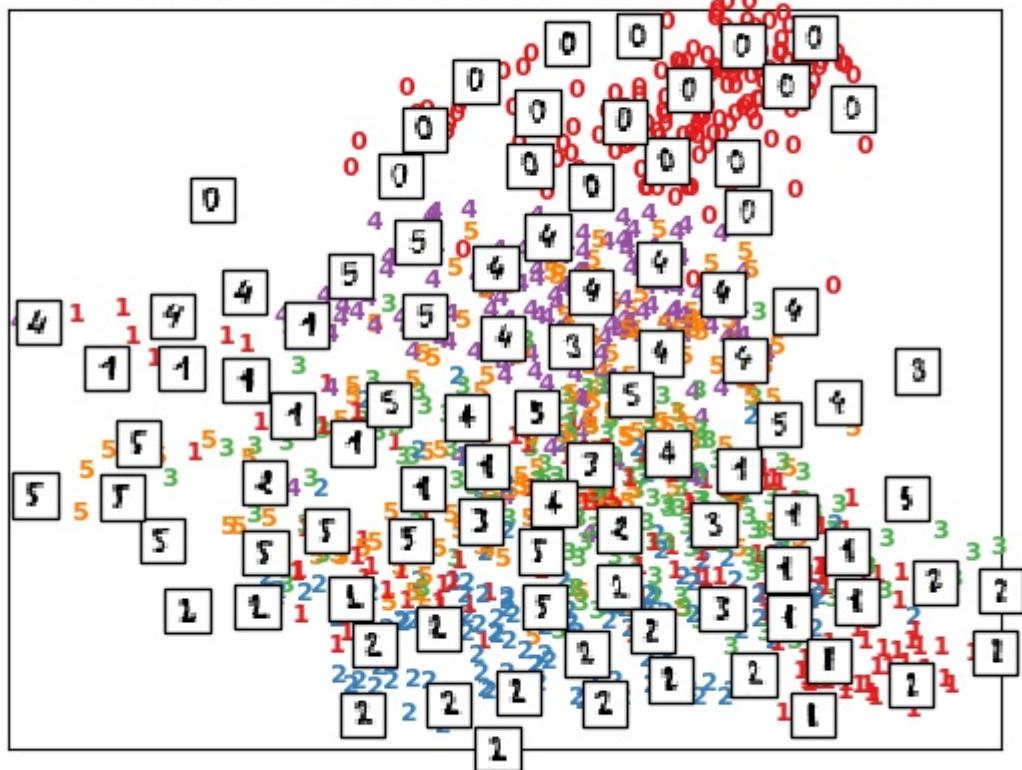
Local Tangent Space Alignment of the digits (time 1.11s)



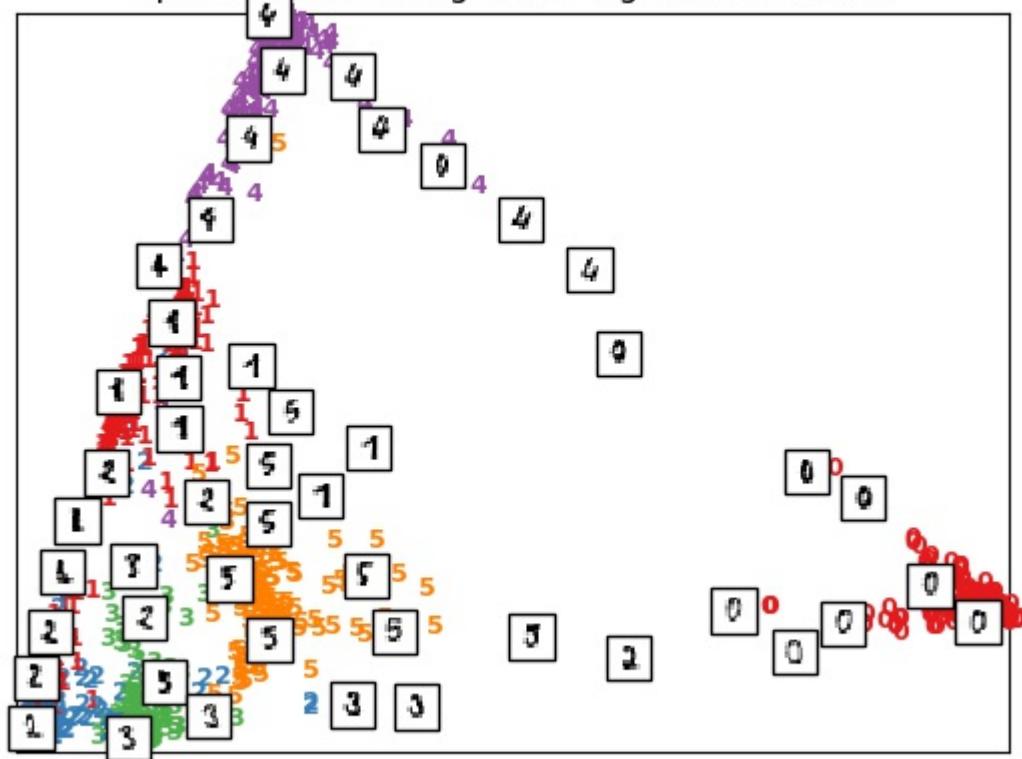
MDS embedding of the digits (time 2.65s)



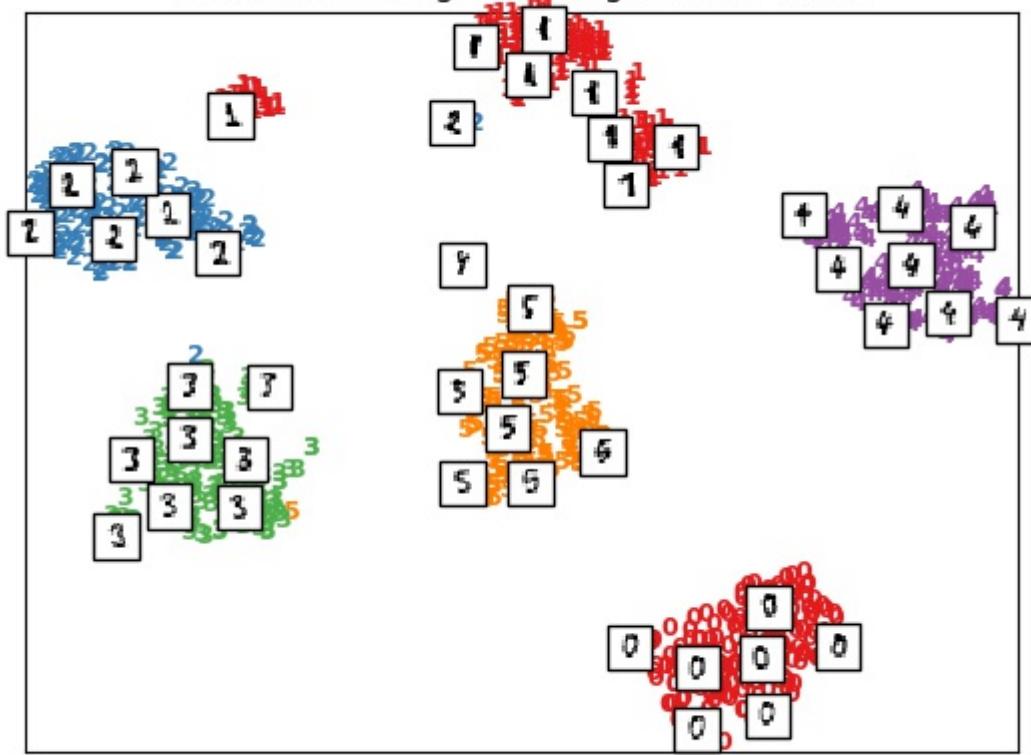
Random forest embedding of the digits (time 0.37s)



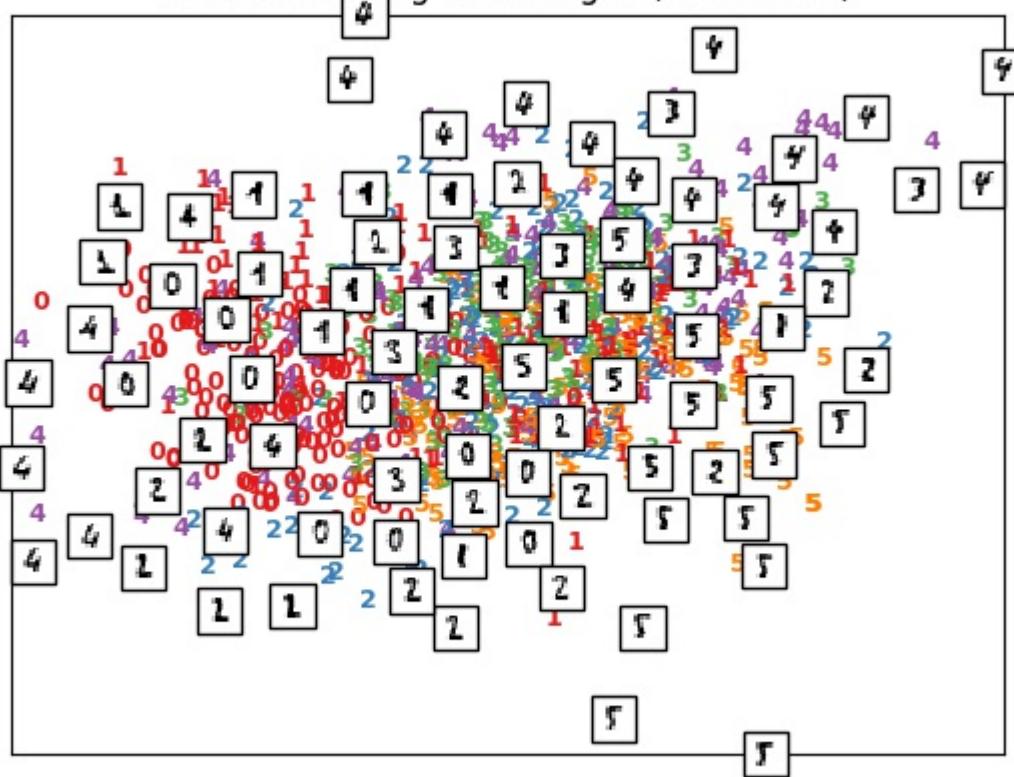
Spectral embedding of the digits (time 0.56s)



t-SNE embedding of the digits (time 4.57s)



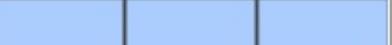
NCA embedding of the digits (time 5.36s)



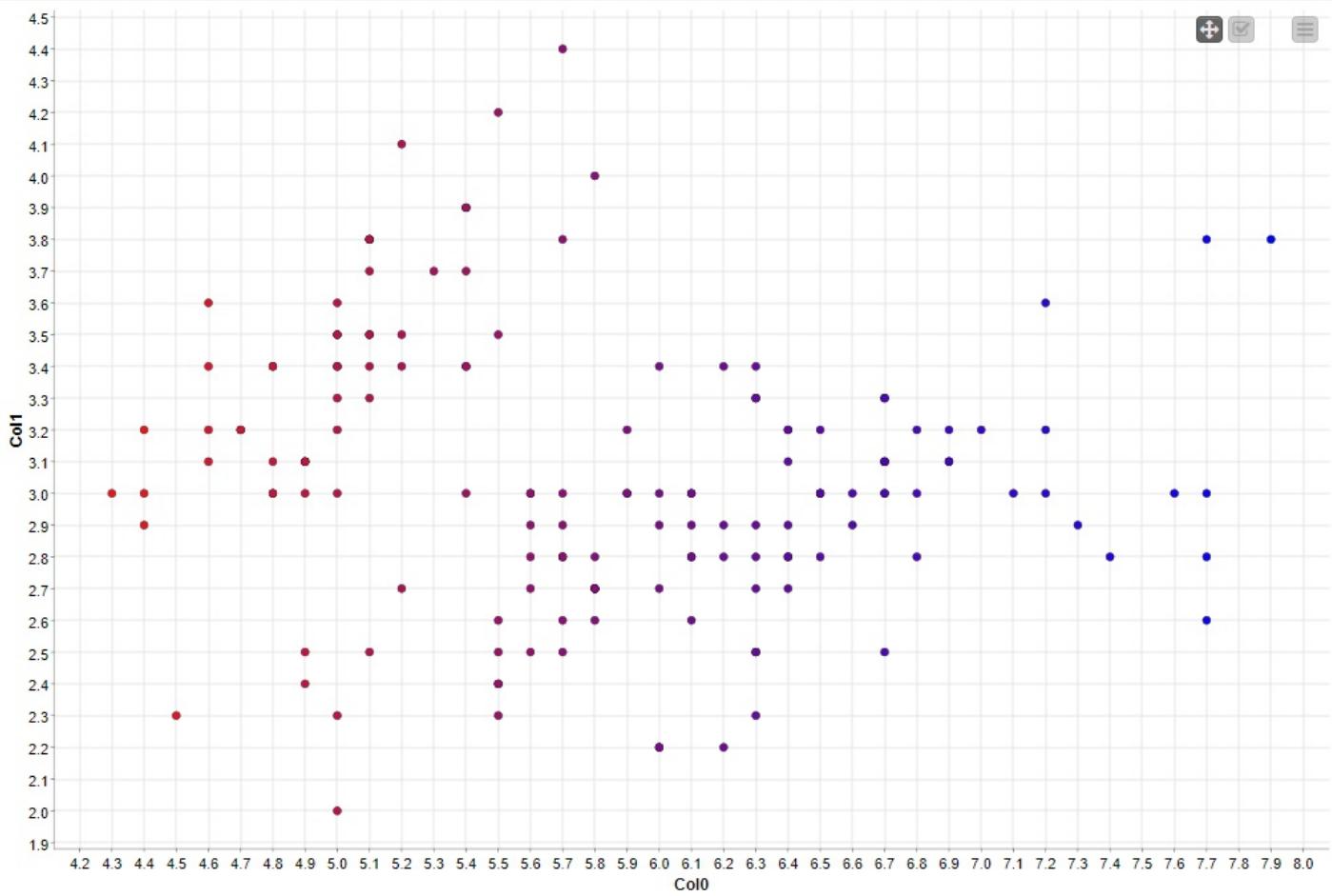
Partie 2

Exercice 1

1. Download the Knime software with all the packages v3.0.0
2. Load the Iris dataset using the 'File Reader' node in a knime project and tests use several nodes as Statistics and Plot to make a small analysis of the dataset.
 - histogramme du noeud "Statistics"

| Row ID | S Column | I No. mis... | Histogram |
|--------|----------|--------------|--|
| Col0 | Col0 | 0 |  |
| Col1 | Col1 | 0 |  |
| Col2 | Col2 | 0 |  |
| Col3 | Col3 | 0 |  |
| Col4 | Col4 | 0 |  |

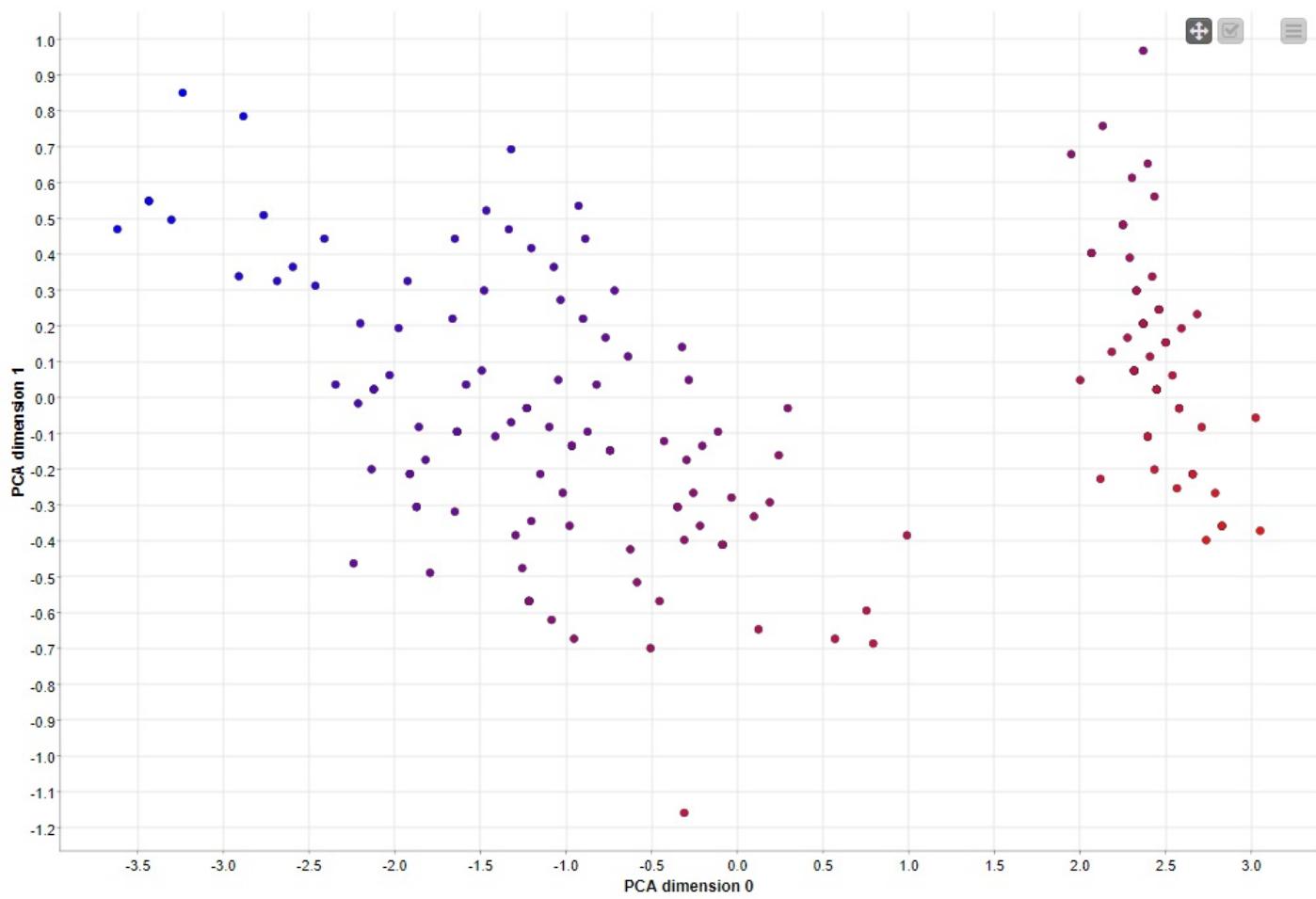
- Nuage de points



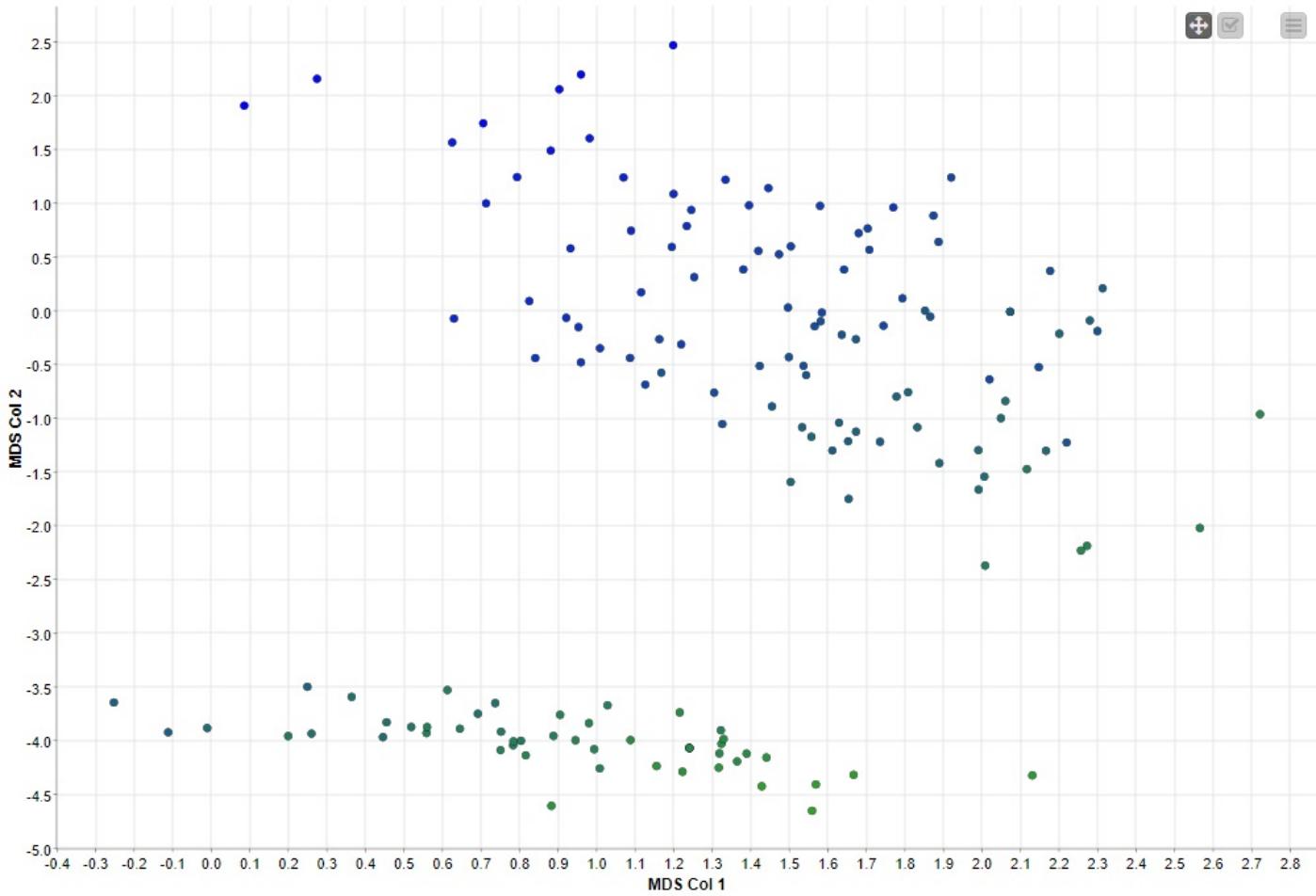
3. Compute the Correlation using the 'Linear Correlation' node and reduce the dimension by using the 'Correlation Filter'. Plot the data using the selected variables.

AF

4. Use the PCA and MDS dimensional reduction techniques to reduce the dimension of the data to 2 variables and plot the results. Make an analysis between the result in question 3 and those obtained here.
- Voici le résultat pour le PCA



- Voici le résultat pour le MDS



On peut constater que la méthode MDS regroupe mieux certains point (notamment ceux proche de l'Axe X) par rapport a la méthode PCA.

Exercice 2

- **Reading full small data set** : Read large data set from DB contained nodes
- **Target Selection** : Selection: Allow to rename some columns and then select or them.
- **Baseline Evaluation** : Choose the best algorithm between three classification algorithms: MLP, decision tree and Naïve Bayes.
- **Reduction based on LDA** : Reduces the number of columns in the input data by linear discriminant analysis
- **Auto Encoder based Reduction** :
- **Reduction based on t-SNE** : Create a probability distribution capturing the relationships between points in the high dimensional space and find a low dimensional space that resembles the probability dimension as well as possible
- **Reduction based on High Corr.** : Identifies pairs of columns with a high correlation (i.e. greater than a given threshold), and removes one of the two columns for each identified pair.
- **Tree Ensemble based Reduction** : Generate a large and carefully constructed set of trees against a target attribute and then use each attribute's usage statistics to find the most informative subset of features.
- **Reduction based on PCA** : Reduce all values to those which are the most accurate for the PCA method.
- **Row Sampling** :
- **Column Splitter** : This node splits the columns of the input table into two output tables.
- **Column Selection by Missing Values** : Remove columns with excessive values.
- **Column Appender** : Takes two tables and quickly combines them by appending the columns of the second table to the first table.
- **Backward Feature Elimination** : Elimine une par une les valeurs les moins pertinente en les comparants 2 par 2.
- **Forward Feature Selection** : Elimine tout et récupère une par une les valeurs pertinente en les comparants 2 par 2.
- **Joiner** : This node joins two tables in a database-like way.
- **ROC Curve** : This node draws ROC curves for two-class classification problems.
- **Positive class probabilities** : Select the value from the class column that stands for the "positive" class
- **Accuracies** :
- **Bar Chart** : Visualizes one or more aggregated metrics for different data partitions with rectangular bars where the heights are proportional to the metric values.

Il y a 233 colonnes et 50 000 lignes

Du a manque de ressource nous n'avons pas pu executer le workflow, il nous est impossible d'analyser les résultats.

TP 3 Classification non supervisée avec l'approche k-means

Exercice 1

- Ouvrez le module «k-means» sur Knime et étudiez toutes les options proposées.

Ce nœud produit les centres de clusters pour un nombre prédéfini de clusters (pas de nombre dynamique de clusters). K-means effectue une mise en cluster rigoureuse qui attribue un vecteur de données à exactement une cluster. L'algorithme se termine lorsque les affectations de clusters ne changent plus.

L'algorithme de mise en cluster utilise la distance Euclidienne sur les attributs sélectionnés. Les données ne sont pas normalisées par le nœud

Nombre de clusters : Le nombre de clusters (centres de clusters) à créer.

Centroid initialisation :

- First k row : Initialise les centroïdes en utilisant les premières lignes du tableau d'entrée.
- Random initialization : initialise les centroïdes à l'aide des lignes aléatoires du tableau d'entrée.
- La case Use static random seed permet d'avoir des résultats qui sont reproductibles.

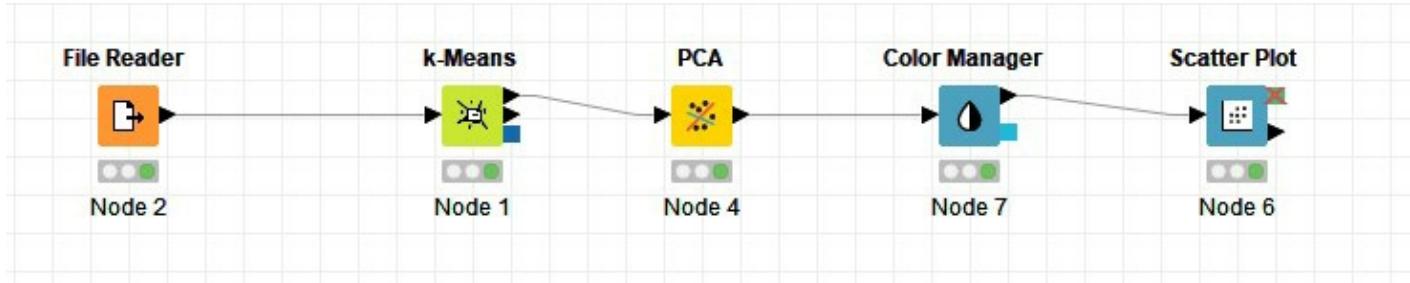
Max number of iterations : Le nombre maximum d'itérations après lequel l'algorithme se termine s'il n'a pas trouvé de solution stable auparavant.

Enable Hilite Mapping : l'hilitage d'une ligne du cluster (2ème sortie) hilitera toutes les lignes de ce cluster dans le tableau d'entrée et le tableau de la 1ère sortie.

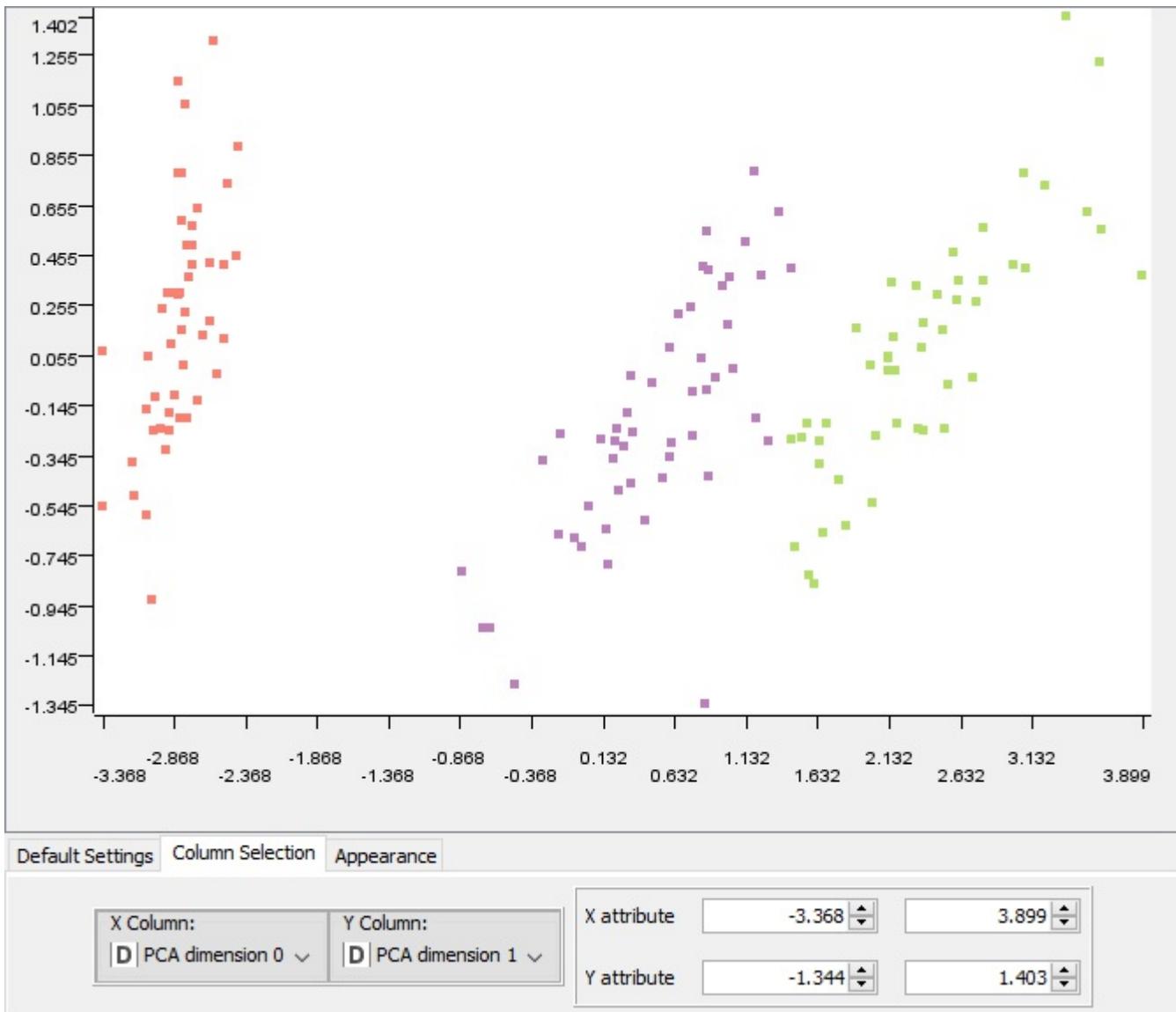
Exercice 2

Nous proposons un scenario suivant, avec une application du PCA et son affichage par scatterplot.

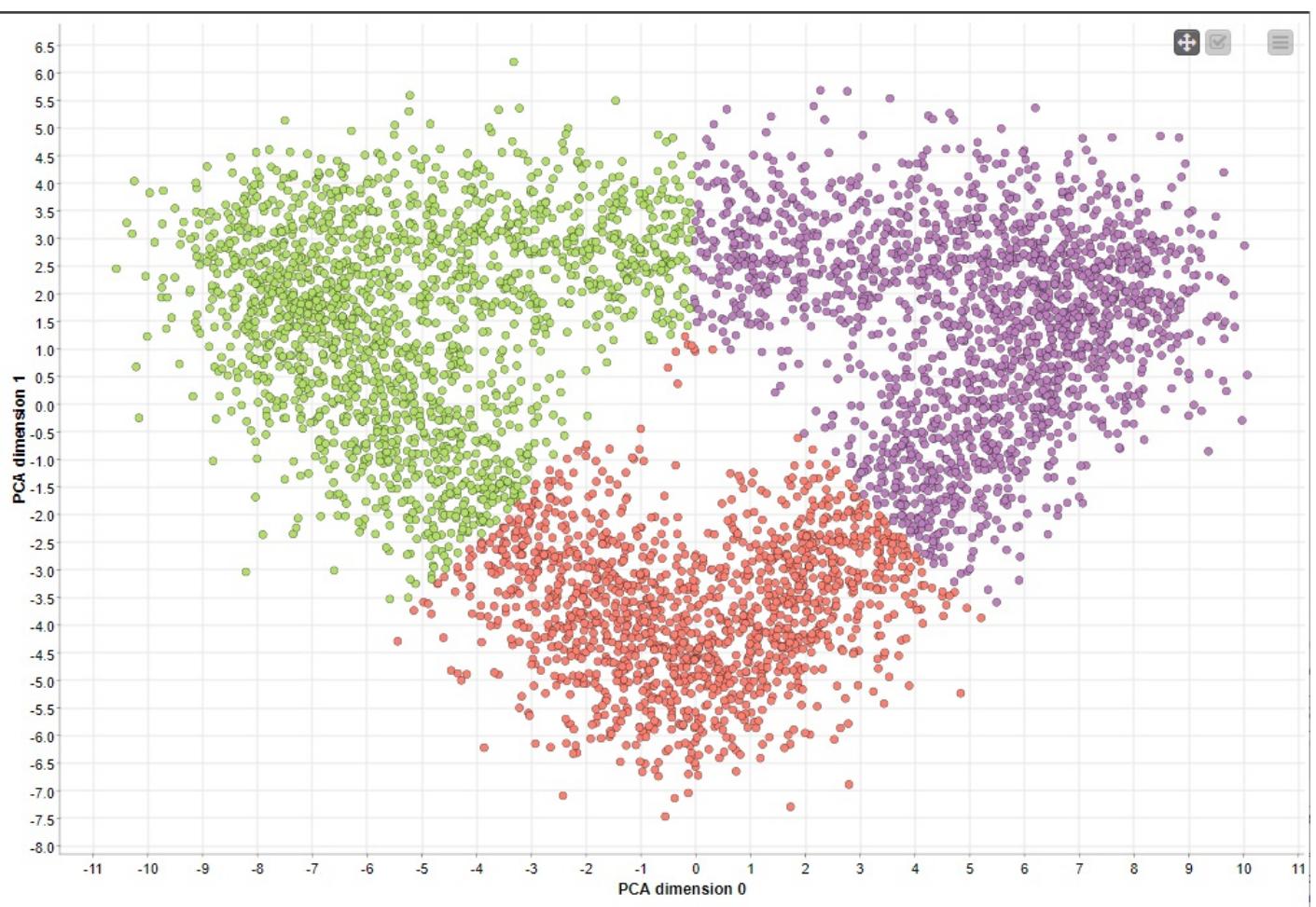
- Cluster 3



Nous obtenons alors le résultat suivant avec la base d'iris.



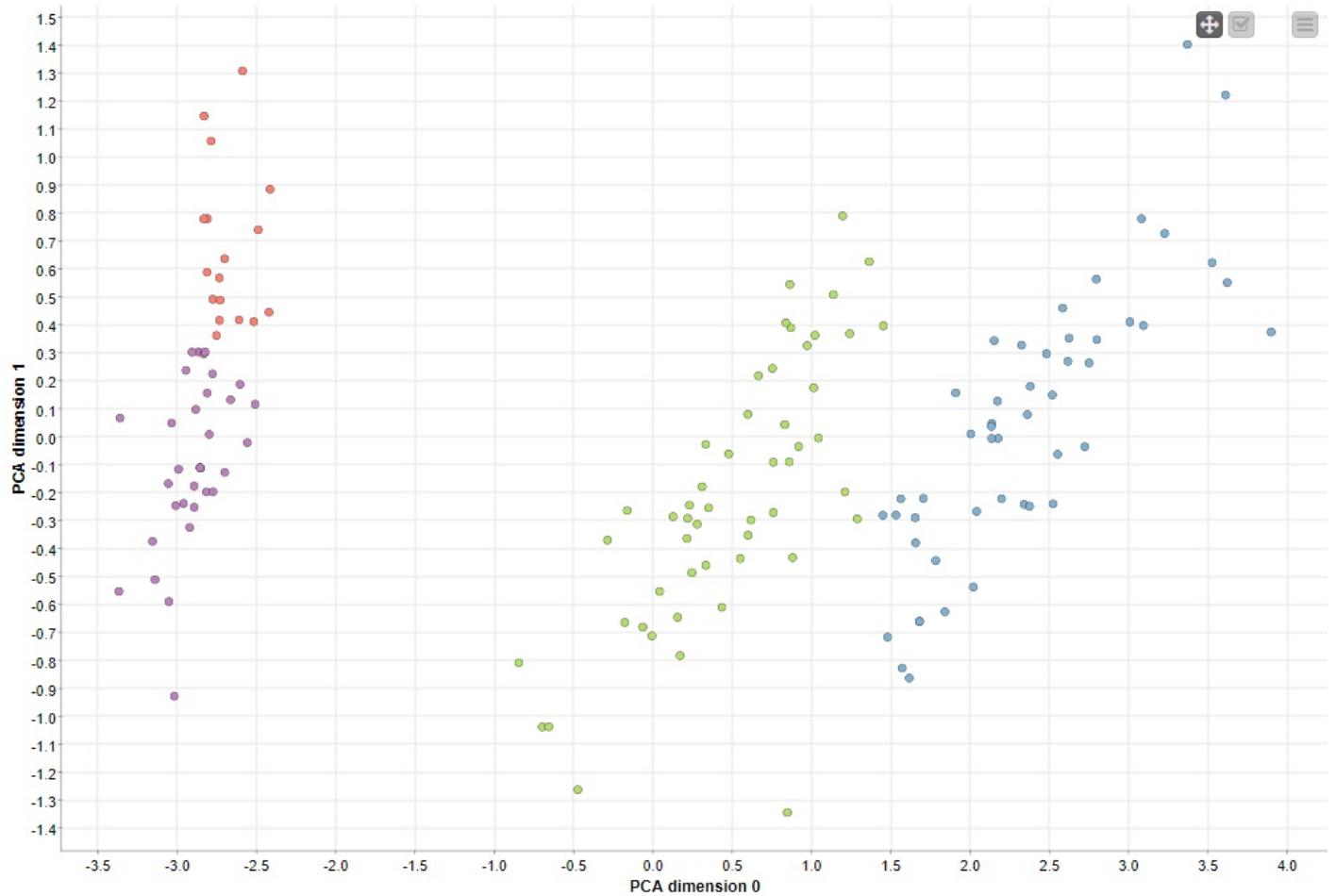
Nous obtenons alors le résultat suivant avec la base waveform



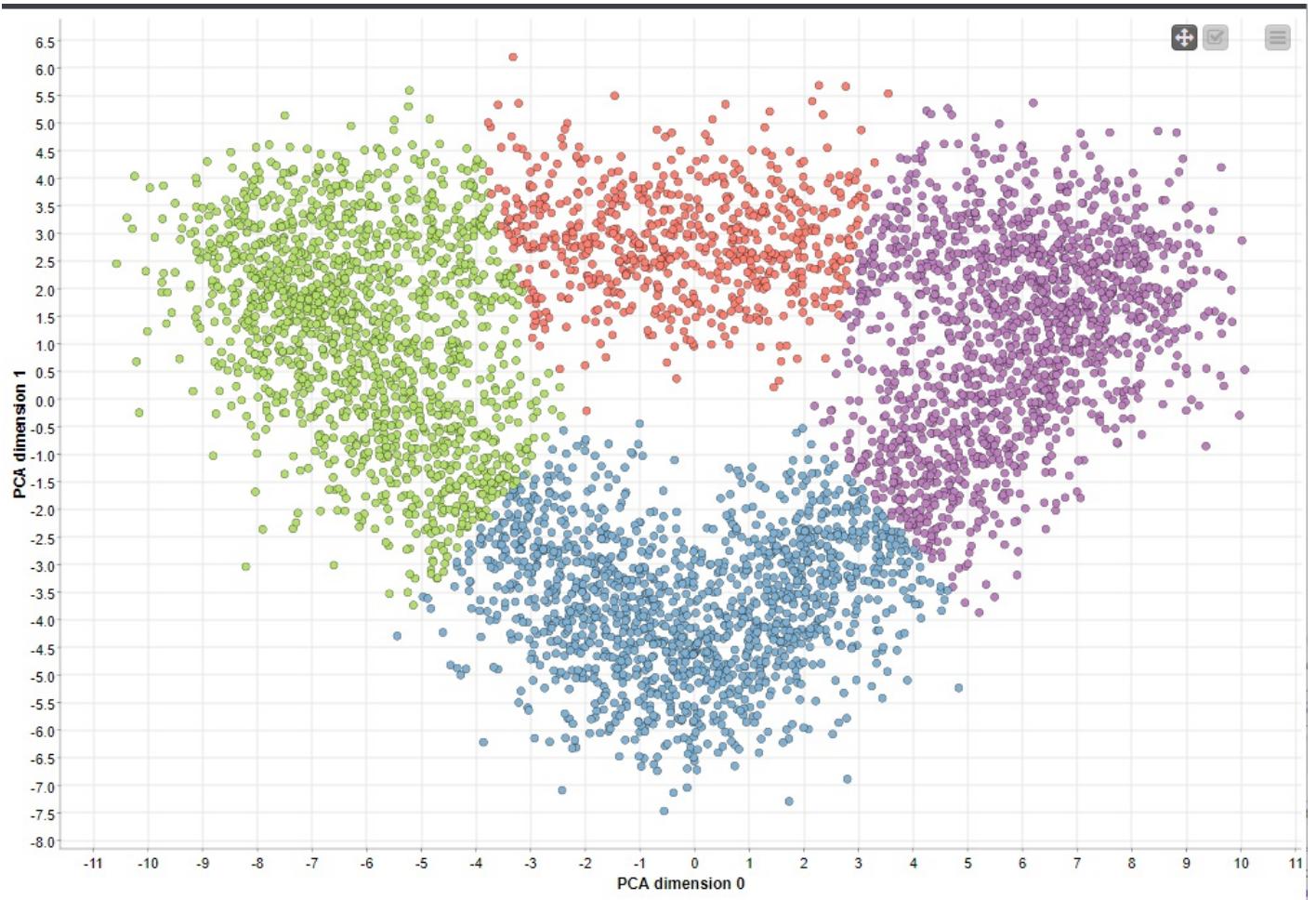
Nous remarquons que l'on obtient 3 clusters qui ne se chevauchent pas après le traitement algorithme du cluster k-means.

- Cluster 4

Nous obtenons alors le résultat suivant avec la base d'iris.



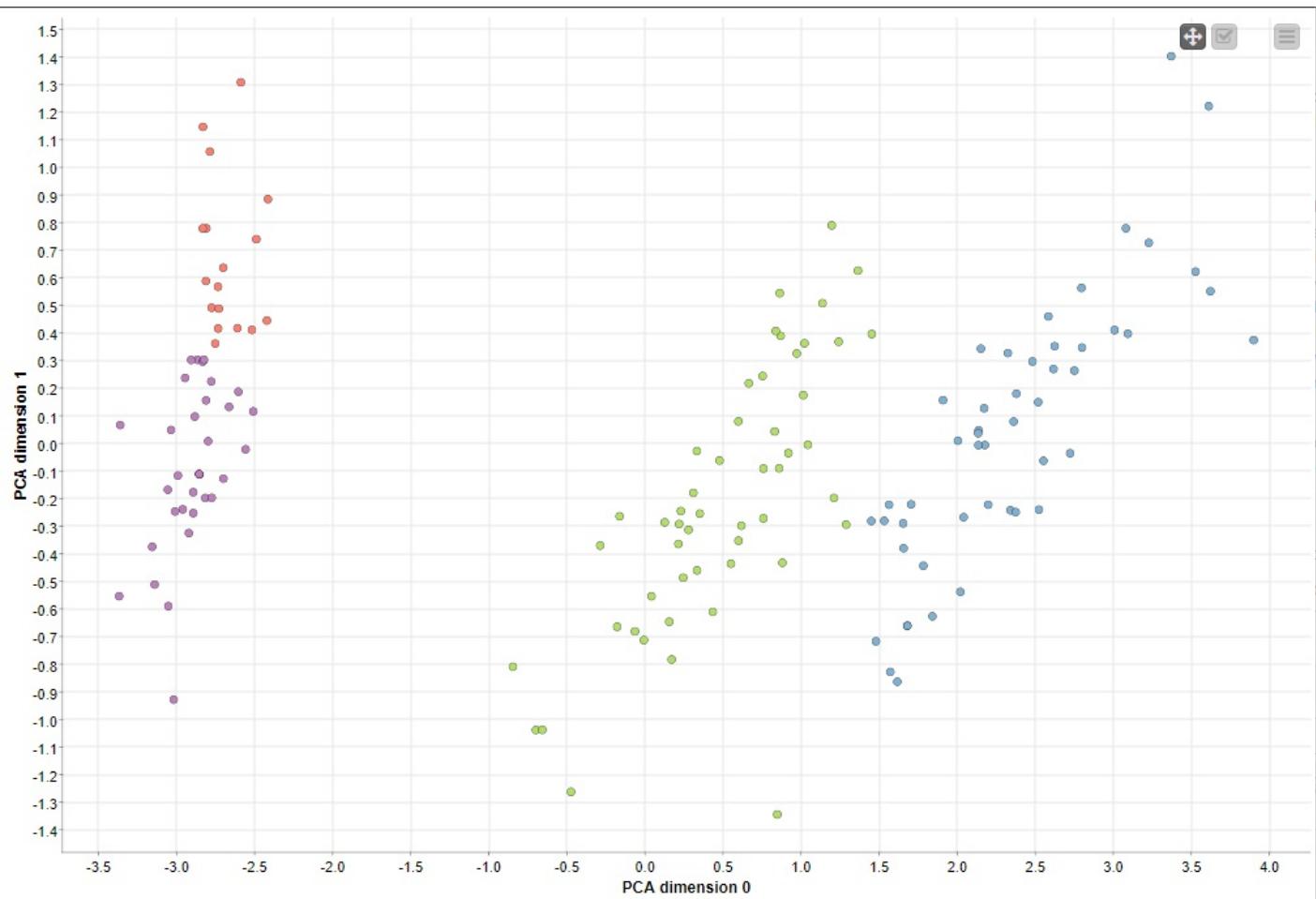
Nous obtenons alors le résultat suivant avec la base waveform



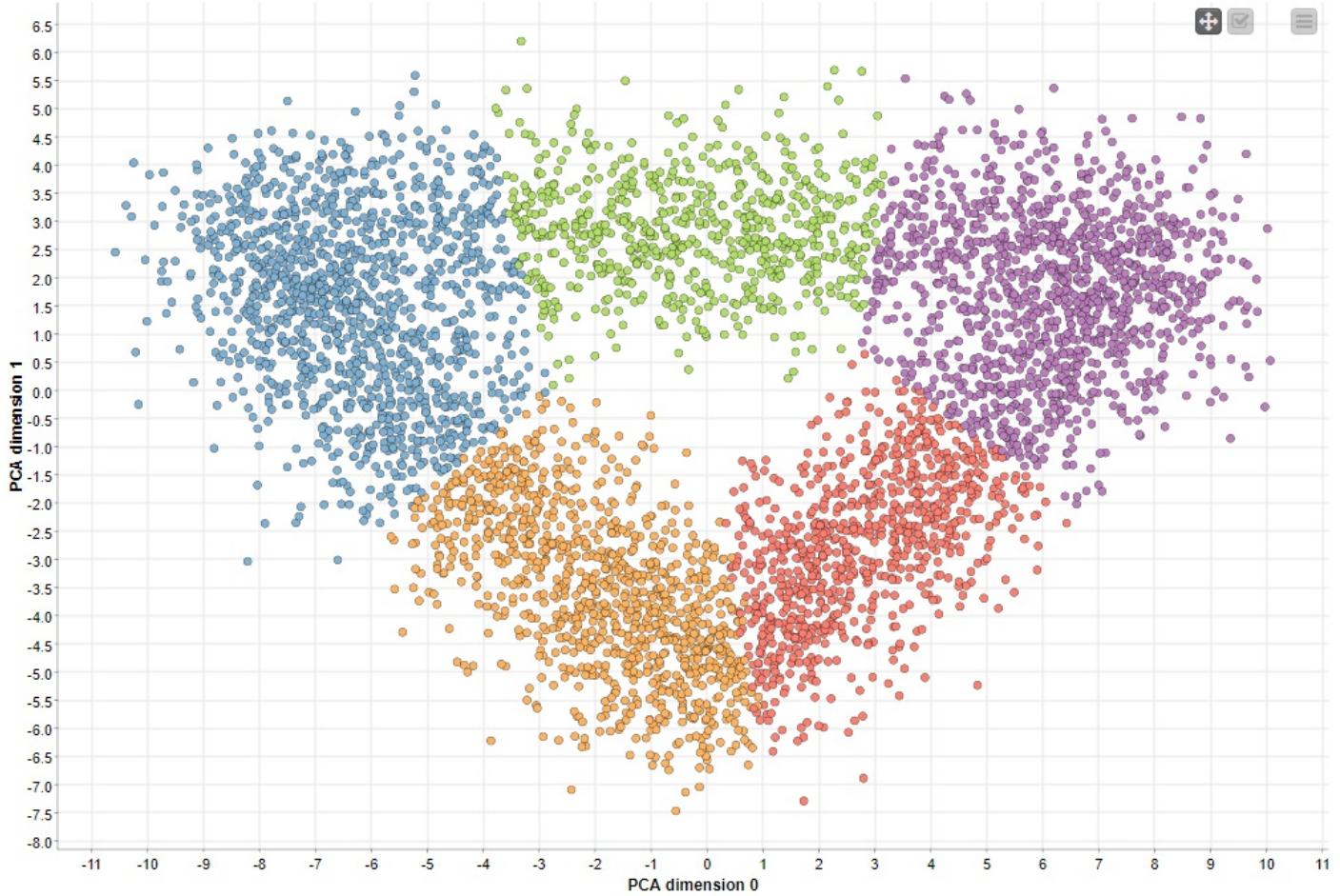
Nous remarquons que l'on obtient 4 clusters qui ne se chevauchent pas après le traitement algorithme du cluster k-means. Avec un PCA de 2 pour la représentation en scatterplot.

- Cluster 5

Nous obtenons alors le résultat suivant avec la base d'iris.



Nous obtenons alors le résultat suivant avec la base waveform



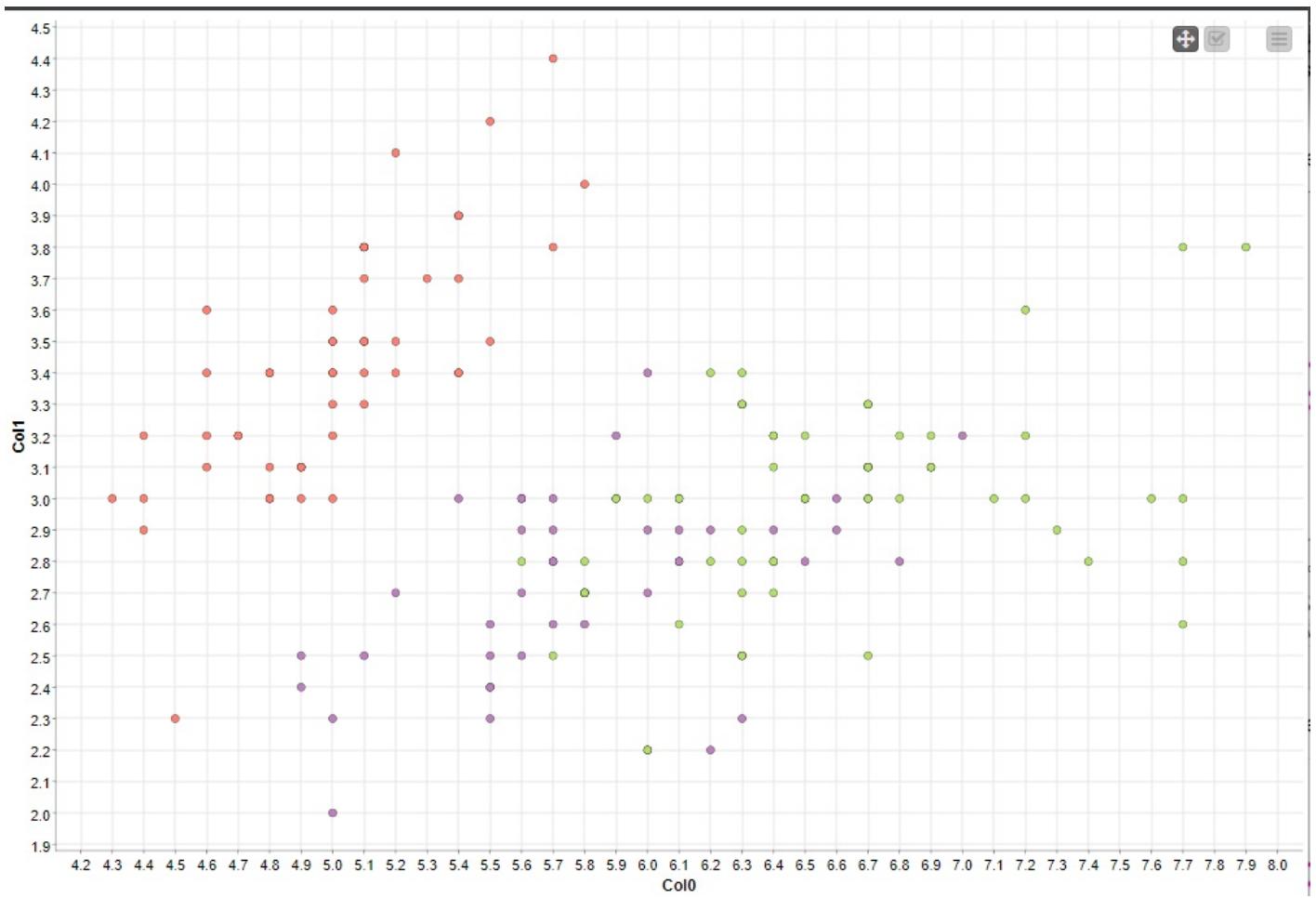
Nous remarquons que l'on obtient 5 clusters qui ne se chevauchent pas après le traitement algorithme du cluster k-means.

En ce qui concerne pour la base de données d'iris, on est limité à 4 cluster.

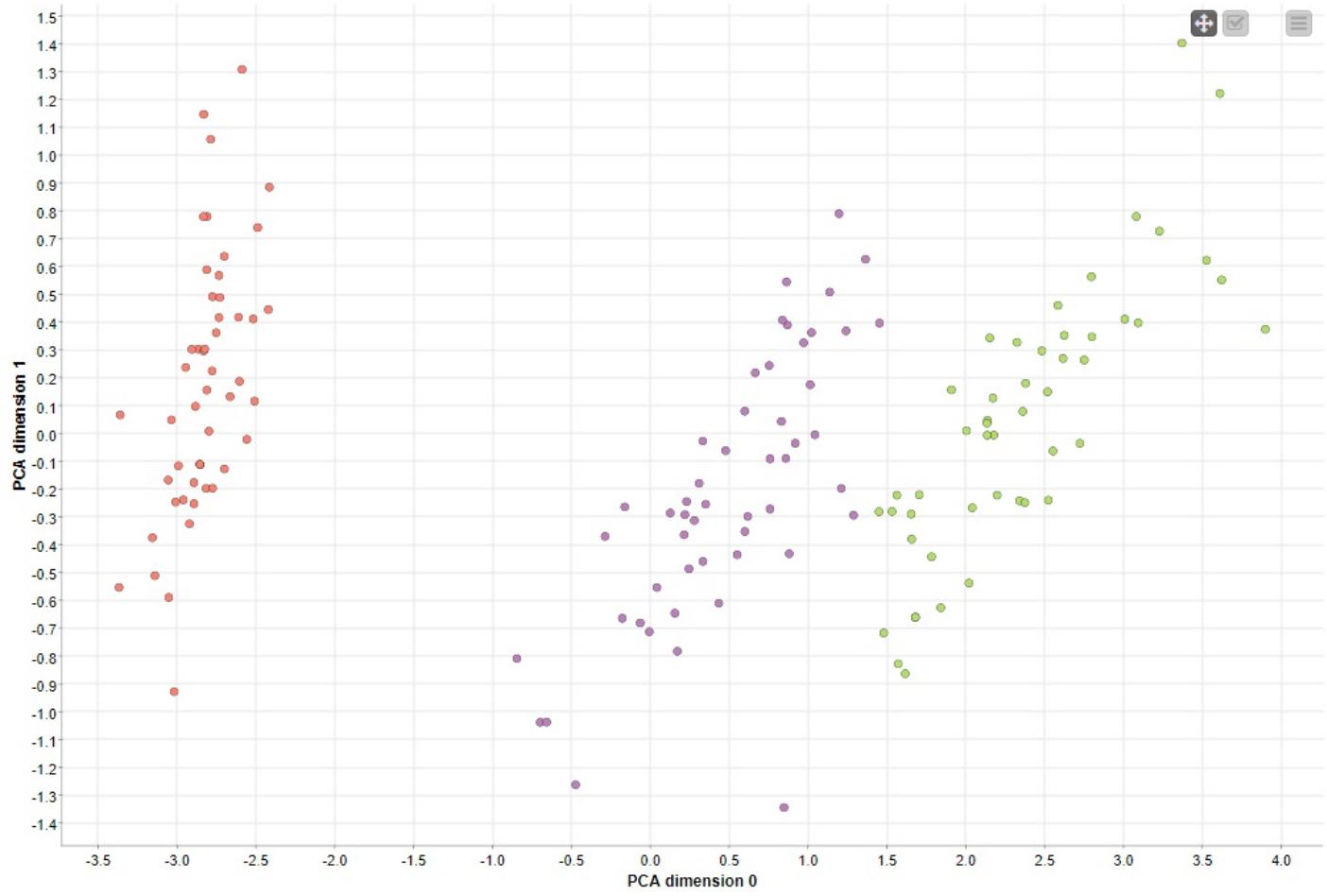
Exercice 3

Visualisez les données initiale set ensuite visualisez les résultats du clustering.Déterminez les centres des clusters résultants. Projetez les barycentres de chaque cluster sur les résultats obtenus.

Pour la base de données iris, on a les données initiales suivantes.



Méthode clustering :



Valeurs du barycentre :

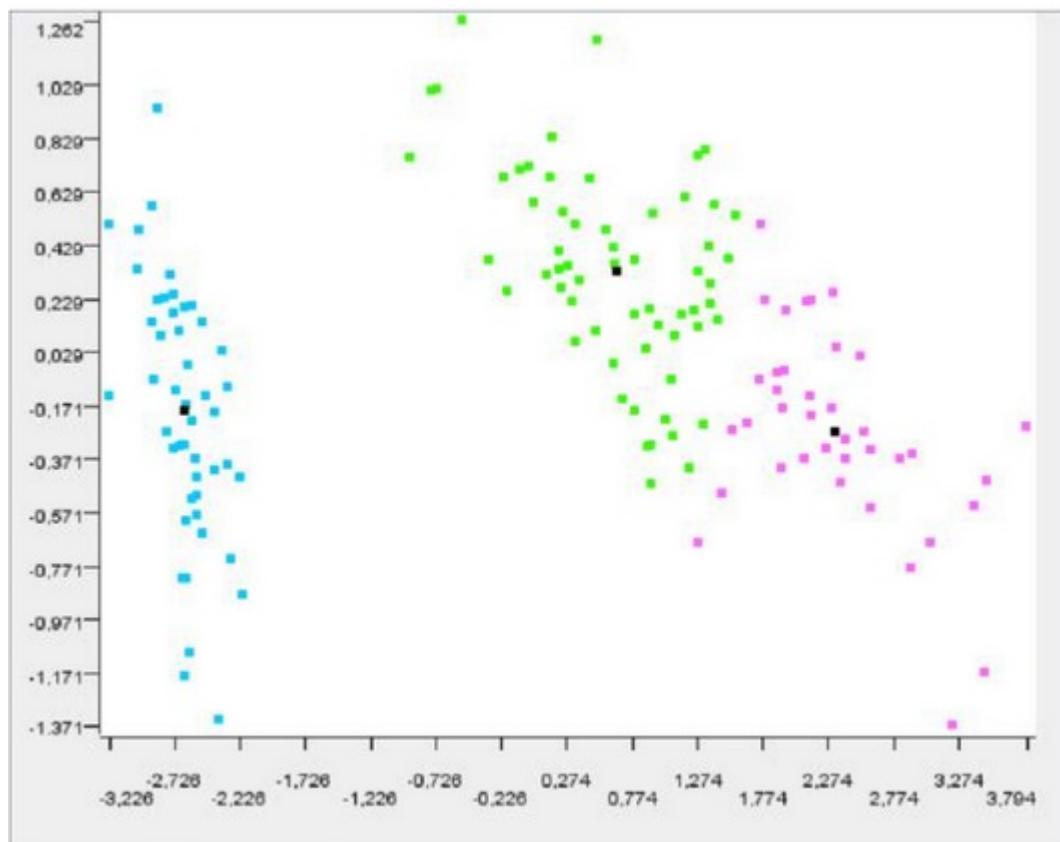
Cluster View - 2:1 - k-Means

File Hilite

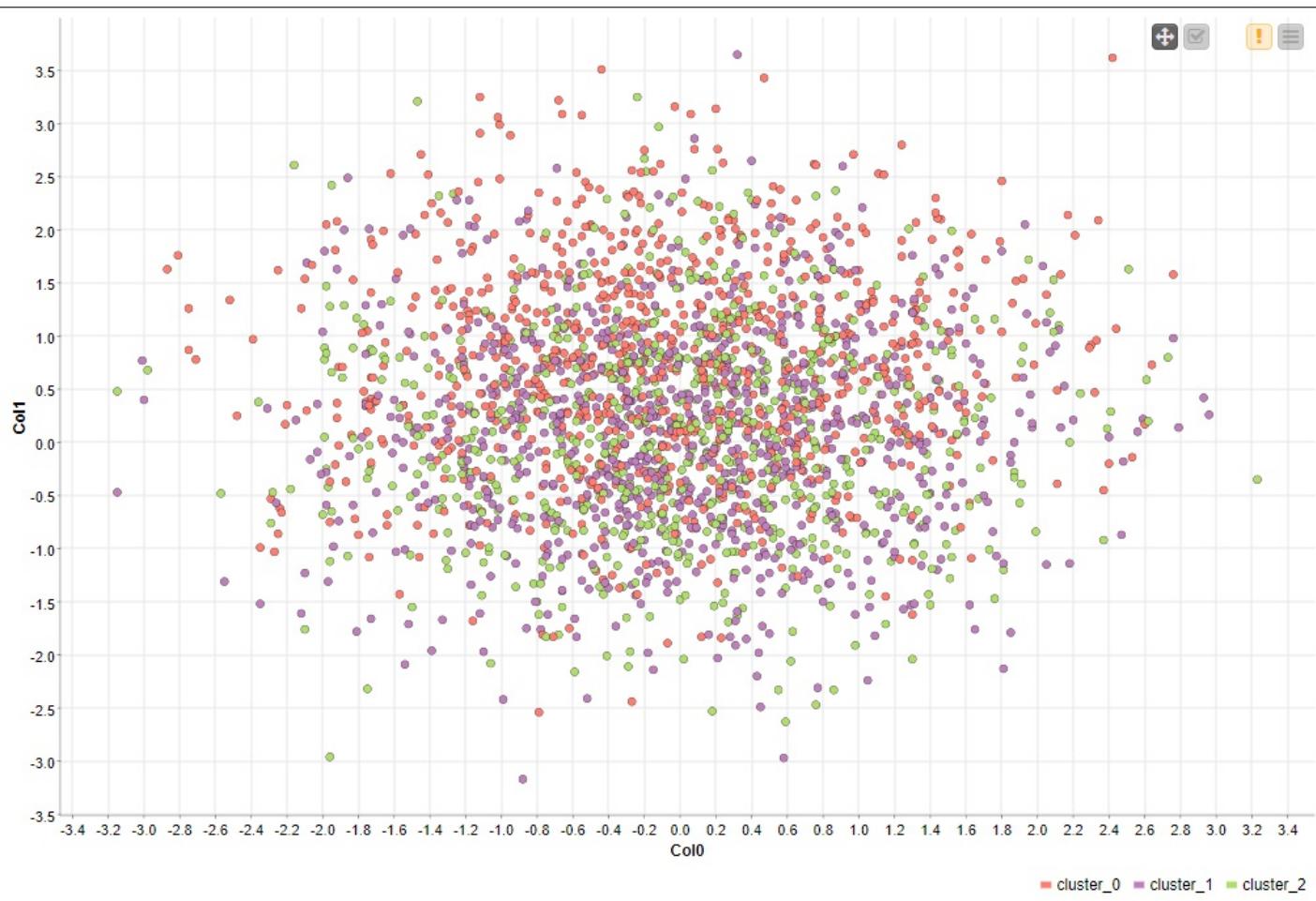
3 Clusters

- cluster_0 (coverage: 50)
 - Col0 = 5.005999999999999
 - Col1 = 3.4180000000000006
 - Col2 = 1.464
 - Col3 = 0.2439999999999999
 - Col4 = 1.0
 - PCA dimension 0 = -2.8244415138607293
 - PCA dimension 1 = 0.17263588799042434
- cluster_1 (coverage: 49)
 - Col0 = 5.8795918367346935
 - Col1 = 2.7530612244897967
 - Col2 = 4.23673469387755
 - Col3 = 1.3224489795918366
 - Col4 = 2.020408163265306
 - PCA dimension 0 = 0.46736603617212014
 - PCA dimension 1 = -0.24081767424508488
- cluster_2 (coverage: 51)
 - Col0 = 6.629411764705881
 - Col1 = 2.9862745098039207
 - Col2 = 5.549019607843136
 - Col3 = 2.0156862745098034
 - Col4 = 2.9607843137254903
 - PCA dimension 0 = 2.3200223513843636
 - PCA dimension 1 = 0.062122973303688184

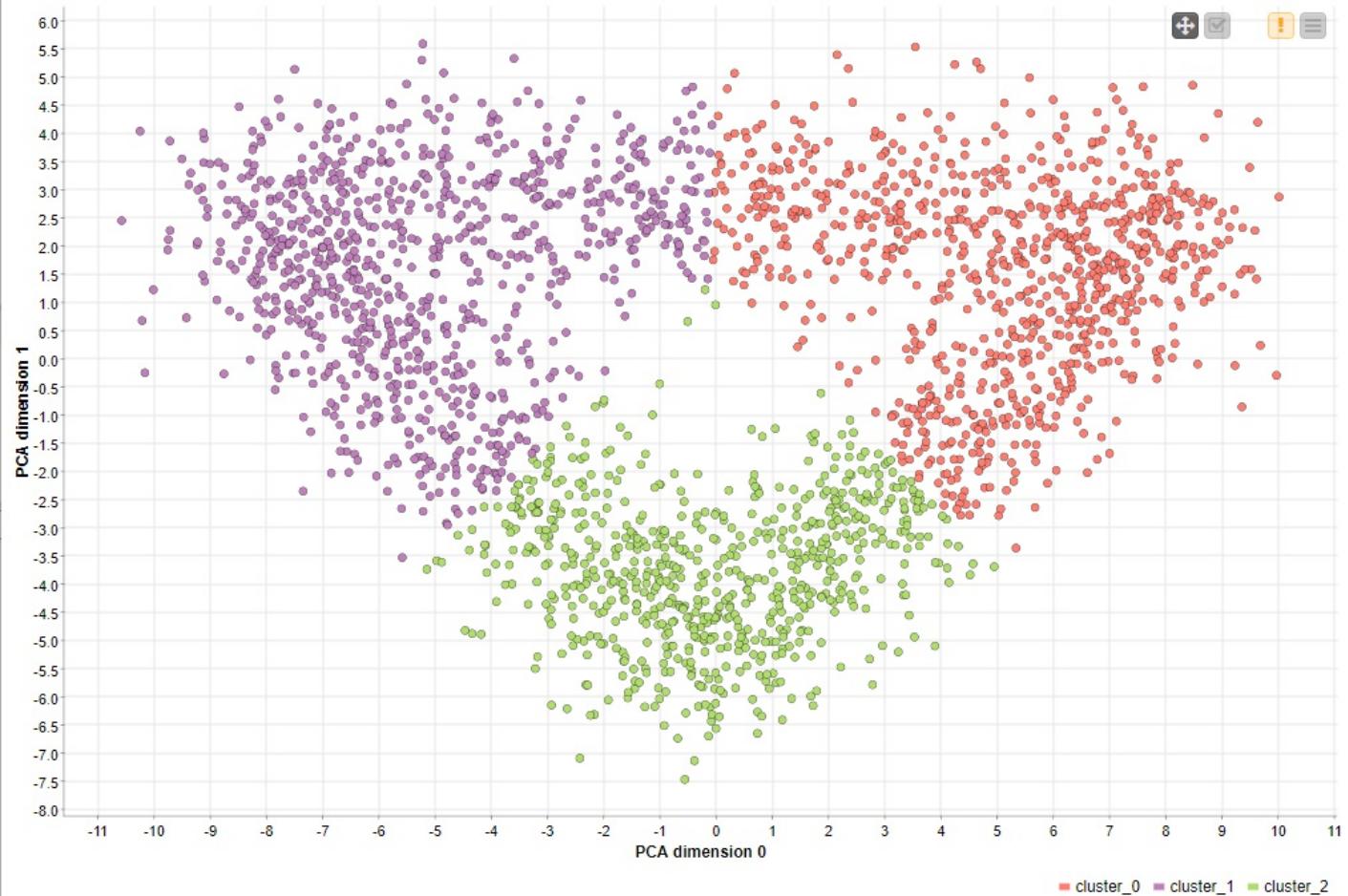
Représentation du barycentre



Pour la base de donnée de waveform, on a les données initiales suivantes.



Méthode clustering



Valeur du barycentre :

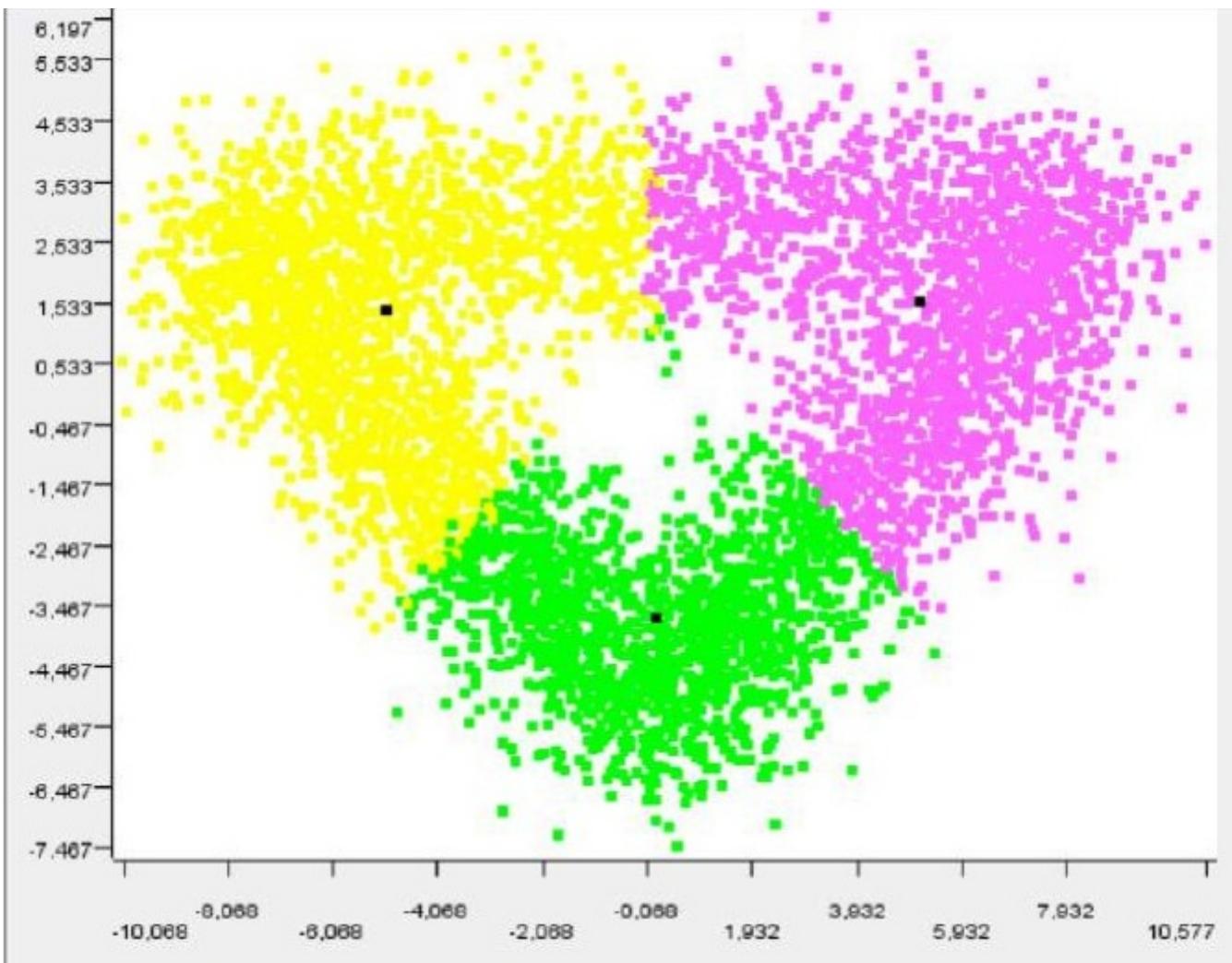
Cluster View - 0:3 - k-Means

File Hilite

3 Clusters

- cluster_0 (coverage: 1820)
 - Col0 = -0.04571428571428572
 - Col1 = 0.7046703296703287
 - Col2 = 1.4113241758241755
 - Col3 = 2.2113021978021945
 - Col4 = 2.952104395604394
 - Col5 = 3.810274725274728
 - Col6 = 4.676642857142854
 - Col7 = 4.126807692307688
 - Col8 = 3.5404725274725295
 - Col9 = 3.035659340659335
 - Col10 = 2.5543021978022002
 - Col11 = 1.8079560439560454
 - Col12 = 1.0662252747252736
 - Col13 = 1.0642692307692319
 - Col14 = 1.054263736263737
 - Col15 = 0.7660109890109882
 - Col16 = 0.48250549450549535
 - Col17 = 0.3613901098901096
 - Col18 = 0.2542912087912087
 - Col19 = 0.1331208791208792
 - Col20 = -0.01593406593406595
 - Col21 = -0.023181318681318747
 - Col22 = 0.009791208791208787
 - Col23 = 0.021241758241758273
 - Col24 = -0.014318681318681309
 - Col25 = -1.5384615384615442E-4
 - Col26 = 0.014329670329670313
 - Col27 = -0.0032472527472527614
 - Col28 = -0.012318681318681347
 - Col29 = 0.0033131868131868218
 - Col30 = -0.028725274725274755
 - Col31 = 0.021835164835164803
 - Col32 = -0.007445054945054939
 - Col33 = 0.051368131868131876
 - Col34 = 0.022082417582417562
 - Col35 = 0.025225274725274697
 - Col36 = -0.04747802197802195
 - Col37 = 0.020576923076923076
 - Col38 = 0.03127472527472515
 - Col39 = 0.050538461538461525
 - Col40 = 0.5126373626373626
- cluster_1 (coverage: 1760)
- cluster_2 (coverage: 1420)

Représentation graphique du barycentre



Exercice 4

Voici le score du clustering :

| Col40 \ Clu... | 2 | 0 | 1 | cluster_0 | cluster_2 | cluster_1 |
|----------------|---|---|---|-----------|-----------|-----------|
| 2 | 0 | 0 | 0 | 960 | 0 | 695 |
| 0 | 0 | 0 | 0 | 809 | 876 | 7 |
| 1 | 0 | 0 | 0 | 0 | 963 | 690 |
| cluster_0 | 0 | 0 | 0 | 0 | 0 | 0 |
| cluster_2 | 0 | 0 | 0 | 0 | 0 | 0 |
| cluster_1 | 0 | 0 | 0 | 0 | 0 | 0 |

Correct classified: 0

Wrong classified: 5 000

Accuracy: 0 %

Error: 100 %

Cohen's kappa (κ) 0

Et voici le score du modèle avec la validation croisée :

| Col40 \ Clu... | 1 | 2 | 0 | cluster_2 | cluster_0 | cluster_1 |
|----------------|---|---|---|-----------|-----------|-----------|
| 1 | 0 | 0 | 0 | 669 | 227 | 757 |
| 2 | 0 | 0 | 0 | 287 | 892 | 476 |
| 0 | 0 | 0 | 0 | 861 | 561 | 270 |
| cluster_2 | 0 | 0 | 0 | 0 | 0 | 0 |
| cluster_0 | 0 | 0 | 0 | 0 | 0 | 0 |
| cluster_1 | 0 | 0 | 0 | 0 | 0 | 0 |

Correct classified: 0

Wrong classified: 5 000

Accuracy: 0 %

Error: 100 %

Cohen's kappa (κ) 0



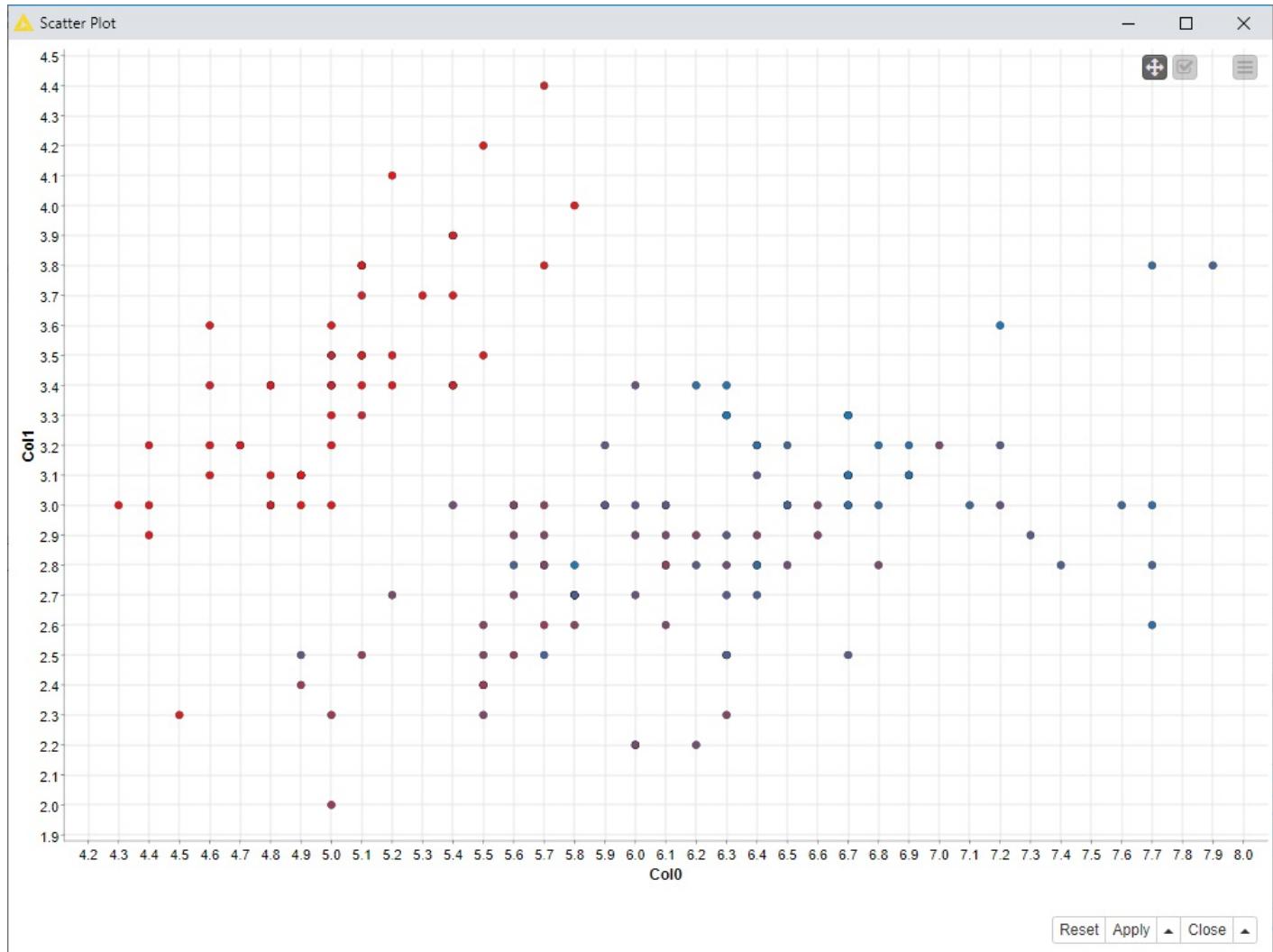
TP 4

Exercice 1

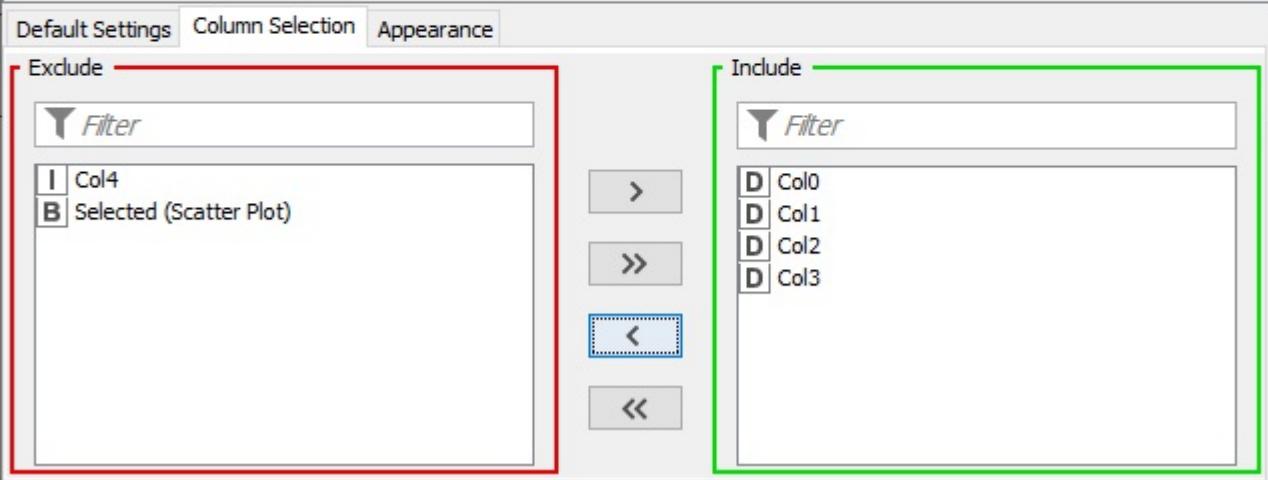
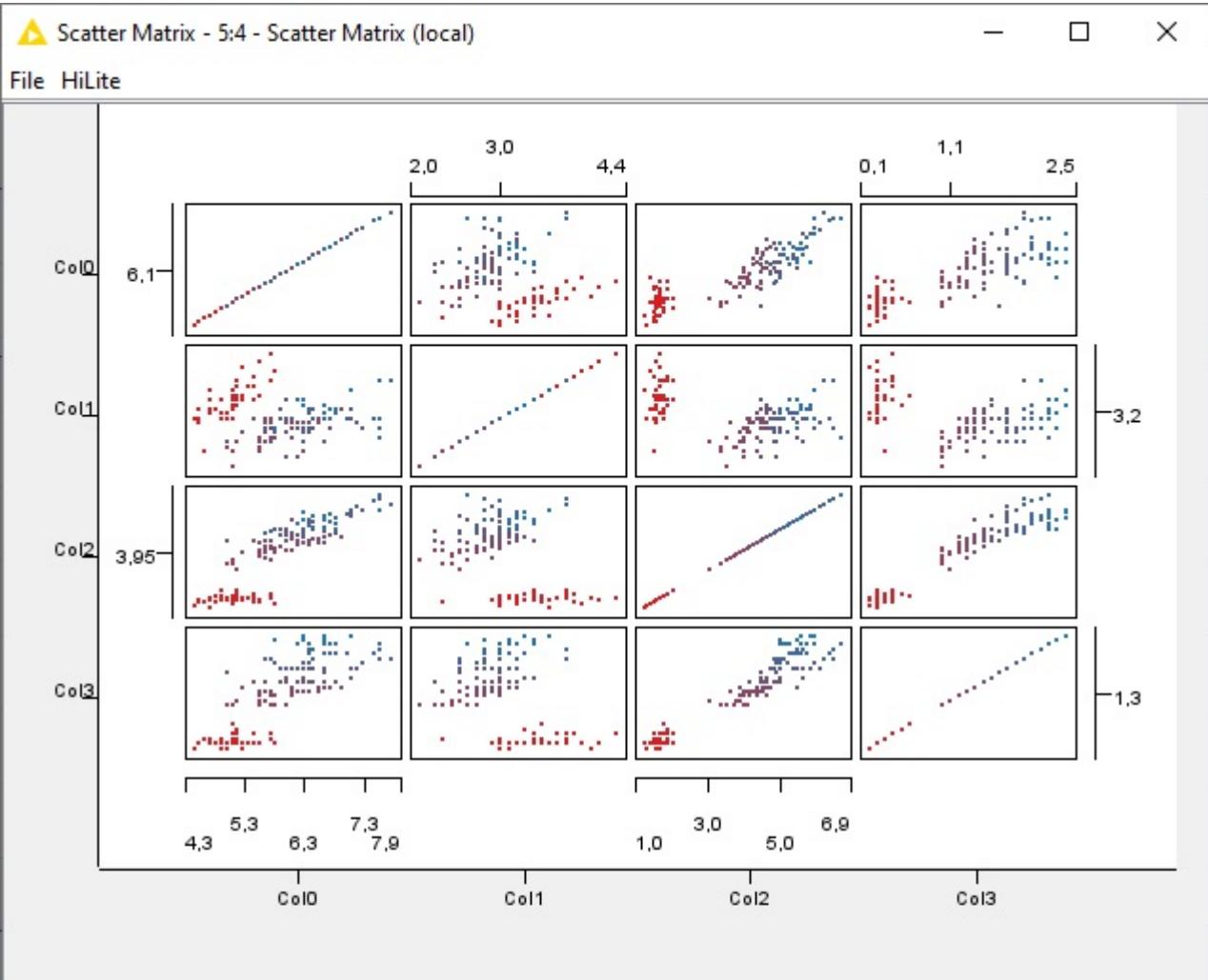
Le PCA permet de faire une bijection des données sur une paire pour pouvoir les visualiser en 2D (dimension to reduce: 2, si besoin, exclure les données non pertinentes).

Exercice 2

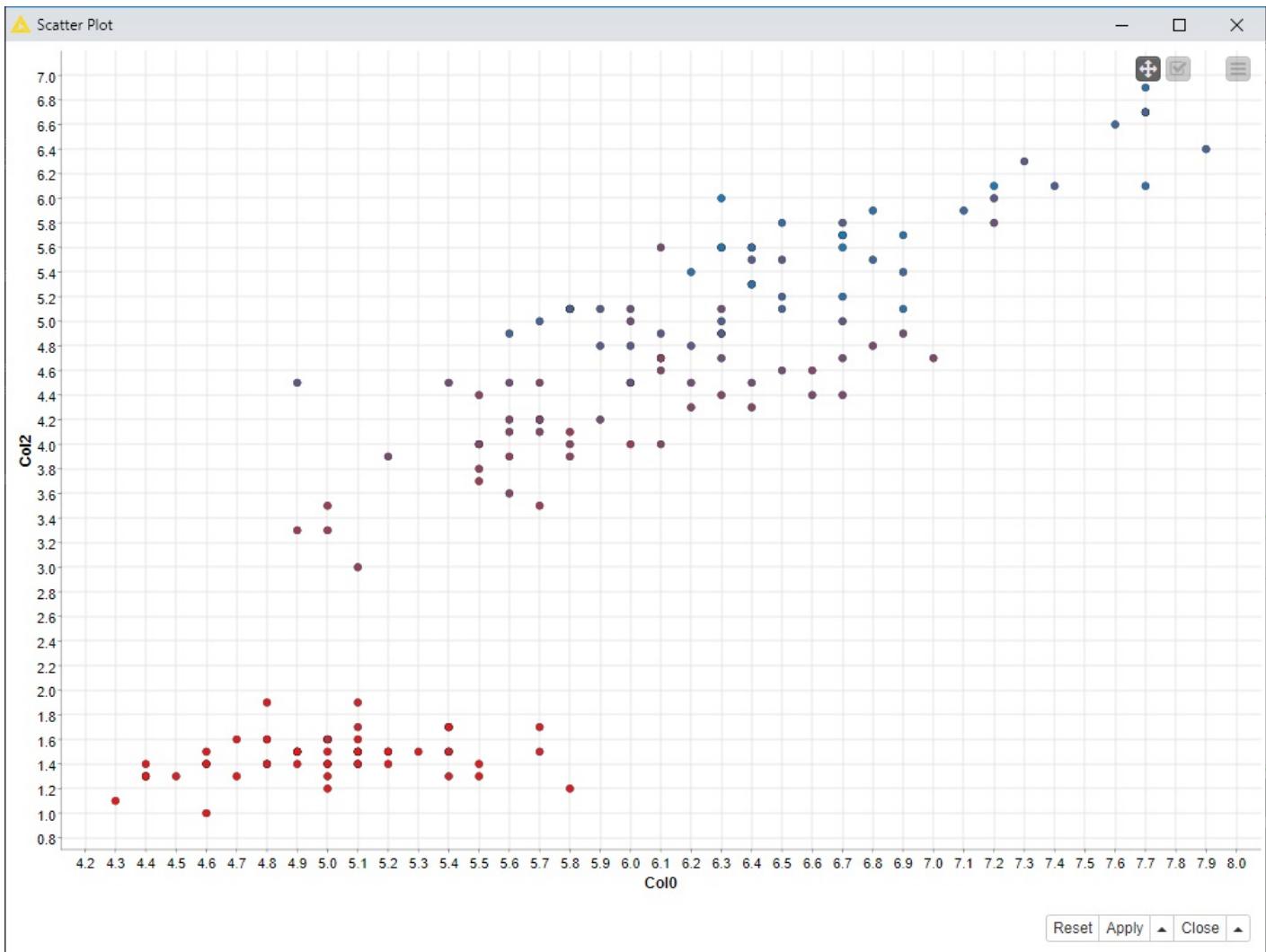
Voici ce que nous obtenons avec le premier scatter plot



Afin d'obtenir toutes les combinaisons de paires pour le scatter plot nous devons utiliser le noeud "Scatter Matrix" ce qui nous permet d'obtenir les graphes suivant :



On remarque que c'est la paire Col0:Col2 qui est la meilleure, car dans cette vue les valeurs sont mieux séparées que dans les autres vues.



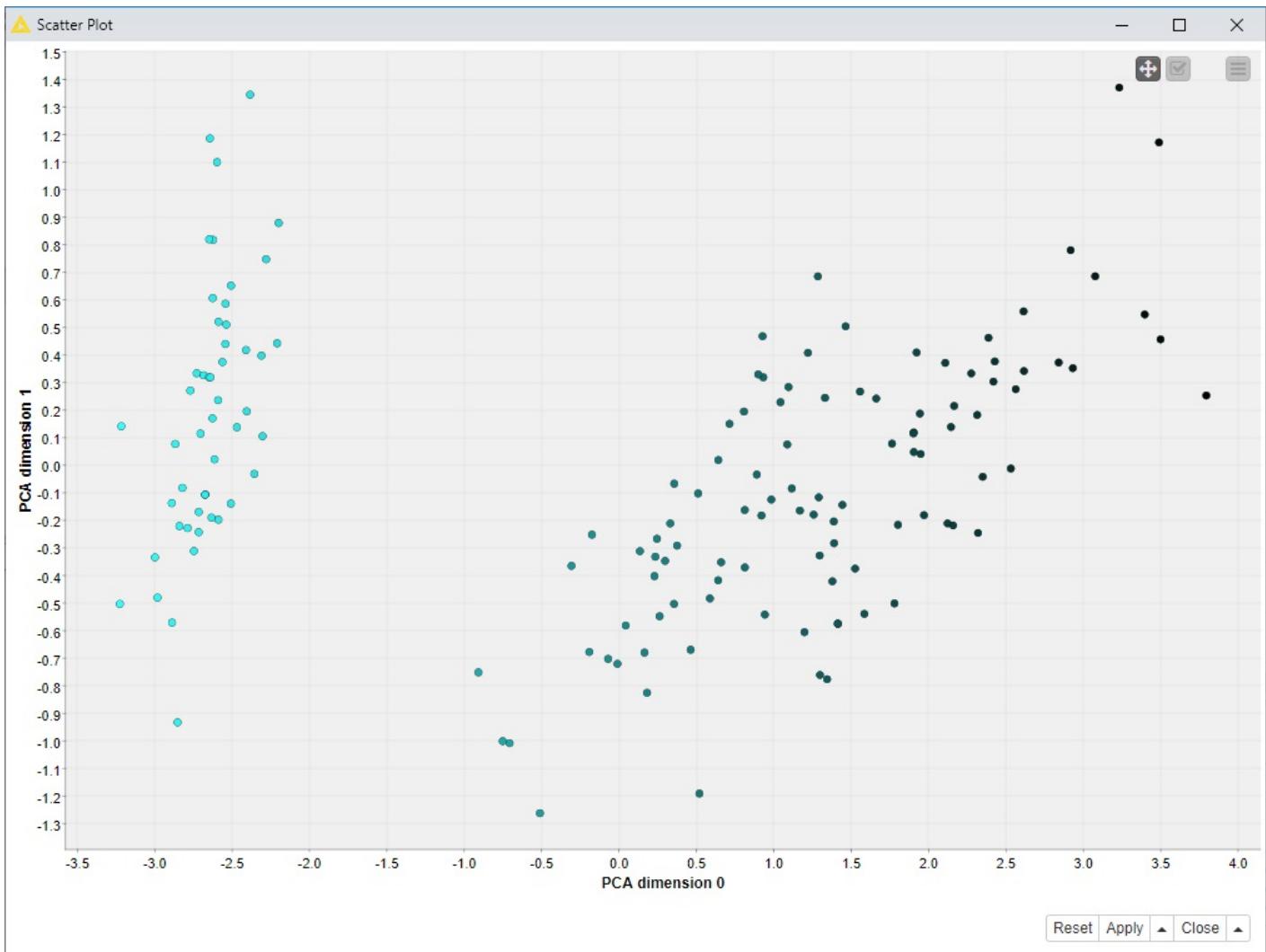
Exercice 3

a) On utilise un **File Reader** afin de lire un data set puis on le relie à un noeud **PCA** afin de réduire le data set à deux dimensions

Afin de faciliter la visualisation, on utilise **Color Manager** relié avec **PCA** pour coloriser les données.

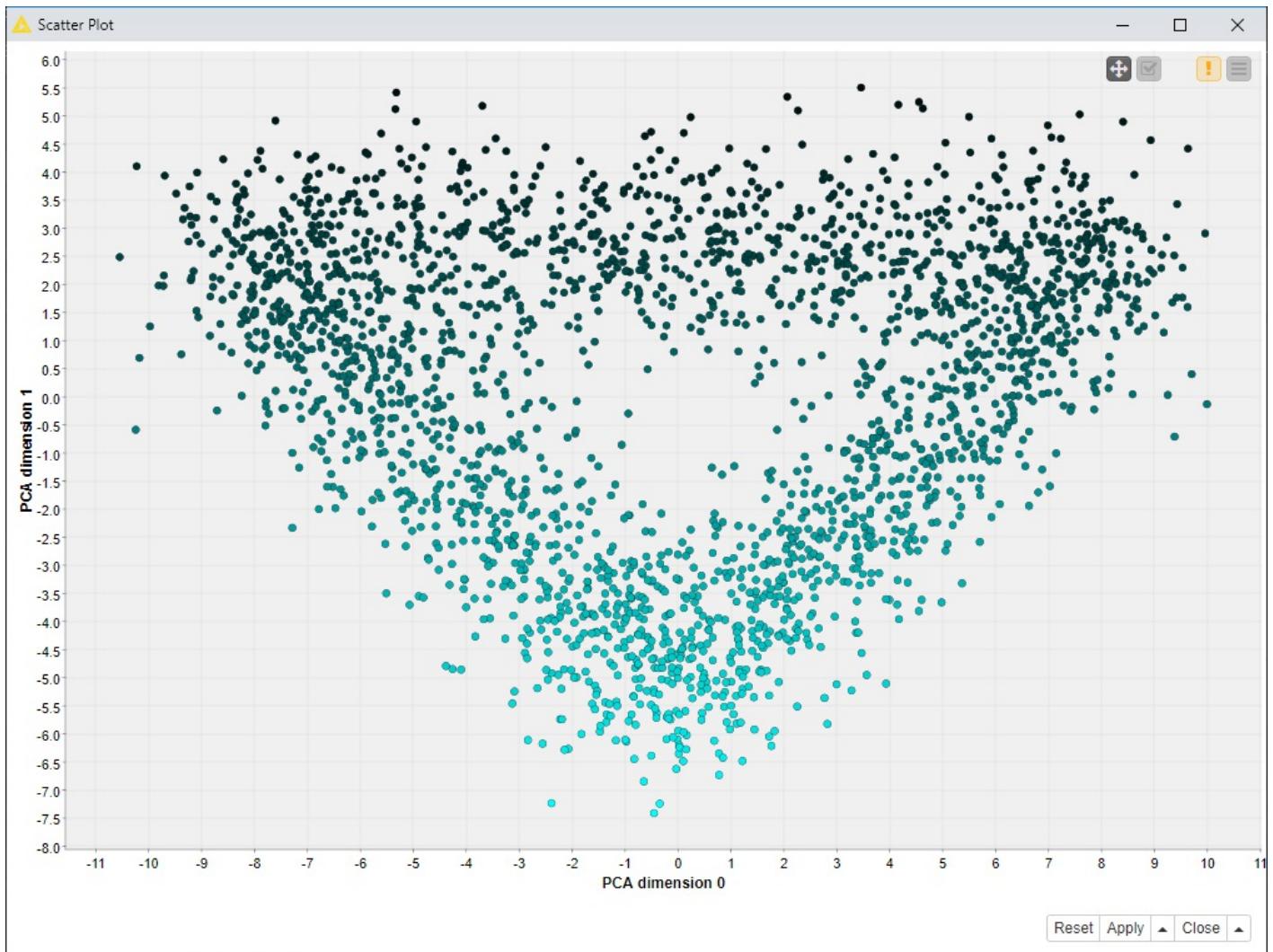
Enfin on utilisera **Scatter Plot** afin de générer un nuage de point et visualiser les données.

b)



Nous constatons que les données sont mieux séparées par classes. Nous distinguons bien les 3 classes (bleu clair / bleu foncé / noir). La méthode PCA permet donc d'avoir chaque type d'iris mieux séparé et groupé.

c.d.e)



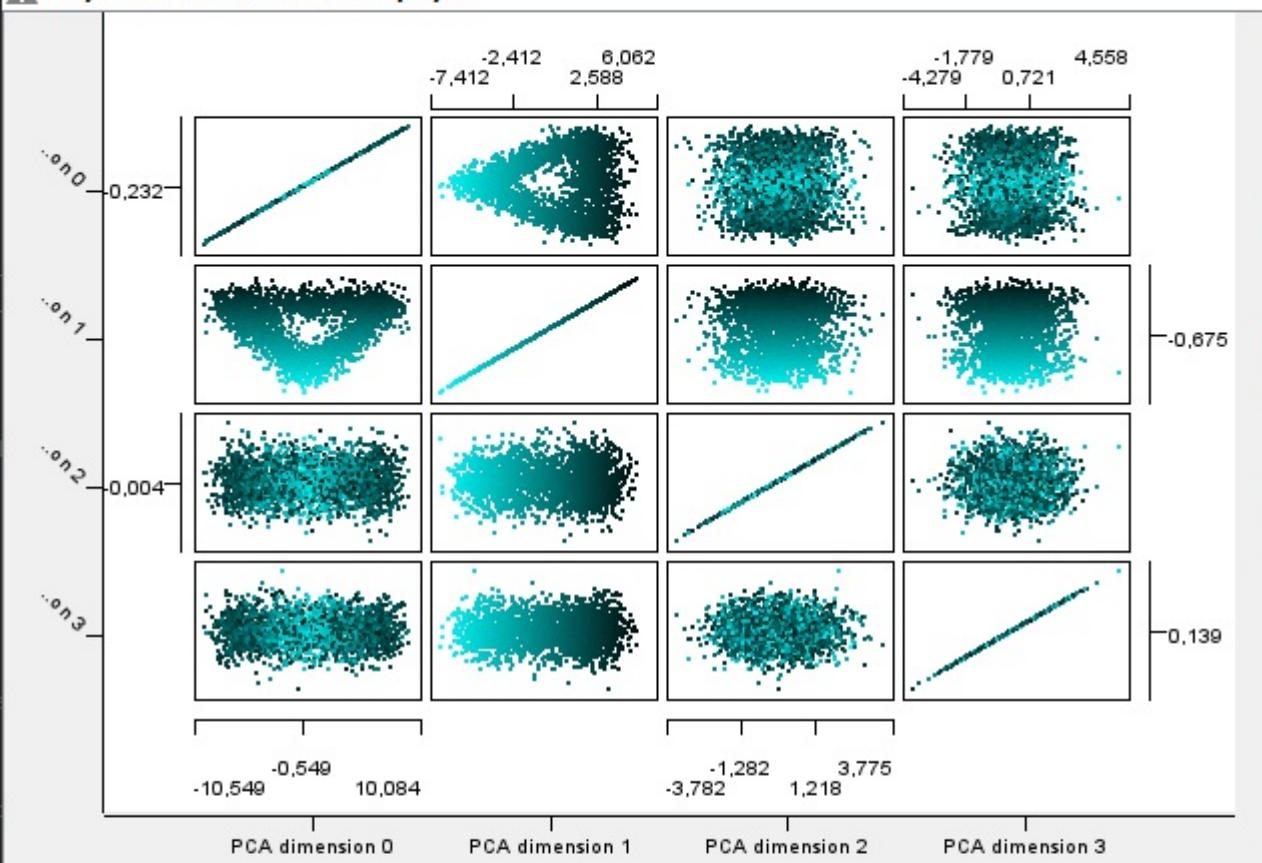
- Ci-dessus le nuage de point du PCA du data set **Waveform** a deux dimensions.

Scatter Matrix - 5:13 - Scatter Matrix (local)

- □ ×

File HiLite

⚠ Only the first 2500 rows are displayed.



Default Settings Column Selection Appearance

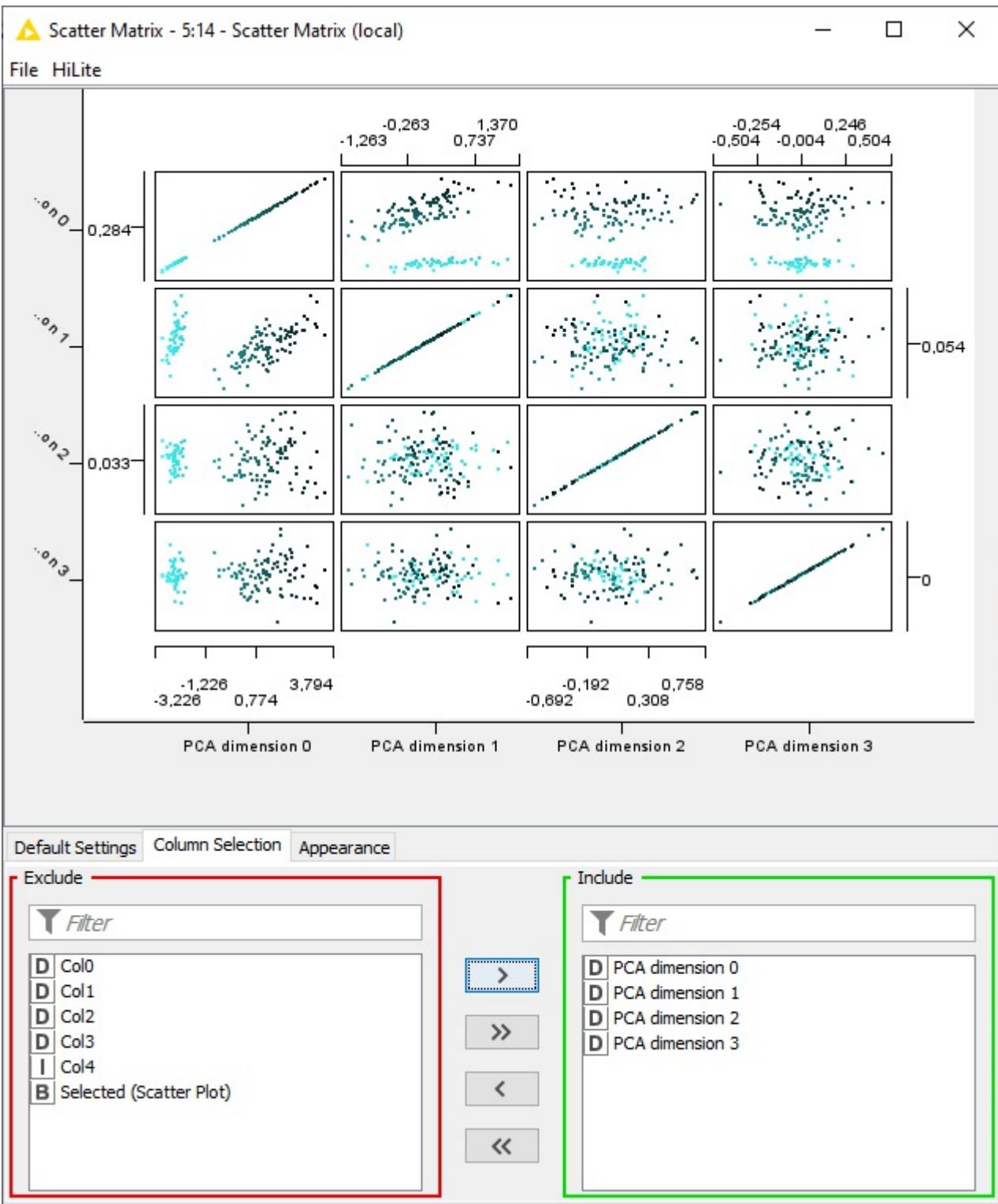
Exclude

- Filter
- Col33
- Col34
- Col35
- Col36
- Col37
- Col38
- Col39
- Col40
- Selected (Scatter Plot)

Include

- Filter
- PCA dimension 0
- PCA dimension 1
- PCA dimension 2
- PCA dimension 3

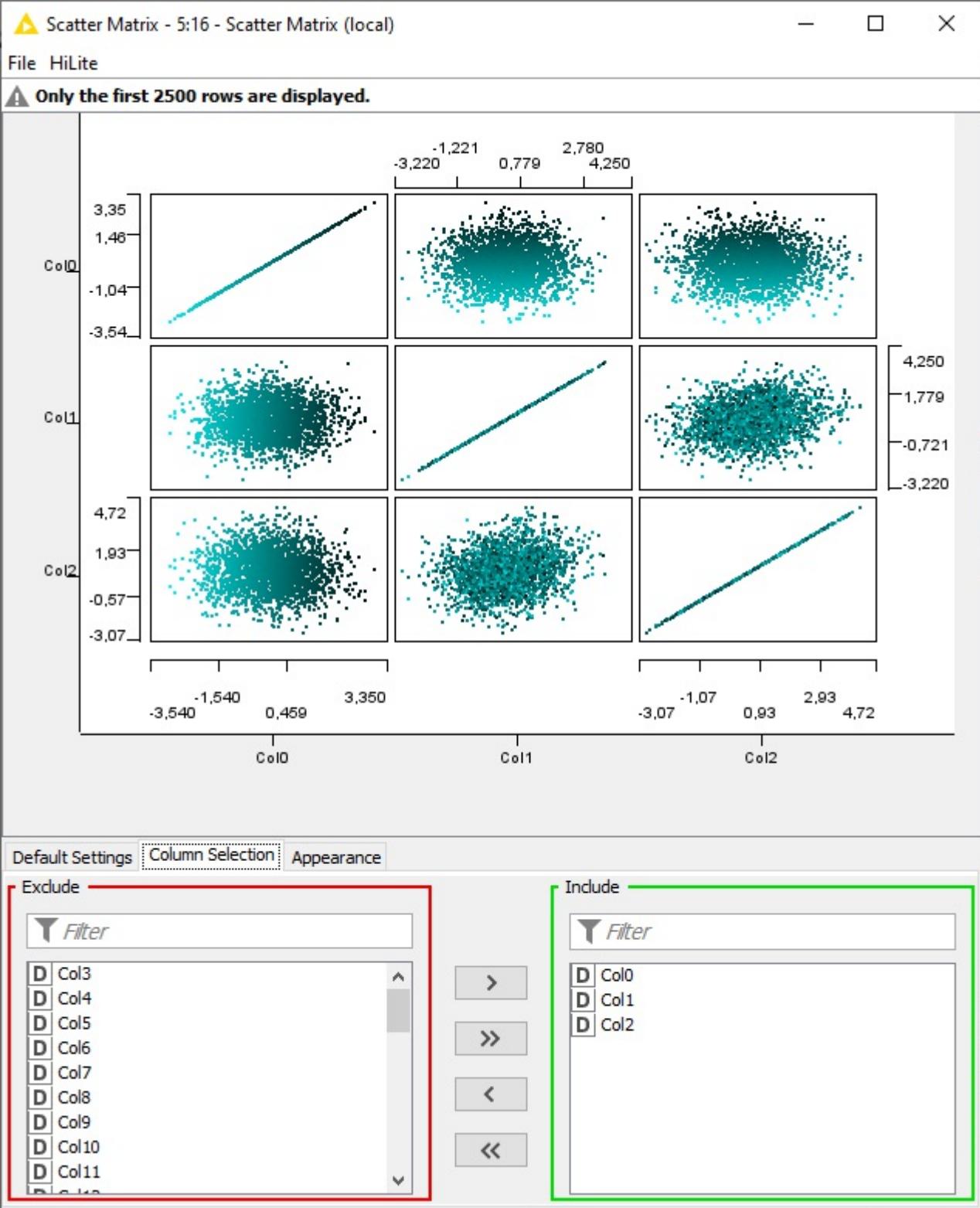
- Ci-dessus la matrice des nuages de point du PCA du data set **Waveform** à quatre dimensions.



- Ci-dessus la matrice des nuages de point du PCA du data set **Iris** à quatre dimensions.

Exercice 4

- a) Voici toutes les combinaisons de nuage de point que nous pouvons obtenir à partir du data set **Waveform** :



Nous remarquons que les classes sont toutes mélangées et par conséquent aucune analyse n'est possible.

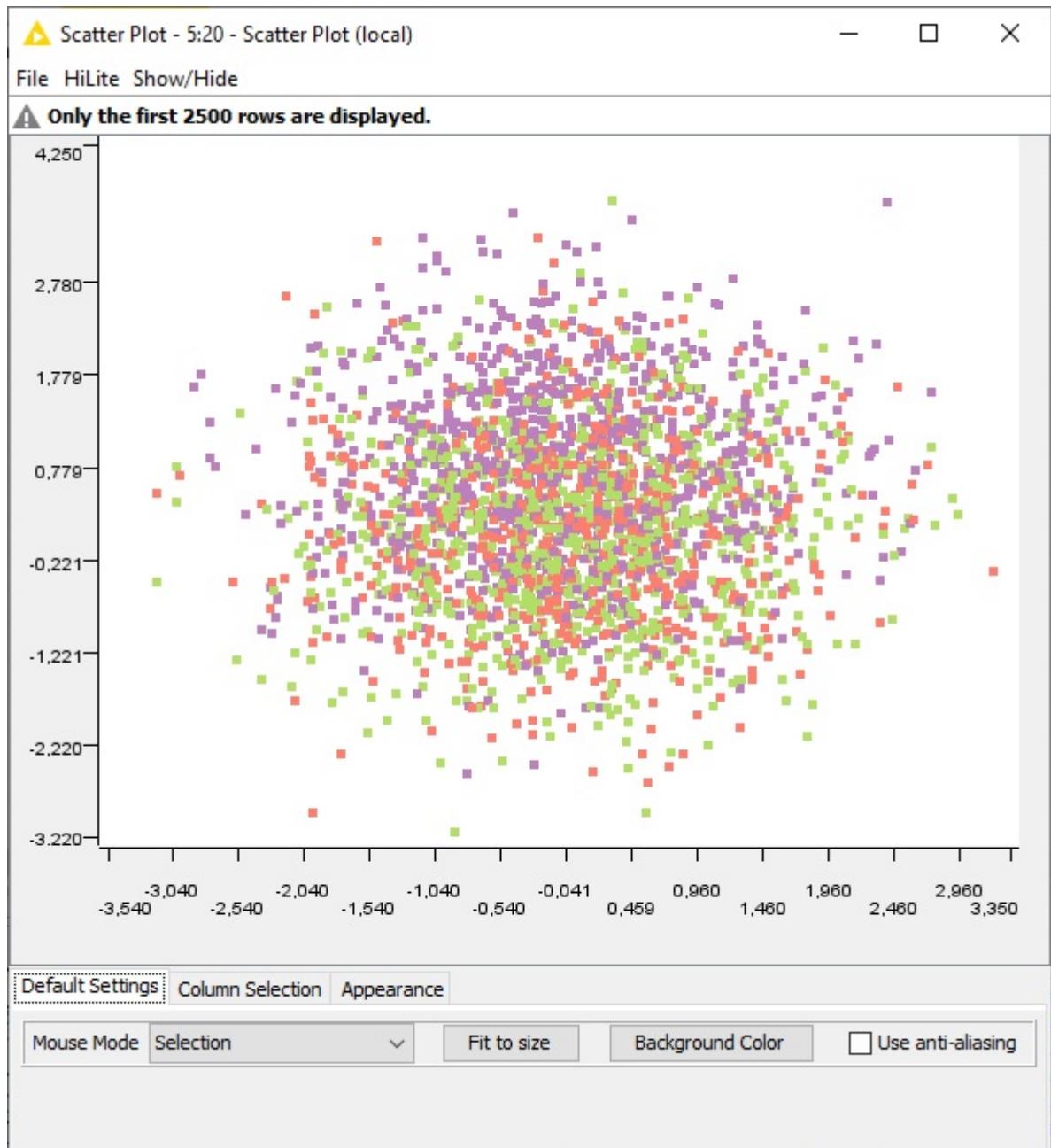
b) (cf les figures de l'exercice précédent)

Nous constatons qu'en utilisant un PCA il est plus facile de distinguer les classes de données, c'est donc une méthode à favoriser pour la lecture de données.

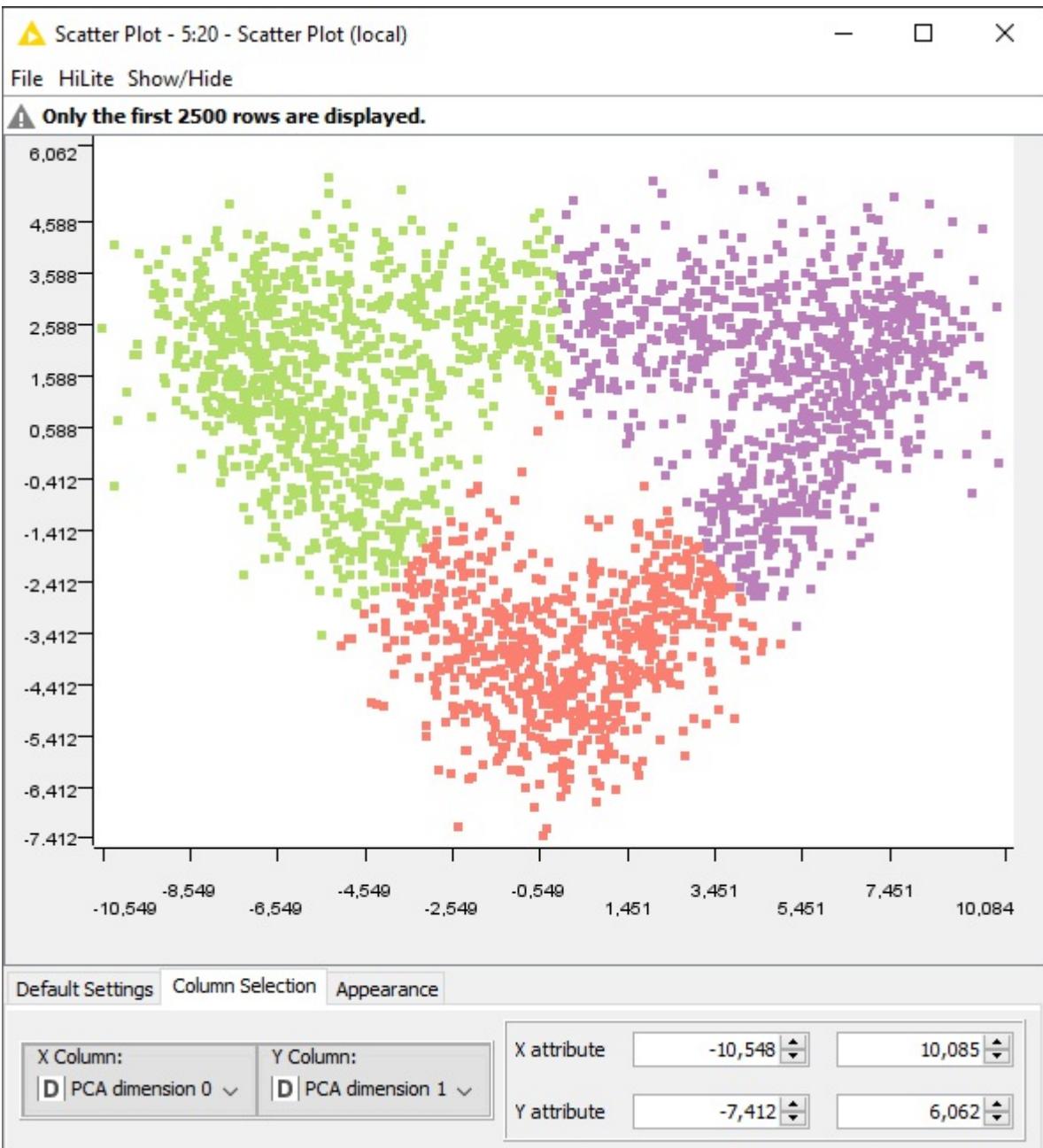
Exercice 5

a) (cf exercice 4)

b) Voici le résultat obtenu avec les clusters :

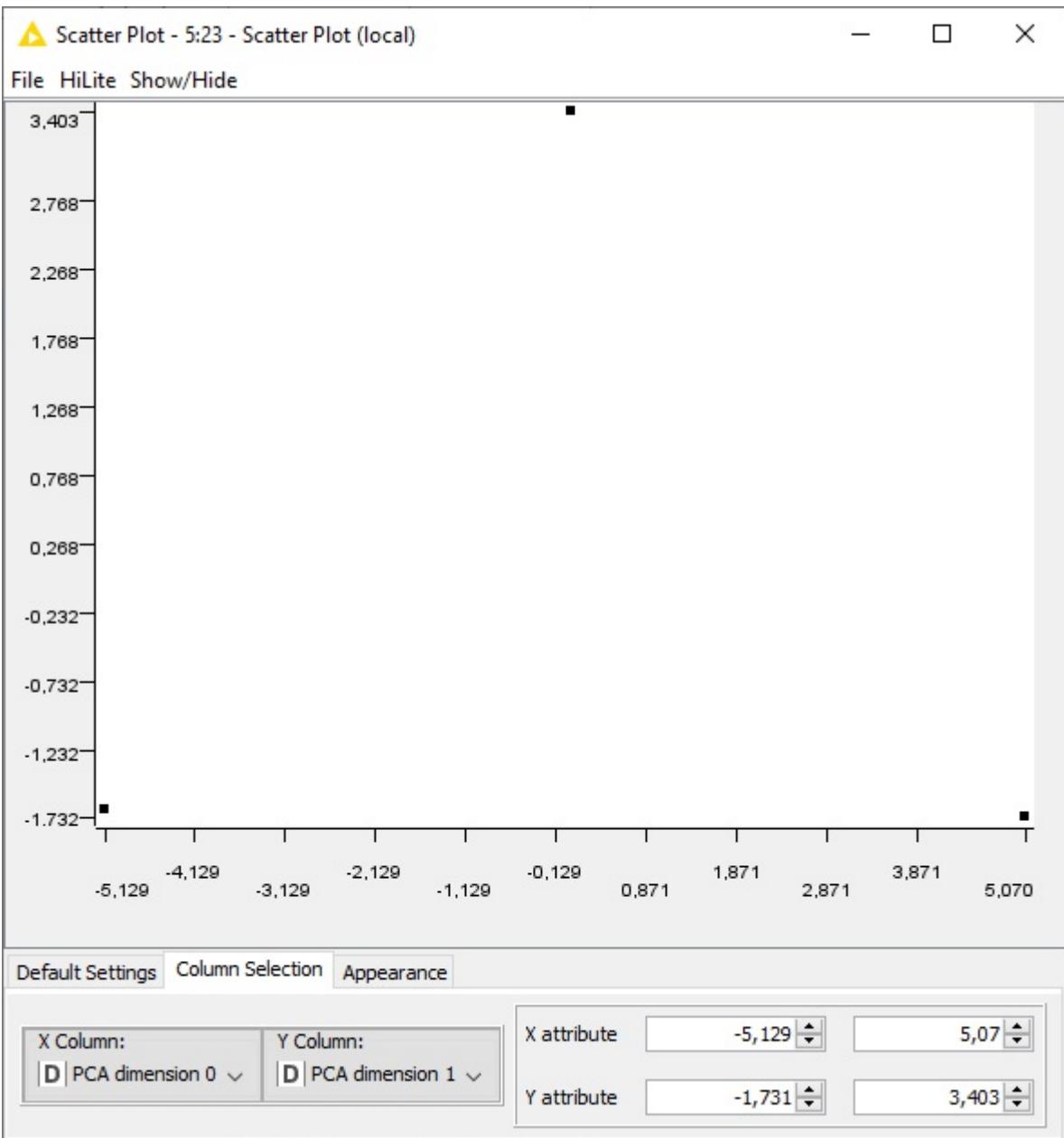


c) Voici le résultat avec PCA

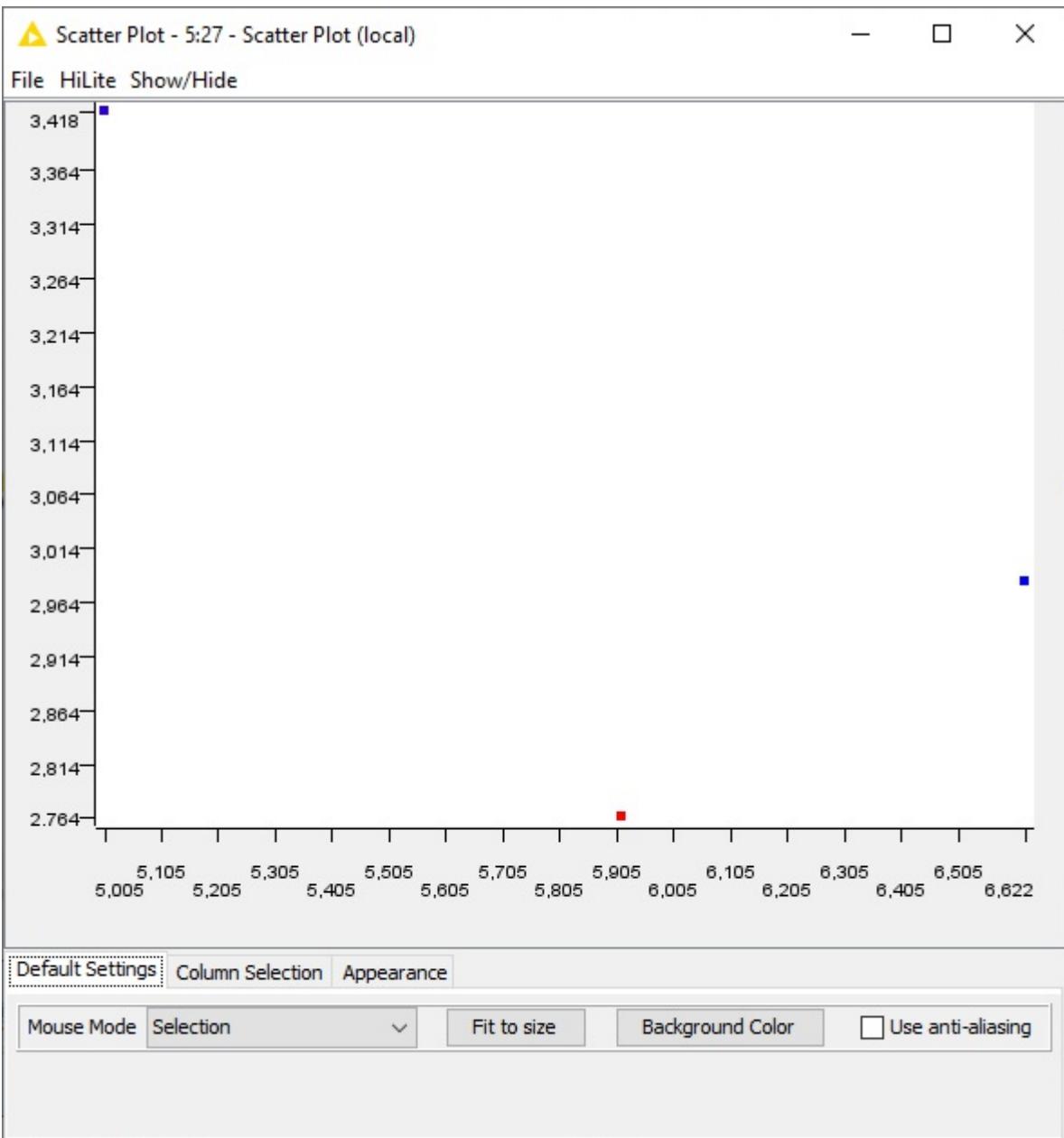


d) Dans la méthode Cluster+PCA nous remarquons que les données sont regroupées autour d'un centre alors qu'avec la méthode PCA seule les données sont regroupées par classes

e)



Ci-dessus le nuage de point pour representer les barycentre Nous n'avons pas réussi à combiner les deux nuages de point afin de superposer les cluster et leurs barycentre respectif.



Ci-dessus les barycentre de la base Iris.