Data Analysis - TP 4

ISEP – November 5th, 2019

Instructions: Prepare a report including the source code and the results. Deposit your report on Moodle and don't forget your binome's name or to make 2 deposits if you did not work alone.

Libraries

This lab may require the following libraries : pandas, matplotlib, numpy, scipy et sklearn.

A Analysing Fiher's Iris with the K-Means algorithm

1. Open the file *iris.csv* from the previous lab, or import them directly from the datasets module of sklearn.

2. The last column in your data contains the labels matching with the Iris specie to which each data belongs. Remove these labels from the main set and store them in another vector. For instance, you could have a vector X with just the data, and a vector Y with just the classes.

3. Use the command **PCA(·)** from sklearn.decomposition to do a Principal Component Analysis on your data. Then use the following lines to retrieve the dataset projected on the two principal components:

```
pca = PCA(n_components=2)
PCA_val = pca.fit_transform(data)
df_iris_PCA = pd.DataFrame(data = PCA_val,columns = ['PC1', 'PC2'])
df_iris_PCA_class = pd.concat([df_iris_PCA, Y], axis = 1)
```

You should now have 2 different data sets, the original one stored in a first variable (e.g. "X"), and the same data projected on 2 components stored in the variable *df_iris_PCA*.

4. Use the K-Means algorithm (library *sklearn.cluster*) on your data *df_iris_PCA* to obtain a partition with 3 clusters and visualize your results. To do so, you can use the code below and explain the parameter of the KMeans function.

```
#Kmeans code, change X by your dataset
kmeans = KMeans(n_clusters=3, n_init=5, max_iter=300).fit(X)
kmeans.score(X)
prediction = kmeans.predict(X)
```

5. Repeat question 4) several times. What happens ? Comment.

6. Project the labels that you stored in a separate vector in question 2). Compare these results with the partitions from your K-Means experiments. Comment.

7. Display the confusion matrix between your results with the theoretical labels. Comment.

8. Choose a solution that seems good enough for you and compute the silhouette index (command *silhouette_score()* of sklearn.metrics). Comment.

9. Start again questions 4) to 8) using the original data (with 4 variables) instead of the projected ones. How different are the results ? Explain the pros and cons of using the projected data or the original ones.

B Hierarchical clustering

This exercise is a tutorial on how to use scipy's hierarchical clustering.

1. First you are going to need to do some important imports to have the right packages:

```
# needed imports
from matplotlib import pyplot as plt
from scipy.cluster.hierarchy import dendrogram, linkage, fcluster
from scipy.cluster.hierarchy import cophenet, inconsistent, maxRstat
from scipy.spatial.distance import pdist
import numpy as np
```

2. First we are going to generate a sample of random data that we will use to try our clustering algorithm. Use the code below and try to modify the different parameters of the functions to create a and b to see what happens:

```
# np.random.seed(0)  # uncomment for repeatability
a = np.random.multivariate_normal([10, 0], [[3, 1], [1, 4]], size↘
    =[100,])
b = np.random.multivariate_normal([0, 20], [[3, 1], [1, 4]], size↘
    =[50,])
X = np.concatenate((a, b),)
plt.scatter(X[:,0], X[:,1])
plt.title('My data distribution')
plt.show()
```

3. The code below is used to generate the linkage matrix and visualize the dendrogram with a given linkage. After fixing the seed in the code above for repeatability, run the code below with different types of linkages and see what happens. Comment.

```
# alternative linkage methods: 'single', 'complete', 'average', '↘
    euclidean' (default), 'cityblock' aka Manhattan, 'hamming', 'cosine↘
    '...

Z = linkage(X, 'ward', optimal_ordering=True)


c, coph_dists = cophenet(Z, pdist(X))
print('Cophenetic Correlation: %1.2f' % c)


plt.figure(figsize=(25, 10))
plt.title('Hierarchical Clustering Dendrogram (full)')
plt.xlabel('sample clusters')
plt.ylabel('distance')
```

```
14
15  dendrogram( Z, leaf_rotation=90., leaf_font_size=8.,)
16
17  plt.show()
```

4. In the code above, what does the variable Z contain ? Display it and try to explain it.

5. Search for the definition and role of the cophenetic coefficient described in the code above. Explain how it could be interpreted.

6. As you can see, the dendrogram can be pretty big and pretty messy. You can display a simplified version using the code below. In this code, explain the roles of parameters "truncate_mode" and "p".

```
1  #display truncated dendrogram
2  plt.title('Hierarchical Clustering Dendrogram (truncated)')
3  plt.xlabel('sample index')
4  plt.ylabel('distance')
5  dendrogram(
6      Z,
7      truncate_mode='lastp', #to explain
8      p=12,  #to explain
9      show_leaf_counts=False,
10     leaf_rotation=90.,
11     leaf_font_size=12.,
12     show_contracted=True,
13 )
14 plt.show()
```

7. Finally, we want to cut the dendrogram to obtain the clusters. To do so, we can use two methods shown in the codes below. Explain how they differ, what the different parameters do, and the pros and cons of each. Try to modify the parameters to see what happens.

```
1  max_d = 14
2  clusters = fcluster( Z, max_d, criterion='distance')
3
4  plt.figure(figsize=(10, 8))
5  plt.scatter(X[:,0], X[:,1], c=clusters, cmap='prism')
6  plt.show()
```

OR

```
1  k=4
2  clusters = fcluster(Z,k,criterion='maxclust')
3
4  plt.figure(figsize=(10, 8))
5  plt.scatter(X[:,0], X[:,1], c=clusters, cmap='prism')
6  plt.show()
```

8. Repeat the questions 3 to 7 with a different linkage to see the differences. Comment.

C Optimal cluster number in exoplanet data

In this exercice, we will try to guess the optimal number of clusters to be found in an artificial data set describing the atmospheric characteristics of exoplanets.

1. Open the file *exo4_atm_extr.csv* and remove the last column containing the labels.

2. Write down the different properties of the Calinski-Harabasz and Davies-Bouldin indexes.

3. Use sklearn.metrics to determine the optimal number of clusters with the K-Means algorithm based on the Davies-Bouldin or Calinski-Harabasz index. Explain with your own words what this function does.

4. Comment yours results. You can use PCA and the real labels in your explanations.