Data Analysis - Lab 3

ISEP – October 15th, 2019

Instructions: Prepare a report and send it to the deposit box on moodle

Preliminary steps: Install and load all the following libraries: scikit (should be already installed), SOMPY (zip-archive downloaded from moodle, installation instructions are given in ex.).

A PCA and LDA on the Iris data

Fisher's Iris are one of the most famous dataset: it describes the characteristics of 150 iris flowers based on their petals and sepals. The dataset contains 3 balances classes representing 3 subspecies of the Iris flower (50 of each): Iris Virginica, Iris Setosa, and Iris Versicolor. One of the 3 class can easily be distinguished from the other 2 that are less separable. The goal of this exercice is to visually study this dataset using PCA and LDA.

1. Open the Iris data using the panda library.
2. Associate each class to a color for display purposes.
3. Center and reduce your data. You may do so using the package *Standard-Scaler* from sklearn, and the function of the same name.

```
#Normalize data
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
data = scaler.fit_transform(data)
```

5. Visualize the data in 2 dimensions using PCA. Comment.
6. Write a function to draw a correlation circle similar to the ones we saw in class for these data (you may use code from the internet). Based on the result, comment on the relations between the different attributes.

We are now going to check the influence of the different attribute to properly separate the 3 classes using LDA.

7. Using the function *train_test_split*, split your dataset into to subset with the proportions 80/20. The dataset with 80% of the data will be your training set, and the other one your test set. Try to have a good class repartition on both subsets.

8. Using the *LinearDiscriminantAnalysis* library from sklearn, train and test an LDA model. Display the confusion matrices for both training and test sets. Compare your visual results with the ones from PCA. Comment.

9. Same question with the QDA algorithm from *QuadraticDiscriminantAnalysis*.

B MDS, LLE and Isomap

B.1 golub data

We are now going to study the "golub" data. These data contains the level of expression of 7129 genes based on 72 sample. Each sample is linked to a variant of Leukemia: AML(25) and ALL(47). Our goal is to visualize the 72 samples on a 2D plan.

1. Load the file *Golub_data* while being very careful with the reading parameters. Compare the data that you just opened with the above description and comment. Modify the data if necessary, then normalize them.

2. Open the label in the file *Golub_class2*. Once again, be very careful with the opening parameters.

3. Run a PCA on the *Golub* data, and visualize the result. You may want to use the labels to color your figure. Comment.

4. Use the *MDS* function from the package *MDS* to run a MDS on these data. Compare with the PCA result and comment.

5. Using the *LocallyLinearEmbedding* package from sklearn, apply LLE to the *Golub* data, with 3, 5, 8, 10, 12 and 15 neighbors. Analyze the results and determine the best neighborhood model. Use the **plt.subplot** function rather than the regular plot function for this question.

6. Same question for the *Isomap()* function from the *Isomap* package. Use 4, 8, 10, 13, 16 and 20 neighbors instead.

B.2 Données Alon

Load the *Alon* data, describe their content and apply the same algorithms as in the previous exercise.

C SOM analysis with the sompy library

1. Install the SOMPY library SOMPY-master.zip using the following instructions: (zip was downloaded at at https://github.com/sevamoo/SOMPY, a bug was manually corrected in the archive.
   Info about the issue here: https://github.com/sevamoo/SOMPY/issues/95)

```
# Open the terminal (if you are using Anaconda on Windows, open Anaconda Promp
# Go to the directory with the downloaded zip
cd /directory_path
pip install SOMPY-master.zip
pip install ipdb==0.8.1
```

Some examples of library usage can be found here:
https://github.com/sevamoo/SOMPY/tree/master/sompy/examples
Use these examples for the following exercise.

2. Build your *iris* data model using the function **SOMFactory().build(......)** with a 10 by 10 grid using a hexagonal neighbourhood topology.

3. Plot the components map using the following code:

   ```
   from sompy.visualization.mapview import View2D
   view2D = View2D(10, 10, " ", text_size=10)
   view2D.show(sm, col_sz=7, which_dim="all", denormalize=True)
   plt.show()
   ```

   What can you say about these prototypes?

4. Display the number of data per neuron (use **BmuHitsView()** function). Comment. Explain zero values. How can we lower their number?

5. Display the U matrix using the **UMatrixView()** function. Precise *distance* = 2 where needed. Comment on the obtained results.