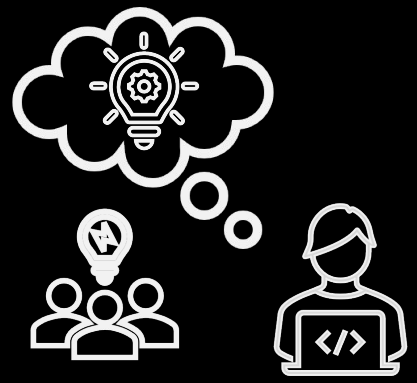
Lab session #2:
Proximity measures

Giulia Cisotto

Department of Informatics, Systems and Communication

University of Milan-Bicocca

giulia.cisotto@unimib.it



Academic year 2023/2024 – II semester

Motivation

Steps:

1. Visualize and describe the dataset **[TASK 1]**
2. Use Pandas DataFrame and compute centroids **[TASK 2a, 2b]**
3. Compute the proximity matrix using different proximity measures **[TASK 3a, 3b]**
4. Compare different proximity measures **[TASK 4a, 4b]**

Motivation

This second lab session aims to start dealing with tools and Python code to compute different proximity measures (using SciPy).

To know more about Python coding for this lab, you can read the **SciPy documentation** where you can find the explanations of the packages used during the lab (e.g., scipy.spatial.distance).

Read the step-by-step instructions below carefully and write your own code to fill the missing steps in the Colab notebook (instructions are also reported in the notebook).

Here is the link to **the Python code @Colab for today**

Make your own copy and work on it!

The **data** to work on will be **available on Moodle** at the beginning to the lab session.

Useful **packages**: numpy, pandas, scipy.spatial.distance, matplotlib, seaborn

Useful Python **data structures**: list, ndarray, Series, DataFrame

Motivation

Steps:

1. Visualize and describe the dataset **[TASK 1]**
2. Use Pandas DataFrame and compute centroids **[TASK 2a, 2b]**
3. Compute the proximity matrix using different proximity measures **[TASK 3a, 3b]**
4. Compare different proximity measures **[TASK 4a, 4b]**

[From row 4] Insert the Python code to generate the dataset (available on eLearning at the beginning of the lab session)

VISUALIZE AND DESCRIBE THE DATASET

The dataset is generated synthetically and is characterized by
-*N* objects
-*M* attributes
-*K* groups (or classes, or categories)

Using the command `print()`, make the code print the complete sentences below.

```
The matrix has shape = ...
It has ... objects and ... attributes.
```

Afterwards, print also the matrix X on the screen.

Then, complete the code to visualize the dataset in three different visualizations (image, line plot, scatterplot), and report them in your lab report with a comment.

Motivation

Steps:

1. Visualize and describe the dataset **[TASK 1]**
2. Use Pandas DataFrame and compute centroids **[TASK 2a, 2b]**
3. Compute the proximity matrix using different proximity measures **[TASK 3a, 3b]**
4. Compare different proximity measures **[TASK 4a, 4b]**

COMPUTE "CENTROIDS"

Import the two common packages `pandas` and `seaborn`.

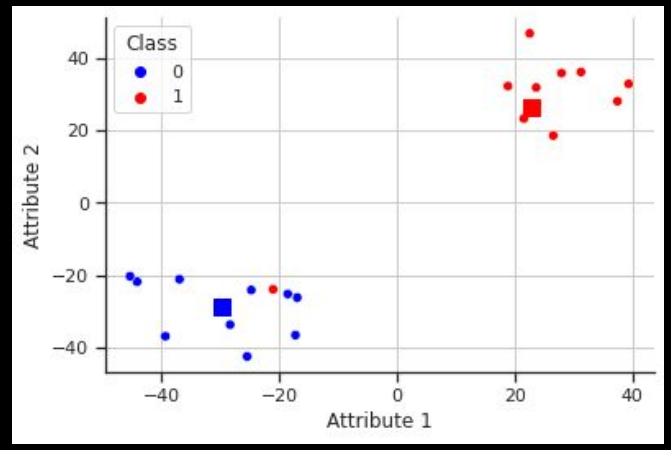
Transform the 2D matrix into a Pandas **DataFrame** data structure, and add a column for assigning each object to a "group" ('0' or '1', numerical class type).

Use `.plot(kind='scatter', …)` to plot the objects with different colors, depending on their group.

Compute the "centroid", i.e., the mean point, of each group (or class):

```
centroik = np.zeros((2,3))
#initialization
```

Use `sns.scatterplot` as an alternative to plot the objects with different colors, depending on their group, and add the centroids with a different 'marker' (e.g., a square).



Motivation

Steps:

1. Visualize and describe the dataset **[TASK 1]**
2. Use Pandas DataFrame and compute centroids **[TASK 2a, 2b]**
3. Compute the proximity matrix using different proximity measures **[TASK 3a, 3b]**
4. Compare different proximity measures **[TASK 4a, 4b]**

Compute the proximity matrix

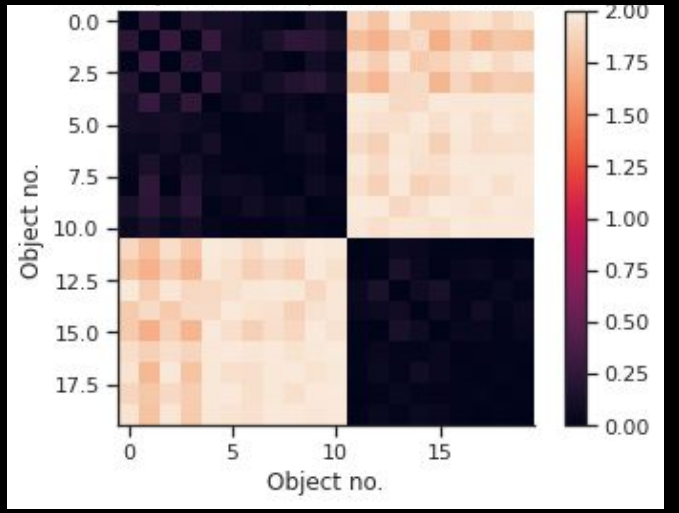
Choose a proximity metric:

```
'euclidean', 'minkowski', 'mahalanobis',  'hamming'/'matching', 'jaccard', 'cosine', 'correlation
'
```

Import the package `scipy.spatial.distance`, and use the function `pdist()` to compute the proximity measure between every two objects of the dataset.

Compute the full **proximity matrix** and visualize it using ***imshow()***

Visualize the values included in the matrix.



Repeat the same for two different choices of the metric.

Motivation

Steps:

1. Visualize and describe the dataset **[TASK 1]**
2. Use Pandas DataFrame and compute centroids **[TASK 2a, 2b]**
3. Compute the proximity matrix using different proximity measures **[TASK 3a, 3b]**
4. Compare different proximity measures **[TASK 4a, 4b]**

Compare different proximity measures

Choose three objects (P1, P2, P3) and retrieve their pair-wise distances (P1-P2, P2-P3, P1-P3) from the three proximity matrices above.

Object	(x,y) on PM1	(x,y) on PM2	(x,y) on PM3
P1	*to be filled*	…	…
P2	…	…	…
P3	…	…	…

Metric	P1-P2	P1-P3	P2-P3
Metric_name_1	*to be filled*	…	…
Metric_name_2	…	…	…
Metric_name_3	…	…	…

Answer to the question: "Do you think proximity reflects class membership?"

Next Lab on *k-Means*

March, 26th – 2.30 p.m.

Meanwhile, if you have any questions…

use the Lab Forum @eLearning!