11752 Machine Learning
Master in Intelligent Systems
Universitat de les Illes Balears

Handout #2: **Unsupervised Learning**

NOTE 1: The following problems require loading dataset `dsggp.txt` where `gg` is the group number and `p` is the dataset number:

```
import numpy as np
group = '01' # assuming group 1
ds = 1       # dataset 1
data = np.loadtxt('ds'+group+str(ds)+'.txt')
X = data[:, 0:2]
y = data[:, 2:3]
```

Class labels are 0, 1, 2, etc. All datasets include the true labelling for all samples.

NOTE 2:

- All problems will require the use of `scikit-learn` (`https://scikit-learn.org`, `https://scikit-learn.org/stable/modules/classes.html`) and `matplotlib` (`https://matplotlib.org/`).
- Pages `https://scikit-learn.org/stable/modules/clustering.html#clustering` and `https://scikit-learn.org/stable/modules/clustering.html#clustering-performance-evaluation` will be particularly useful.
- You can make use of other functions from scikit-learn or any other Python library which may be useful.

P1. **Given datasets** `dsxx1.txt` and `dsxx2.txt`, cluster them according to the following combinations:

- the *single linkage*, the *complete linkage*, the *average linkage* and the *ward* algorithms
- 2, 3, 4, and 5 clusters

and

(a) considering the **V-measure**:
- calculate the metric value for all cases
- select the best number of clusters according to this metric for each algorithm
- select the best algorithm and number of clusters according to this metric
- plot separately the dataset using the true labelling and the labelling derived from the best algorithm and number of clusters
- find the contingency table for the best algorithm and number of clusters

(b) considering the **Davies-Bouldin score**:
- calculate the metric value for all cases
- select the best number of clusters according to this metric for each algorithm
- select the best algorithm and number of clusters according to this metric
- plot separately the dataset using the true labelling and the labelling derived from the best algorithm and number of clusters
- find the contingency table for the best algorithm and number of clusters

To finish, comment on whatever you consider adequate regarding the results obtained, e.g. the score is not indicating the real number of clusters, whether the score for the real number of clusters is close or not to the optimum one and therefore ..., etc.

P2. (a) Following the pseudocode at slide 7 of the lecture notes, write a **crisp clustering function** for straight line-shaped clusters, 2D samples and assuming that the true straight lines contain point (0,0). This function must match the following definition:

`def do_crisp_clust(X, M, n_iter, n_attempts, eps)`

where: X is the dataset, M is the number of clusters, *n_iter* is the maximum number of iterations per attempt, *n_attempts* is the number of attempts to perform (to counteract the random initialization of the parameters of the clusters, $\Theta(0) = \{\theta_j(0)\}$; the clustering leading to the best final value of the cost function J has to be returned) and *eps* is such that the clustering is stopped as soon as $J(t) - J(t-1) < eps$.

See the appendix for the description of the proximity function to use and the derivation of the calculation of the cluster parameters.

(b) Following the pseudocode at slide 18 of the lecture notes, write a **fuzzy clustering function** for the same case as the previous point, performing the necessary adaptations of the previous function. This function must match the following definition:

`def do_fuzzy_clust(X, M, n_iter, n_attempts, eps, q)`

where q is the fuzzyfier.

(c) **Given dataset** `dsxx3.txt`:

- Using the crisp clustering function, set $M = 3$ and 10 attempts for clustering, adequate values for the maximum number of iterations, e.g. 100, and for the termination criterion value, e.g. 10^{-3}, and
  - plot the value of J along the iterations performed for the best clustering (that leading to lowest J)
  - plot separately the dataset using the true labelling and the labelling derived from the best clustering
  - find the contingency table and calculate the V-measure for the best clustering
- Using the fuzzy clustering function, set $M = 3$ and 10 attempts for clustering, adequate values for the maximum number of iterations, e.g. 100, and for the termination criterion value, e.g. 10^{-3}, $q = 2$, and
  - plot the value of J along the iterations performed for the best clustering (that leading to lowest J)
  - plot separately the dataset using the true labelling and the labelling derived from the best clustering
  - find the contingency table and calculate the V-measure for the best clustering

P3. (a) Read the description of the **density-based** clustering algorithm DBSCAN available in Aula Digital and have a look at the function available in `scikit-learn` regarding this algorithm (`https://scikit-learn.org/stable/modules/clustering.html#dbscan`). NOTE: Notice this algorithm does not need to set the number of clusters M.

(b) **Given dataset** `dsxx4.txt`:

- choose the best performing values for the two main parameters of DBSCAN: $\epsilon \in \{0.1, 0.2, 0.3, 0.4\}$ and $q \in \{3, 5\}$
- plot the labelling resulting from every configuration and calculate the V-measure
- find the contingency table for the best configuration

(c) Provide the same data as before for the *single linkage* algorithm ($M = 2$) so that we can compare with DBSCAN.

- A report of the work done, problem by problem and point by point, has to be released by February 24, 2021 in electronic form using the templates provided through *Aula Digital*. Notice that there is a *template for results* and another *template for the source code*. Zip the corresponding PDF files, together with an executable source file, either a notebook file (.ipynb) or a python file (.py), whatever you have used for solving the different problems (3 files in total).
- Provide the requested data and plots/figures at each point above, using different colours/markers for the classes. For figures, use appropriate titles, axis labels and legends to clarify the results reported.

- Suitable comments are expected in the source code.
- This work has to be done individually (see the number of group in *Aula Digital*).
- IMPORTANT NOTICE: An excessive similarity between the reports released can be considered a kind of plagiarism.

The following objects/functions of `scikit-learn` will be necessary/useful (among many others):

`sklearn.cluster.AgglomerativeClustering`

`sklearn.cluster.DBSCAN`

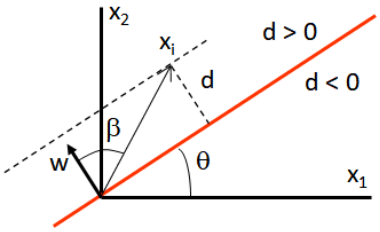
`sklearn.metrics.v_measure_score`

`sklearn.metrics.davies_bouldin_score`

`sklearn.metrics.cluster.contingency_matrix`

Appendix: distance between a point and a straight line

Let us consider a straight line containing point (0,0), such as the following one:



The unit vector aligned with the straight line direction is given by $(\cos\theta, \sin\theta)$, and, thus, the two unit normal vectors are $w = \pm(-\sin\theta, \cos\theta)$.

Given a point $x_i = (x_{i1}, x_{i2})$, the orthogonal distance d between x_i and the straight line is given by:

$$w^T x_i = \|w\|\|x_i\| \cos\beta = \|x_i\| \cos\beta = d$$

where d is, respectively, positive or negative depending on whether x_i is on the same half-plane w is pointing to or not.

Consequently, for this problem, we can define as proximity function between a sample x_i and a cluster C_j described by its normal vector w_j or, alternatively, by the corresponding angle θ_j:

$$\wp(x_i, C_j) = (w_j^T x_i)^2 = ((-\sin\theta_j)x_{i1} + (\cos\theta_j)x_{i2})^2 = (\mathrm{s}\,\theta_j)^2 x_{i1}^2 + (\mathrm{c}\,\theta_j)^2 x_{i2}^2 - 2(\mathrm{c}\,\theta_j)(\mathrm{s}\,\theta_j)x_{i1}x_{i2}$$

(We consider the squared distance in order to get rid of the sign. Notice also that $\mathrm{s}\,\theta = \sin\theta$ and $\mathrm{c}\,\theta = \cos\theta$.)

Step 3.3 of GHAS involves *solving for* $\theta(j+1)$ *in* $\sum_{i=1}^{N} u_{ij}(t)\frac{\partial \wp(x_i,\theta_j)}{\partial \theta_j} = 0$. For this case:

$$\frac{\partial \wp(x_i,\theta_j)}{\partial \theta_j} = 2(\mathrm{s}\,\theta_j)(\mathrm{c}\,\theta_j)x_{i1}^2 - 2(\mathrm{c}\,\theta_j)(\mathrm{s}\,\theta_j)x_{i2}^2 - 2(-\mathrm{s}\,\theta_j^2 + \mathrm{c}\,\theta_j^2)x_{i1}x_{i2}$$

$$\sum_{i=1}^{N} u_{ij}\frac{\partial \wp(x_i,\theta_j)}{\partial \theta_j} = 0 \Rightarrow 2(\mathrm{s}\,\theta_j)(\mathrm{c}\,\theta_j)\sum_{i=1}^{N} u_{ij}(x_{i1}^2 - x_{i2}^2) - 2(\mathrm{c}\,\theta_j^2 - \mathrm{s}\,\theta_j^2)\sum_{i=1}^{N} u_{ij}x_{i1}x_{i2} = 0$$

$$\Rightarrow 2(\mathrm{s}\,\theta_j)(\mathrm{c}\,\theta_j)\sum_{i=1}^{N} u_{ij}(x_{i1}^2 - x_{i2}^2) = 2(\mathrm{c}\,\theta_j^2 - \mathrm{s}\,\theta_j^2)\sum_{i=1}^{N} u_{ij}x_{i1}x_{i2}$$

$$\Rightarrow (\mathrm{s}\,2\theta_j)\sum_{i=1}^{N} u_{ij}(x_{i1}^2 - x_{i2}^2) = 2(\mathrm{c}\,2\theta_j)\sum_{i=1}^{N} u_{ij}x_{i1}x_{i2}$$

$$\Rightarrow \theta_j = \frac{1}{2}\tan^{-1}\frac{2\sum_{i=1}^{N} u_{ij}x_{i1}x_{i2}}{\sum_{i=1}^{N} u_{ij}(x_{i1}^2 - x_{i2}^2)}$$

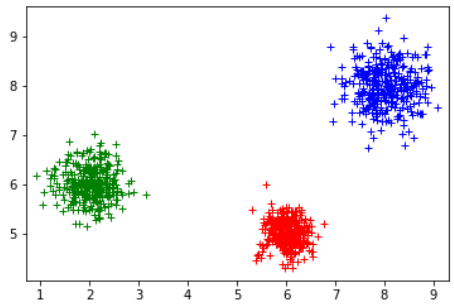
(When calculating $\tan^{-1}$ it is better to use function `math.atan2(num,den)` in order to handle correctly the angle quadrant.)

Step 3.3 of GFAS involves *solving for* $\theta(j+1)$ *in* $\sum_{i=1}^{N} u_{ij}^q(t)\frac{\partial \wp(x_i,\theta_j)}{\partial \theta_j} = 0$. Analogously to the previous case:

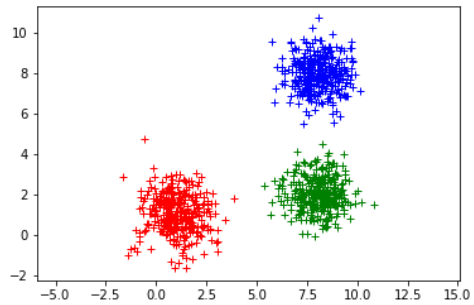
$$\theta_j = \frac{1}{2}\tan^{-1}\frac{2\sum_{i=1}^{N} u_{ij}^q x_{i1}x_{i2}}{\sum_{i=1}^{N} u_{ij}^q(x_{i1}^2 - x_{i2}^2)}$$

Appendix: examples of datasets

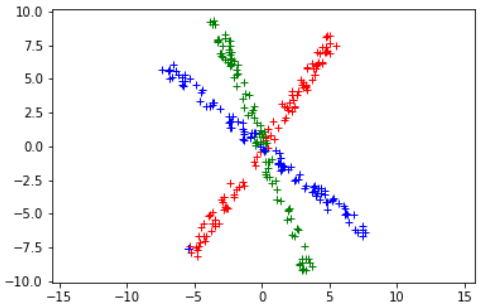
dsxx1



dsxx2



dsxx3



dsxx4