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

Handout #3: **Instance-based Learning**

NOTE 1: Problems P3 and P4 require *training* and *test datasets*. They are, respectively, stored in `dsxx34tr.txt` and `dsxx34te.txt` files:

```
import numpy as np
group = '01' # assuming group 1
ds = 34      # assuming problems 3 and 4
data = np.loadtxt('ds'+group+str(ds)+'tr.txt')
X_train = data[:, 0:2]
y_train = data[:, 2]
data = np.loadtxt('ds'+group+str(ds)+'te.txt')
X_test = data[:, 0:2]
y_test = data[:, 2]
```

Class labels are 1 for ω_1 and 0 for ω_2.

NOTE 2: Problems P3 and P4 also require the use of `scikit-learn` (`https://scikit-learn.org`) and `matplotlib` (`https://matplotlib.org/`). Apart from considering the library functions suggested at certain points, you can make use of others which may be relevant at each point (to this end, page `https://scikit-learn.org/stable/modules/classes.html` will be useful; sections `sklearn.svm`, `sklearn.neighbors`, `sklearn.preprocessing`, `sklearn.metrics` and `sklearn.model_selection` are of particular relevance).

P3. **Given datasets `dsxx34tr.txt` and `dsxx34te.txt`**, find a suitable SVM classifier adopting a *soft-marging* approach. You have to define the classifier design strategy, including data normalization, e.g. *min-max* scaling, and setting up the classifier hyper-parameters, e.g. by means of *grid-search*, as well as estimate the classifier performance by means of *n-fold cross validation*.

a) Define the design strategy: input data normalization, combinations of hyper-parameters considered (kernel and its parameters, and C), number of folds for the cross-validation process.

NOTE: typical values for C are $10^{-2}, 10^{-1}, 10^{0}, 10^{1}, 10^{2}$ and 10^{3}.

b) Using the *training dataset*, find the best performing classifier according to the design strategy and employing the accuracy as performance metric for the cross-validation process.

c) Generate the following plots in the original space:

1. a first plot with the *training samples*, highlighting the *support vectors* and plotting the 2D *decision curve*; and
2. a second plot with the *classification map*, i.e. evaluate the *decision function* for a 'regular' subset (grid) of points.

Use different markers and/or colours for each class. See the appendix for examples of the requested plots.

d) Report on the classifier performance using the *test dataset*:

1. measure the *test accuracy*, *test precision*, *test recall* and *test f1-score*; and
2. in a single figure, plot the *test samples* over the already calculated *classification map* (use different markers and/or colours for each class).

e) Obtain an improved estimation of the *accuracy*, *precision* and *recall* measures by means of *5-fold cross-validation*. To this end, put together the *training* and *test* datasets, so that the corresponding function can build the *folds* from all available data.

P4. **Given datasets `dsxx34tr.txt` and `dsxx34te.txt`**, find a suitable k-NN classifier (`KNeighborsClassifier` object of `scikit-learn`). You have to define the classifier design strategy, including data normalization, e.g. *min-max* scaling, and setting up the classifier hyper-parameters, e.g. by means of *grid-search*, as well as estimate the classifier performance by means of *n-fold cross validation*.

a) Define the design strategy: input data normalization, combinations of hyper-parameters considered (number of neighbours and distance function), number of folds for the cross-validation process.

b) Using the *training dataset*, find the best performing classifier according to the design strategy and employing the accuracy as performance metric for the cross-validation process.

c) Plot the training samples on top of the *classification map*, i.e. evaluate the decision function for a 'regular' subset (grid) of points of the feature space. Use different markers and/or colours for each class.

d) Report on the classifier performance using the *test dataset*:

1. measure the *test accuracy*, *test precision*, *test recall* and *test f1-score*; and
2. in a single figure, plot the *test samples* over the already calculated *classification map* (use different markers and/or colours for each class).

e) Obtain an improved estimation of the *accuracy*, *precision* and *recall* measures by means of *5-fold cross-validation*. To this end, put together the *training* and *test* datasets, so that the corresponding function can build the *folds* from all available data.

- A report of the work done has to be released by December 29, 2021 in electronic form as a notebook file (.ipynb).
- Provide the requested data and plots/figures at each point above. 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*).