**MLSM 2023 - Quiz 5**
**03/11/2023 - 15/11/2023 at 23:59**

---

(1) (1.0 pt.)

Which option is a valid statement relating to the Multi-layer perceptrons.

(1) It is NP-hard to find the parameters minimizing the empirical error for a network with one single hidden layer that contains 10 neurons.
(2) The function NOT(XOR(.,.)) can only be represented by a Multi-layer perceptron of at least 3 hidden layers.
(3) Learning an arbitrary Boolean function with MLP can be realized with polynomial number of nodes and data examples.
(4) If the activation functions are linear in all layers in a Multi-layer perceptron then that network can not be replaced with a network containing only a linear function mapping the input layer into the output.

(2) (1.0 pt.) Assume 15 base learners with independent true risk 0.35 ($\epsilon = 0.35$), are grouped to form a strong learner via majority voting. What is the probability of having an incorrect aggregated prediction (using the formula given on the slide "Why can model combination work? A thought experiment" of Lecture about "Ensemble Learning" ):

(1) 0.45
(2) 0.05
(3) 0.15
(4) 0.11

(3) (1.0 pt.)

Which option is a valid statement relating to the ensemble learning.

(1) The decision tree learners can not be applied in the AdaBoost as weak learners.
(2) The weights of the data examples, $D_t(i)$, might take negative values in the AdaBoost algorithm.
(3) An ensemble which is combined from base learners of a given hypothesis class can only learn patterns which are learnable by at least one element of that class.
(4) In the AdaBoost algorithm the weight of a learner, $\alpha_t$, could be negative if the error of the corresponding weak learner, $\epsilon_t$ is greater than 0.5.

(4) (2.0 pt.)

In this question the AdaBoost, an ensemble learner mentioned in Lecture 9, is applied on Multi-layer perceptrons(MLP) in classification problems. The distinction between MLP learners is expressed by the number of hidden nodes. The programming environment, the hyper-parameters and the data generation process are described in the example file `quiz_5_question_4_hint.py`. Similarly to the Question 4 of Quiz 4 you can follow the structure of the example code in developing your version of the solution.

The main programming task is the implementation of the AdaBoost applied on a set of MLP learners which needs to contain the training and prediction functions corresponding to the AdaBoost. Finally, the performance of ensemble learner needs to be measured on two datasets by F1 score, and to provide those learners whose weight are the highest in the ensembles corresponding to the two datasets. In the answer the learners are identified by the number of hidden nodes.

Here is the summary of the task, further details can be found in the example code.

The datasets are taken from sklearn:

```
from sklearn.datasets import make_circles, make_moons
make_circles(n_samples= 1000, noise=0.3, factor=0.5, random_state=0),
make_moons(n_samples = 1000, noise=0.3, random_state=0)
```

The learners, classifiers, are given by the help of the sklearn methods:

```
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.neural_network import MLPClassifier

## coonstruct the list of learners
learner_names = []
classifiers = []
for nnode in lnodes:
  classifiers.append(
    make_pipeline(
      StandardScaler(),
      MLPClassifier(
        solver="adam",
        alpha=0.0001,
        random_state=0,
        max_iter=iteration,
        early_stopping=False,
        hidden_layer_sizes=[nnode]),   ## set the layers
    )
  )
  learner_names.append("mlp_node_"+str(nnode))
```

Question: Run the training and test of the AdaBoost on the two datasets and provide the two F1 scores computed on the datasets, and the learners whose weight in the ensemble is the highest one for both datasets. The learner is identified by the number of hidden nodes.

The answer consists of 4 numbers:

- F1 score on `make_circles` dataset,
- F1 score on `make_moons` dataset,
- Number of hidden nodes of that learner whose alpha value is the highest on the `make_circles` dataset,
- Number of hidden nodes of that learner whose alpha value is the highest on the `make_moons` dataset.

(1) 0.77, 0.90, 10, 10
(2) 0.75, 0.92, 100, 100
(3) 0.82, 0.88, 20, 50
(4) 0.80, 0.93, 10, 100