MALIS Project 2:
The perceptron

The perceptron is a machine learning algorithm for learning a binary classifier of the form:

$$\hat{y} = h(\boldsymbol{x}) = sign(\mathbf{w}^T \boldsymbol{x} + \boldsymbol{b})$$

where w is a vector of real-valued weights and b is denoted as the bias.

Objectives

By executing this project, you will be able to

1. Gain understanding on the principles governing the perceptron algorithm
2. Strengthen your understanding of the gradient descent algorithm.
3. Familiarize with the process of training, validation and testing
4. Improve proficiency in the use of python, Jupyter, libraries used in machine learning (pandas, numpy, matplotlib, etc) and programming, in general.
5. Test your creativity by designing a solution to the stated problem

Part I – Implementing a perceptron algorithm:

Task 1. Your first task will be to code the perceptron algorithm. In attachment you have been provided with the file `perceptron.py`. This contains the skeleton of a Python class that you will need to complete with the necessary code. In particular, you will need to implement the following functions:

1. `train` – with the necessary steps to train a perceptron
2. `predict` – with the necessary steps to predict the labels y of a group of samples X.

You may add any other functions that you consider necessary.

Make sure you debug your code, to verify that it works accordingly. You may consider comparing it against scikit-learn implementation for validation.

Task 2. Document your code

Part II – Using the perceptron:

You will now use your coded perceptron to classify digits from the UCI ML handwritten digits dataset (best known as MNIST). To this end, create a jupyter notebook denoted `experiment.ipynb`. This will contain all your experiments.

Task 3. Load the dataset and using the load_digits function in scikit-learn. Check its documentation to see how you can use it (link). The dataset contains digits from 0 to 9.

Task 4. You need to implement an algorithm to classify the MNIST digits using a perceptron (no multi-layer perceptron). Consider splitting your data into training, validation and test. You can use the train_test_split function from scikit-learn.

Task 5. Train your perceptron using the training set. Use the validation set to choose a model. Report the selected model's accuracy using the testing set. Report your results.
Hint: The digits dataset is a dataset of images. You need to convert them from a 2D array to a 1D one.

Report:

You need to prepare a 1000 words (max) report explaining your solution. How does your method work? Include information about how your model was trained, the results obtained in the validation set and the strategy you used to choose a model (if any).

Contributions: Please include a section that describes what each team member worked on and contributed to the project. This is to make sure team members are carrying a fair share of the work for projects. This section does not count towards the total word count.

Deliverables:

Upload a zip file containing the report, the perceptron.py file, experiments.ipynb and any instructions required to run your code.

Important:

- Failing to submit a report leads to a mark of zero (0).
- If ChatGPT is used, failing to report it and explaining its use leads to a mark of zero (0).