Opened: Tuesday, 18 November 2025, 4:00 PM
Due: Monday, 1 December 2025, 11:59 PM

- Task 1: Obtain and prepare the data
- Task 2: Task 2: Build a plain autoencoder network, use it for embedding mages of digits
- Task 3: Perform cross-validation and model selection

Task 1: Obtain and prepare the data

Download the mnist data set. See Assignment 2, and recall that you can simply do:

```
from tensorflow.keras.datasets import mnist
(X_train, y_train), (X_test, y_test) = mnist.load_data()
```

In the data preprocessing section of your code, select two classes among the available ten (0 1 2 3 4 5 6 7 8 9). Make this selection parametric so that you can perform experiments with different pairs, e.g.

```
class0 = 5; class1 = 7
```

or

```
classes = (4, 2)
```

(or whatever you like, how you store classes is irrelevant and the above are just suggestions, not prescriptions).

Task 2: Build a plain autoencoder network, use it for embedding mages of digits

Here you have to create a shallow neural network. It should have:

- As many output units as input units (for use with the MNIST data, that means d = 784)
- Two (2) hidden units
- Sigmoid activation on the hidden layer
- Linear activation on the output layer

Train the network with this configuration on the MNIST data, using X_train in place of y_train, and MSE as the loss.

Hints:

- You may have to use a small learning rate and do more than one learning trial to obtain convergence to a low local minimum.
- If you are using Keras, the easiest way to be able to use just a part of the network (take the output at the hidden layer, not using the output layer) is to use the .

Task 3: Perform cross-validation and model selection

Here you have to create another program, based on the previous one. It should:

- Select a list of possible values of *h*, and for each:
  - Run k-fold cross-validation and obtain *k* estimates of quality
  - For each "fold", repeat training several times (multi-start approach) and collect the best result, saving it in a list
  - Perform statistics on the list and store the result in *another* list, as a typical `mse` value and a typical range `[mselower, mseupper]` (e.g. median, 25th percentile, 75th percentile)
- Select the best value for *h* based on the cross-validated test results.

Remark: Note that this is a multi- (two-)criterion optimisation: we want the best typical mse (low error), but also the narrowest range (good generalisation).

This is an instance of the bias/variance dilemma. As such, there is no hard-and-fast rule to decide. Explain your choice.

Hints:

- Note that your code will have a certain number of nested loops: outermost, on values of h; then on k "folds"; then on a number of restarts.
- Larger errors may have larger variability. It may then be advisable to compare not directly the range extension, but the ratio `(mseupper-mselower)/msetypical`. This is equivalent to reasoning in log scale. Take care of possible numerical issues (e.g. what if typical mse = 0?)
- For each value of h, store all the histories to plot. Decide afterward, by inspection, which ones are interesting and should be plotted.

Add submission

Submission status

?

Attempt number	This is attempt 1.