Assignment 1
E-mail your solutions *and* a zip-file with your python scripts to a.schoenhuth@cwi.nl before November 13, 2019, 3 pm. In your email include your 7-digit UU username. Include a python script for each sub question (except b)), so your zip file will eventually contain the following scripts: exercisea.py, exercisec.py, exercised.py and exercisee.py.

Exercise
During the tutorial we built a basic neural network for the MNIST dataset. In this exercise you will expand and improve upon that network. Use (only when applicable!) 50 epochs, a batch size of 100, 5 fold cross validation and gradient descent optimization with a learning rate of 0.5. Use cross entropy as a loss function with softmax activation in the output layer. Use your 7-digit UU username as the random seed for both cross validation and the variable initializers.

a) Build a neural network with three hidden layers. The first hidden layer has 512 nodes, the second layer has 256 nodes, and the third layer has 128 nodes. Use the sigmoid function as the activation function in each layer. *[5 points]*

b) Train the network from exercise a), plot the training loss over the epochs and evaluate and report the final performance of the model (indicate which metric you used). Describe the steps you took for training. *[3 points]*

c) Optimize the network architecture by finding the best among the list of architectures below. Describe the steps you took for training. Report the loss values for each of the architectures, choose the best model (and motivate your decision) and report its final performance (indicate which metric you used). Use the sigmoid function as the activation function in each layer. *[5 points]*
(i) One layer with 128 nodes;
(ii) One layer with 256 nodes;
(iii) One layer with 512 nodes;
(iv) Two layers where layer 1 has 64 nodes and layer 2 has 32 nodes;
(v) Two layers where layer 1 has 256 nodes and layer 2 has 128 nodes.

d) Continue with the best architecture from exercise c). Apply a dropout to each of the hidden layers, where the probability that a node is kept is 0.5. Train the model and report the relevant loss values. Does dropout improve the performance of the model? Motivate your answer. *[4 points]*

e) Continue with the best architecture from exercise c) (so without dropout). Add l2 regularization. Train and validate the model for the following scaling factors of the l2 regularization: 0 (so no regularization, same as the model in c)), 0.01, and 0.001. Report the relevant loss values. Which model performs best? Motivate your answer. Report the best model's final performance. *[4 points]*