MACHINE LEARNING FROM DATA
Fall 2018

Report: Lab Session 5 – Neural Networks

Names: Santagiustina Francesco, Simonetto Adriano

Instructions

- Answer the questions
- Save the report and Matlab code in a folder, and upload the compressed folder (zip, rar).

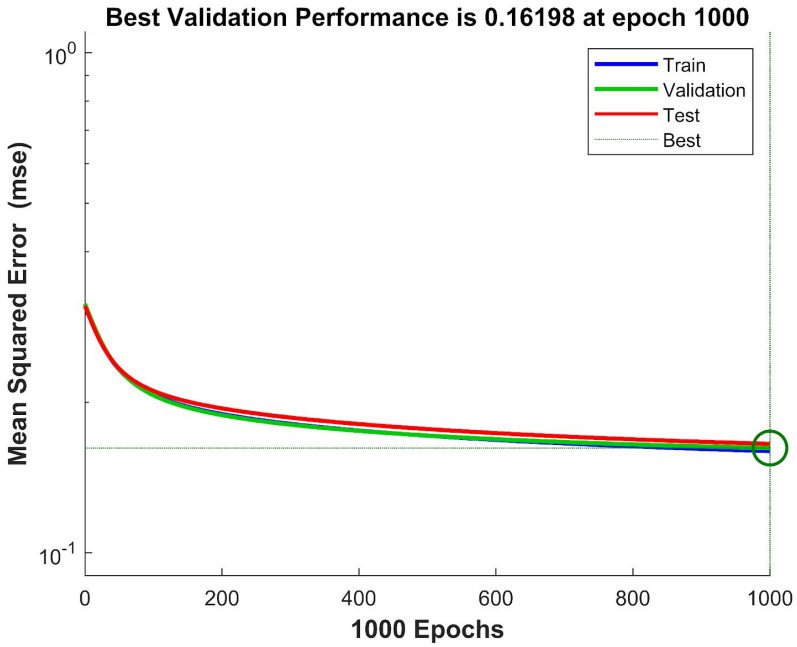
Questions

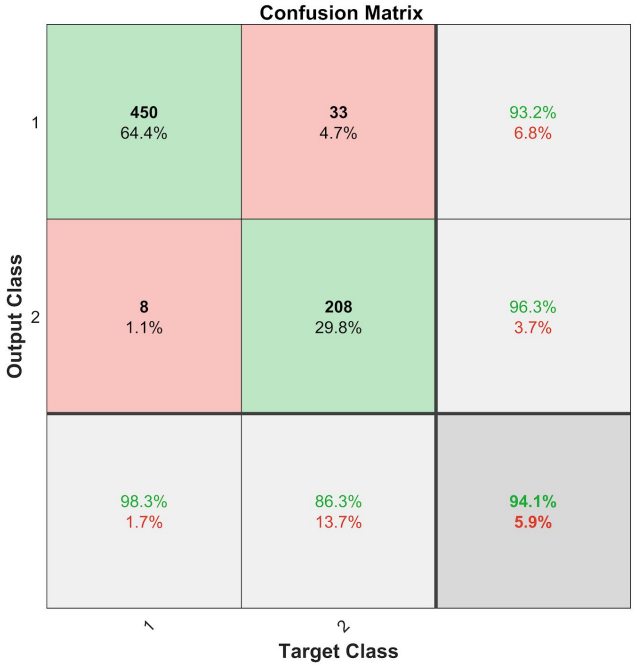
Q1: List the main network features: training function, performance function, number of hidden layers, learning rate (for traingd), epochs, early stopping criterion. You will need to search Matlab help documentation to find the default values of some of the hyperparameters (ex. help traingd)

Training function: Gradient descent
Performance function: Mean squared error
Number of hidden layers: 10
Learning rate of gradient descent: 0.01 (default)
Number of epochs: 1000 (default)
Early stopping criterion: stop if the validation performance has increased more than 6 times (default) since the last time it decreased

Q2: Train the network using gradient descent. Copy Performance and Confusion figures and discuss the results.

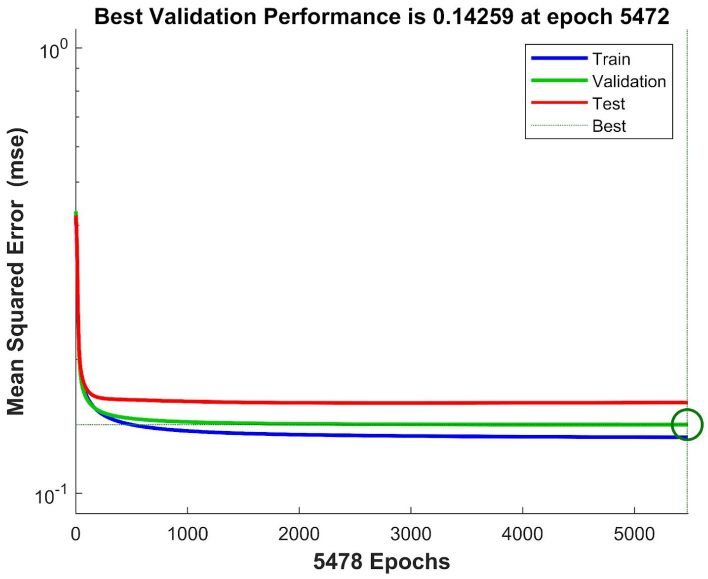
In the performance plot we can see how the errors on training, validation and test set, keep decreasing monotonically up to 1000 epochs. The fact that the error on the test set follows such behavior, means that there is still room for improvement, and suggests that using a higher number of epochs would lead to a better performance. In practice, we would like the network to stop training not because the maximum number of epochs has been reached, but because the error on the validation set is not decreasing anymore (which means that we found the set of parameters minimizing the error on the validation set, and if everything went right, also on the test set).
Observing the confusion matrix, we can see that 6.8% of the benign cells are misclassified as malign. The issue is amplified by the fact that the number of benign cells is much higher than that of the malign ones (483 vs. 216). This leads to a not so good precision (86.3%). Other parameters as recall and specificity have instead a better performance (around 95%).

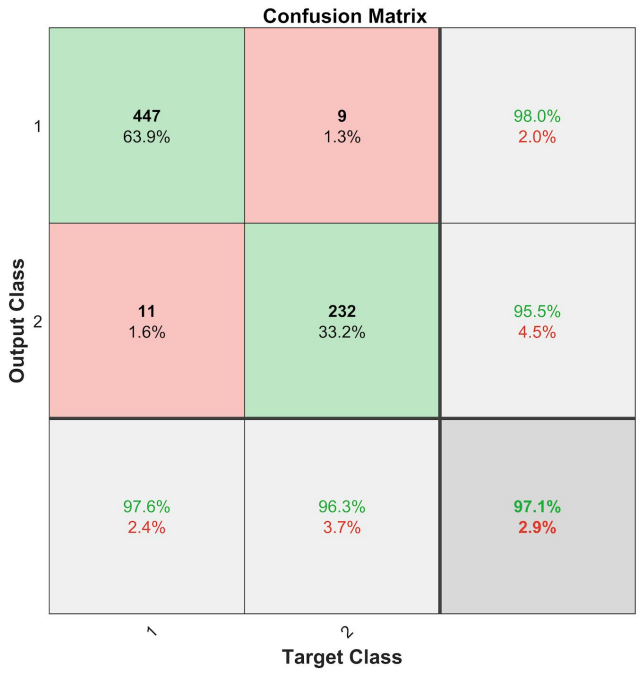




Q3: Repeat the experiment in Q2 changing the values of the learning rate (ex. 0.05 or 0.1). If necessary, change the default number of epochs so that the training stops when there are no improvements on the validation set. Discuss the results

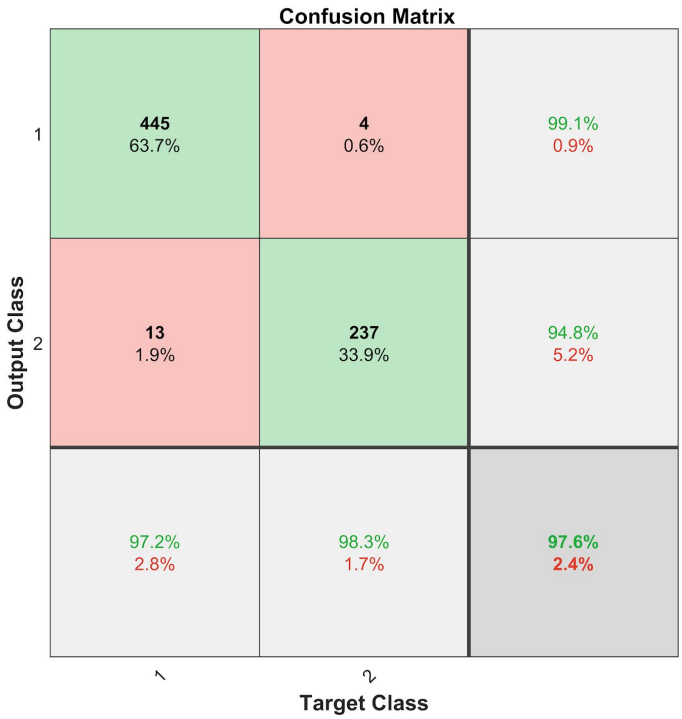
Using a learning rate of 0.1 and a maximum amount of epochs of 10000, we see that this time the training stops after 5472 epochs (due to the early stopping criterion).
We see that, as expected, the validation error is lower than before (0.14259 vs. 0.16198) and the same can be said about the error on the test set (0.1599 vs. 0.1759). Looking at the confusion matrix we notice how all the metrics more or less improved or kept the same value (all above 95%) and in particular the precision jumped from a value of 86.3% to 96.3%, showing a much better performance.





Q4: Train the network using Levenberg-Marquadt (net.trainFcn='trainlm'). Compare with gradient descent in terms of convergence speed and performance.

The adoption of Levenberg-Marquadt leads to a huge improvement concerning the convergence speed. In practice, even if we change the early stopping criterion (using a value of 10 instead the default 6), the algorithm stops after just 5 epochs. Moreover, we can see how this, far from affecting the overall performance, leads to results comparable, if not better, than the ones found adopting gradient descent. In this case validation and test error are respectively 0.13353 and 0.1430, lower than before (0.14249 and 0.1599).
Same can be said for the confusion matrix, where we can see that all performance metrics are extremely high (where just the recall is at 94.8%, and all the other metrics peak at around 97-99%).

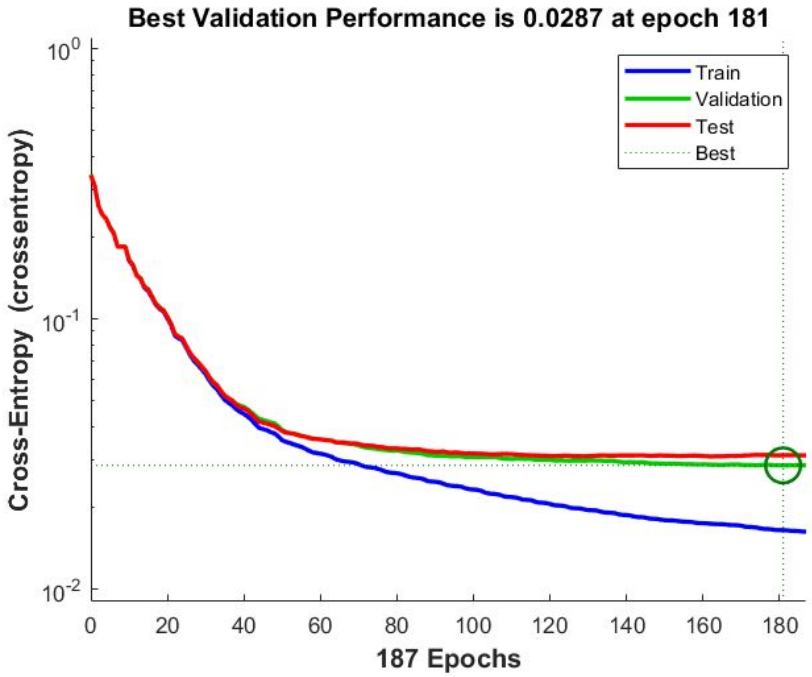


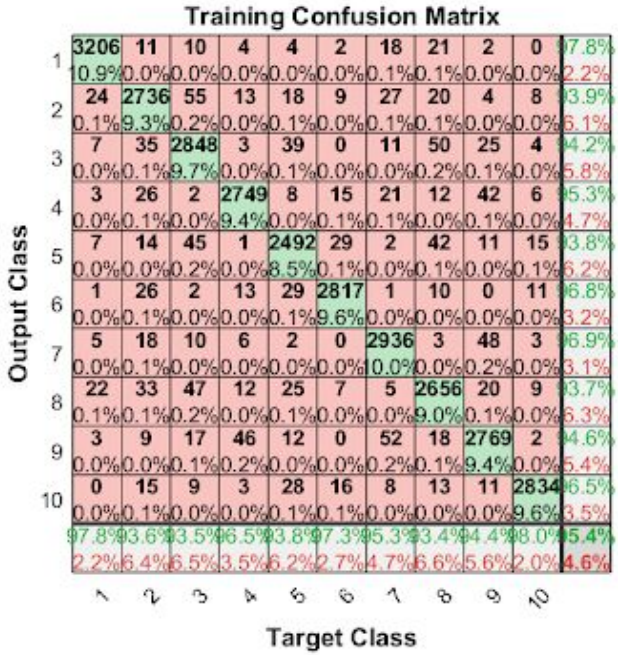
Q5: Which are the network settings and hyperparameters? (training algorithm, performance function, number of epochs, etc.)

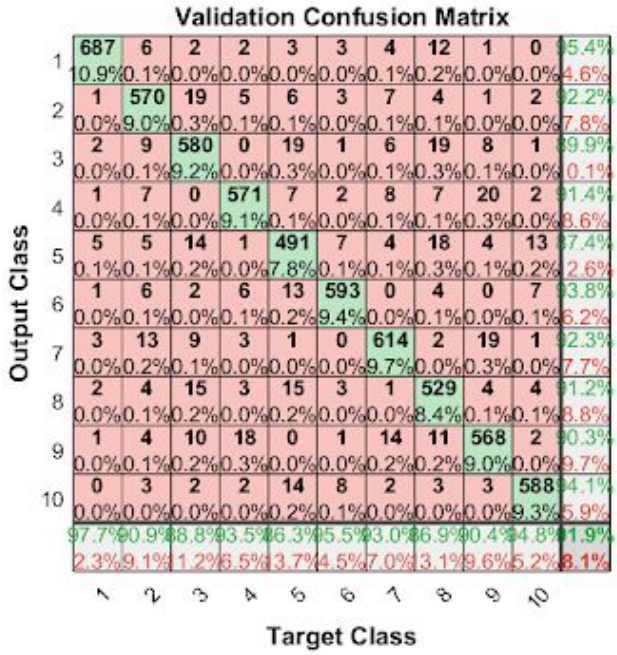
Training algorithm: Scaled conjugate gradient
Performance function: Cross-Entropy
Number of hidden layers: 10
Number of epochs: 1000 (default)
Minimum performance gradient : 1e-6 (default)
Early stopping criterion: stop if the validation performance has increased more than 6 times (default value) since the last time it decreased ; or if the performance gradient falls below 1e-6 (default value).

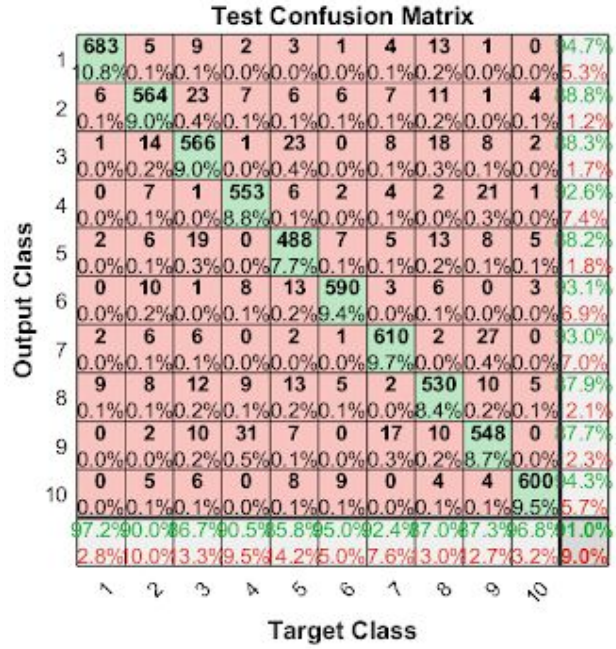
Q6: Compute confusion matrices and performance plots for the training, validation and test sets. Insert the figures in the report. Discuss results.

The early stop is reached quite quickly, at epoch 181. There, we obtain a validation error of 0.0809 and a test error of 0.0903 so about an error each 10 predictions. Confusion matrices nevertheless show us that errors are not homogeneously distributed. For instance it is much easier to correctly recognize a 1 (only 2.8% of error in the test set) than a 5 or an 8 (14.2% and 13.0% of errors in the test set).









Q7: Try to improve the network performance by increasing the number neurons in the hidden layer. Edit the script, add the code to validate the number of neurons. Choose the number of neurons that minimize the classification error on the validation set. Try with the following numbers: 10, 50, 100, 150, 200, 250, 300. Compute the error on the test set.

We found that the best results on the validation set are obtained with 200 hidden layers. For this value we obtain on the test set an error of 0.0357. Approaching the optimal number of layers by validation allowed us to divide the probability of error by almost 3!

Q8: Plot the network accuracy vs the number of hidden layers