B(E)3M33UI — Exercise M: Trainning simple MLP using Backpropagation

Petr Pošík

May 23, 2017

The goals of this exercise:

- learn about multi-layer perceptron (MLP) networks
- implement backpropagation algorithm
- use MLP from scikit-learn library

1 Backpropagation for simple MLP

Suppose that we are given the following training data, cf. Figure 1. The task is to create a simple neural network classifier that would be able to discriminate between red and blue.

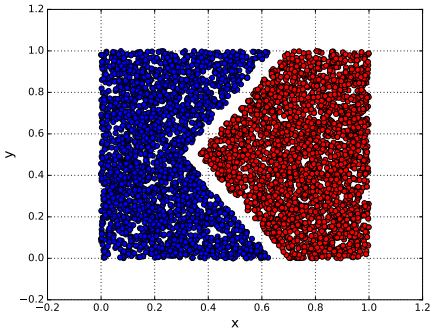

Figure 1: Trainning data

1.1 Preliminary questions

Task 1: Can one simple perceptron classify the data correctly?

Task 2: How many linear boundaries do we need here?

Task 3: What architecture of the network will be suitable?

Task 4: We need to tune the NN to the training set. What is actually tuned?

1.2 Design of neural network

The network architecture is 2-2-1 and is shown in Figure 2

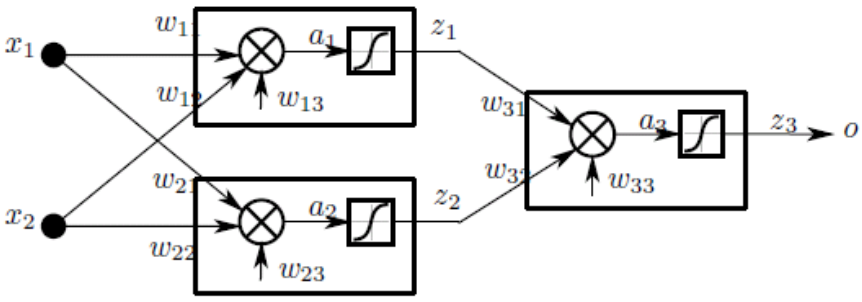

Figure 2: Architecture of simple MLP

Task 5: How many weights have to be tuned in our network?

Task 6: For the given multi-layer perceptron implement the backpropagation algorithm to estimate weights $\mathbf{w}$.

Notes:

- MLP learning procedure (cf. lectures on NN, slide 18):
  1. Starting at the input layer, we forward propagate the patterns of the training data through the network to generate an output.
  2. Based on the network's output, we calculate the error that we want to minimize using a cost function that we will describe later.
  3. We backpropagate the error, find its derivative with respect to each weight in the network, and update the model.

Task 7: Visualize the decision boundary for the training set.

2 MLP from scikit-learn

Task 8: Use the `MLPClassifier` from scikit learn with the same architecture (2-2-1) and plot the decision boundary.

3 Experiments

Consider another training set, the so called XOR problem in Figure 3. These data are in the file:`data_xor.csv`.

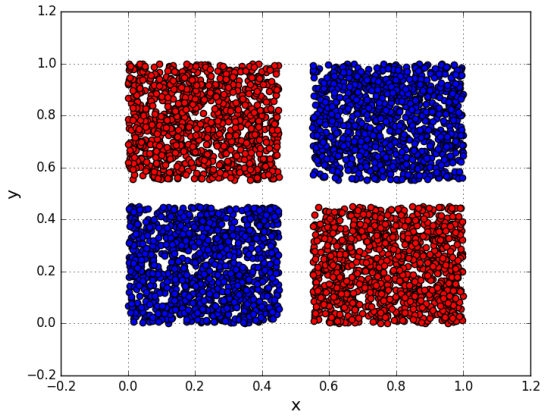

Figure 3: Trainning data - The XOR problem

Task 9: Use the same architecture. Can you classify the data correctly?

Task 10: Increase the complexity of network by increasing number of hidden units.

Task 11: Visualize the decision boundaries.

Hints:

- Try to zoom out from figure, e.g. by changing limits in plotting.py.
- Are you happy with the decision boundaries outside of available data?

4 Have fun!

Complete the exercise as a homework, ask questions on the forum, and upload the solution via Upload system!