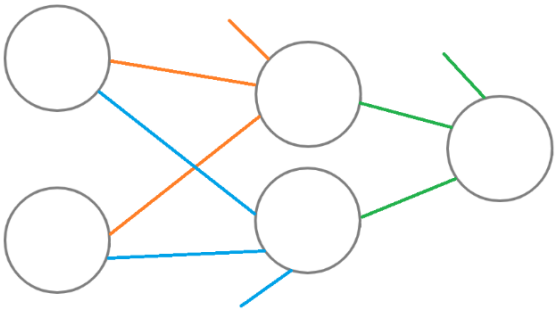
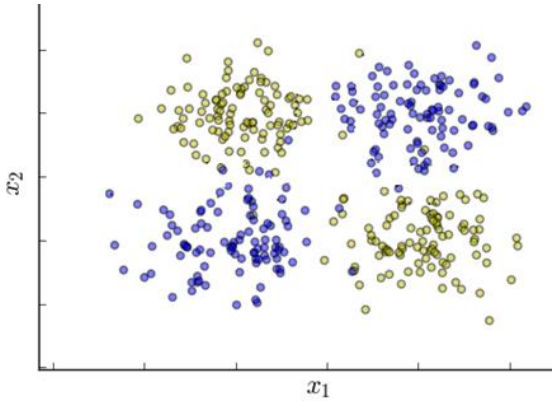
Neural Networks
Homework 3

In this homework, you will implement backpropagation to train a neural network (Figure 1.a) for classifying the noisy XOR data (Figure 1.b). Do not implement a generic code that will run for all the network architectures, instead implement the feed-forward and backpropagation just for the specific neural network given in Figure 1.a. Use hyperbolic tangent activation function and quadratic loss. Note that, necessary details are given in the backpropagation lecture. In that lecture, derivatives for weights are shown for single data for the sake of simplicity. Use batch mode and calculate the derivative for all the data by simply averaging the derivatives over all data (see homework 1). Please try to reuse the computations to improve computational efficiency of your implementation. Also, use backpropagation for MLP provided in Scikit-learn (https://scikit-learn.org/stable/modules/neural_networks_supervised.html) and verbally compare it with your implementation (i.e. differences, similarities).



a. Neural Network



b. Noisy training data

Figure 1. Neural Network that solve XOR data

- Create your own data that resembles above data set and visualize it (10 points)
- Implement feed-forward phase (10 points)
- Implement (weight and bias update via gradient descent) backpropagation (30 points)
- Do training in batch mode with your implementation, (15 points)
- Show training error and accuracy per epoch (5 points)
- Train another network with the same network structure using Scikit-learn (20 points)
- Compare your implementation and Scikit-learn implementation (10 points)

Submit your solution as a *Jupyter Notebook* file (write your comparison or notes as comments). In this, homework I expect you to use pair-programming practices (if possible) since one can make simple mistakes easily. Only one student should submit the homework, through Aybuzem (state student names and surnames at the top of the *Jupyter Notebook* code).