SGN-13006 Introduction to Pattern Recognition and Machine Learning
Department of Signal Processing, Tampere University of Technology
Joni Kamarainen and Katariina Mahkonen
Exercise 5

Complete / start preparing your solutions before the exercise session. During the exercise session you can consult the teacher, and once finished your work, show your solutions to the teacher to get the exercise points.

1. **Perceptron as logical unit** (1 point)

   Consider a Perceptron unit that implements Boolean functions. The unit has two binary inputs, x_1 and x_2, two weights w_1 and w_2, and a threshold θ. The binary output is computed as follows:

   $$\begin{aligned} y &= 1, \text{ if } w_1 x_1 + w_2 x_2 \geq \theta; \\ &= 0, \text{ otherwise.} \end{aligned}$$

   We interpret the value 1 as true, 0 as false.

   Determine the perceptron weights and thresholds to implement the following Boolean functions:

   a) $NOT x_1$

   b) $x_1 OR x_2$

   c) $x_1 AND x_2$

2. **MLP classifier** (2 points)

   Fig 1 shows two classes of 2-dim. points that should be classified using an MLP neural network. Sketch the structure of a MLP network that has two inputs, two outputs and some suitable number of neurons in the hidden layer. The output vector should be (1,0) whenever the input belongs to class 1, and (0,1) whenever the input belongs to class 2. Determine the weight ($w_1 w_2$) and bias (θ) values for each hidden neuron according to the fact that a (decision boundary) line $x_2 = k * x_1 + b$ is implemented as $w_1 * x + w_2 * x_2 - \theta = 0$ by a neuron (perceptron). The output neurons should be able to handle the combination of the outputs from the linear classifiers (i.e. the perceptrons) of the hidden layer to produce correct outputs.

3. **Neural Network** (3 points)

   Data: We are using the tiny subset of Imagenet dataset. Use the Matlab template `imagenet_tiny5_process.m` to handle the data.

   Feature extraction: For Neural Network, we can use all kinds of features. Even using 3x64=192 pixel values may work. Try that, and/or invent new features.

   The neural Network works best so that there is one output for each class. Each output estimates the likelihood of one class. To do that, the class labels should be changed in following way:

   $$\begin{aligned} 1 &\rightarrow [10000] \\ 2 &\rightarrow [01000] \\ 3 &\rightarrow [00100] \\ 4 &\rightarrow [00010] \\ 5 &\rightarrow [00001] \end{aligned}$$

   Classifier training: There exists the Graphical User Interface of Matlab's Neural Networks Toolbox (`nnstart`), but for more insight, use the command line functions starting from network definition `doc patternnet` to train and use a neural network. Look at the net structure and find the neuron weight and bias values before and after training. Check also, what kind of transfer functions are used at each layer. Try different setups (number of layers and number of neurons at each layer) to search for the best performance.

   Note that Matlab's Neural Networks Toolbox expects both input and output data to be column vectors.

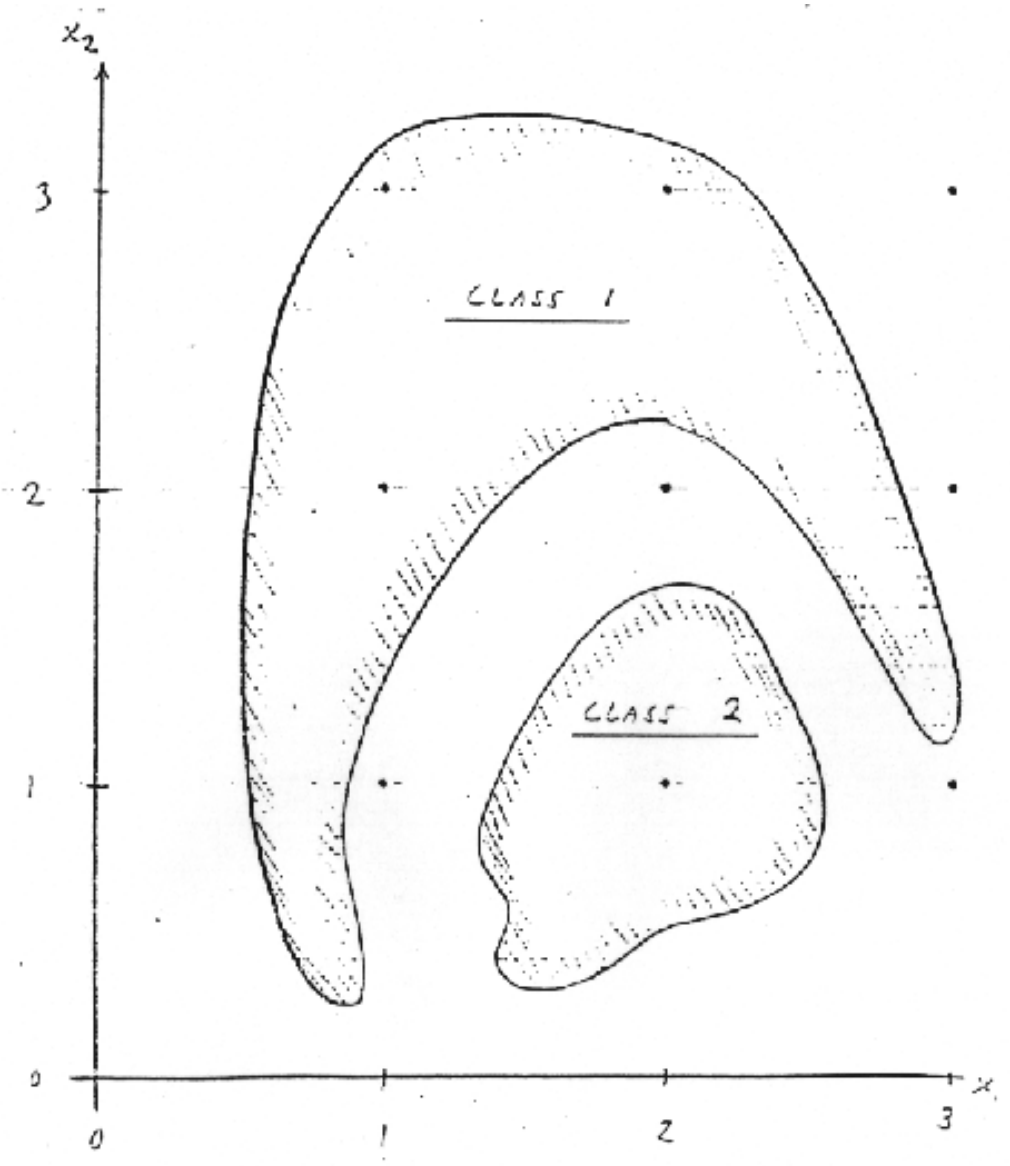

Figure 1: Classes to be classified.