

# Einführung in Künstliche Intelligenz SS 2014, 2.0 VU, 184.735

## Exercise Sheet 2 – Learning and Neural Networks

For the presentation part of this exercise, mark your solved exercises in **TUWEL** until Friday, May 16, 13:00 CET. Be sure that you tick only those exercises that you can solve and explain on the blackboard!

Please ask questions in the **TISS** Forum or visit our tutors during the tutor hours (see **TUWEL**).

**Exercise 1 (3 pts.):** Consider a classification problem with three attributes  $A, B, C$  with the domains  $V(A) = \{a_1, a_2\}$ ,  $V(B) = \{b_1, b_2\}$ ,  $V(C) = \{c_1, c_2\}$  and two classes  $K = \{T, F\}$ . For this problem, build a decision tree using the rule from the lecture of choosing the attribute maximising information gain in each step.

	$A$	$B$	$C$	Class
1	$a_1$	$b_2$	$c_1$	$F$
2	$a_2$	$b_2$	$c_1$	$F$
3	$a_1$	$b_2$	$c_2$	$F$
4	$a_1$	$b_1$	$c_1$	$T$
5	$a_1$	$b_1$	$c_2$	$F$
6	$a_2$	$b_1$	$c_2$	$F$

**Exercise 2 (2 pts.):** Compute the information gain for an attribute  $A$  that takes a different value for each example. What problem do you see?

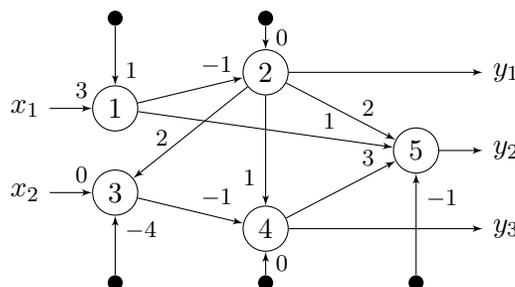
**Exercise 3 (2 pts.):** Consider again the problem in a). Again construct a decision tree, but in every step choose the argument that maximises the *relative information gain*, i.e., the ratio between their gain and their own intrinsic information.

$$GainR(A) = \frac{Gain(A)}{H(A)}$$

Thereby we have  $H(A) := \sum_{a \in V(A)} \frac{|E_a|}{V} \log_2 \frac{V}{|E_a|}$ , where  $V(A)$  denotes the set of possible values for the attribute  $A$  and  $E_a$  is the set of samples with  $A = a$  and  $V := \sum_{a \in V(A)} |E_a|$  is the total number of samples.

**Exercise 4 (2 pts.):** Suppose that an attribute splits the set of examples  $E$  into subsets  $E_k$  and that each subset has  $p_k$  positive and  $n_k$  negative examples. Show that the attribute has zero information gain if the ratio  $\frac{p_k}{p_k + n_k}$  is equal for all  $k$ .

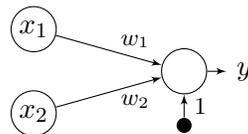
**Exercise 5 (1 pt.):** Consider a neural network with five nodes of the following form:



The numbers beneath the arrows denote the respective weights. Nodes 1 and 3 have the *identity function* as their activation function. Nodes 2 and 4 use the hard limiter with threshold 0 and node 5 uses a sigmoid function ( $1/(1 + e^{-x})$ ). The arrows from the black dots give the respective bias weights. What is the output produced by the network when the input is  $(x_1, x_2) = (1, 1)$ ? Note that the fixed input  $a_0$  is 1 for each node.

**Exercise 6 (2 pts.):** Construct a neural network with 4 input nodes  $x_1, x_2, x_3, x_4$  and one output node  $y$  which produces a parity bit of the respective input values, namely  $y = x_1 \text{ XOR } x_2 \text{ XOR } x_3 \text{ XOR } x_4$  holds. You are allowed to introduce hidden layers with additional nodes of arbitrary depth, but every node should use the hard limiter with threshold 0 as activation function. Please note that the fixed input  $a_0$  is 1, as presented in the lecture.

**Exercise 7 (2 pts.):** Consider a 2-layer perceptron with two input neurons and one output neuron of the following form:



Train the perceptron using the *Perceptron learning rule* and the identity transfer function on the following training data:  $f(0, 0) = 1$ ,  $f(1, 0) = 0$  and  $f(0, 1) = 0$ . The weights are initialised to 0 and the learning rate  $\alpha$  is 1. The bias weight will not be learned and stays 1.

**Exercise 8 (1 pt.):** Suppose you have a neural network with linear activation functions. That is, for each unit the output is some constant  $c$  times the weighted sum of the inputs. For simplicity, assume that the activation function is the same linear function at each node:  $g(x) = c \cdot x + d$ .

Assume that the neural network has one hidden layer. For a given assignment to the weights  $w$ , write down equations for the value of the units in the output layer as a function of  $w$  and the input layer  $x$ , without any explicit mention of the output of the hidden layer. Show that there is a network with no hidden units that computes the same function.