Exercise: 4

Meeting on January 26th / 28th

Problem 1: Maximin Strategy

In this problem you will look at various two-player-games with different reward and loss distributions. Using the maximin technique from the lecture, compute the respective maximin strategy for each of these games, where mixed strategies are allowed (aside from (d)). The tables are set up so that the first number of a cell belongs to the row player and the second number to the column player.

(a) First look at a simple coin toss, where both players choose a side of the coin simultaneously. Player B wins if they choose the same side, player A wins if the choose different sides.

	A: Heads	A: Tails
B: Heads	$1, -1$	$-1, 1$
B: Tails	$-1, 1$	$1, -1$

Table 1: Matching Pennies

(b) Now look at Two-Finger-Morra with different rewards:

	O: One	O: Two
E: One	$4, -4$	$-6, 6$
E: Two	$-4, 4$	$6, -6$

Table 2: Two-Finger-Morra variant

(c) Finally look at a variant of rock-paper-scissors. In this variant, gains and losses of rock are doubled and for scissors even tripled (i.e. if player A uses rock and player B uses paper, the loss of $A = -2$ the gain of $B = 1$). Model the game in its normal form first and then compute the maximin strategy, if only pure strategies are allowed.

(d) Again, list the normal form of rock-paper-scissors, this time with even gains and losses, and compute the maximin strategy for pure strategies.

Problem 2: Linear Regression (Theory) - Part 1

It was shown in the lecture that linear regression can be solved approximately with a Hill-Climbing-Algorithm.

(a) What is the linear modell for approximating the data $\{y_i, \vec{x}_i\}$?

(b) What is the equation for the error to be minimized by linear regression?

(c) What are the parameters that need to be optimized by the model?

(d) What property is given at the minimum?

(e) What is formula for the update rule of Gradient-Descent (Hill-Climbing)?

(f) Compute explicitly the derivative with respect to the optimization paramaters.

(g) Explain, how to integrate the bias w_0 into the parameter vector $\vec{w}$. What is the linear model then?

Problem 3: Linear Regression (Theory) - Part 2

Following, the bias will be integrated into the weight vector $\vec{w}$, which simplifies the following computations but keeps them exact. Linear regression is a special case of a differential equation which can be solved analyitically. This differential equation is:

$$\nabla_{\vec{w}} L = 0$$

(a) Compute the solution to the optimization problem analytically. Potentially, compute the problem first with only one component (i) of the gradient. (Solution: $\vec{w} = \vec{y}\bar{X}^{-1}$)

(b) Why does a large number of examples/data points pose a problem?

(c) Why does inserting a smart identity reduce the problem:

$$\vec{w} = \vec{y}\bar{X}^T \left(\bar{X}\bar{X}^T\right)^{-1}$$

Problem 4: Activation function

A Multilayer-Perceptron (MLP) is a kind of neural network and hence a simple, non-linear classifier, consiting of an input, a hidden and an output layer of "nodes"/vectors (see Fig. 1)

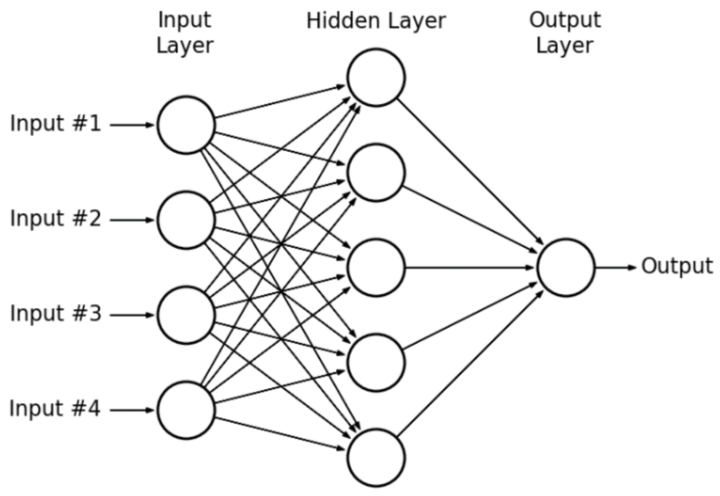

Figure 1: Beispiel für ein Multilayer-Perceptron

The hypothesis room $\mathcal{H}$ of MLPs is given by

$$y = f(\vec{x}) = \vec{w} \cdot r(\bar{V}\vec{x}) \ . \tag{4}$$

Where is r a non linear function, since the above equation could otherwise be reduced to a linear equation.

(a) Show this by setting $r(x) = x$. Following that, why is a non-linear activation function r necessary?

(b) Show that:

- $\sigma'(x) = \sigma(x)(1 - \sigma(x))$ with $\sigma(x) = \frac{1}{1+e^{-x}}$
- $\tanh'(x) = 1 - \tanh^2(x)$ with $\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$
- $\text{ReLU}'(x) = \begin{cases} 0 \text{ for } x < 0 \\ 1 \text{ for } x > 0 \\ \text{not defined for } x = 0 \end{cases}$ with $\text{ReLU}(x) = \max(0, x)$

Problem 5: Programming a perceptron

Given is a sequence of numbers $Z_i \in \{0,1\}$. Program and learn a multi-class perceptron with gradient descent as an optimizer, which predicts the next number from the sequence of the last 10 numbers. This means, the net has 10 input nodes with values $Z_{t-9} \rightarrow Z_t$ and should predict Z_{t+1}. Use softmax as the output function and one node for 0 and 1 respectively. The sequence for testing the algorithm is given by random numbers which are entered manually (by the command line) to the program.

Use the following sequence as a clue:

1. Initialize the saved sequence (i.e. last 10 elements) by e.g. random numbers.
2. Initialize the weight vector W of the perceptron with e.g. random numbers.
3. Repeat:
   (a) Compute the next prediction of the net p_{t+1}.
   (b) Read the next number z_{t+1} from the console.
   (c) Compare p_{i+1} and z_{i+1} and perhaps adjust W.
   (d) Save z_{t+1} into the sequence of the last 10 elements (input data).
   (e) Ouput of the percentage of all correctly predicted numbers.

What is the perceptron's accuracy? Hence, are you a reliable random number generator?