Exercise 4 - Backpropagation and unsupervised Learning

Winter term 2019/2020

name1, name2, name3

Important

Please solve the assignments in groups of 3 to 4 students. The solutions are going to be presented and discussed after the submission deadline. Sample solutions will not be uploaded. However, you are free to share correct solutions with your colleagues **after they have been graded**. Please submit your solutions via Moodle **and** in printed form. Only one member of the group has to submit the solutions. Therefore, make sure to specify the names of all group members. Please do not submit hand-written solutions, rather use proper type-setting software like LaTeX or other comparable programs.

Your homework will be corrected and given back to you. Correct solutions are rewarded with a bonus for the exam (max. 10 percent, if all solutions submitted are correct). **Please note:** You have to pass the exam **without the bonus points**! *(i.e. it is not possible to turn 5.0 into 4.0)* The solutions have to be your own work. If you plagiarize, you will lose all bonus points!

Further remarks:

- Code assignments have to be done in `Python`
- The following packages are allowed: `numpy`, `pandas`
  (please ask, if you want to use a specific package not mentioned here)
- **Do not use already implemented models** (e. g. from `scikit-learn`)

1 Backpropagation

a) Backpropagation by Hand (5 points)

You are given the neural network depicted below. Compute one forward-pass and one backward-pass based on the labeled training example $\langle \boldsymbol{x} = [0.05, 0.10], \boldsymbol{y} = [0, 1] \rangle$. Employ the squared error loss function:

$$\mathcal{J}(\boldsymbol{\Theta}) = (y_{pred} - y)^2$$

$$\mathcal{J}'(\boldsymbol{\Theta}) = 2 \cdot (y_{pred} - y)$$

The weight matrices and bias weights are initialized as follows (where for instance weight $\Theta_{12}^{(1)}$ connects the input x_2 with the first neuron in the hidden layer):

$$\boldsymbol{\Theta}^{(1)} = \begin{pmatrix} 0.15 & 0.20 \\ 0.25 & 0.30 \end{pmatrix} \qquad \boldsymbol{\Theta}^{(2)} = \begin{pmatrix} 0.40 & 0.45 \\ 0.50 & 0.55 \end{pmatrix}$$

$$\boldsymbol{b}^{(1)} = \begin{pmatrix} 0.35 & 0.35 \end{pmatrix} \qquad \boldsymbol{b}^{(2)} = \begin{pmatrix} 0.60 & 0.60 \end{pmatrix}$$

Use the ReLU and sigmoid activation functions in the hidden layer and output layer, respectively. Write down all necessary computations. What are the gradients for the weights $\Theta_{12}^{(1)}$, $\Theta_{21}^{(2)}$?

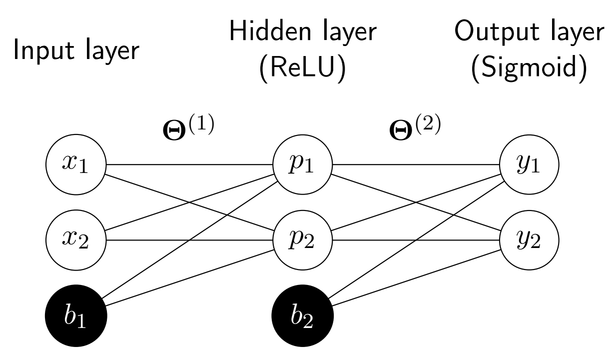

Figure 1: MLP architecture

Solution:

2 Unsupervised Learning

a) k-Means clustering (4 points)

Implement k-Means clustering for image compression. Compress the exemplary image file which can be found under the path `/data/dhbw.jpg`. You can find an explanation of how image compression with k-Means works on this web page.[1]

Solution:

b) PCA (1 point)

Explain how to choose the number of principal components for dimensionality reduction. Why does this work?

Solution:

[1] https://medium.com/@agarwalvibhor84/image-compression-using-k-means-clustering-8c0ec055103f

c) Bonus Question: Spectral clustering (1 point)

How can you automatically choose the number of clusters for spectral clustering?

Solution: