# Assignment1 Artificial Neural Network

Lukas Schnittcher Ernest Pokropek

November 2020

## 1 Structure of the Neural Network

Our Neural Network has the following structure.
Input-Nodes: 19 (One for each Input parameter)
Hidden Nodes: 200 (Give us the best Solutions after testing)
Output-Nodes: 1 (If the Neuron is firing the Class is 1, otherwise Class 0)
A Visualization is shown below, due to complexity reasons we only show 25 instead of 200 Hidden-Nodes.

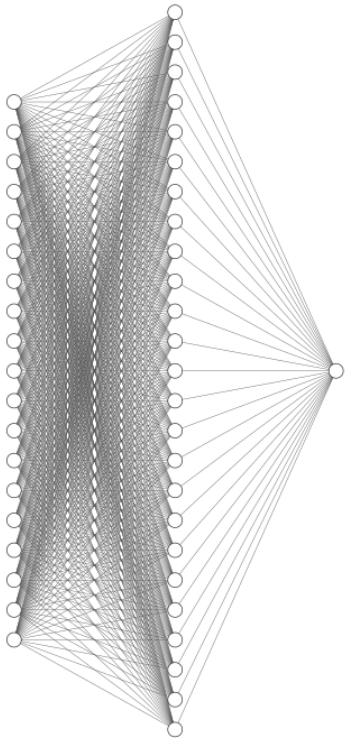

Figure 1: Structure of Neural Network

## 2 Used Equations

Inputvector $(19 \times 1)$ $x_i = \begin{pmatrix} x_1 & x_2 & \cdots & x_{19} \end{pmatrix}$

Weights between Input- and Hidden-Layer $(200 \times 19)$ $W1_{i,h} = \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,19} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,19} \\ \vdots & \vdots & \ddots & \vdots \\ a_{200,1} & a_{200,2} & \cdots & a_{200,19} \end{pmatrix}$

Weights between Hidden- und Output-Layer $(1 \times 200)$ $W2_{h,o} = \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,200} \end{pmatrix}$

Activation-Function (Relu) for Hidden-Layer $\varphi_1(x) = \max(0, x)$

Activation-Function (Sigmoid) for Output-Layer $\varphi_2(x) = \frac{1}{1+e^{-}x}$

Derivation $\varphi_1'(x) = \begin{cases} 1 & x \geq 0 \\ 0 & x < 0 \end{cases}$

Derivation $\varphi_2'(x) = \varphi_2(x) * (1 - \varphi_2(x))$

Output of Hidden-Layer $O1 = \varphi_2(\sum_{i=1}^{19} x_i * W_{i,h})$

Output of Output-Layer $O2 = \varphi_2(\sum_{h=1}^{200} O1_h * W_{h,o})$

Learningrate $\alpha = 0.1$

Target-Output: $T$

Error $\delta_O = \varphi_2'(\sum_{h=1}^{200} O1_h * W2_{h,o}) * (O2 - T)$

Error $\delta_H = \varphi_1'(\sum_{i=1}^{19} x_i * W1_{i,h}) * \delta_O * W1$

$W1 = W1 + \alpha * \delta_H * O1$

$W2 = W2 + \alpha * \delta_O * O2$

# 3 Validation Accuracy over Training Process

The validation was performed on randomly selected 10% portion of the whole dataset. Each epoch, the model's performance was measured on it and the results were saved before further computation. Finally, after 300 epochs, we receive an array of accuracy values over each training epoch. The results might be viewed on Figure 2, where the red line corresponds to polynomial approximation of accuracy over epochs to give a better idea of the overall trend during training.

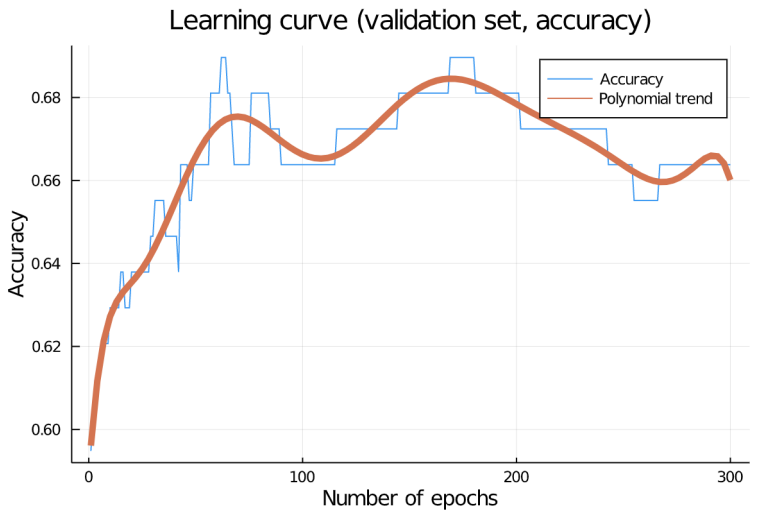


Figure 2: Learning curve based on validation data. Red line is a polynomial approximation of obtained metrics (blue line). Accuracy is defined as percentage of correctly classified examples (true positive + true negative) divided by number of all samples from the validation set.

The learning curve presented of Figure 2 shows a overally good trend for training at the beginning, yielding a huge increase of performance at the beginning. Furthermore, the curve stabilizes, which was expected, just to slightly decrease the performance after some time. This is most probably due to the overfitting phenomena, yet we still included it in this paper for our investigation.

# 4 Percentage of Correctness for test Data Set and both Classes

The accuracy on the test set (15% of the whole data) is equal to **74.71%** for hyperparameters and neural network's architecture as described in section 1. Furthermore, predictions for this set have following form:

- **Actual positives: 83**
- **Actual negatives: 87**
- True Positives: 55
- False Positives: 15
- False Negatives: 28
- True Negatives: 72

Please note that the numbers are corresponding to the number of samples for which model was tested on (170 in total).
Furthermore, Receiver Operating Characteristic (ROC) and Precision-Recall curves has been computed and may be seen on Figures 3 and 4 respectively.

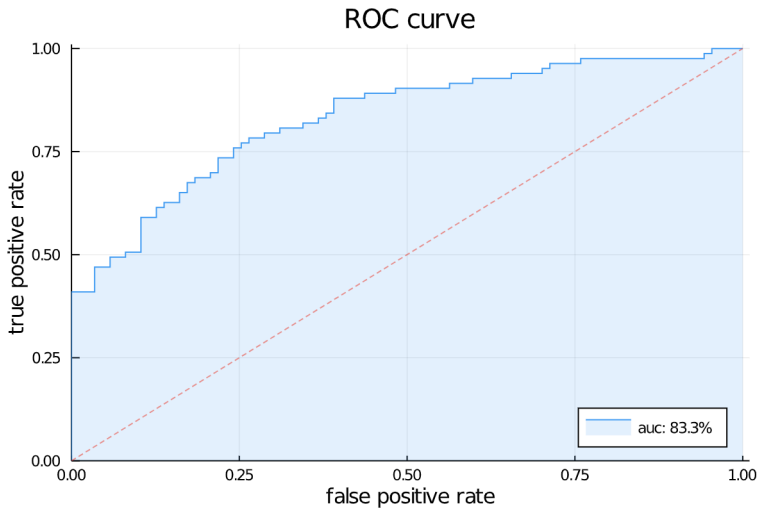


Figure 3: ROC Curve for the model over test data

The model shows a relatively good performance for even low FPR values (given Figure 3) and the overall shape of the ROC curve is sufficient for our application. The model performs way better than if the data was exposed to random choices (red dashed line) which yields promising results in future

development, in terms of hyperparameter tuning and modifying the network's architecture.

Dealing with medical data we have to take into account, that our model should have relatively high alarm ratio - we cannot miss any case of drastic disease, thus it is better to alarm a healthy patient rather than miss the ill one. For this analysis we include the precision-recall curve, visible on Figure 4.

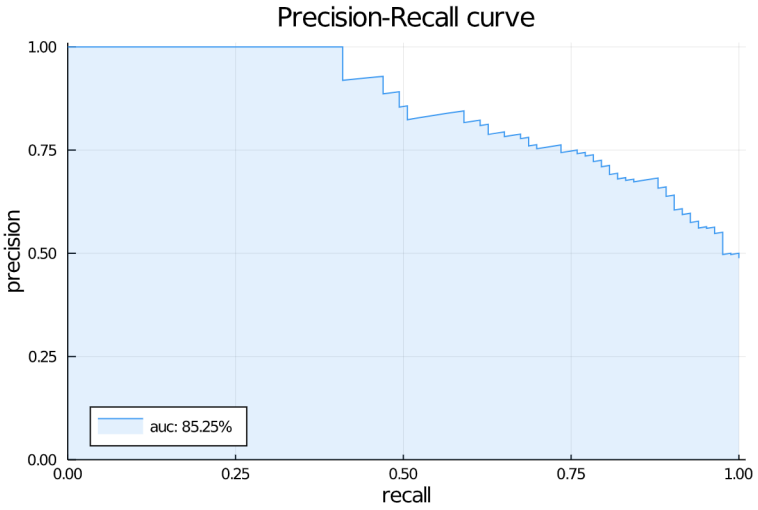


Figure 4: Precision-Recall curve for the model over test data

The precision-recall analysis yields that for even extremely high values of recall we may expect accuracy of our model to be around 50%.