EINDHOVEN UNIVERSITY OF TECHNOLOGY

Foundations of Data Mining (2IMM20)
Department of Computer Science and Mathematics

Homework 3B

Arngrimur Einarsson
Gudmundur Orri Palsson
Nick Geertjens
Stefan Gunnlaugur Jonsson
Troy Maasland

Supervisors:
Decebal Mocanu

Eindhoven, November 2019

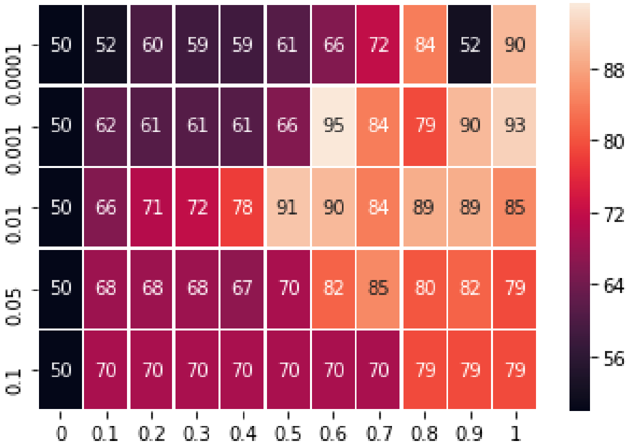
1. Hyperparameters Optimization

In this assignment our goal was to perform hyperparameter optimization to try and understanding the behavior of feed forward neural networks. Using our MLP implementation from HW3A.

1. (10 points) Parameter initialization. Choose the best performing model from HW3A in terms of classification accuracy. For this model, initialize all its parameters (connection weights and biases) to zero and retrain it using all the other settings from HW3A. Report the performance (e.g. loss function over training epochs, classification accuracy and confusion matrix on the validation set after training) of this new trained model. Do you observe any interesting behavior for this new trained model? Discuss the performance of this new trained model in comparison with the performance of the best model from HW3A.

Solution. Our accuracy is stuck at 50% when the weights and bias are initialized to zero. In each iteration of the backpropigation, we update the weights by multiplying the existing weight by a delta determined by backpropagation. If the initial weights value are zeros, multiplying it by any value for delta won't change the weight which means each iteration has no effect on the weights you're trying to optimize.

2. (10 points) Learning rate vs parameter initialization. Retrain several times for the same fixed amount of epochs (e.g. 100 - this value is your choice) the best model from HW3A using exactly the same settings as in HW3A, with the exception of the learning rate and the initialization of the connection weights and biases which have to be different for each retraining. Start with a very small learning rate (e.g. 0.0001) and after that gradually increase it (e.g. next values can be $.001, 0.01, ...$). Initialize the connection weights and biases by sampling from a normal distribution $N(0, \sigma^2)$. σ^2 has to take the following values $\{0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1\}$. Make a heatmap where the x-axis represents σ^2, the $y$axis represents the learning rate, while the colors represent the accuracy obtained on the validation set after each training. Each element in the heatmap matrix represents practically the accuracy obtained with a specific learning rate and a specific initialization. Discuss the heatmap results.



Solution. We can see that the best/highest accuracy can be fount in the middle. This is where neither the learning rate nor the weight initialization are too extreme. Our best results are where the learning rate $\alpha = 0.1$ and standard deviation from $\sigma = 0.4$ to $\sigma = 0.6$. When the learning rate is too low the algorithm either takes a long time to train or it can get stuck in a local extremum in the training data. When it is too big it has a trouble training because it overshoots the minumum and never quite gets to the optimal place. As seen in part 1, when we initialize the weights to 0 we don't get good results. It is the same when we are close to zero, the training gets stuck or the weights just dont respond to the training. When the weights are too big take a longer time to train.

3. (10 points) Activations. Choose the worst (one which still trains and does not give errors during training) and the best performing models from Subproblem 2 in terms of classification accuracy. For both models, visualize the hidden neurons activations for each of the two hidden layer three times (after connection initialization before to start the training process, at half of the training process, and at the end of the training process). Visualization hints (it is optional to follow them, please feel free to be creative): you can compute the activations on the validation set where the data points are ordered per class; you can use a heatmap where x-axis represents the hidden neuron id, the y-axis represents the data points id, and the colors represent the activations. Discuss the results.

Solution.

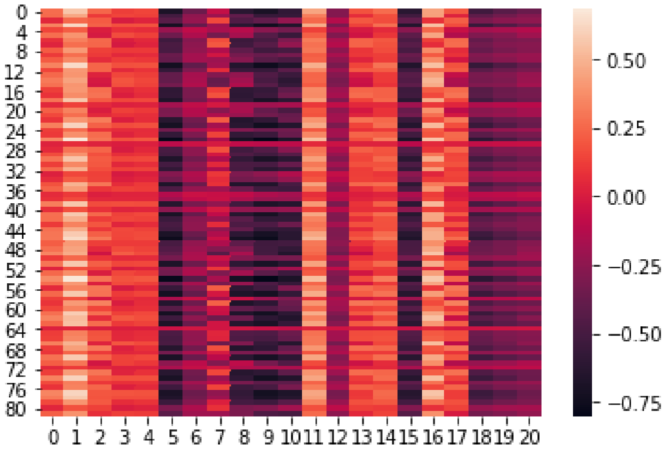


Figure 1.1: Good model: No training

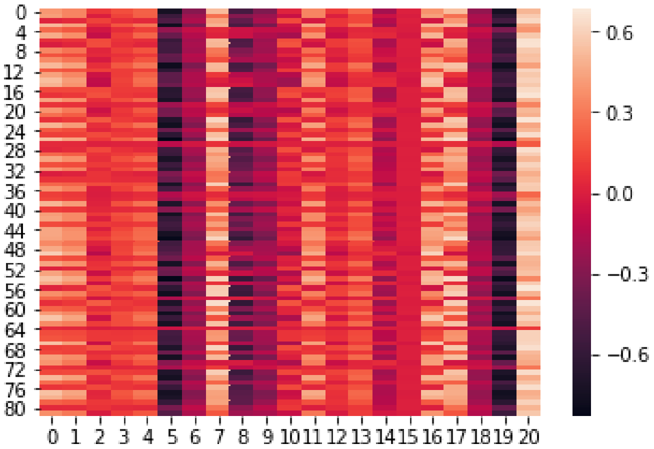


Figure 1.2: Good model: Half trained

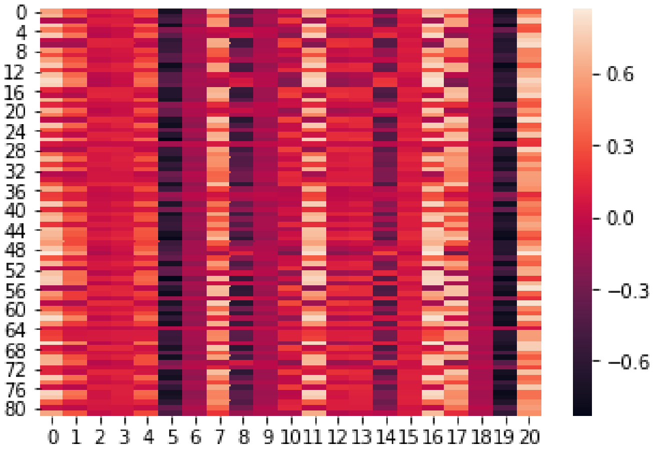


Figure 1.3: Good model: After training process

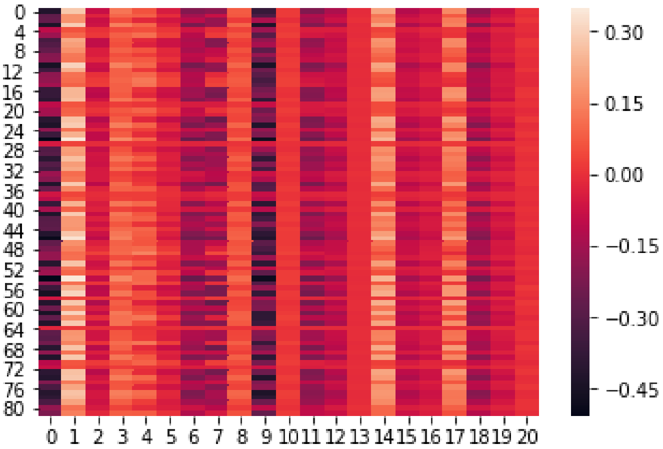


Figure 1.4: Bad model: No training

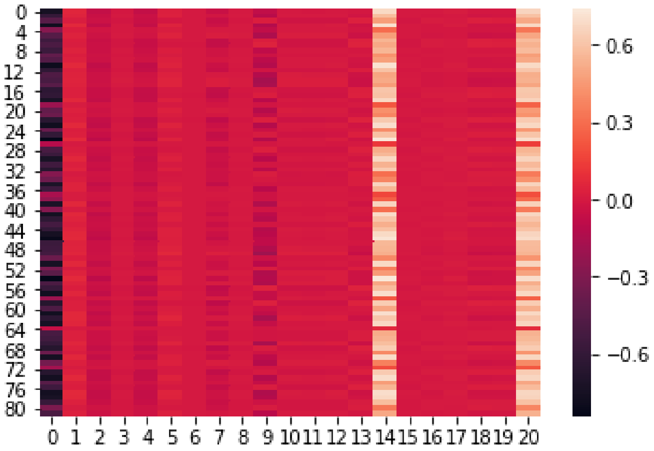


Figure 1.5: Bad model: Half trained

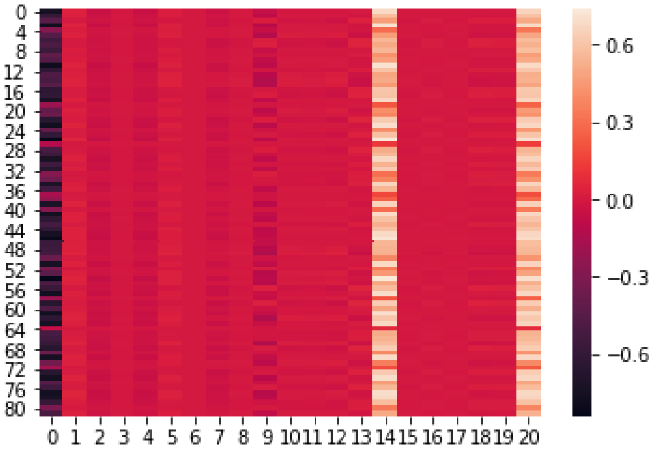
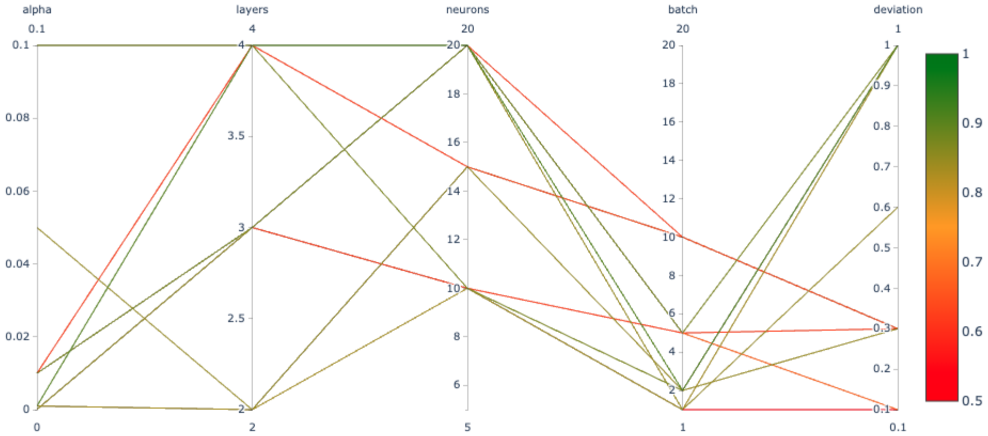


Figure 1.6: Bad model: After training process

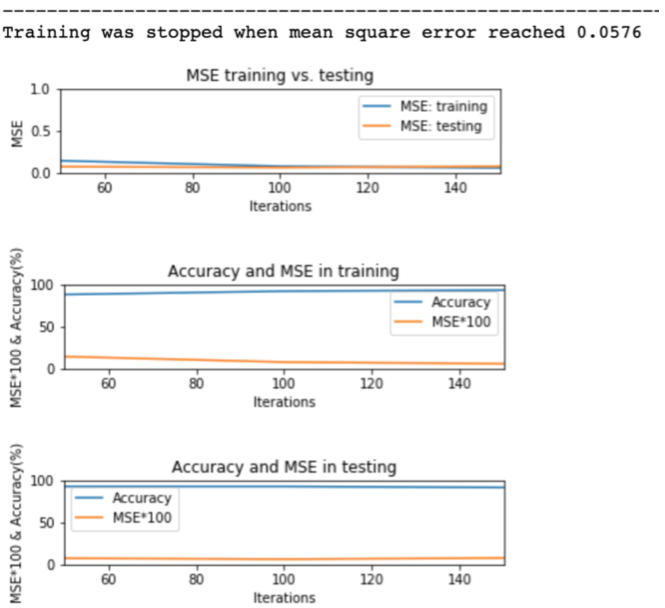
Our "good" model is using $\sigma = 0.6$ and $\alpha = 0.1$ and our "bad" model is using $\sigma = 0.2$ and $\sigma = 0.05$. In our "good" model we can see that the output of the activations tend to gravitate towards the edges of our scale(-1,1). Its kind of like they become more sure of their choices as training goes on. Our bad model on the other hand goes towards the middle(0) and stays there. If we look at the images at before training we can clearly see some patterns, lines in the model going vertically, and some horizontally. The vertical lines represent each neuron. The neuron is the same for every pair of X_0 and X_1 that is run through our model so it makes sense that there is some similarity over every neuron in the net. As time goes on these vertical lines tend to become more clear and strong in our "good" model. They become less so in the bad one and kind of fade out into redness in the "bad" model. The horizontal lines witch seem red are probably some data points that are close to the edges of the scales, either 1 or 0.

4. (20 points) Hyperparameters Optimization. On the dataset from HW3A please train a MLP model on which you perform hyperparameters optimization. For this specific subproblem, you are not constraint any-more to use just two hidden layers, and 10 neurons per hidden layer. You are free to vary any hyperparameter (e.g. learning rate, number of hidden neurons per layer, type of activation function) you consider to be important to maximize the accuracy on the validation set. Please make a Parallel Coordinates Plot to study the effect of the hyperparameters choice on the performance. The colors of the lines which connect various hyperparameter settings represent the accuracy on the validation set. Discuss the results. For the worst and best performing models, please make a plot with the values of the loss function computed separately over the training and validation sets respectively. Discuss the generalization performance (non exclusive suggestions: overfitting, underfitting) of these two extreme models.

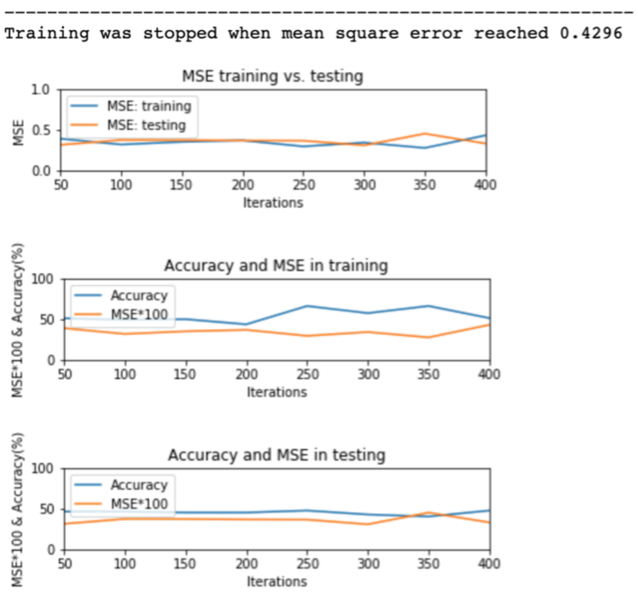
Solution.



On the parallel axis plot the color indicates the accuracy of the model. If the accuracy is really good we get a green line and it gets redder for lower accuracies. Our worst models have learning rates really small and size of layers and neurons rather big, this might be because you need more time to train those models or just better learning rates. The better models seem to have the learning rate around 0.01 or 0.001, four layers and 20 neurons which is pretty large so good models can also be big. The main thing seems to be the batch size and the deviation on the initialization of the weights. The deviations are close to 1 in the better models, so the weights are initialized at a larger interval.



```
------------------------------------------------------------
Training was stopped when mean square error reached 0.0576
```



```
-----------------------------------------------------------
Training was stopped when mean square error reached 0.4296
```

We see that in our bad model that when the error goes up on the testing data then the error on the training data goes down, this suggests overfitting. This happens after 350 iterations.