Homework 5

Point f.

To see if it's possible to get better results I applied 5 individual changes:

- Hack1: change eta.

  To get an idea on the impact of eta I tried a lower (eta=0.001) and a higher (eta=0.1) value. Lower values of eta should ensure a more stable convergence towards the optimal solution which usually takes longer, though. With higher values the convergence is faster, but we may observe oscillations in the updates which could also lead to getting further from the optimal solution.

  In Figure 1 we can observe the expected behavior since the MSE quickly drops in the first epochs to then keeps oscillating around values quite close to the optimal convergence value.

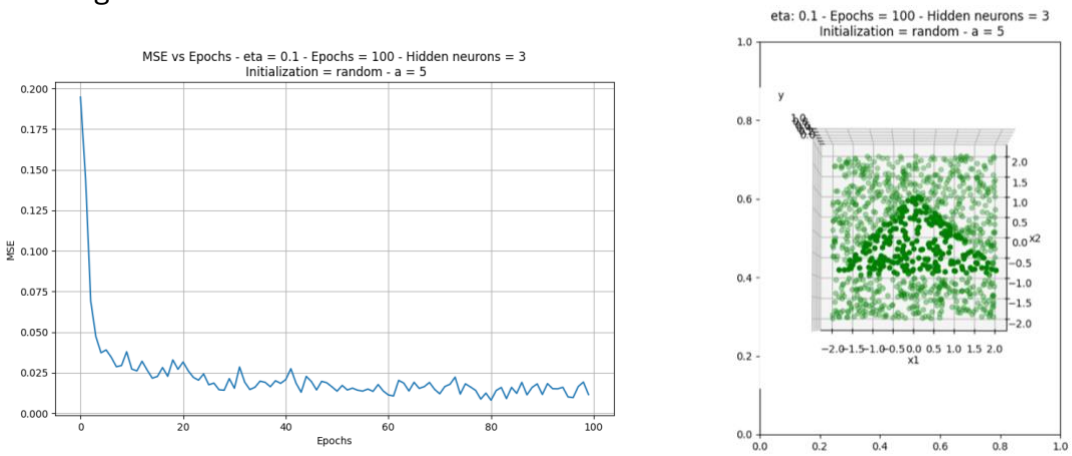


Figure 1

In Figure 2 we can see the behavior for eta=0.001, which confirms our expectations: the MSE slowly and constantly drops towards lower values. We can see that at the 100th epoch it's still decreasing, but slowly, so we can assume that if we ran it for a higher number of epochs it would reach lower values of MSE. Also, the decision boundary is not quite precise since there are many points with a not well-defined prediction value.

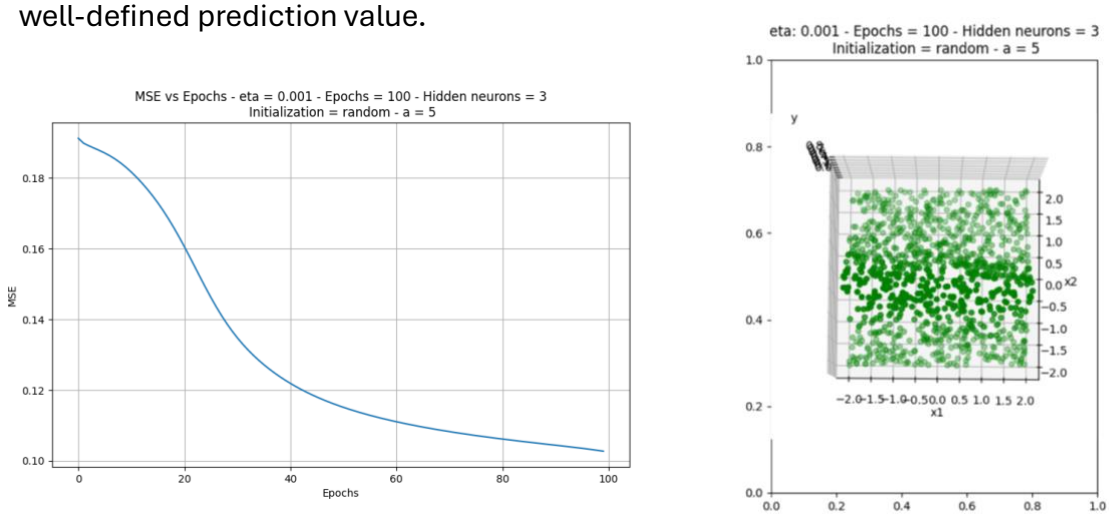
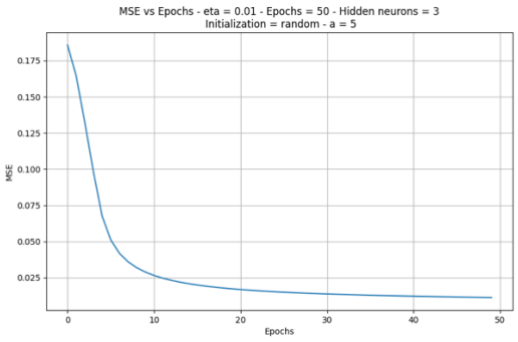


Figure 2

Given these results and the MSE-Epochs plot in point d we can conclude that, in this case, the best choice for eta is 0.01, since it has the best tradeoff between convergence speed and stability.

- Hack2: change the number of epochs.
  A change in the number of epochs could be beneficial for the algorithm performance: if we notice that even with a lower number of epochs the results are quite the same, we can stop the execution earlier saving time. If, instead, we are not quite happy with the results, we can try to run it for a larger number of epochs to see whether we can get to a decent improvement.
  In Figure 3 we see the results obtained with 50 epochs: here we see that the convergence is reached around the 40th-50th epoch so this could suggest that a fair number of epochs is around 40-50.



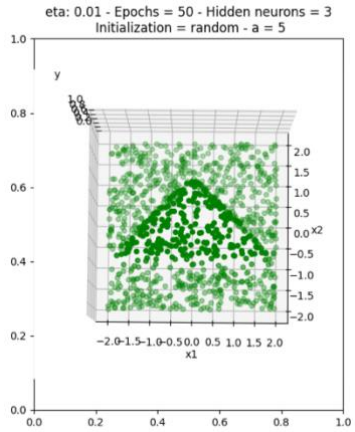
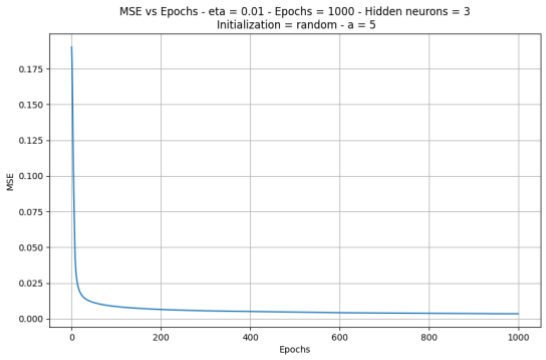


Figure 3

In Figure 4 we see that increasing the number of epochs to 1000 does not impact the final MSE value since the convergence is reached in the first hundreds of epochs. The decision boundary, on the other hand, is very well-defined since almost all points have a 0-1 value and just few of them are in between (0, 1).



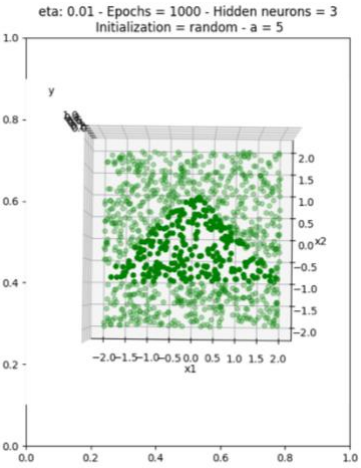


Figure 4

These results suggest that a proper value of epochs is 50 since it's the point where the MSE reaches its asymptotic value. Of course, by choosing higher values of epochs we can expect to find better values of MSE, but the difference should be minimal.

- Hack3: change the number of hidden neurons.
  To understand the impact of the number of hidden neurons I tried to use a lower and a higher value to get an idea of the best direction to move towards to.
  Using just one hidden neuron the obtained results are shown in Figure 5. We can see that choosing the lowest number of hidden neurons leads to a final MSE value that is higher than the previous solutions and the decision boundary is far from the optimal one. This is due to the fact that the neural network is too simple to capture all the important information of the data so it can just provide approximate predictions.

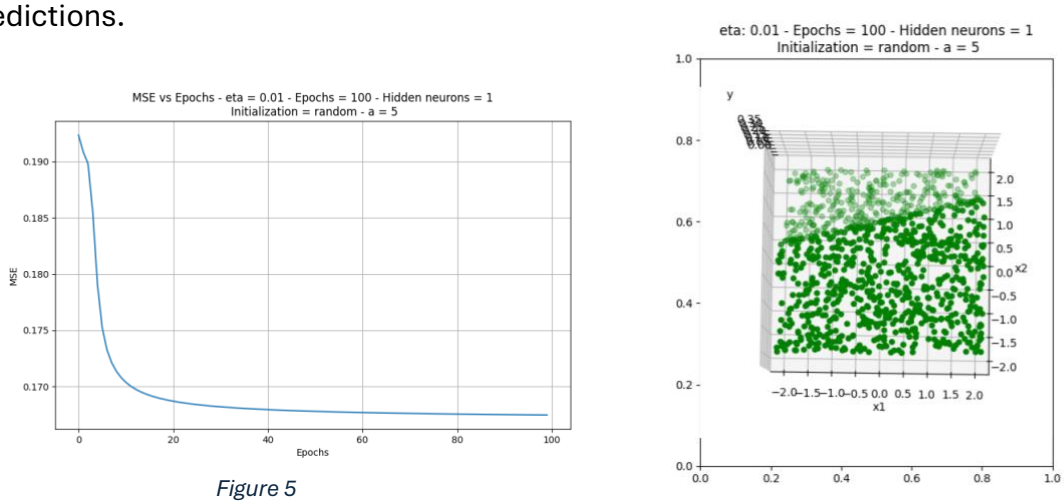


Figure 5

In Figure 6 we can see that choosing a higher number of hidden neurons (20) we can improve the final result by quite a margin.

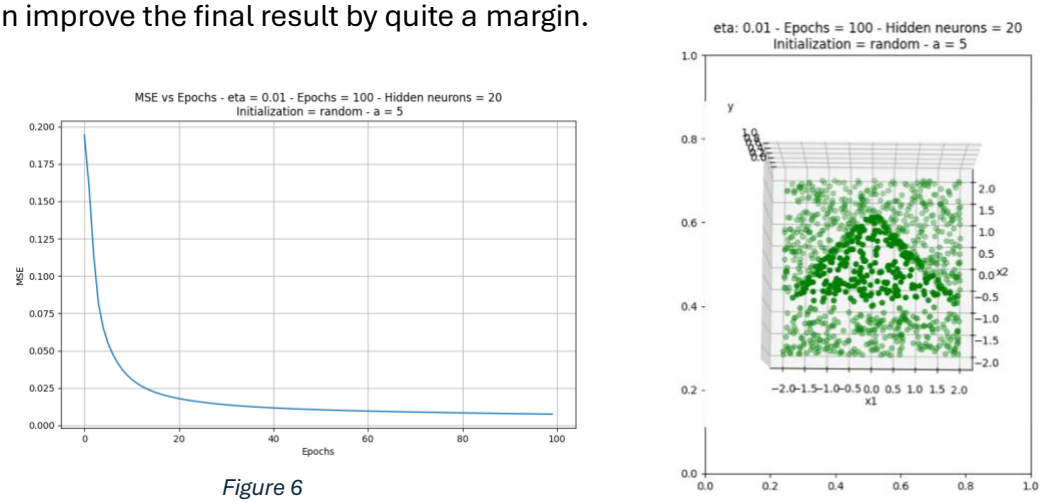
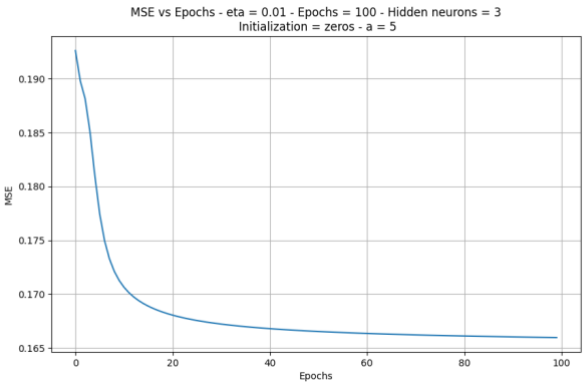


Figure 6

It's important to remember that by adding more hidden neurons we increase the internal computation. So, the final choice is a tradeoff between the improvement in MSE and the further computational complexity. In this case, given the results, it could be a good idea to increase the number of hidden nodes.

- Hack4: change initialization technique of weights and biases.
  We know that the weights and biases initialization is important to guarantee fast convergence and avoiding exploding gradients.
  In Figure 7 we can see the results obtained by initializing the weights and biases to zero. The MSE curve is quite similar to previous solutions, but its final MSE value is higher than previous solutions and the decision boundary is not quite correct.



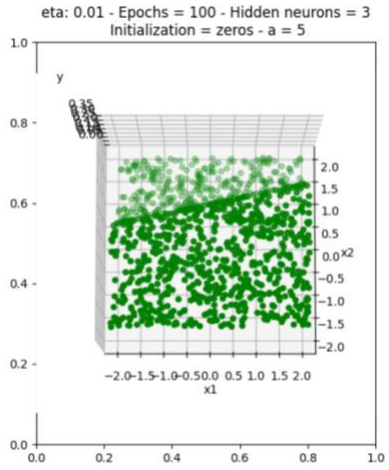
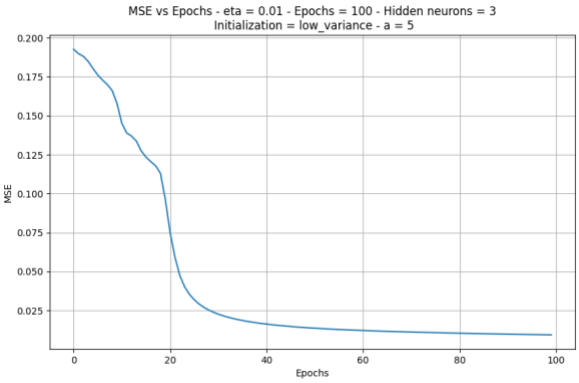


Figure 7

In Figure 8 are reported the results obtained by initializing the weights and biases following a normal distribution with a low standard deviation value. The results are similar to the previous solutions.



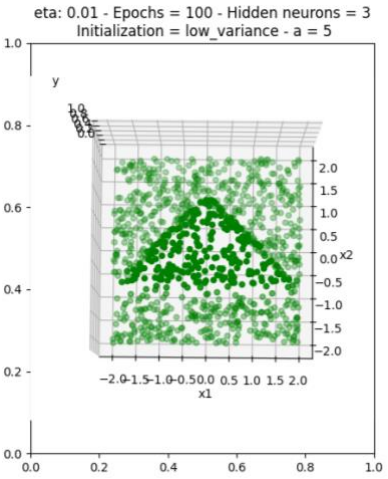
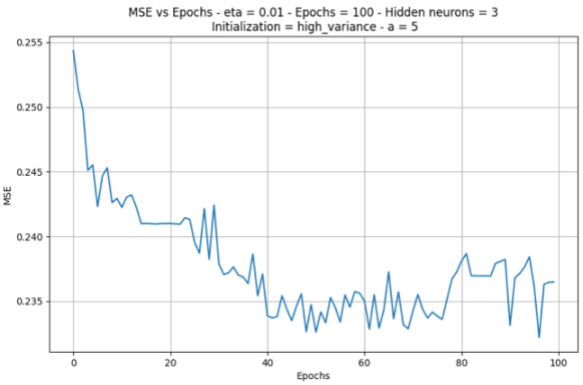


Figure 8

In the last case (Figure 9) I chose a higher standard deviation value. This impacts a lot the final results for two reasons: the final MSE value is worse than the previous solutions and the learning rate is quite irregular.



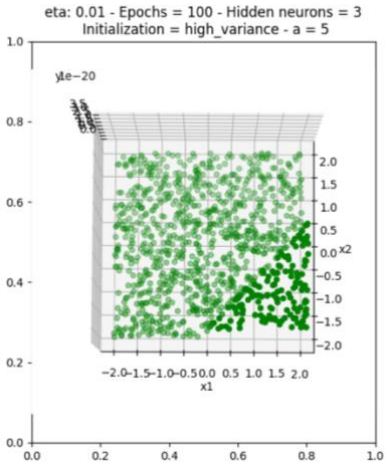
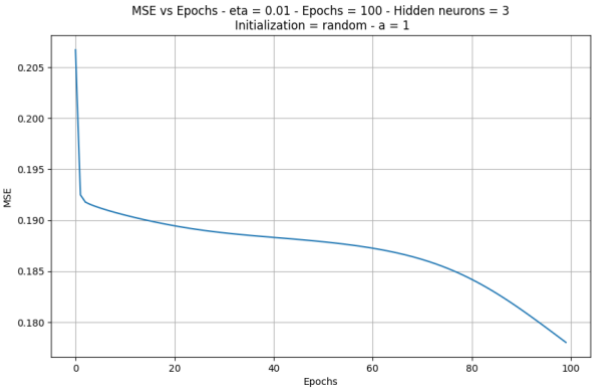


Figure 9

From the previous results we can conclude that a good initialization strategy is to set weights and biases using a normal distribution with a limited standard deviation to keep all the values close to each other.

- Hack5: change the *a* parameter of the sigmoid function.
  To better understand the impact of the *a* parameter we chose a lower and a higher value of *a*.
  In Figure 10 we can see that with *a*=1 the convergence curve is different since in the very first epochs it gets a steep increase and then it smooths out a bit to then reach worse results compared to previous solutions and the decision boundary is far off the desired one.



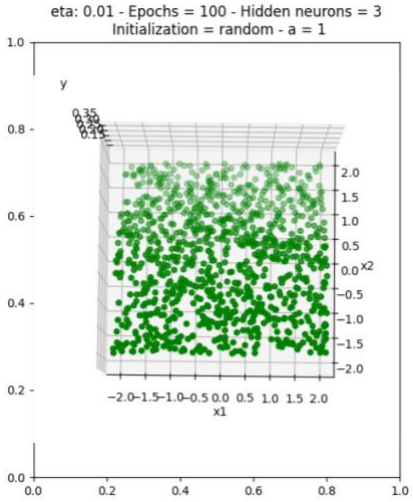
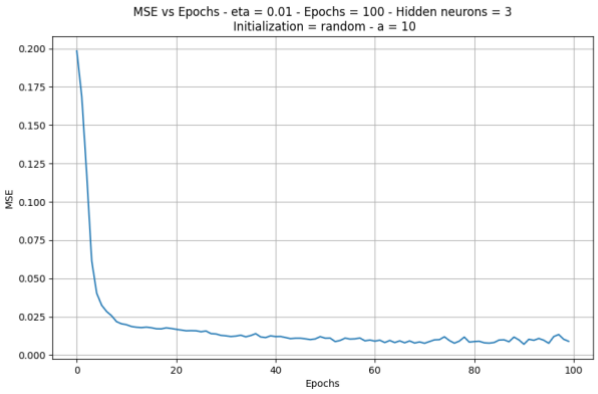


Figure 10

In Figure 11 we have the results of *a*=10. It has a similar behavior to *a*=5 even though the MSE starts oscillating after converging to a good value. These oscillations are quite limited, though.



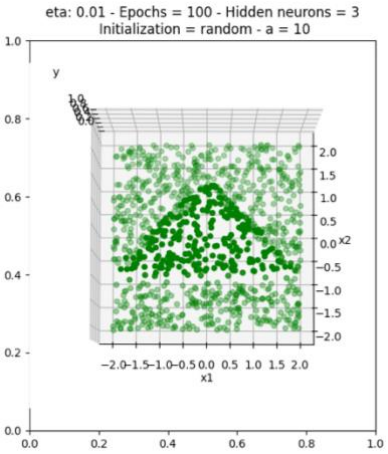


Figure 11

By analyzing the two cases we can conclude that the value of *a* affects the learning rate: for lower values of *a* it's less smooth and also slower.

Given all the previous considerations I tried to run the algorithm combining all the best values for all parameters. I choose to run it with:

- eta = 0.01: tradeoff between stability and convergence speed.
- epochs = 500: get the best results with higher epochs.
- hidden neurons = 20: get the best results with higher hidden neurons.
- initialization = random: from the considerations above it's the best initialization method.
- a = 5: not too high to limit the oscillations at convergence.

The final obtained MSE value is 0.006 (compared to the one obtained in point d which was 0.008) and the MSE curve is shown in Figure 12. We can also see that the decision boundary in Figure 13 is quite precise since there are few points with a predicted value significantly different from 0 or 1 and this confirms the good quality obtained by the last model.

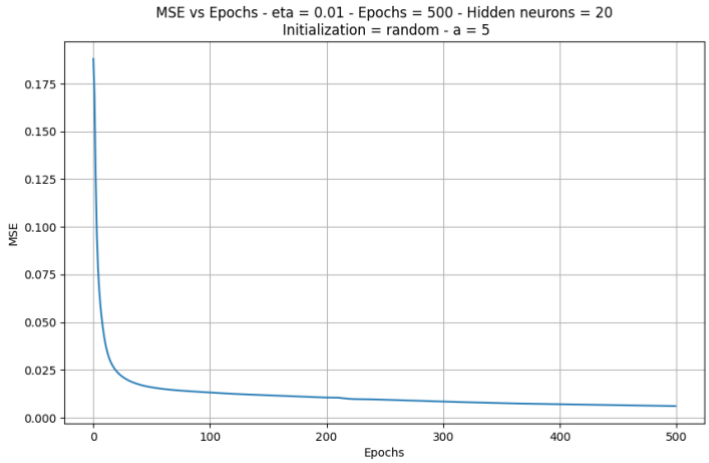


Figure 12

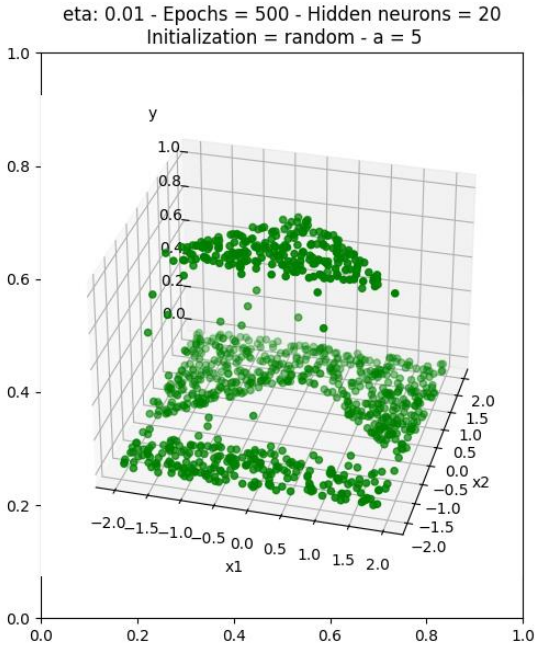


Figure 13