XOR TEST

The XOR was trained with 2 inputs 2 hidden units and 1 output I trained the system for 1000000 and 100000 epochs with a collection of learning rates [1,0.8,0.6,0.4,0.2,0.02] for the full results please see the text document with the results. However, I will provide a quick synopsis of the results. Before training the model, we had a prediction that was quite a bit off predicting the actual value for the target. My best test I will highlight yellow.

```
Target: [0] Output:     [ 0.52985127]
Target: [1] Output:     [ 0.53157742]
Target: [1] Output:     [ 0.53757261]
Target: [0] Output:     [ 0.53901508]
```

After training we can see from the results the value predicted is closer to its target. We can also see from the tables below that the learning rate of 1 with 1000000 epochs is the best case overall. A key part of this is that the error should decrease at every epoch it is a balancing game as setting the learning rate to high or too low adversely effects the learning there is a bit of a trade off as far as I can see between the amount of epochs carried out and the learning rate the are connected and increasing or decreasing either value too much or too little can have e an adverse effect on the overall system making the learning stagnant.

Table 1: Max epochs =1000000

Learning Rate	Target 0	Target 1	Target 1	Target 0	Error at last epoch
1	0.00152	0.99380	0.99380	0.00838	0.0055738
0.8	0.001753	0.9930655	0.9930655	0.0093775	0.00624992
0.6	0.002107	0.991980	0.991980	0.010838	0.0072463
0.4	0.002733	0.990151	0.990151	0.013299	0.008932
0.2	0.00427369	0.985979	0.985979	0.018901	0.012804
0.02	0.01994723	0.95291462	0.95291459	0.06299386	0.044277

Table 2: Max epochs =100000

Learning Rate	Target 0	Target 1	Target 1	Target 0	Error at last epoch
1	0.00671524	0.979968	0.979968	0.02695395	0.0184333
0.8	0.00777864	0.97750634	0.97750633	0.03024605	0.02075311
0.6	0.00941523	0.9738558	0.9738558	0.03512236	0.02420660
0.4	0.01236427	0.9671164	0.9671153	0.04343015	0.0301480
0.2	0.019947	0.952914	0.952914	0.032993	0.044278
0.02	0.1267825	0.80998	0.80998	0.24986	0.189769

SINE TEST

I ran the sin test with 3 different learning rates [0.1,0.01,0.006] we can wee from the results of the error in the training where the MLP begins to loose their learning where the error becomes stagnant or increases instead of decreasing I have marked the learning rate in blue where the epoch with a learning rate becomes stagnant it might be useful in these cases to stop the process early . The one highlighted in yellow is my best test where the training error decreased and there was 20 hidden units. We can see that the error is a little lower when we add additional units.

Table 1: with 8 hidden units. full results in: sintest.txt

	1000 Epochs	**100000**	**1000000**
0.1	**0.309186341133**	**0.288733740053**	**0.290829661614**
0.01	**0.303655101438**	**0.25676945815**	**0.253384771682**
0.006	**0.307007668047**	**0.2605356432**	**0.253488701729**

Table 2: with 20 hidden units. full results in :sintest2.txt

	1000 Epochs	**100000**	**1000000**
0.1	**0.301888120397**	**0.2861578009**	**0.277445486143**
0.01	**0.302680630543**	**0.254823986956**	**0.247377487199**
0.006	**0.308603729524**	**0.255065943243**	**0.244840297305**

If we look at the difference between the actual value – the predicted values we will see that they are closer together at 1000000 epochs and learning rate 0.006 with 20 hidden units and the error at each epoch was still decreasing in value. If we could find the optimal value for the hidden nodes and the learning rate with the number of epochs I am certain that the error produced could be significantly reduced it is about finding the perfect balance. However, I believe that my map shows that it learns well, and I am satisfied with the result