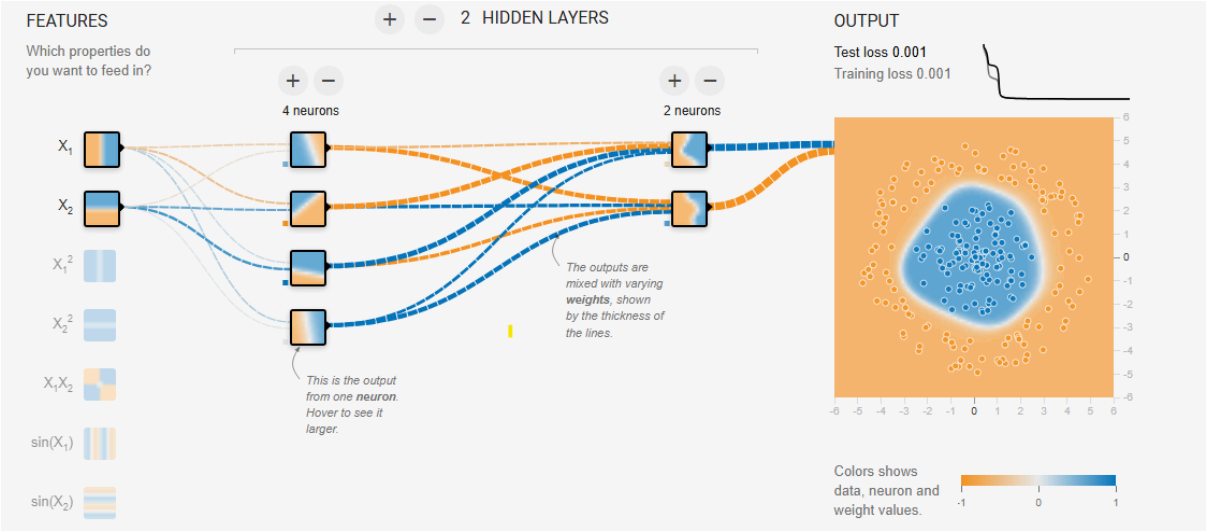
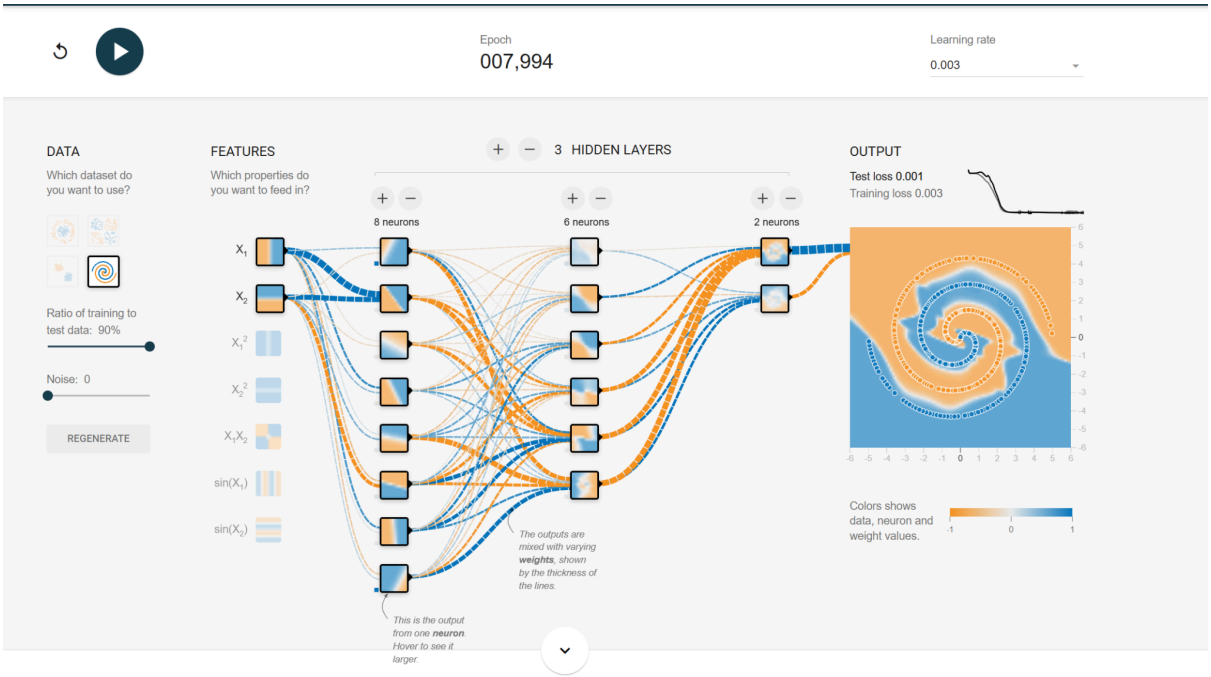


The predictions became a lot more accurate after the training, the value of test and training loss dropped after the training. The connections between the neurons became stronger and turned into meaningful colors, as shown in the screenshot.
After the training, the hidden layer learned patterns from the input (X1, X2), and recognized the pattern (the position of blue and orange dots) after a certain amount of training time, providing critical information to the output.



Data Set change to **spiral**, a much more complicated dataset

Parameters change:
Learning rate: 0.03 -> 0.003
Hypothesis : reduce the learning rate will slow down the process, but more accurate result
Hidden Layers: 2 -> 3

Neurons: 4->8 in the first layer, 2-> 6 in the second layer, 2 in the third layer.
Hypothesis: Adding more hidden layers and neurons can capture more details from the input, so I assumed the output would become more accurate after these changes.

Ratio of training to test data: 50% to 90%
Hypothesis: Increase accuracy, as more sample data is provided.

By lowering the Learning rate
The network learned slower than the previous training, but got a similar test and training loss from the previous test, considering this is a complex dataset (spiral form), having comparable results suggests that the test and training losses are improving. allowing detailed information to pass between neurons, and from neurons to the output.

By adding Hidden layers and neurons
There are more “lines” (connection) between the nodes, as the connection becomes more complex, this allows more details to be processed from the input data, and the connections between nodes also become stronger.

By increase the ratio of training to test data
It provides a larger data sample for the neural network, and accuracy is increased. If the ratio decreases, less samples are provided and the network is only able to learn from a small dataset, affecting its overall accuracy.

Both learning rate and the number of hidden layers and neurons are important for a neural network. The learning rate significantly affect the training speed, but the slower learning rate can capture more details from the input, even though, it will be too inefficient for a simple datasets, so the learning rate needs to be adjusted regarding of the size/complexity of the datasets in order to optimize both accuracy and efficiency.
Similarly, the number of hidden layers and neurons also affects the speed of training. More hidden layers and neurons can break the input into smaller parts and yield a more accurate result. However, if the datasets are small and simple, adding more hidden layers and neurons might be a bad idea, as it will lower the performance with no benefits (big difference only shows in complex datasets). In my example, reducing the learning rate and adding more hidden layers didn’t seem worth it because the output didn’t improve much for a small and simple data set.

In my observation, for a complex dataset, reducing the learning rate and adding more hidden layers would be ideal. For a smaller dataset, increasing the learning rate and reducing the number of hidden layers is suggested to optimize performance, as fewer details are needed for a good result.