Machine Learning (Lecture 8)

UEM/IEM Summer 2018

Introduction to Neural Networks: Anatomy

- Neural networks are a more sophisticated version of feature crosses. In essence, neural networks learn the appropriate feature crosses for you.
- If you recall from the Feature Crosses unit, the following classification problem is nonlinear:

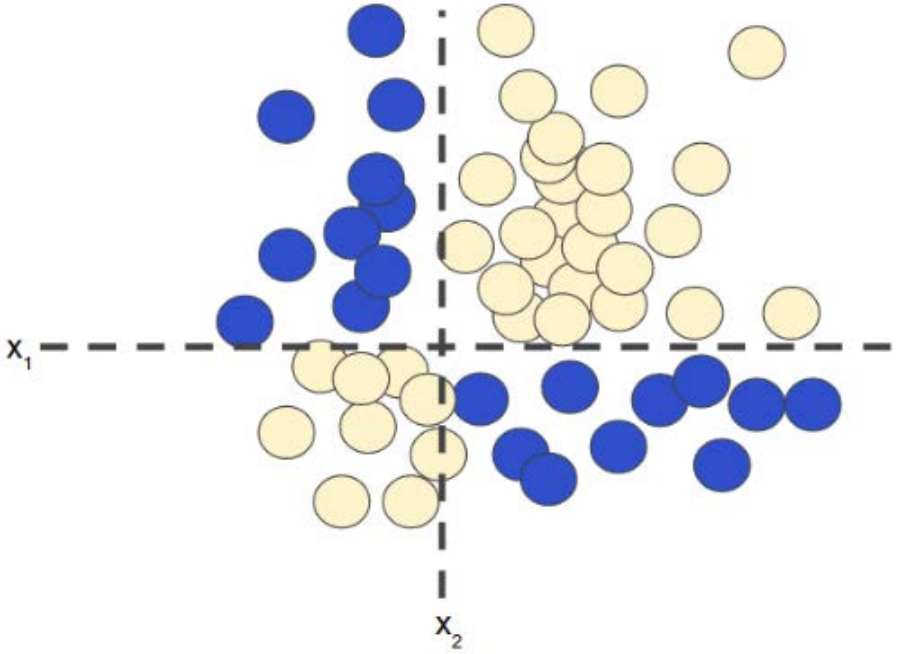
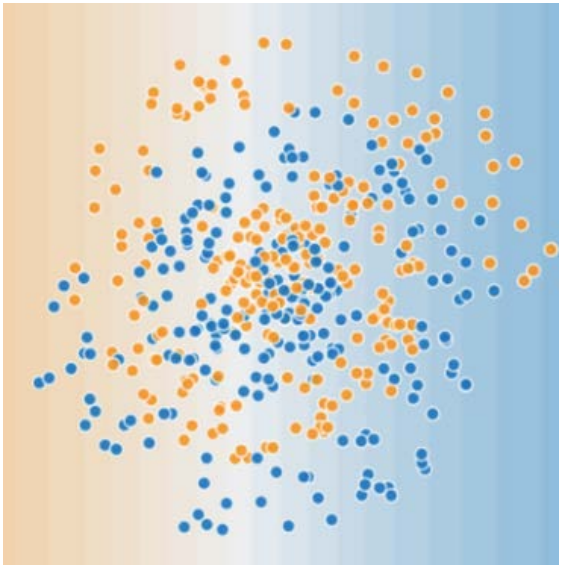


Figure 1. Nonlinear classification problem.

Introduction to Neural Networks: Anatomy

- "Nonlinear" means that you can't accurately predict a label with a model of the form $b+w_1x_1+w_2x_2$.
- In other words, the "decision surface" is not a line.
- Previously, we looked at feature crosses as one possible approach to modeling nonlinear problems.
- Now consider the following data set:



The data set shown in Figure 2 can't be solved with a linear model.

Figure 2. A more difficult nonlinear classification problem.

Introduction to Neural Networks: Anatomy

- The data set shown in Figure 2 can't be solved with a linear model.
- To see how neural networks might help with nonlinear problems, let's start by representing a linear model as a graph:

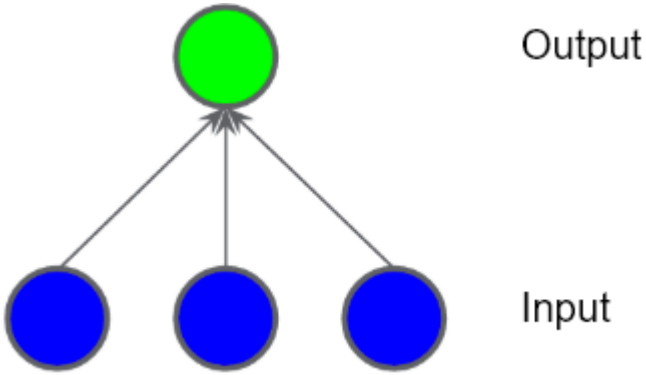


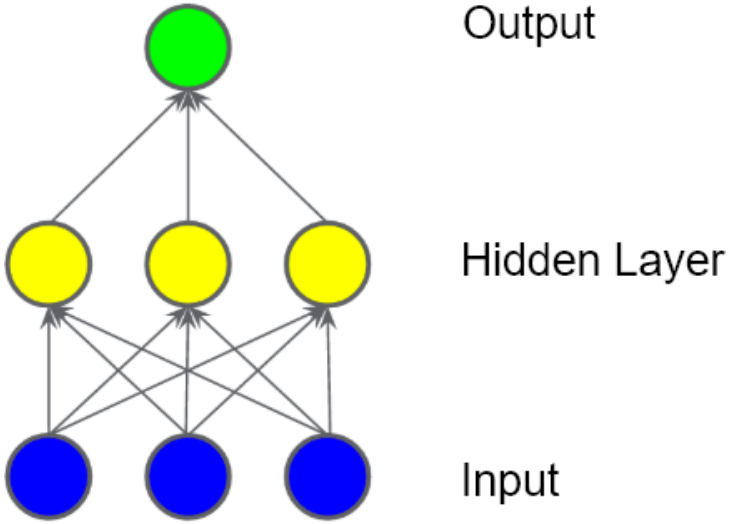
Figure 3. Linear model as graph.

Each blue circle represents an input feature, and the green circle represents the weighted sum of the inputs.

How can we alter this model to improve its ability to deal with nonlinear problems?

Introduction to Neural Networks: Anatomy

- Hidden Layers
  - In the model represented by the following graph, we've added a "hidden layer" of intermediary values.
  - Each yellow node in the hidden layer is a weighted sum of the blue input node values. The output is a weighted sum of the yellow nodes.



Is this model linear? Yes—its output is still a linear combination of its inputs.

Figure 4. Graph of two-layer model.

Introduction to Neural Networks: Anatomy

- In the model represented by the following graph, we've added a second hidden layer of weighted sums.

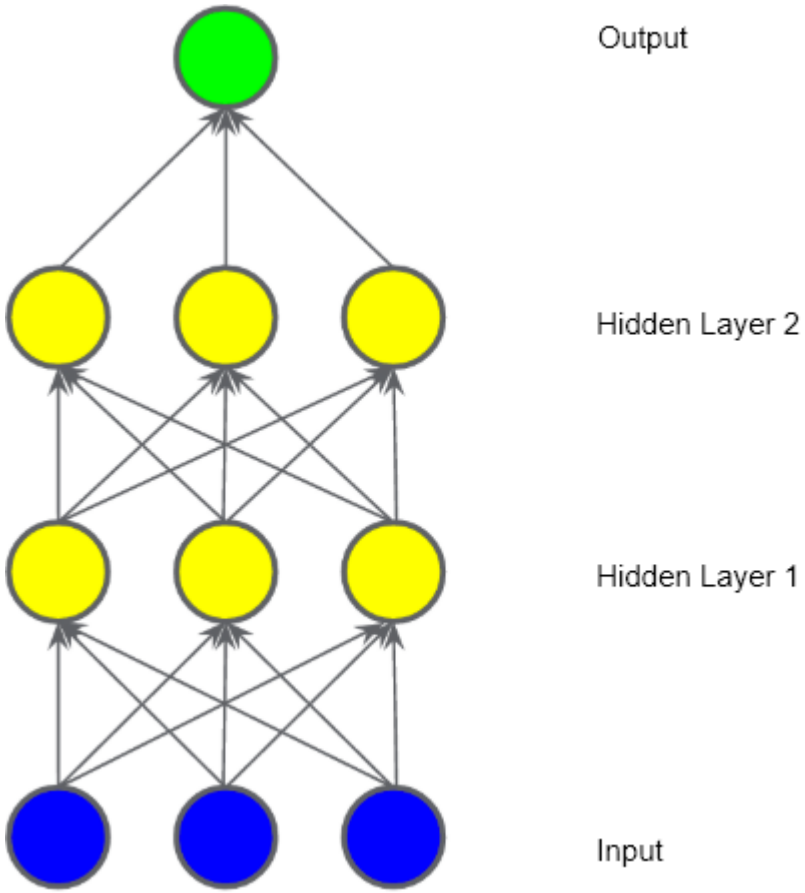


Figure 5. Graph of three-layer model.

Is this model still linear? Yes, it is. When you express the output as a function of the input and simplify, you get just another weighted sum of the inputs. This sum won't effectively model the nonlinear problem in Figure 2.

Introduction to Neural Networks: Anatomy

- Activation Functions
  - To model a nonlinear problem, we can directly introduce a nonlinearity. We can pipe each hidden layer node through a nonlinear function.

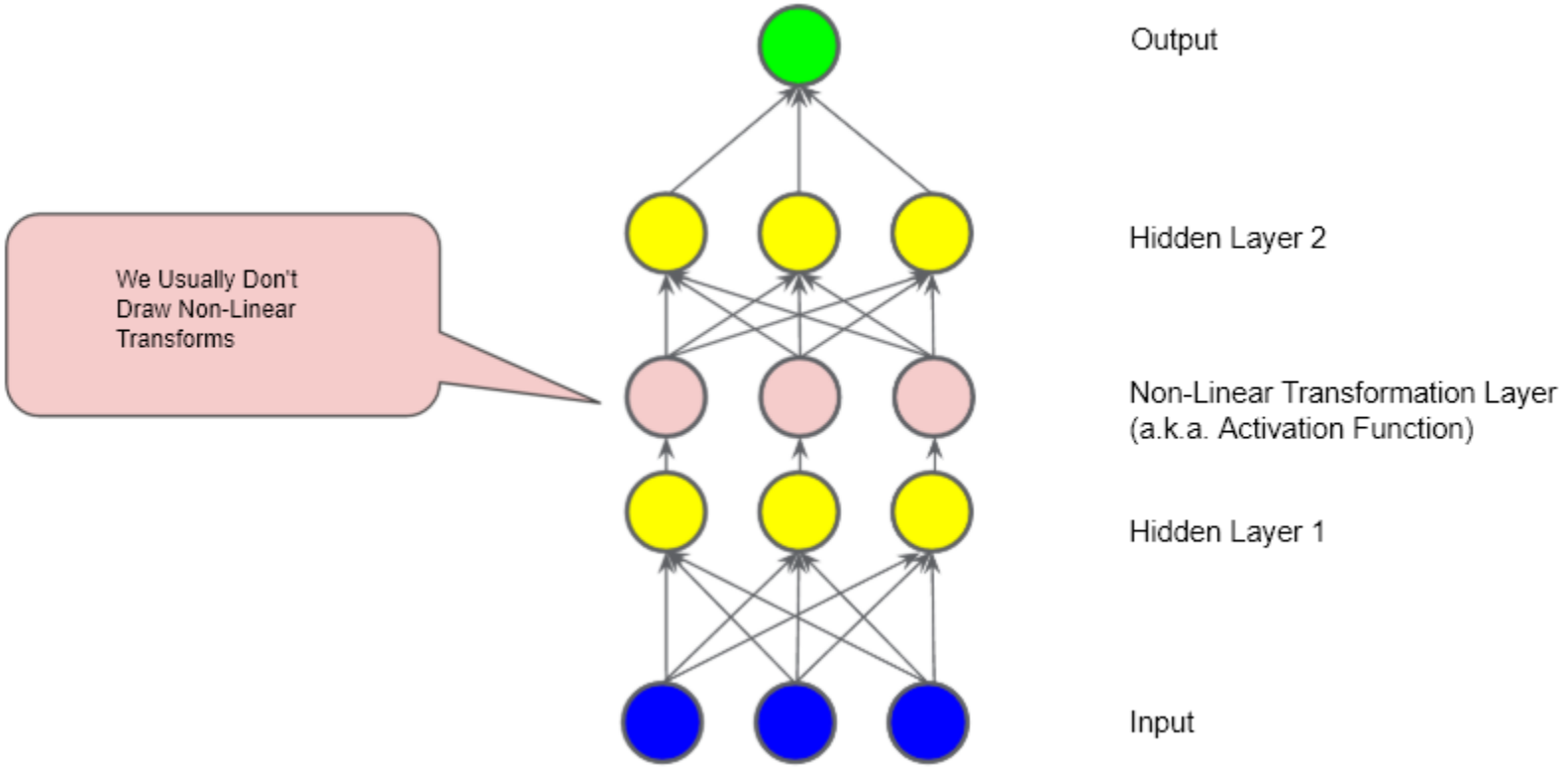


Figure 6. Graph of three-layer model with activation function.

Introduction to Neural Networks: Anatomy

- In the model represented by the previous graph, the value of each node in Hidden Layer 1 is transformed by a nonlinear function before being passed on to the weighted sums of the next layer.
- This nonlinear function is called the **activation function**.
- Now that we've added an activation function, adding layers has more impact.
- Stacking nonlinearities on nonlinearities lets us model very complicated relationships between the inputs and the predicted outputs.
- In brief, each layer is effectively learning a more complex, higher-level function over the raw inputs.

Introduction to Neural Networks: Anatomy

- Common Activation Functions
  - The following **sigmoid** activation function converts the weighted sum to a value between 0 and 1.

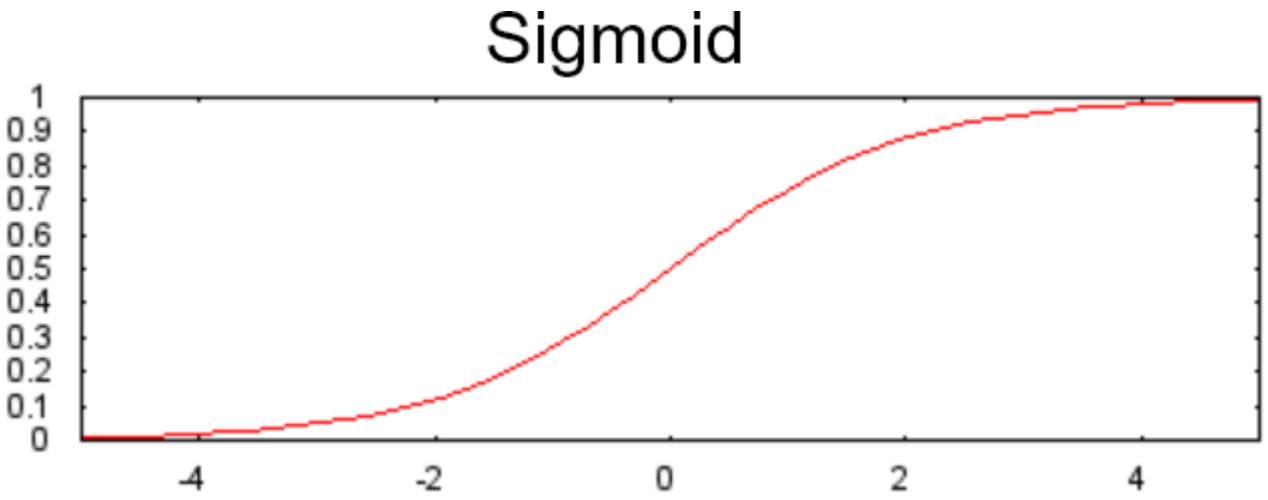


Figure 7. Sigmoid activation function.

Introduction to Neural Networks: Anatomy

- The following **rectified linear unit** activation function (or **ReLU**, for short) often works a little better than a smooth function like the sigmoid, while also being significantly easier to compute.

$$F(x) = max(0, x)$$

- The superiority of ReLU is based on empirical findings, probably driven by ReLU having a more useful range of responsiveness. A sigmoid's responsiveness falls off relatively quickly on both sides.

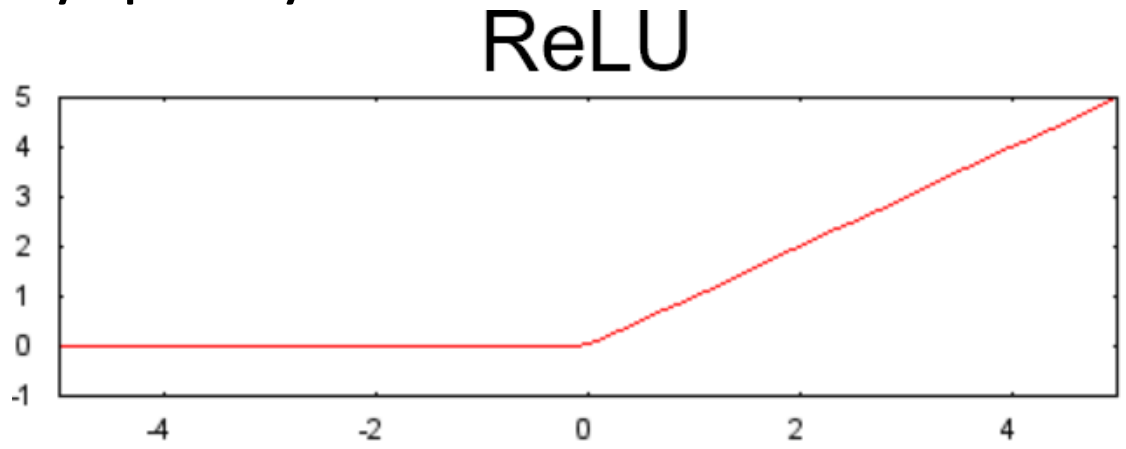


Figure 8. ReLU activation function.

Introduction to Neural Networks: Anatomy

- In fact, any mathematical function can serve as an activation function. Suppose that σrepresents our activation function (Relu, Sigmoid, or whatever). Consequently, the value of a node in the network is given by the following formula:

$$\sigma(\boldsymbol{w} \cdot \boldsymbol{x} + b)$$

- Many off-the-shelf software such as TensorFlow provides out-of-the-box support for a wide variety of activation functions. That said, starting with ReLU is still recommended.

Introduction to Neural Networks: Anatomy

- Now our model has all the standard components of what people usually mean when they say "neural network":
  - A set of nodes, analogous to neurons, organized in layers.
  - A set of weights representing the connections between each neural network layer and the layer beneath it. The layer beneath may be another neural network layer, or some other kind of layer.
  - A set of biases, one for each node.
  - An activation function that transforms the output of each node in a layer. Different layers may have different activation functions.

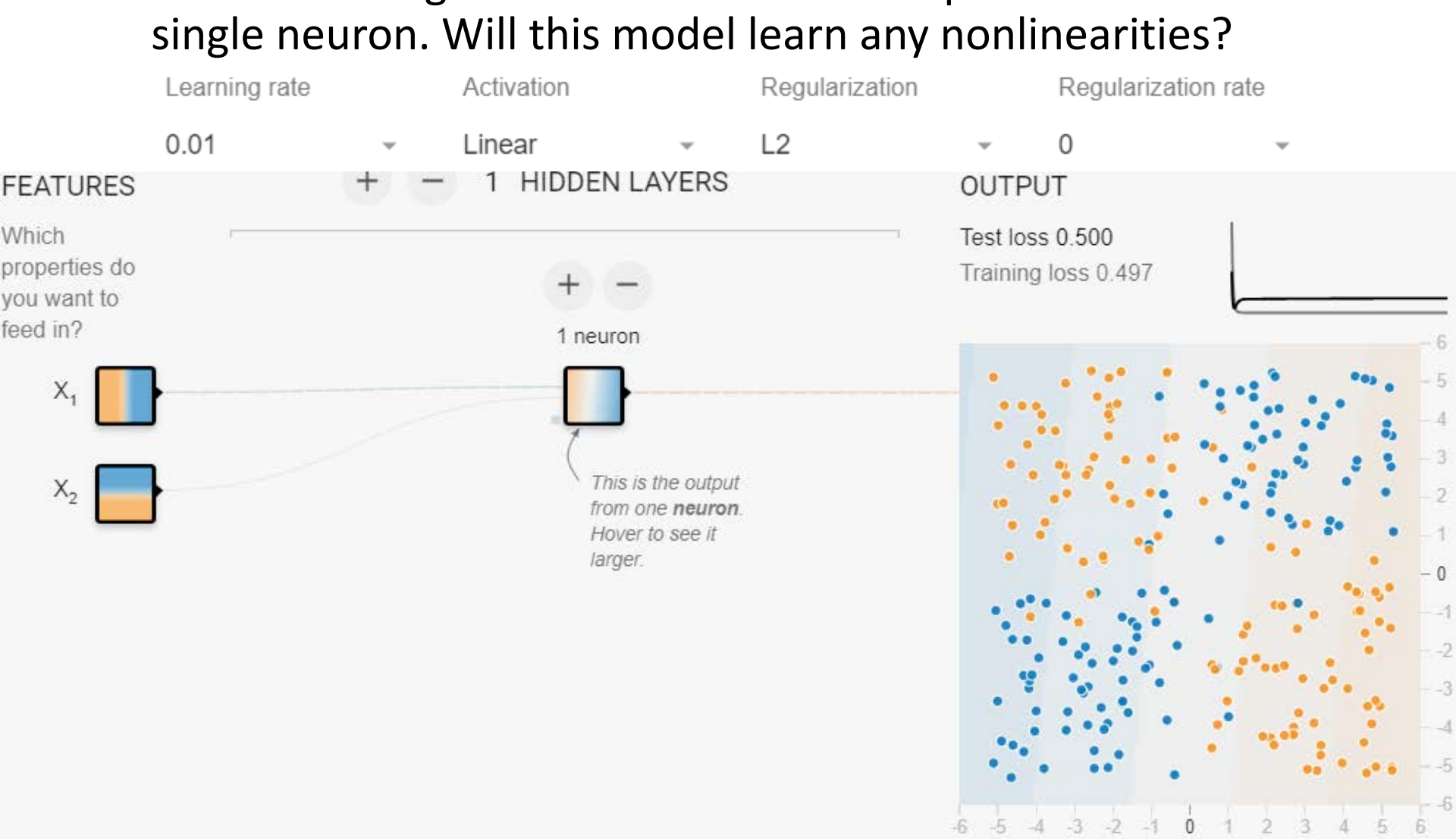
Introduction to Neural Networks

- In these demonstrations, we will train our first little neural net. Neural nets will give us a way to learn nonlinear models without the use of explicit feature crosses.

- A First Neural Network
  - In this demonstration, we will train our first little neural net. Neural nets will give us a way to learn nonlinear models without the use of explicit feature crosses.

Introduction to Neural Networks

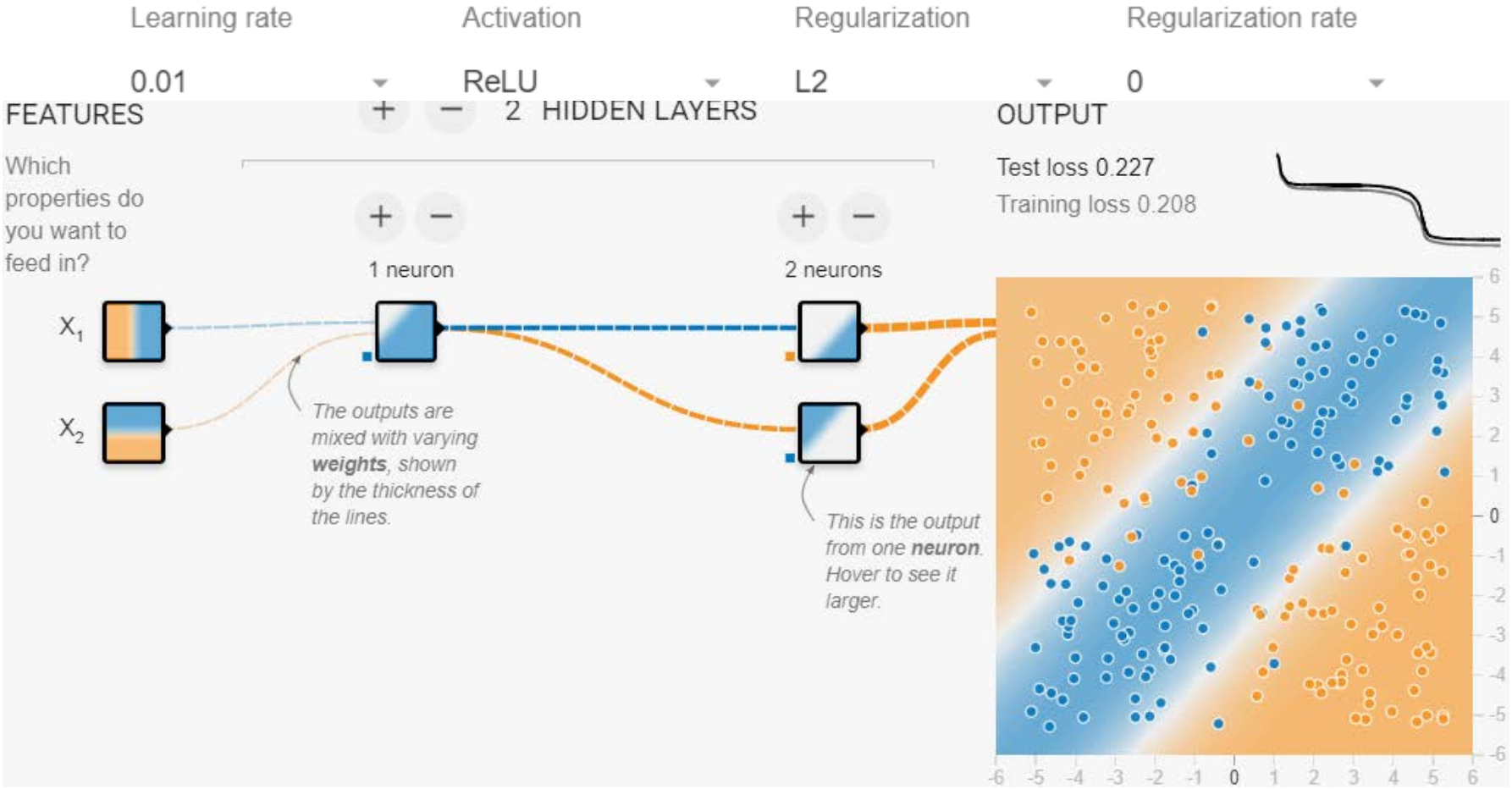
- The model as given combines our two input features into a single neuron. Will this model learn any nonlinearities?



The Activation is set to Linear, so this model cannot learn any nonlinearities. The loss is very high.

Introduction to Neural Networks

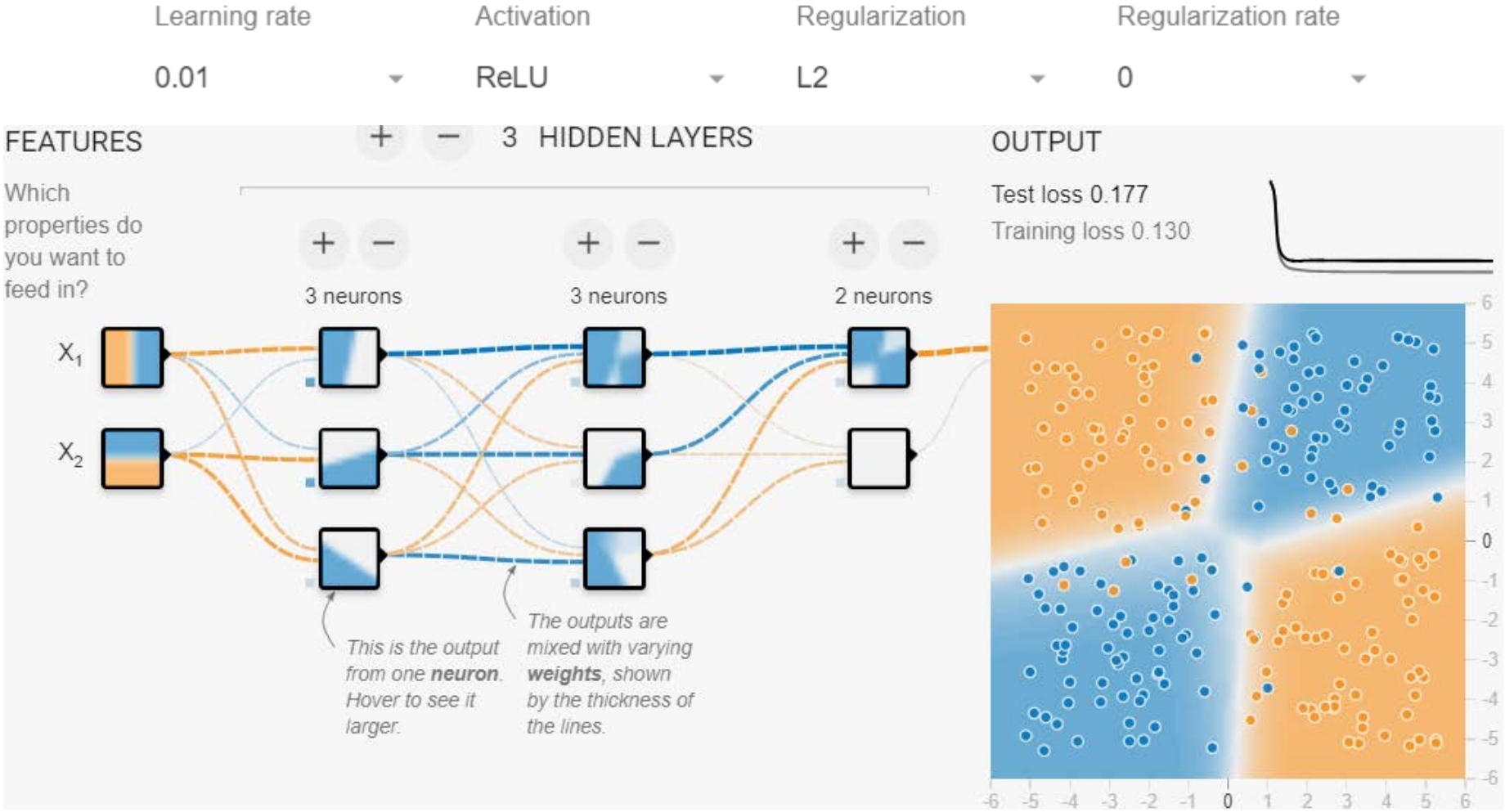
- Try increasing the number of neurons in the hidden layer from 1 to 2, and also try changing from a Linear activation to a nonlinear activation like ReLU. Can you create a model that can learn nonlinearities?



The nonlinear Activation function can learn nonlinear models. However, a single hidden layer with 2 neurons will take a while to learn the model. These demonstrations are nondeterministic, so some runs will not learn an effective model, while other runs will do a pretty good job.

Introduction to Neural Networks

- Add or remove hidden layers and neurons per layer. Also change learning rates, regularization, and other learning settings. What is the smallest number of nodes and layers you can use that gives test loss of 0.177 or lower?



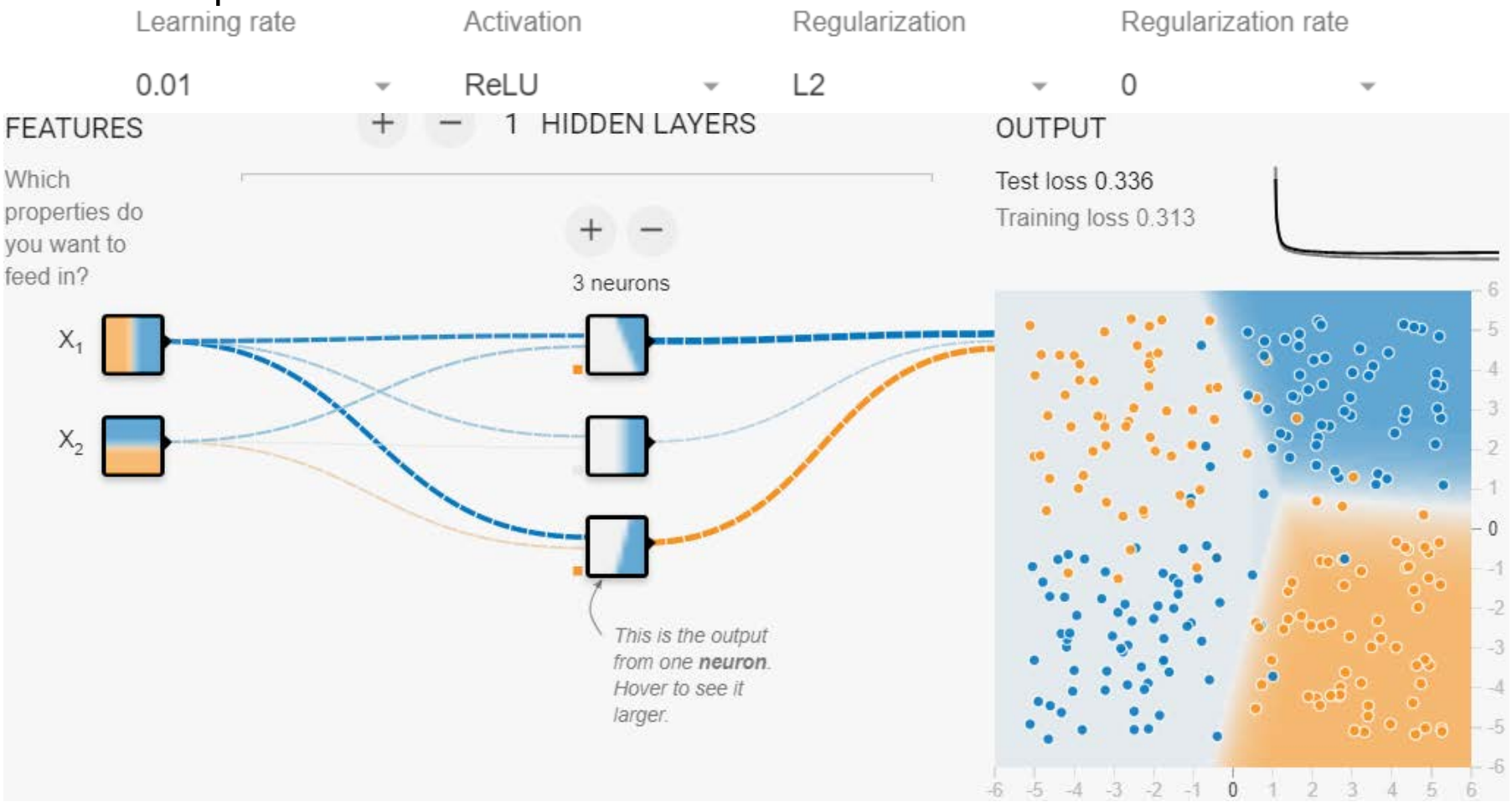
Some runs produce very low test loss with 3 Hidden Layers, arranged as follows: First layer had 3 neurons. Second layer had 3 neurons. Third layer had 2 neurons.

Introduction to Neural Networks

- Neural Net Initialization
  - This demonstration uses the XOR data again, but looks at the repeatability of training Neural Nets and the importance of initialization.

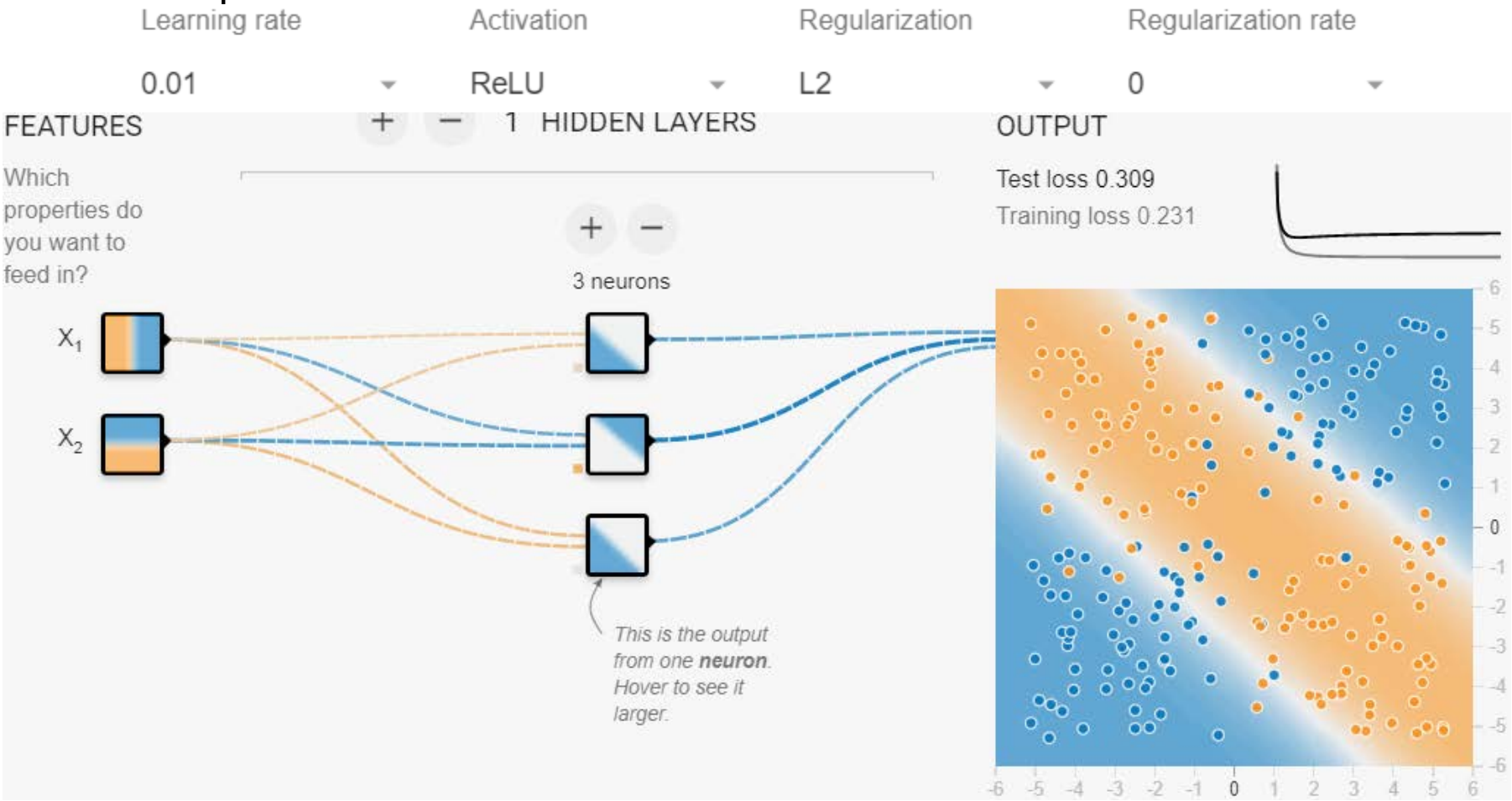
Introduction to Neural Networks

- Run the model as given four or five times. Before each trial, hit the **Reset the network** button to get a new random initialization. What shape does each model output converge to? What does this say about the role of initialization in non-convex optimization?



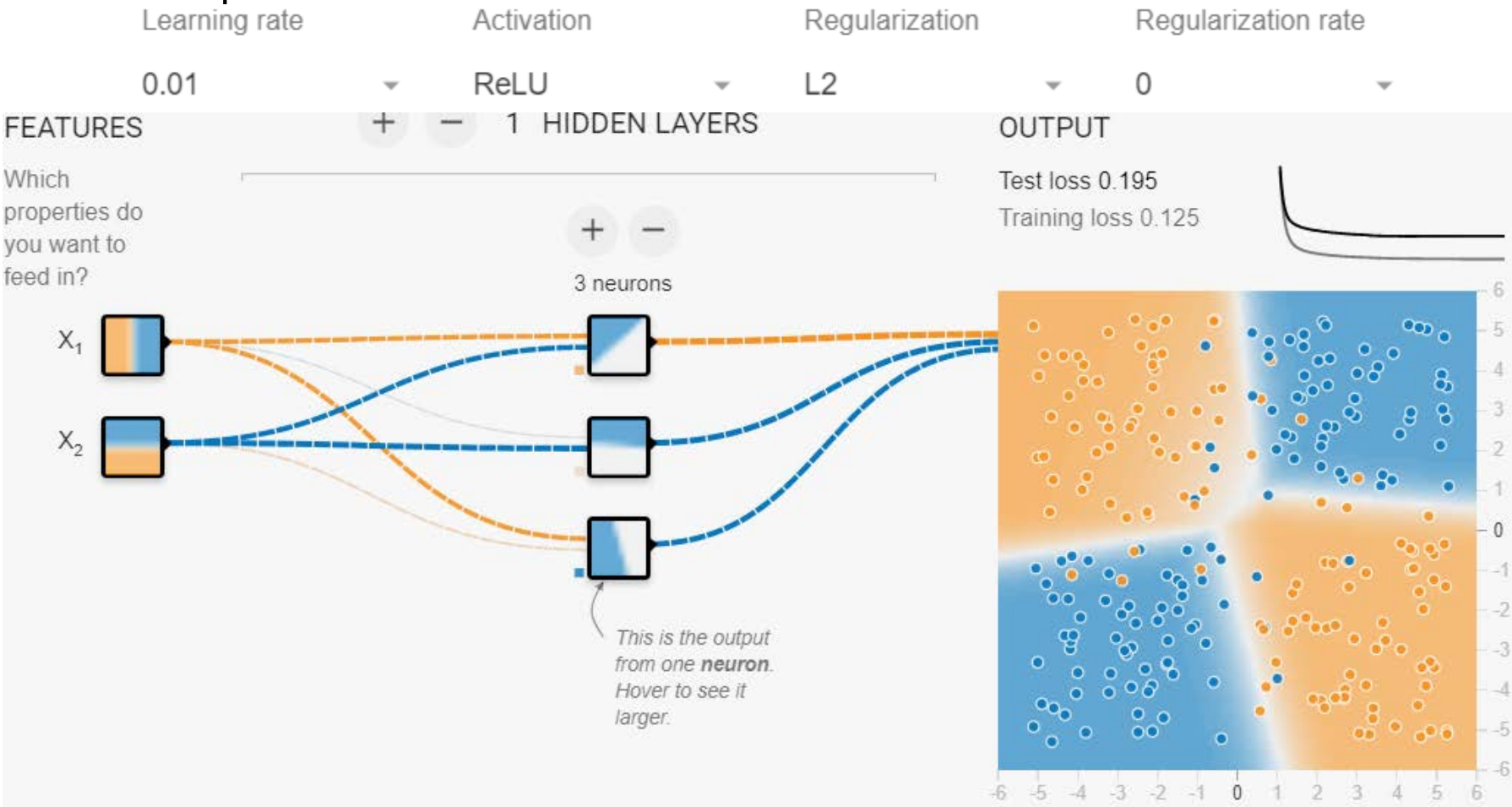
Introduction to Neural Networks

- Run the model as given four or five times. Before each trial, hit the **Reset the network** button to get a new random initialization. What shape does each model output converge to? What does this say about the role of initialization in non-convex optimization?



Introduction to Neural Networks

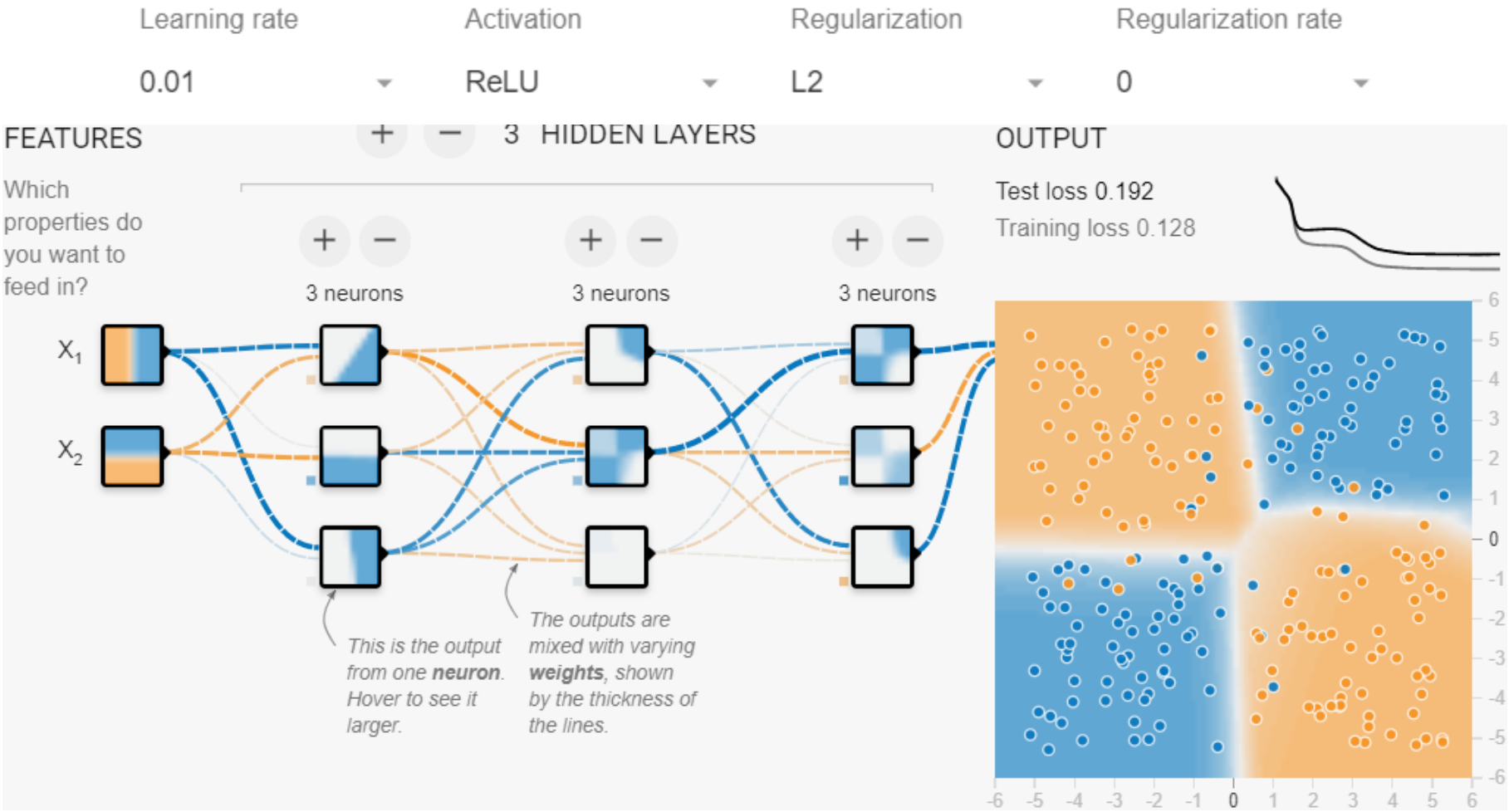
- Run the model as given four or five times. Before each trial, hit the **Reset the network** button to get a new random initialization. What shape does each model output converge to? What does this say about the role of initialization in non-convex optimization?



The learned model had different shapes on each run. The converged test loss varied almost 2X from lowest to highest.

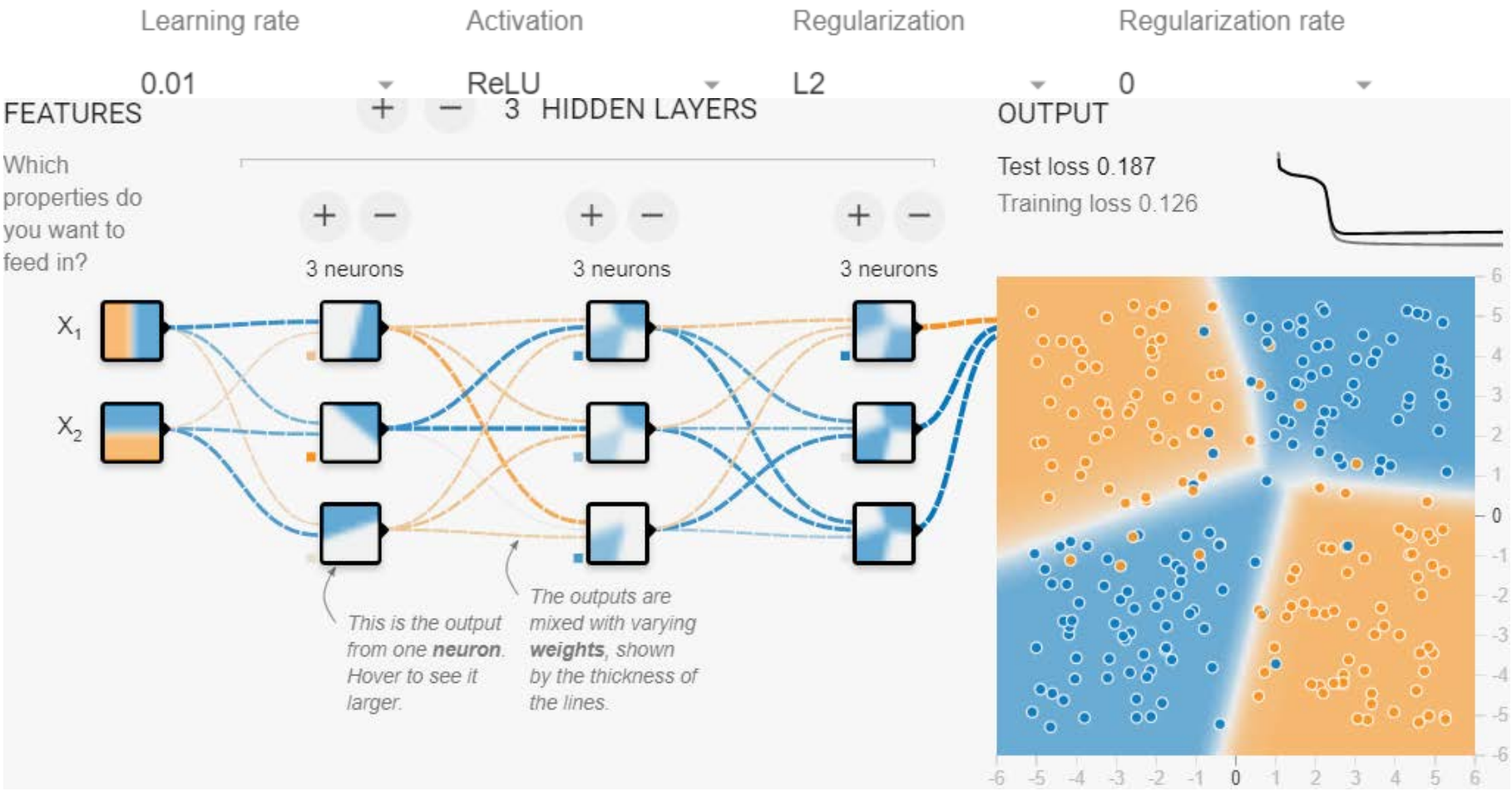
Introduction to Neural Networks

- Try making the model slightly more complex by adding a layer and a couple of extra nodes. Does this add any additional stability to the results?



Introduction to Neural Networks

- Try making the model slightly more complex by adding a layer and a couple of extra nodes. Does this add any additional stability to the results?



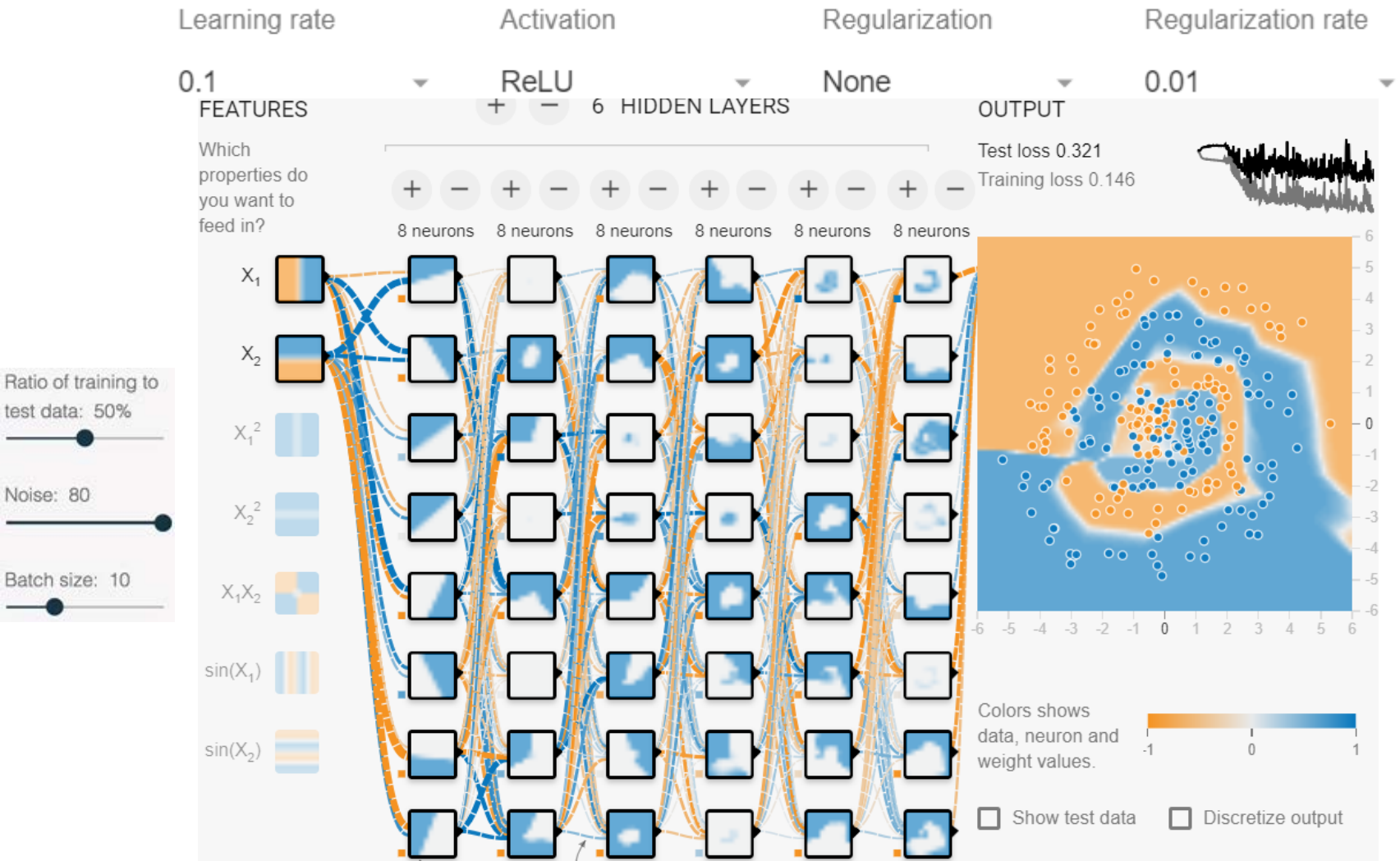
Adding the layer and extra nodes produced more repeatable results. On each run, the resulting model looked roughly the same. Furthermore, the converged test loss showed less variance between runs.

Introduction to Neural Networks

- Neural Net Spiral
  - This data set is a noisy spiral. Obviously, a linear model will fail here, but even manually defined feature crosses may be hard to construct.

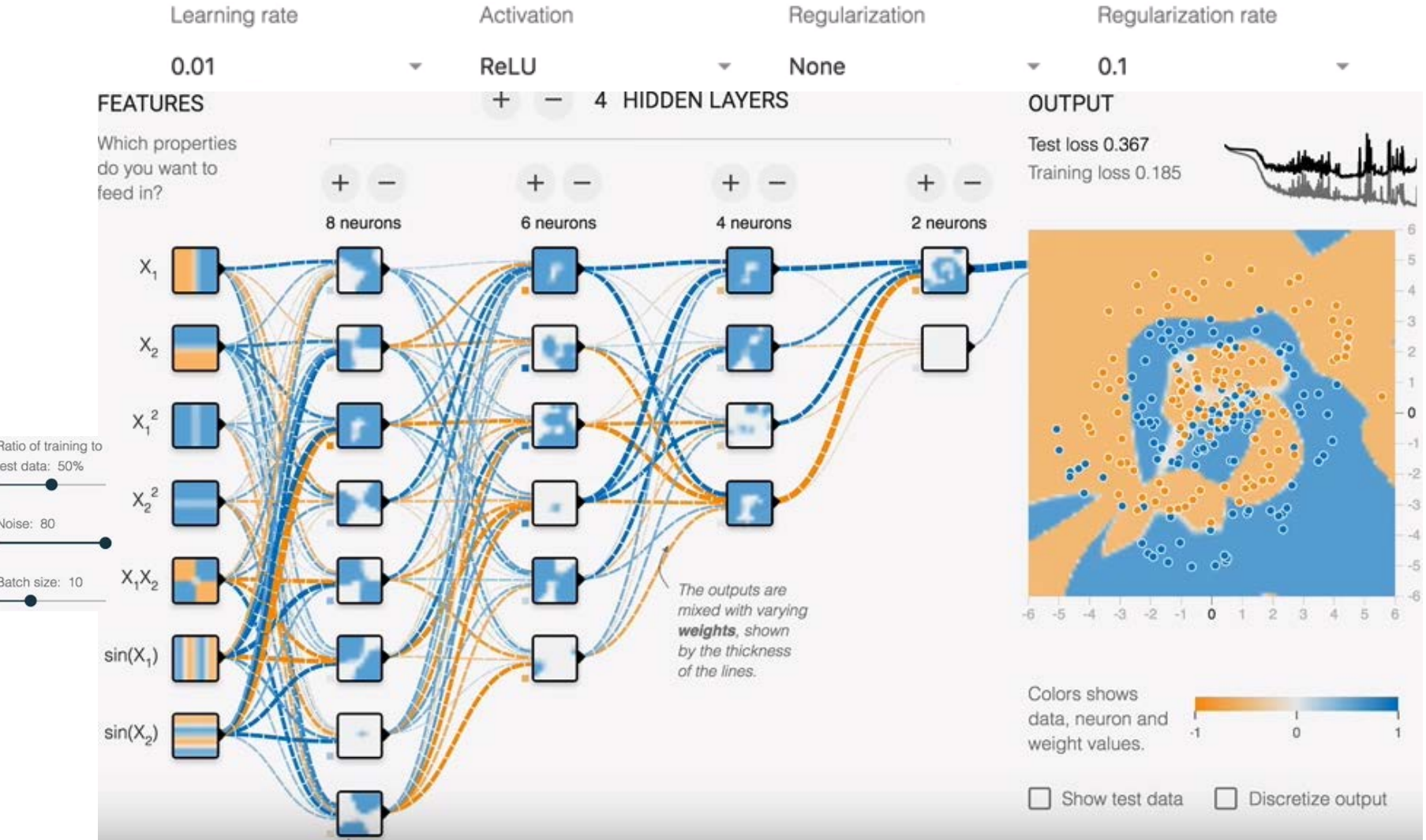
Introduction to Neural Networks

- Train the best model you can, using just X1 and X2. Feel free to add or remove layers and neurons, change learning settings like learning rate, regularization rate, and batch size. What is the best test loss you can get? How smooth is the model output surface?



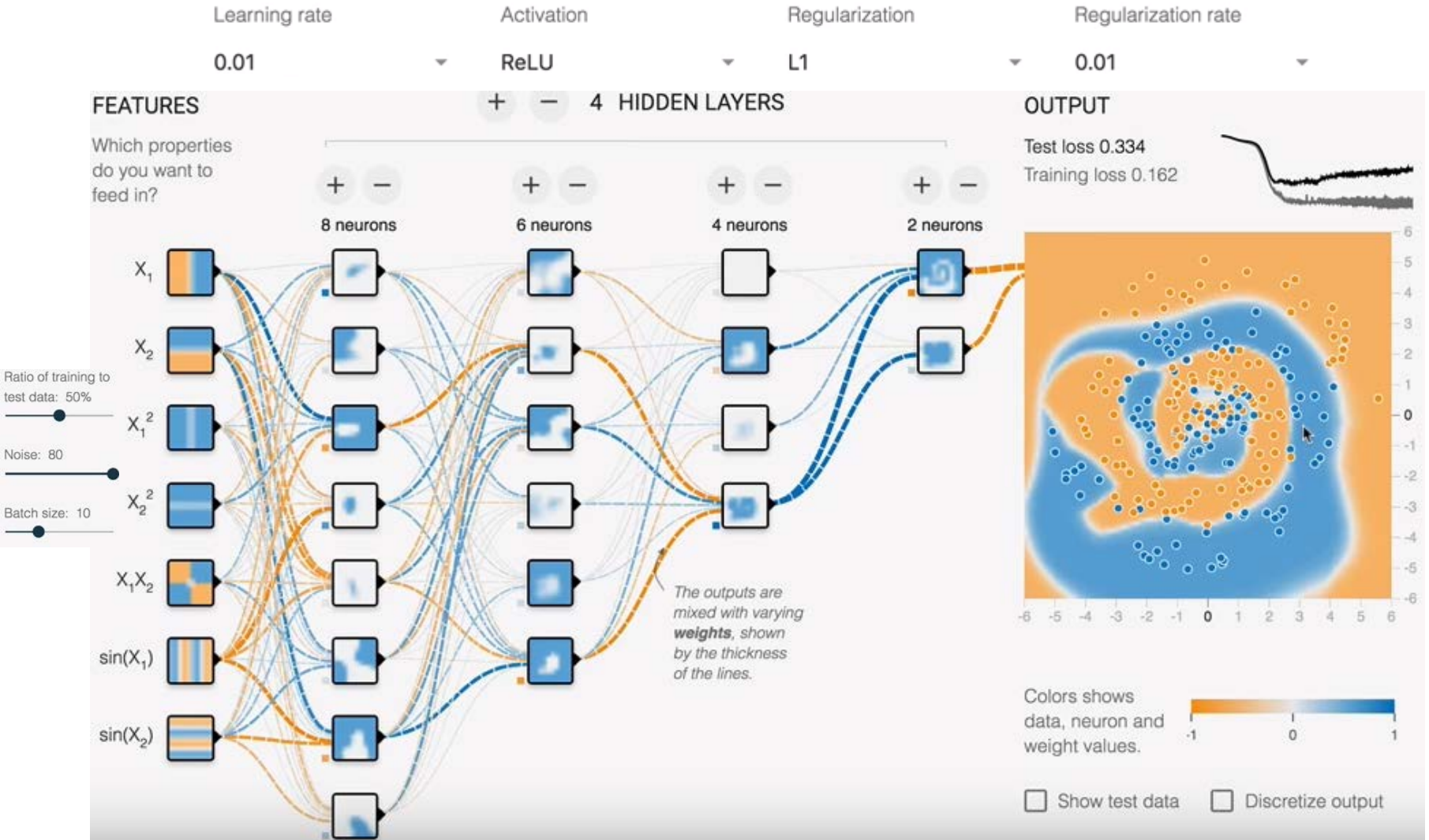
Introduction to Neural Networks

- Train the best model you can, using just X1 and X2. Feel free to add or remove layers and neurons, change learning settings like learning rate, regularization rate, and batch size. What is the best test loss you can get? How smooth is the model output surface?



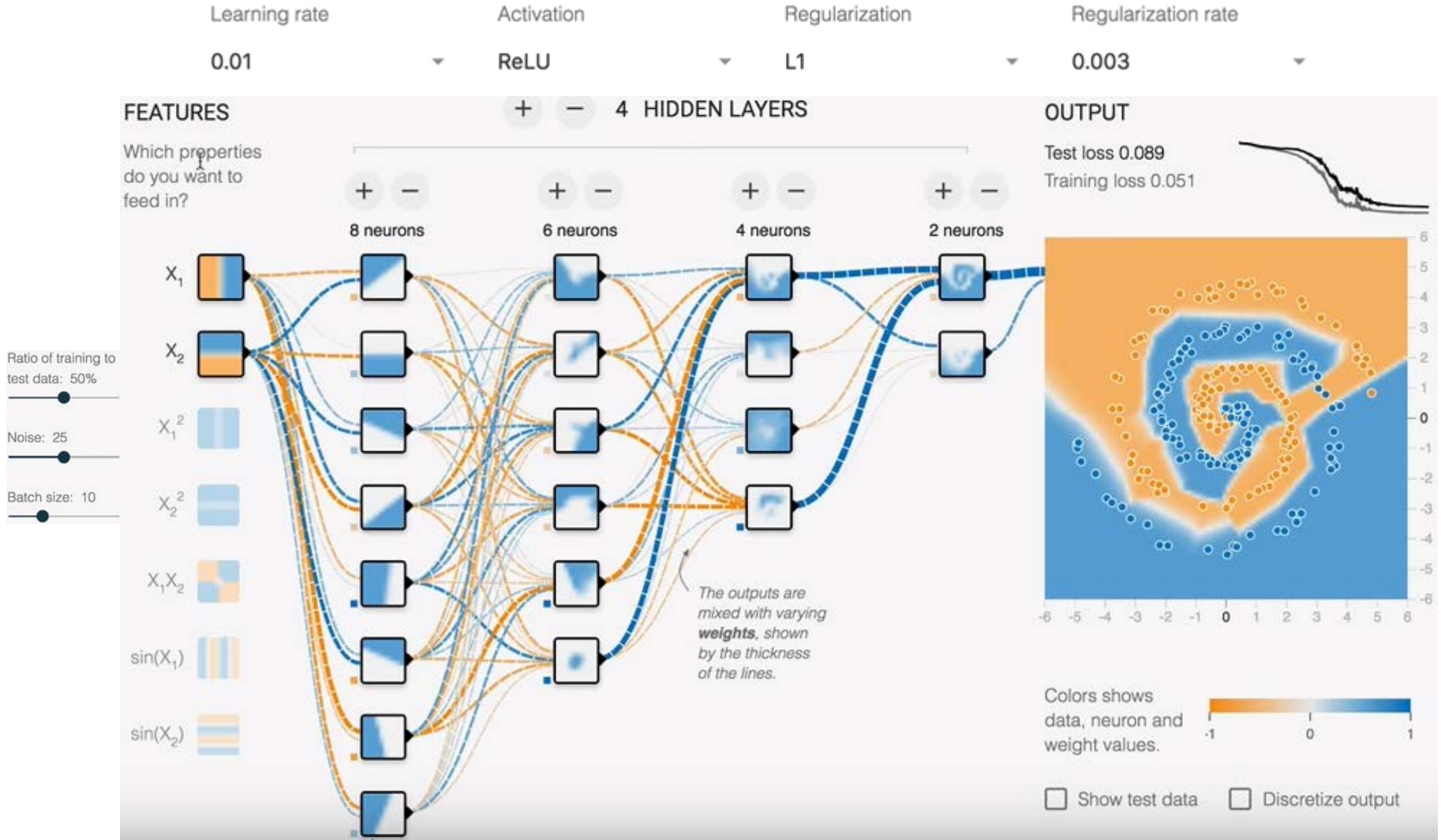
Introduction to Neural Networks

- Train the best model you can, using just X1 and X2. Feel free to add or remove layers and neurons, change learning settings like learning rate, regularization rate, and batch size. What is the best test loss you can get? How smooth is the model output surface?



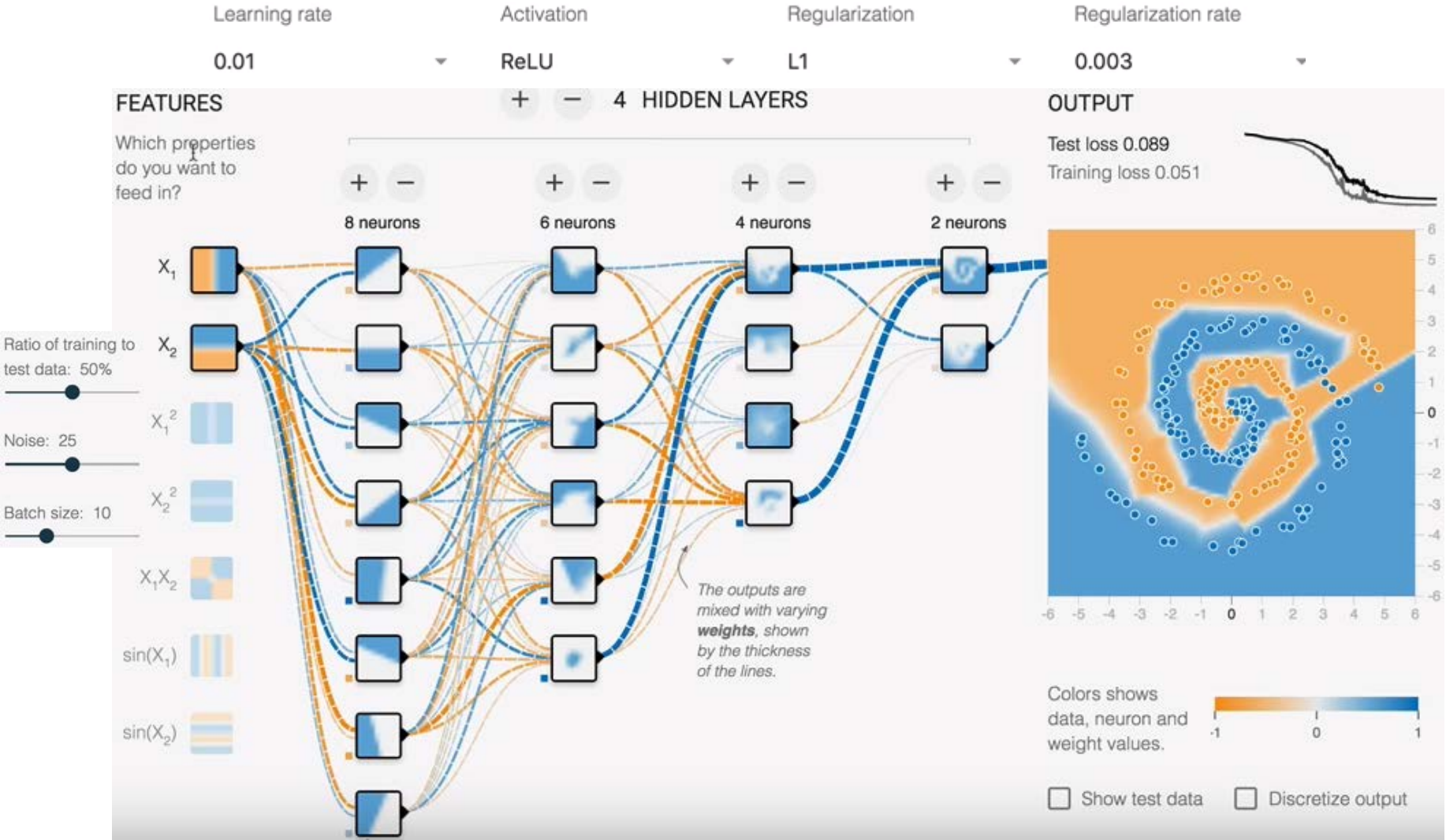
Introduction to Neural Networks

- Train the best model you can, using just X1 and X2. Feel free to add or remove layers and neurons, change learning settings like learning rate, regularization rate, and batch size. What is the best test loss you can get? How smooth is the model output surface?



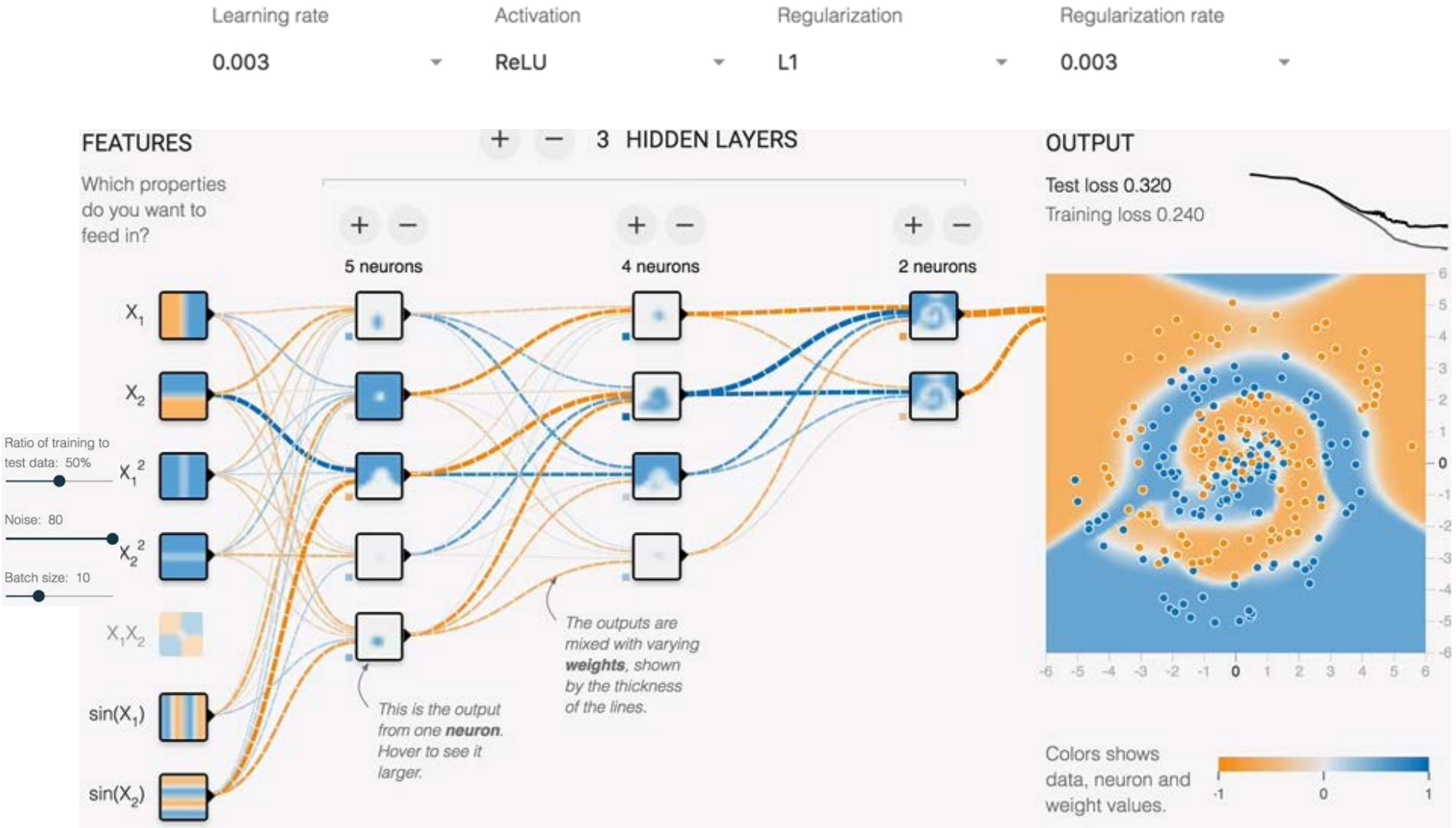
Introduction to Neural Networks

- Train the best model you can, using just X1 and X2. Feel free to add or remove layers and neurons, change learning settings like learning rate, regularization rate, and batch size. What is the best test loss you can get? How smooth is the model output surface?



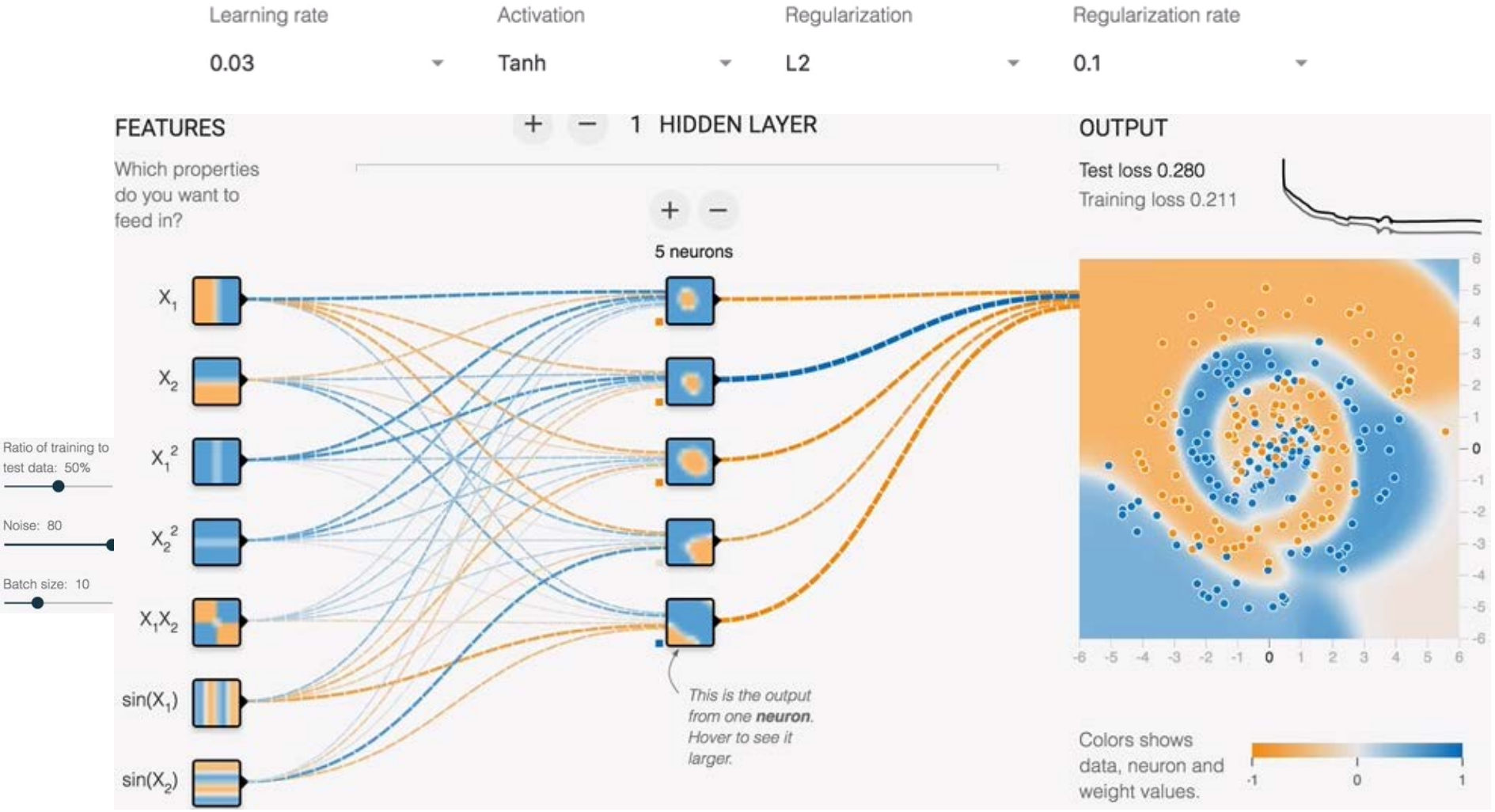
Introduction to Neural Networks

- Train the best model you can, using just X1 and X2. Feel free to add or remove layers and neurons, change learning settings like learning rate, regularization rate, and batch size. What is the best test loss you can get? How smooth is the model output surface?



Introduction to Neural Networks

- Train the best model you can, using just X1 and X2. Feel free to add or remove layers and neurons, change learning settings like learning rate, regularization rate, and batch size. What is the best test loss you can get? How smooth is the model output surface?



Reference

- This lecture note has been developed based on the machine learning crash course at Google, which is under *Creative Commons Attribution 3.0 License*.