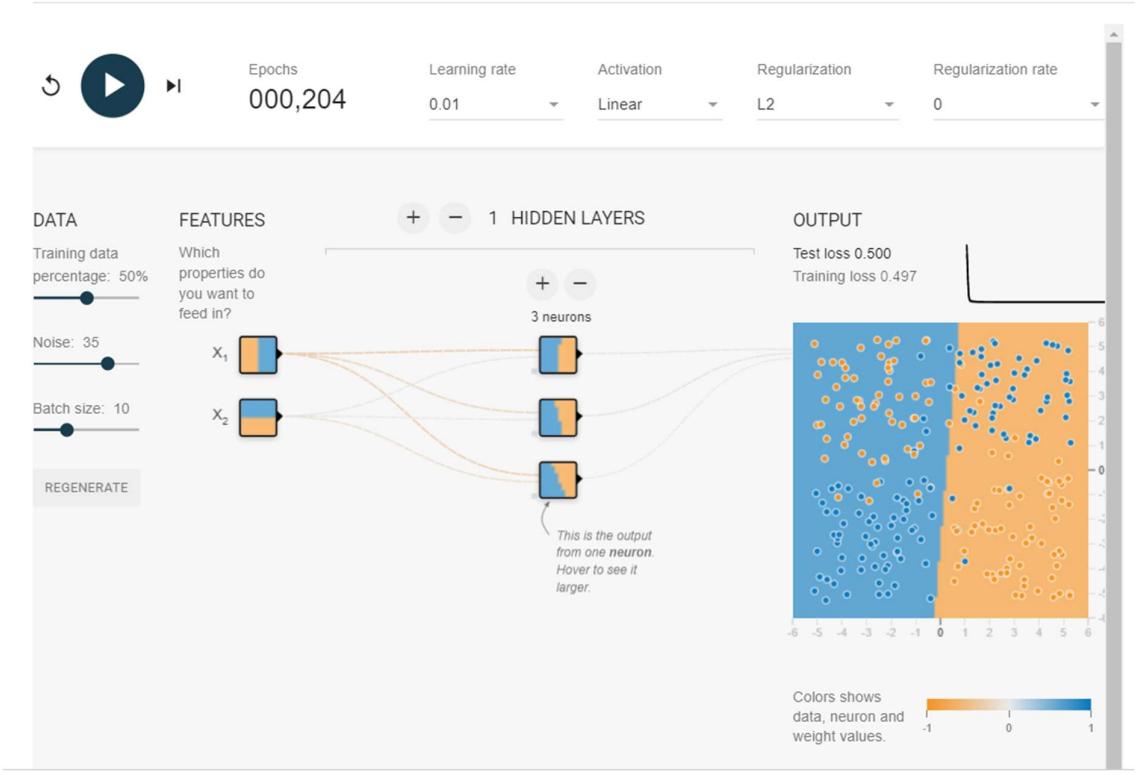
PRML LAB Q-3

Shalin Jain (B21CS0070)

Google Playground

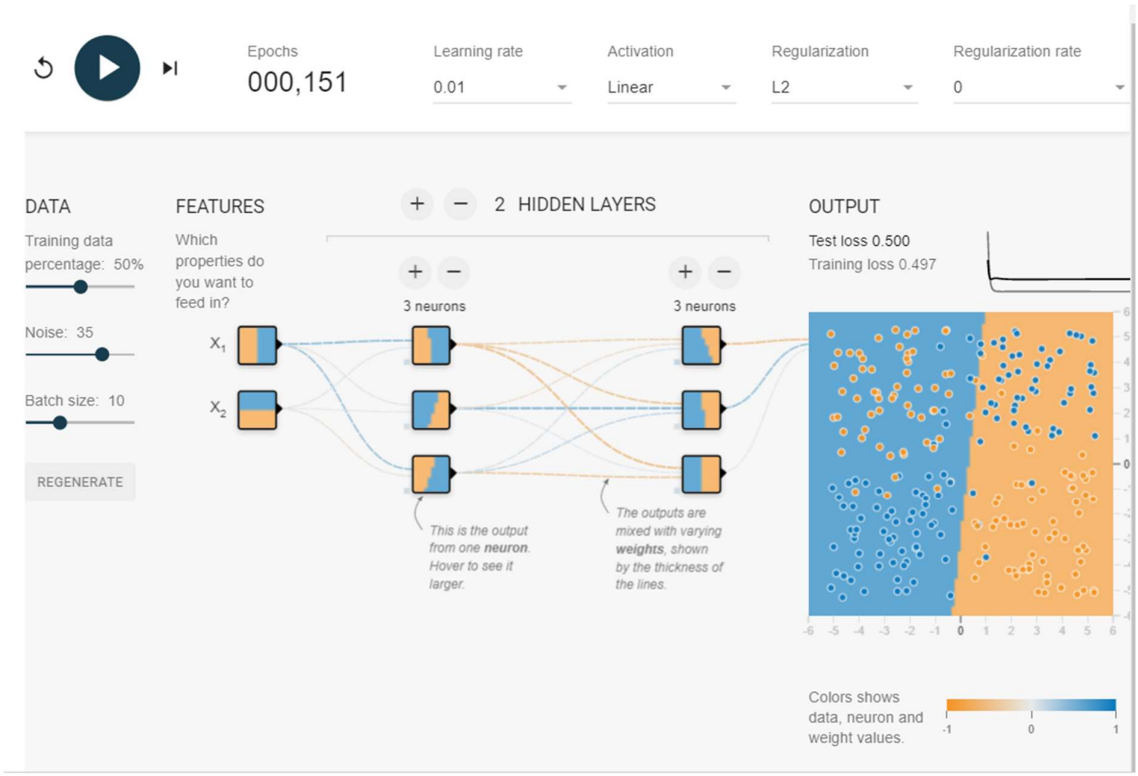
A) Does increasing the model size improve the fit, or how quickly it converges? Does this change how often it converges to a good model? For example, try the following architecture:

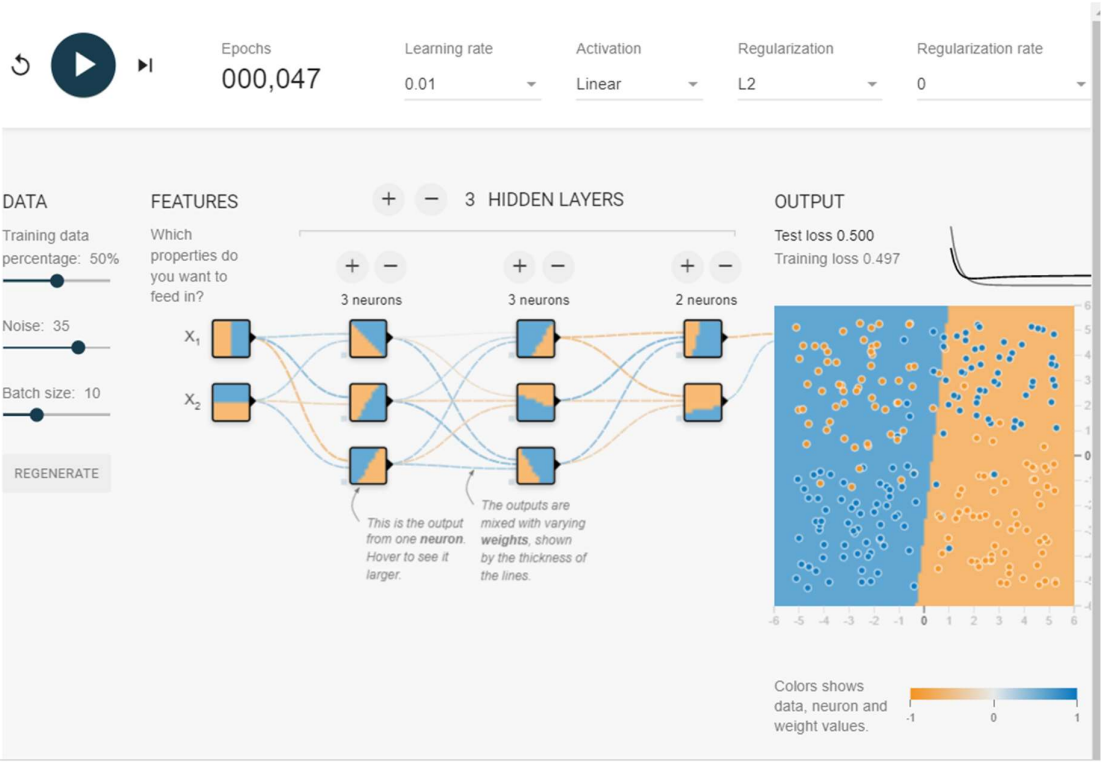
- First hidden layer with 3 neurons.
- Second hidden layer with 3 neurons.
- Third hidden layer with 2 neurons.



As we increase the number of neurons and the number of layers the model get converges in lesser number of epochs but it take a little more time than the lesser number of the neurons. This is also depicted by the screenshots attached below.

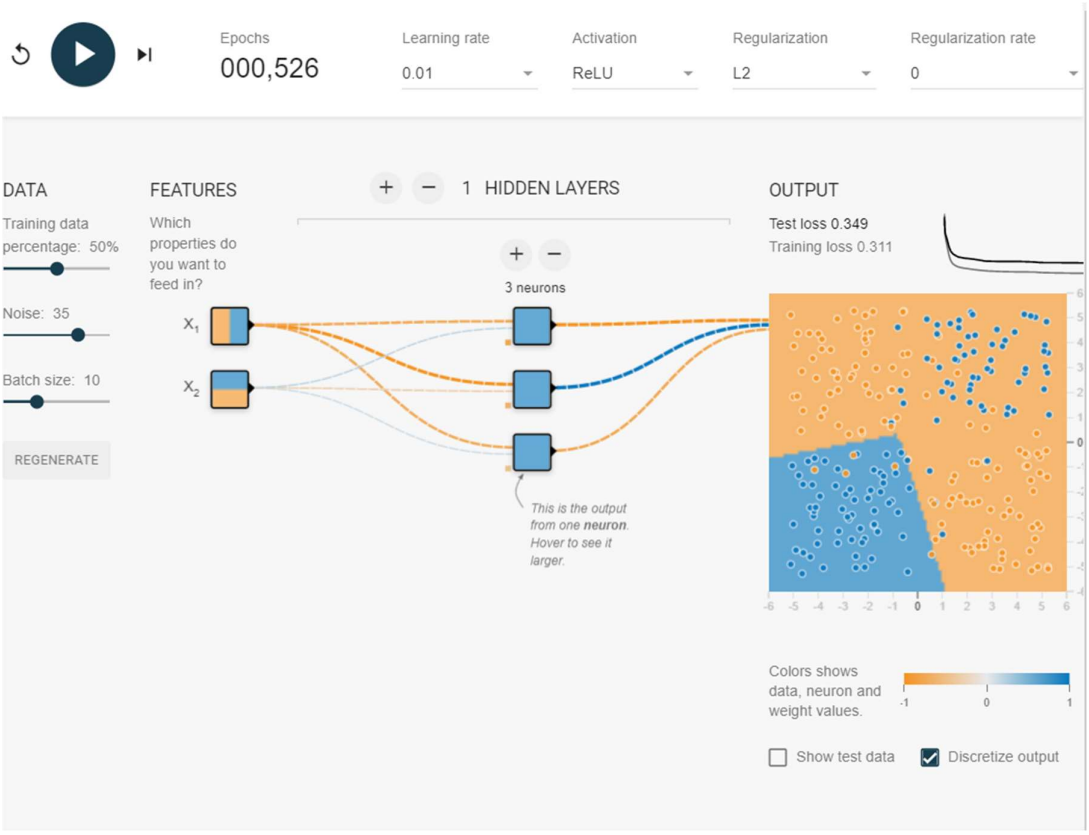
(Answers appear just below the exercise.)

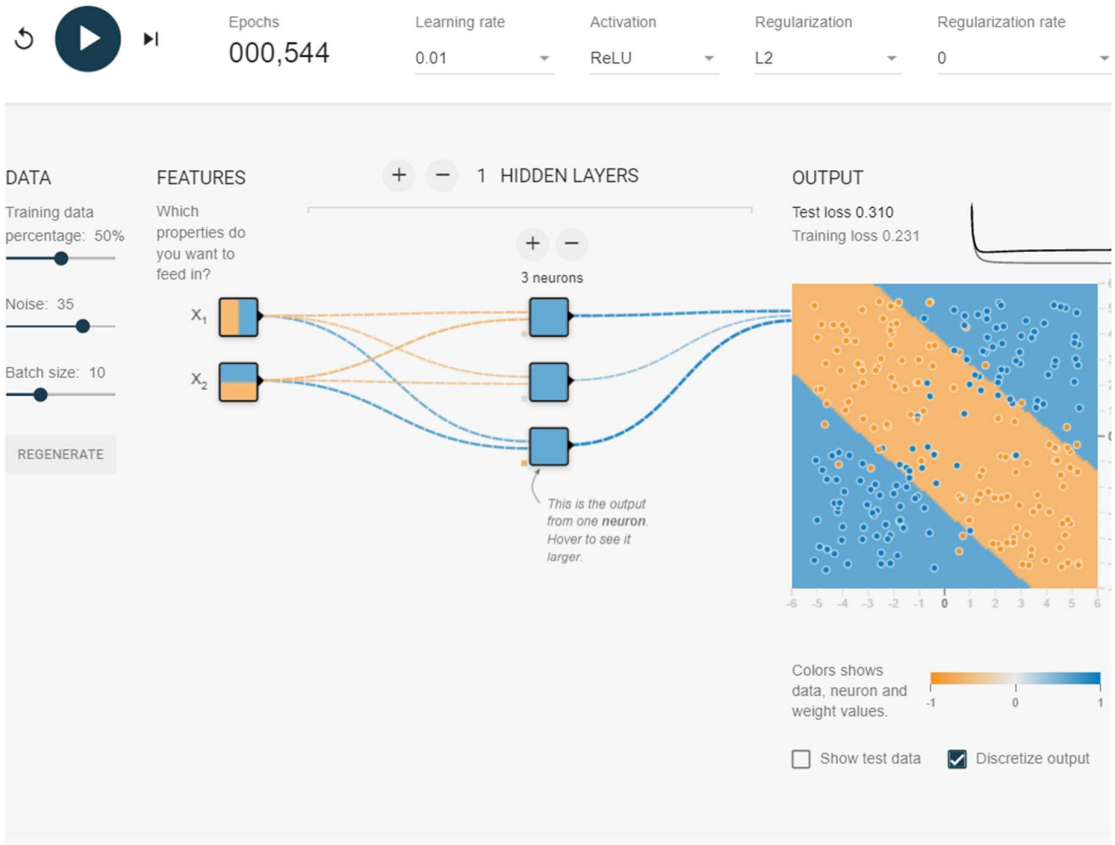


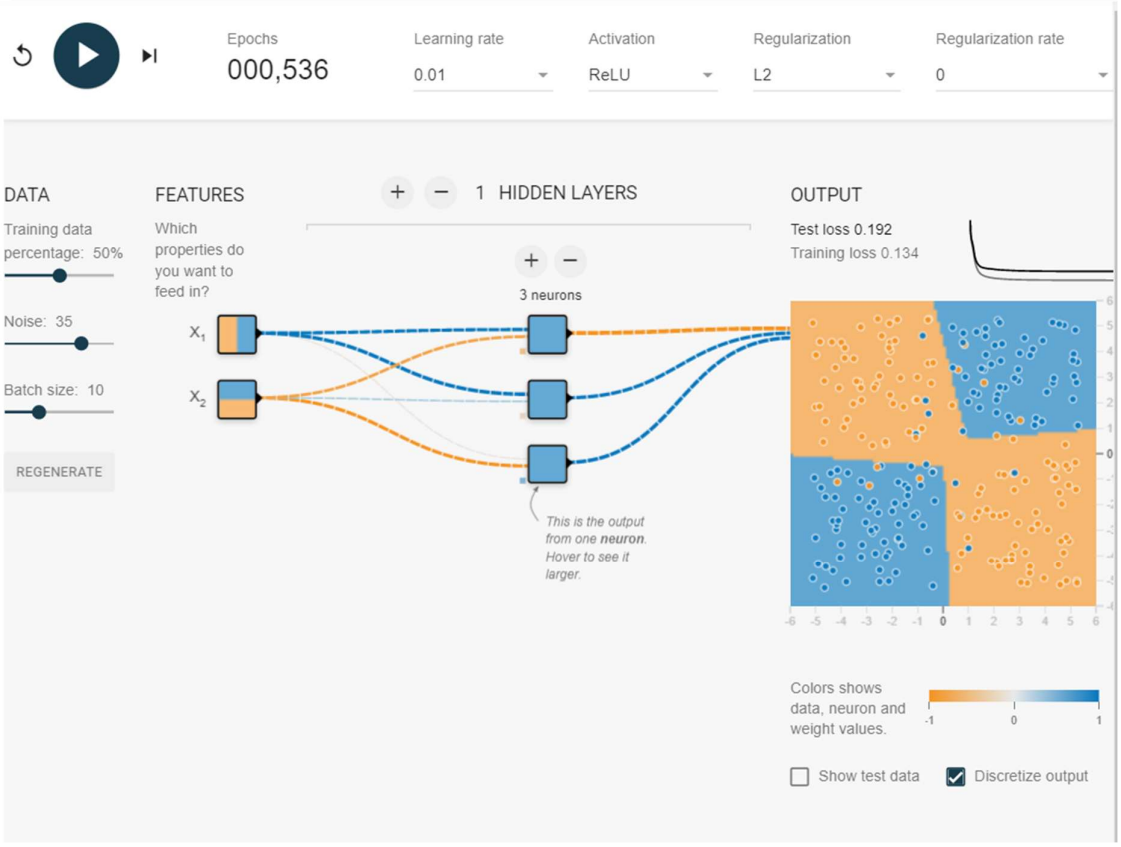


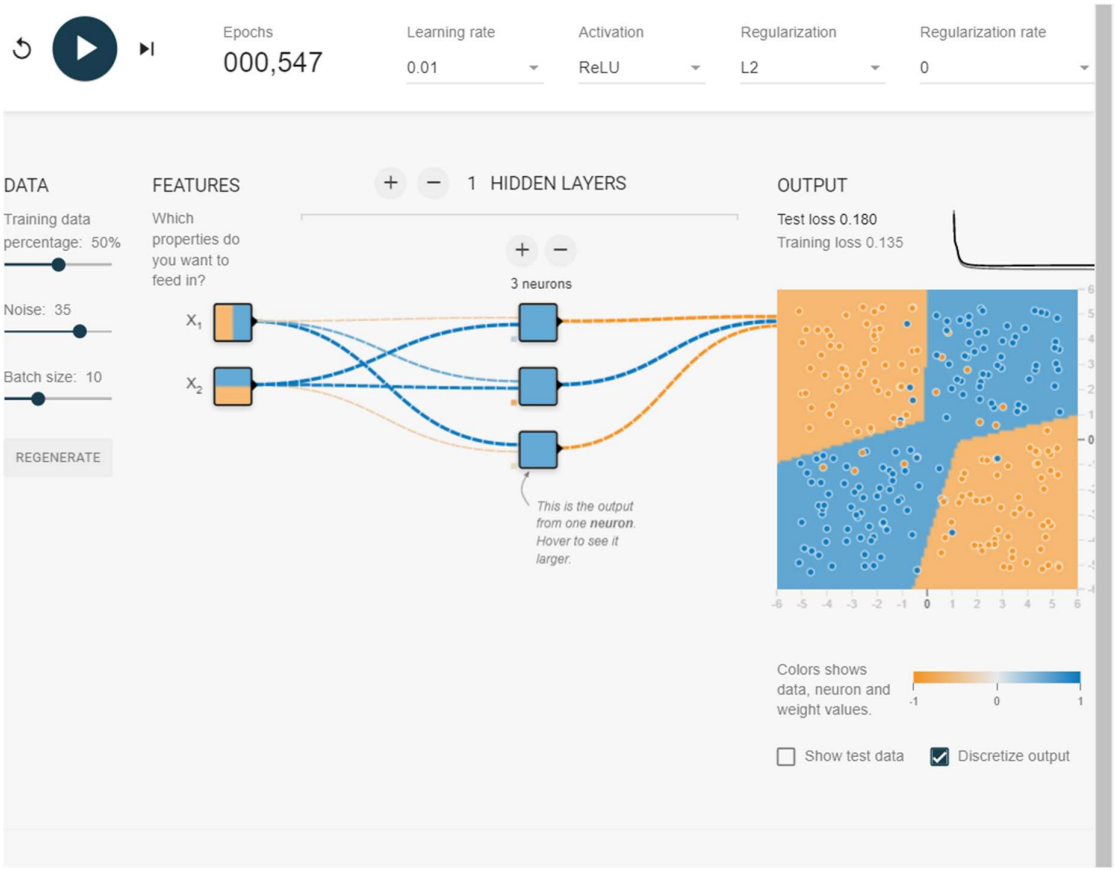
B) Run the model as given four or five times. Before each trial, hit the Reset the network button to get a new random initialization. (The Reset the network button is the circular reset arrow just to the left of the Play button.) Let each trial run for at least 500 steps to ensure convergence. What shape does each model output converge to? What does this say about the role of initialization in non-convex optimization? Try making the model slightly more complex by adding a layer and a couple of extra nodes. Repeat the trials from Task 1. Does this add any additional stability to the results?

In this part we can see that every boundary is different from the other one and on adding one more layer to the network the accuracy might not increase to a large extent.

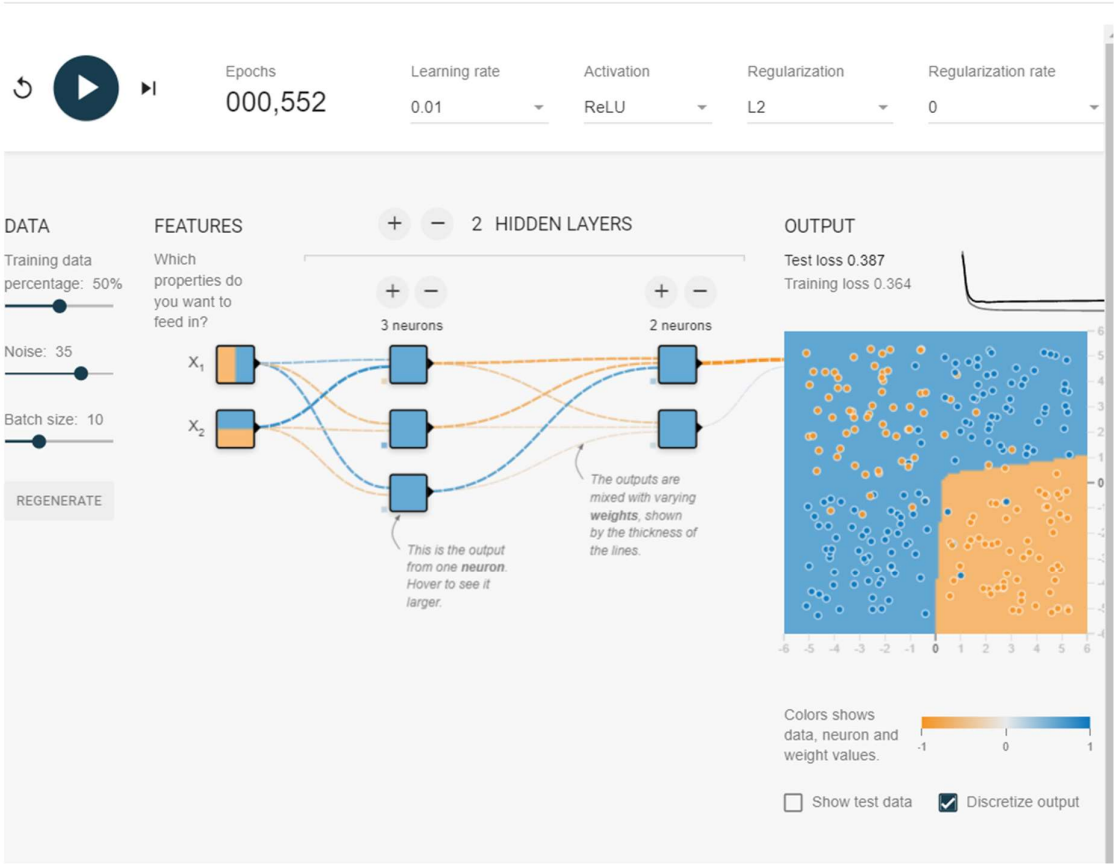


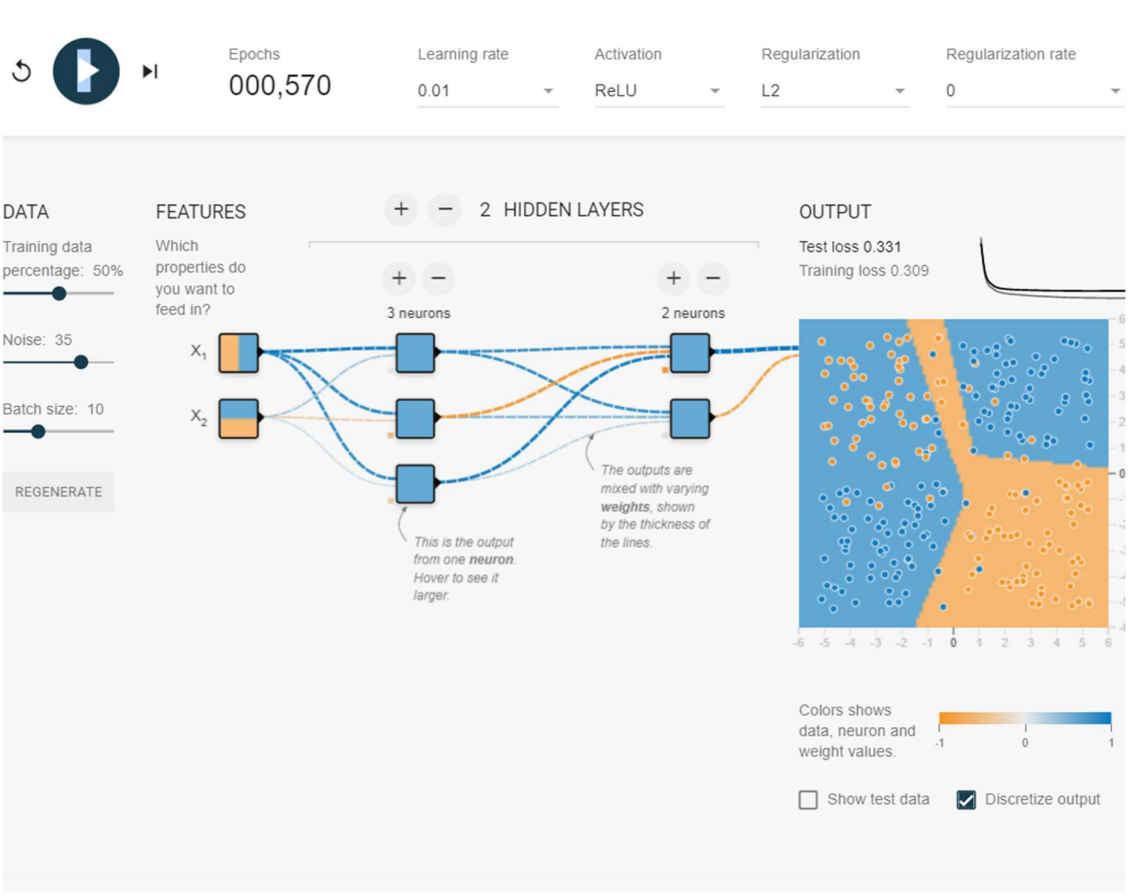

Epochs
000,544
Learning rate
0.01
Activation
ReLU
Regularization
L2
Regularization rate
0
DATA
Training data percentage: 50%
Noise: 35
Batch size: 10
REGENERATE
FEATURES
Which properties do you want to feed in?
1 HIDDEN LAYERS
3 neurons
X1
X2
This is the output from one neuron. Hover to see it larger.
OUTPUT
Test loss 0.310
Training loss 0.231
Colors shows data, neuron and weight values.
Show test data
Discretize output

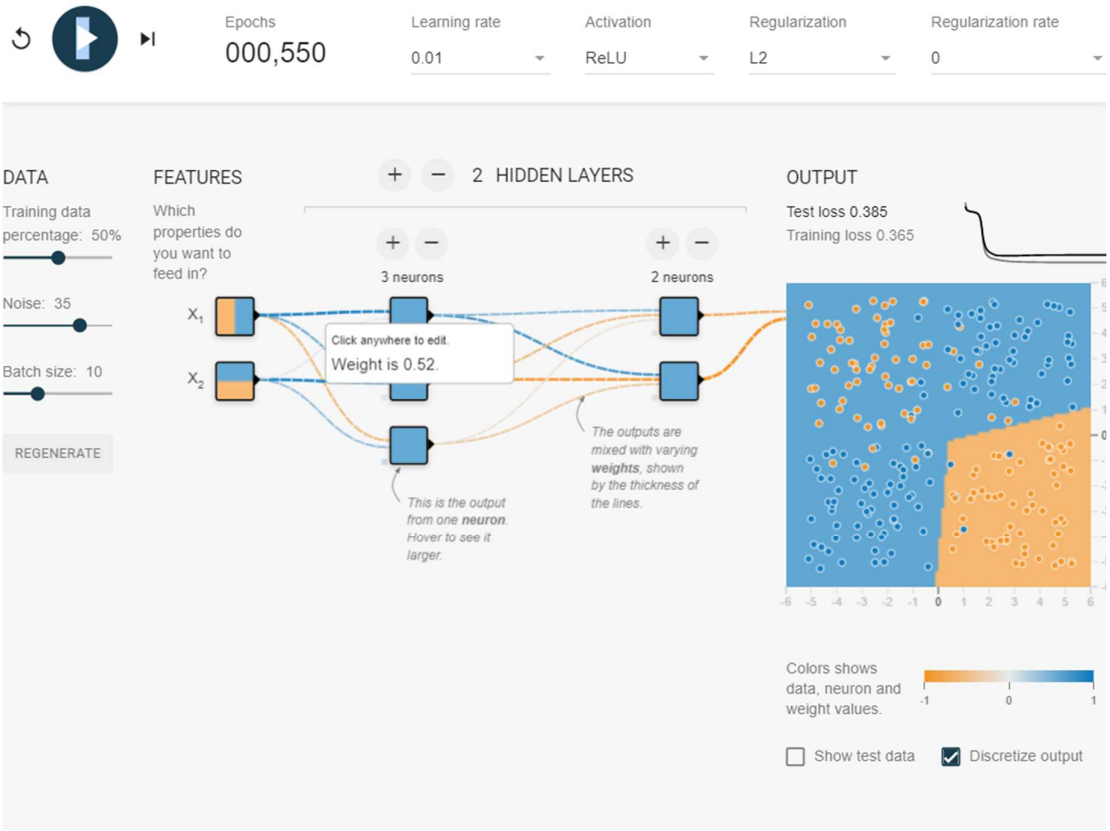

Epochs
000,536
Learning rate
0.01
Activation
ReLU
Regularization
L2
Regularization rate
0
DATA
Training data percentage: 50%
Noise: 35
Batch size: 10
REGENERATE
FEATURES
Which properties do you want to feed in?
1 HIDDEN LAYERS
3 neurons
X1
X2
This is the output from one neuron. Hover to see it larger.
OUTPUT
Test loss 0.192
Training loss 0.134
Colors shows data, neuron and weight values.
Show test data
Discretize output

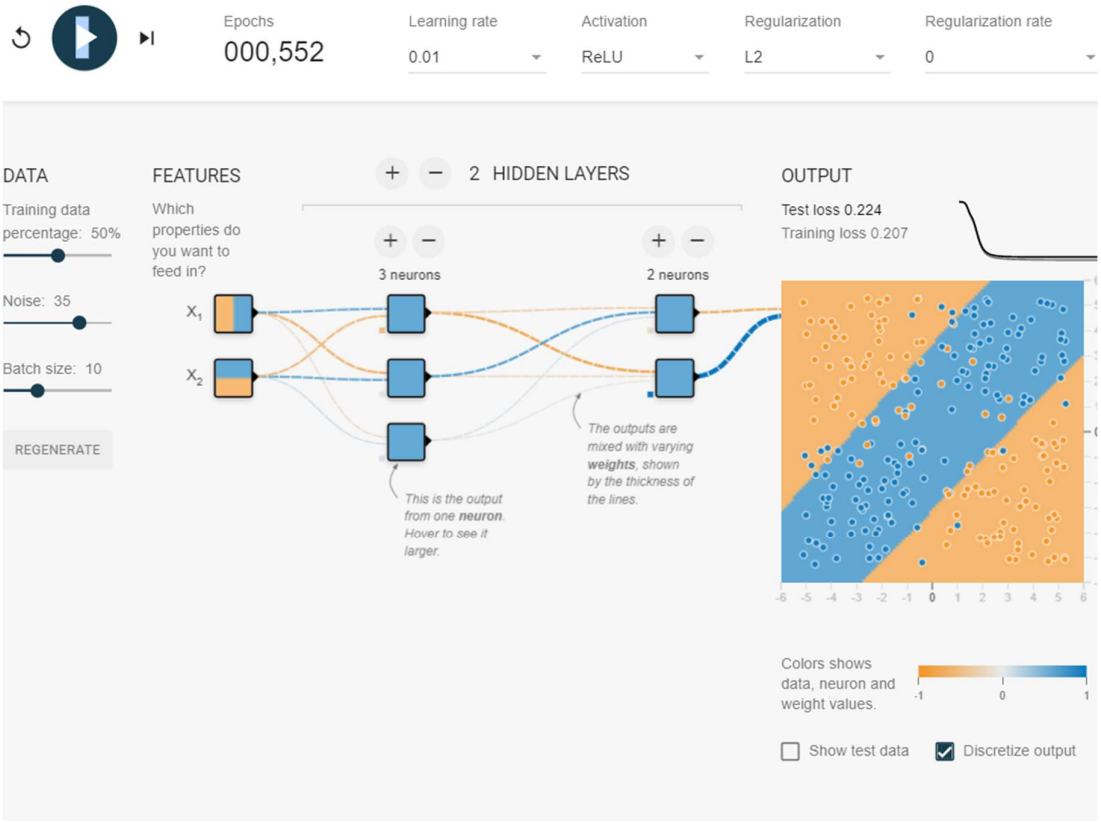


More layers and nodes

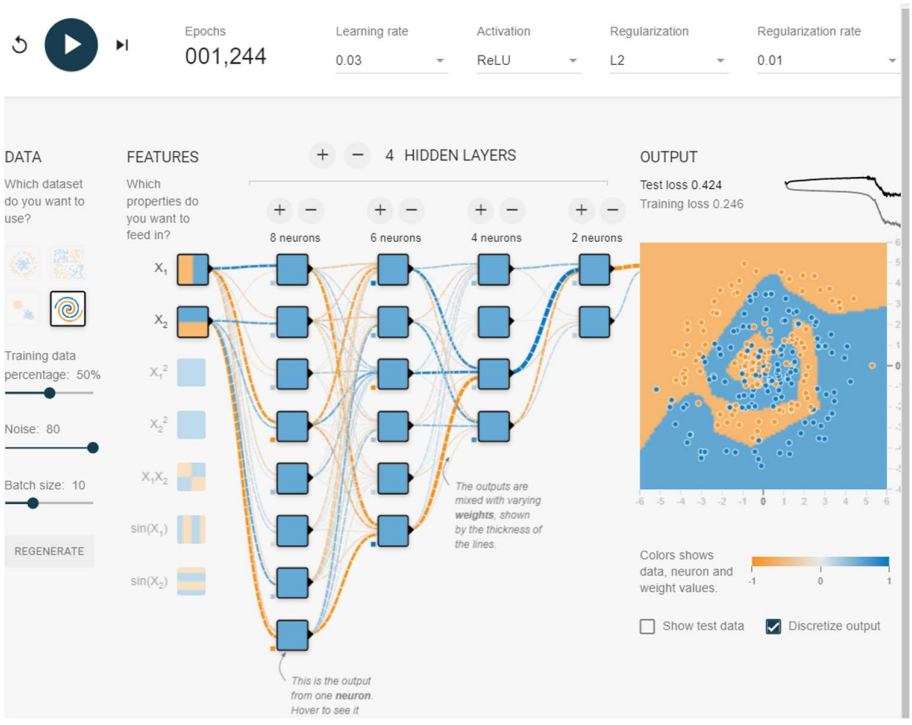


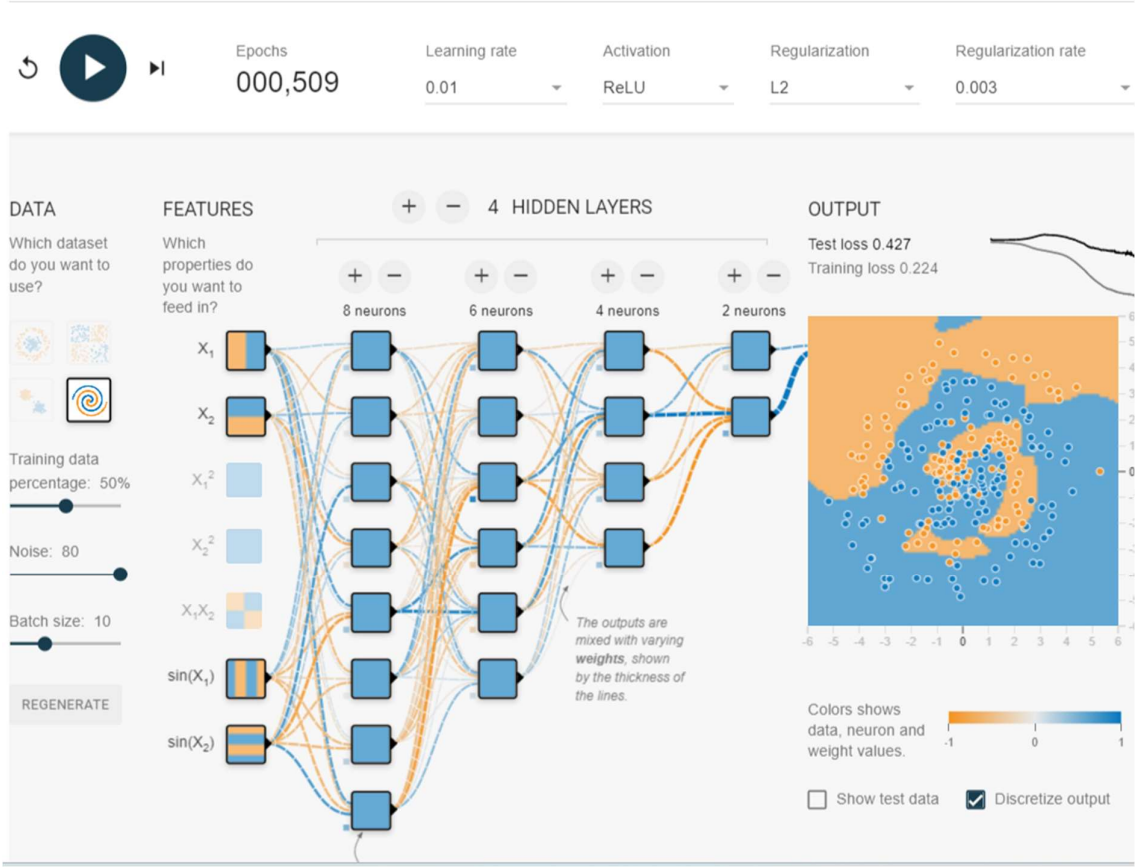

Epochs
000,570
Learning rate
0.01
Activation
ReLU
Regularization
L2
Regularization rate
0
DATA
Training data percentage: 50%
Noise: 35
Batch size: 10
REGENERATE
FEATURES
Which properties do you want to feed in?
2 HIDDEN LAYERS
3 neurons
2 neurons
This is the output from one neuron. Hover to see it larger.
The outputs are mixed with varying weights, shown by the thickness of the lines.
OUTPUT
Test loss 0.331
Training loss 0.309
Colors shows data, neuron and weight values.
Show test data
Discretize output


Epochs
000,550
Learning rate
0.01
Activation
ReLU
Regularization
L2
Regularization rate
0
DATA
Training data percentage: 50%
Noise: 35
Batch size: 10
REGENERATE
FEATURES
Which properties do you want to feed in?
2 HIDDEN LAYERS
3 neurons
2 neurons
Click anywhere to edit.
Weight is 0.52.
This is the output from one neuron. Hover to see it larger.
The outputs are mixed with varying weights, shown by the thickness of the lines.
OUTPUT
Test loss 0.385
Training loss 0.365
Colors shows data, neuron and weight values.
Show test data
Discretize output



C) 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? Even with Neural Nets, some amount of feature engineering is often needed to achieve best performance. Try adding in additional cross product features or other transformations like sin(X1) and sin(X2). Do you get a better model? Is the model output surface any smoother?

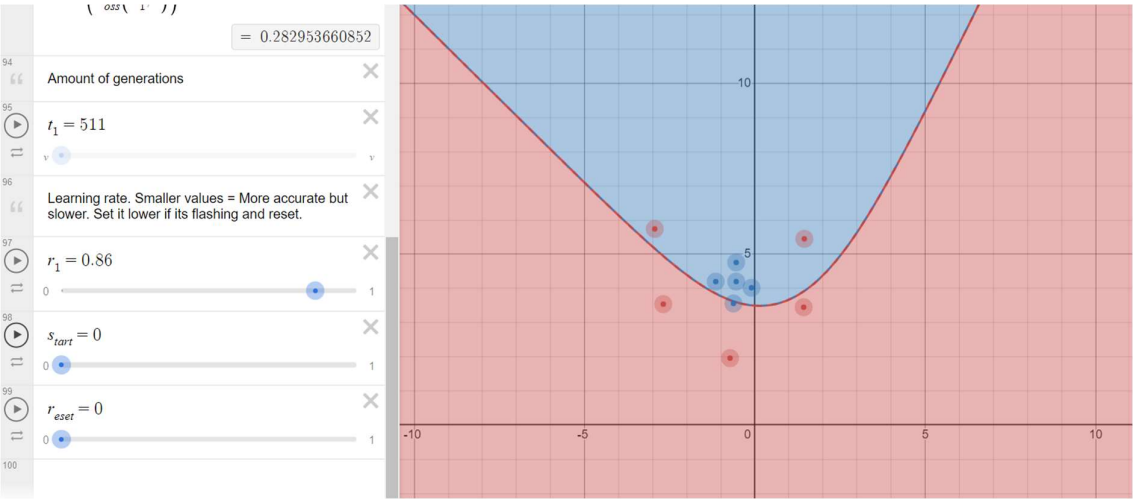




On normally using two features is accurate but take a large amount of time but if we introduce a complex feature function then the time take for getting the boundary is very less.

Neural Nets

Try tweaking the learning rate and report your observations regarding the stability of training by observing the decision boundary.



We see that at learning rate of 0.86 and epochs > 500 the almost perfectly shows the boundaries.