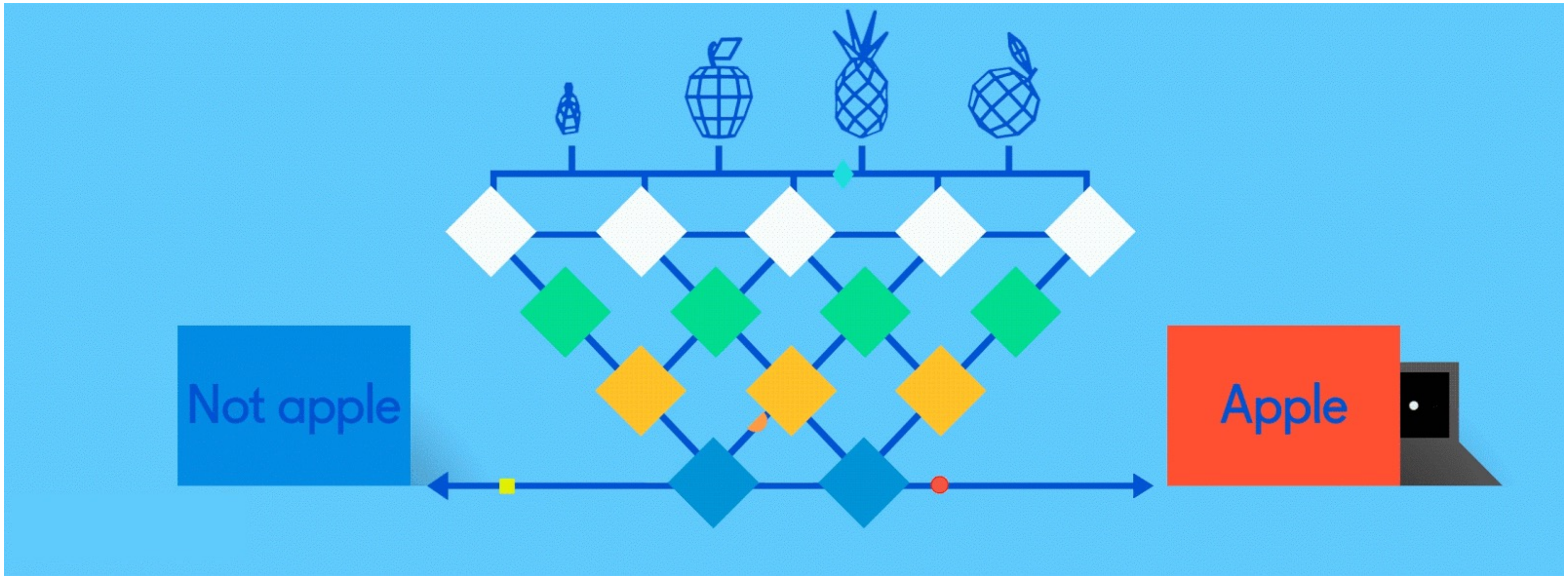
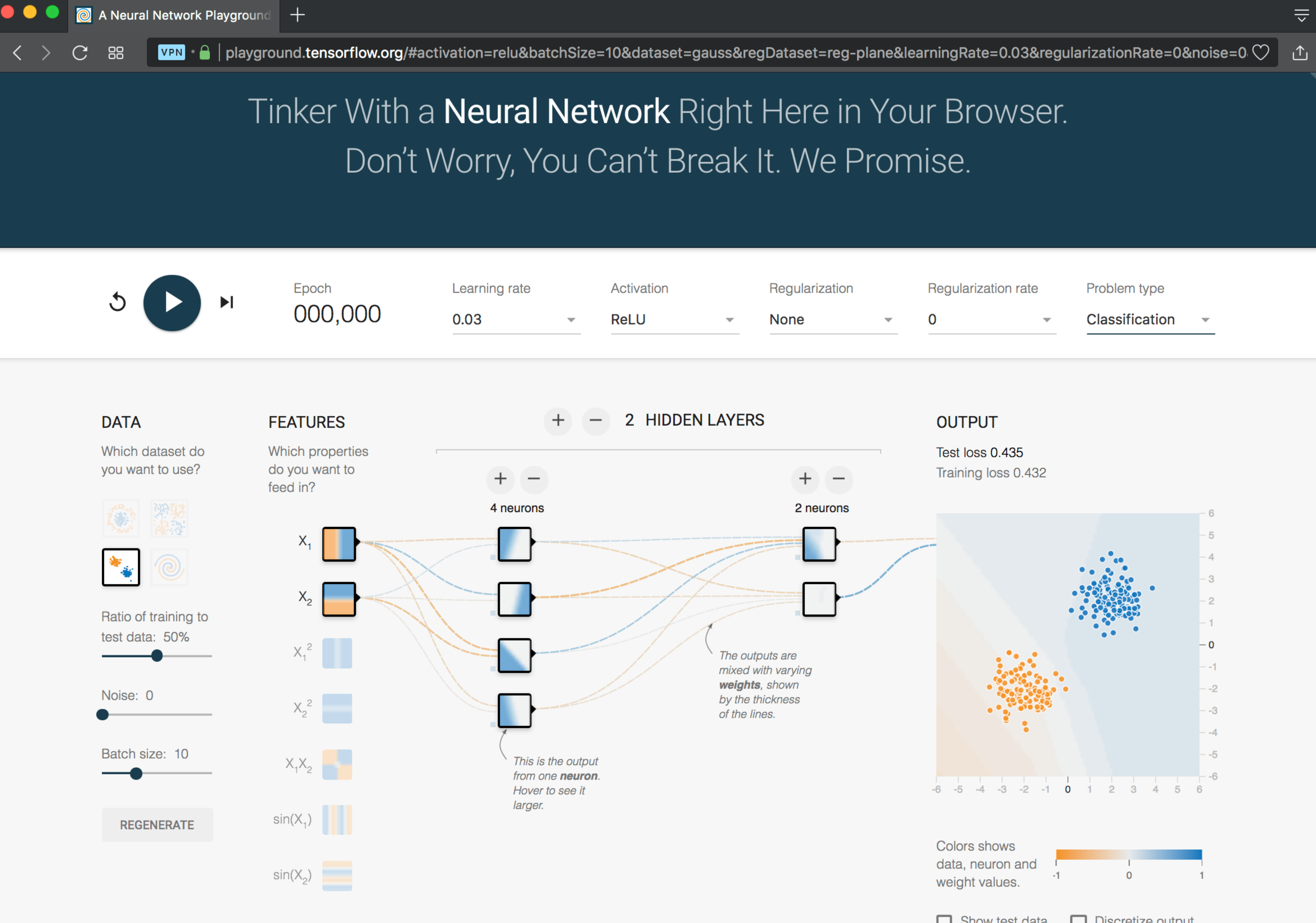
Neural Networks for Machine Learning

tensorflow playground



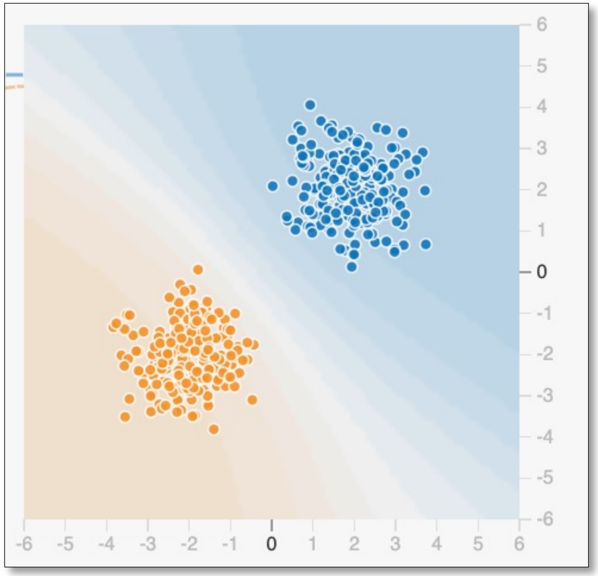
TensorFlow Playground

- Great javascript app demonstrating many basic neural network concepts (e.g., MLPs)
- Doesn't use TensorFlow software, but a lightweight js library
- Runs in a Web browser
- See **http://playground.tensorflow.org/**
- Code also available on GitHub
- Try the playground exercises in Google's machine learning crash course



[HTTP://PLAYGROUND.TENSORFLOW.ORG/](http://playground.tensorflow.org/)

Datasets



- Six datasets, each with 500 (x,y) points on a plane where x and y between -5 and +5
- Points have *labels* of positive (orange) or negative (blue)
- Two possible machine learning *tasks*:
  - Classification: Predict class of test points
  - Regression: find function to separate classes
- *Evaluation*: split dataset into training and test, e.g., 70% training, 30% test

Available Input features

X_1 Point's x value

X_2 Point's y value

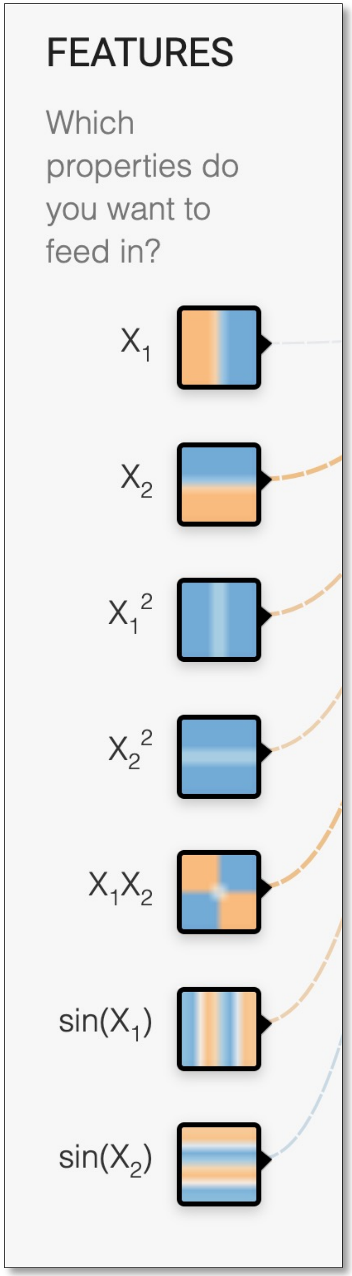
X_1^2 Point's x value squared

X_2^2 Point's y value squared

X_1X_2 Product of point's x & y values

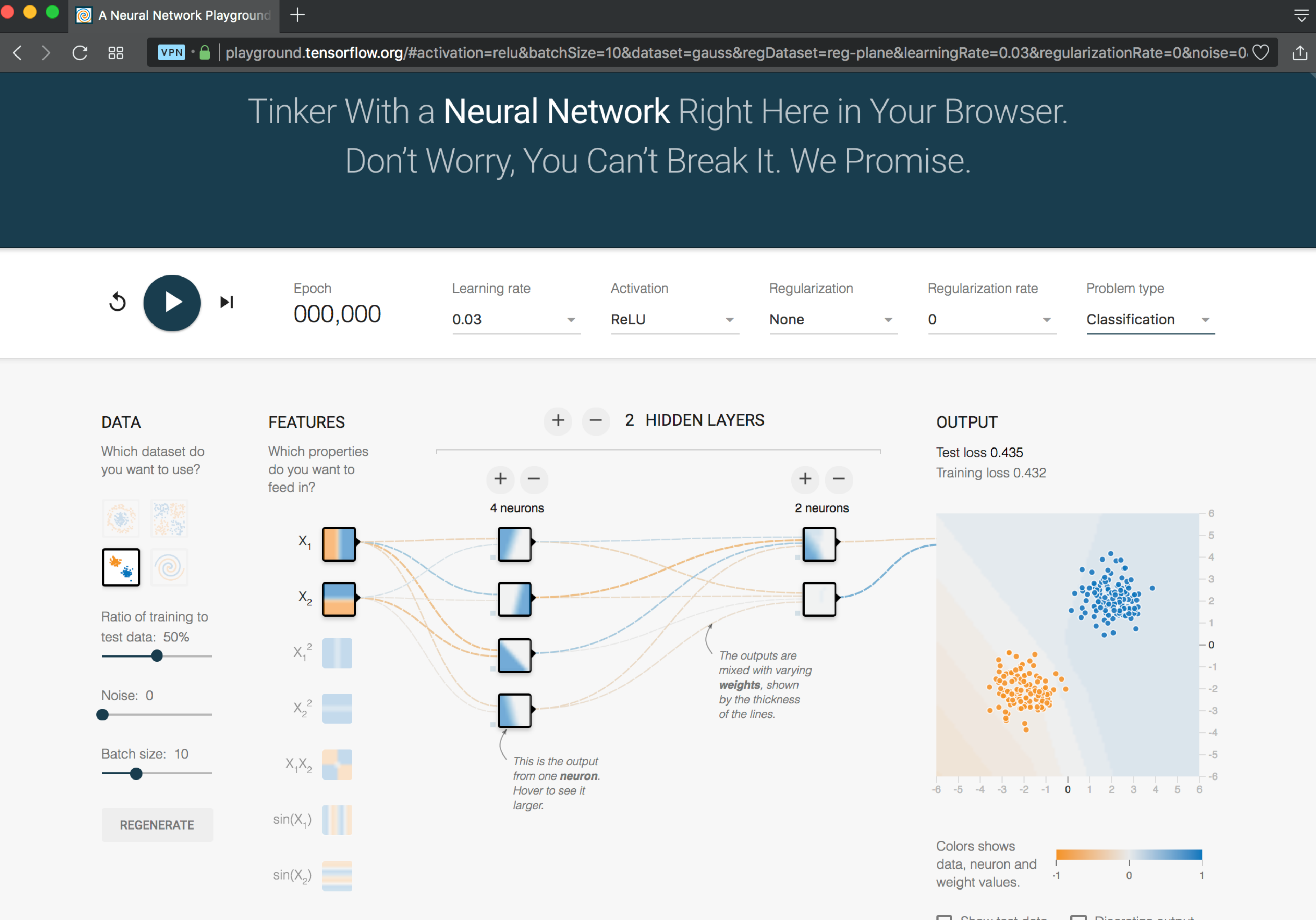
$\sin(X_1)$ Sine of point's x value

$\sin(X_2)$ Sine of point's y value



Designing a neural network

- Simple feed forward NNs have a few choices
  - What input features to use
  - How many hidden layers to have
    - How many neurons are in each layer
    - How each layer is connected to ones before & after
- Complex NNs have more choices
  - E.g., CNNs, RNNs, etc.
- High-level interfaces (Keras, TensorFlow, PyTorch, …) try to make this easier



[HTTP://PLAYGROUND.TENSORFLOW.ORG/](http://playground.tensorflow.org/)

Training a Neural Network

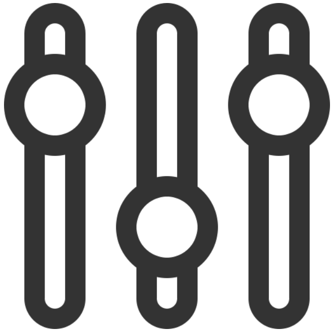
- Neural networks are used for **supervised** machine learning and need to be trained
- The training process is broken done in a series of ***epochs***

  In each epoch, all of the training data is run through the system to adjust the NN parameters
- Process ends after a fixed # of epochs or when error rate flattens or starts increasing

Typical Training Flow

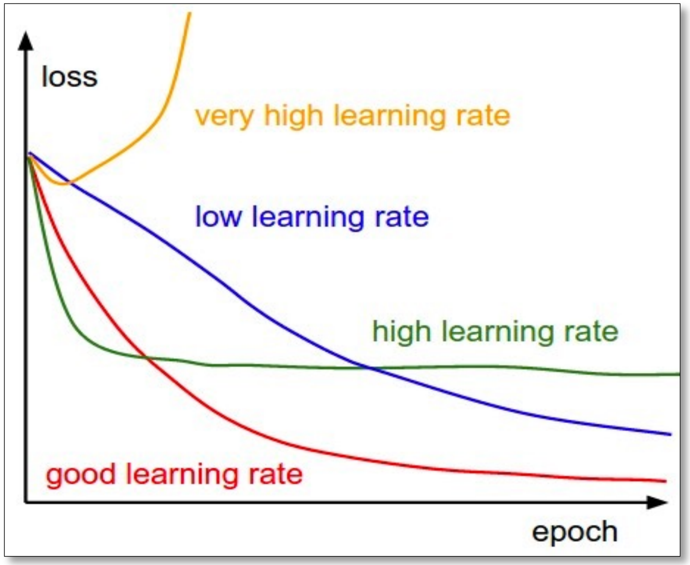
- Divide training data into batches of instances (e.g., batch size = 10)
- For each epoch:
  - For each batch:
    - Instances run through network, noting difference between predicted and actual value
    - Backpropagation used to adjust connection weights
  - Stop when training loss flatten out
- If test loss is too high, then try
  - Adding additional hidden layers
  - Adding more features to inputs
  - Adjusting hyperparameters (e.g., learning rate)
  - Get more training data

Hyperparameters



- Parameters whose values are set before the learning process begins
- Basic neural network hyperparameters
  - Learning rate (e.g., 0.03)
  - Activation function (e.g., ReLU)
  - Regularization (e.g., L2)
  - Regularization rate (e.g., 0.1)

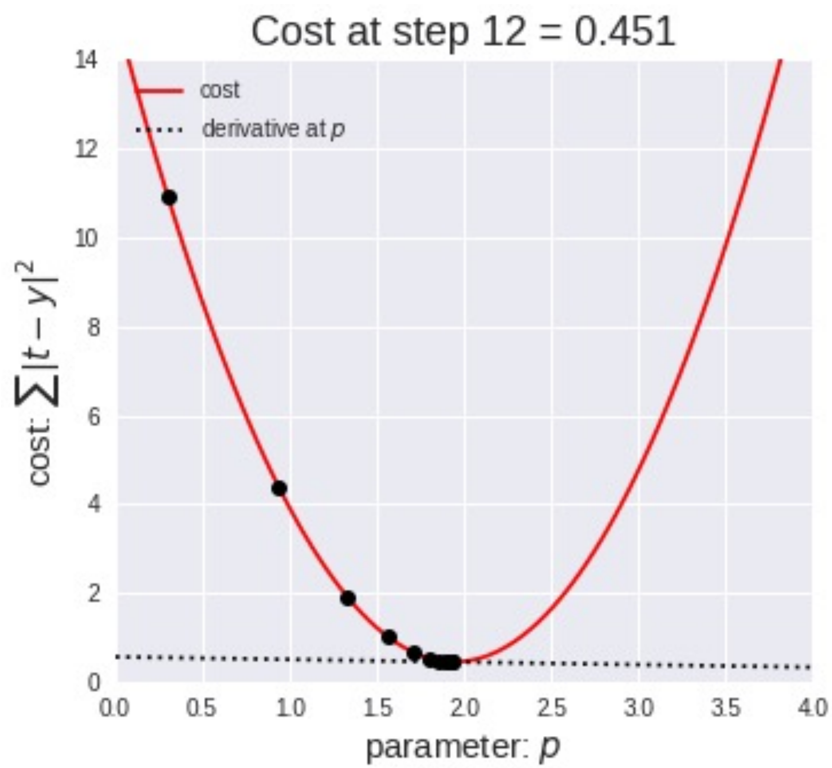
Learning rate



- Gradient descent used in backpropagation to adjust weights to minimize the loss function
- Learning rate determines how much weights are adjusted each time
- If too high, we may miss some or most minima
  - Result: erratic performance or never achieving a low loss
- If too low, learning will take longer than necessary

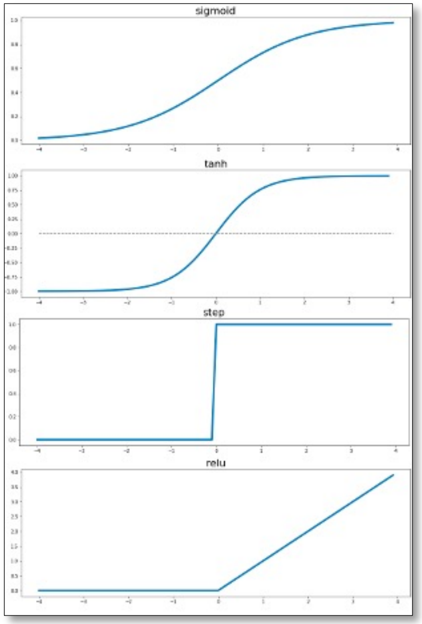
Gradient Descent

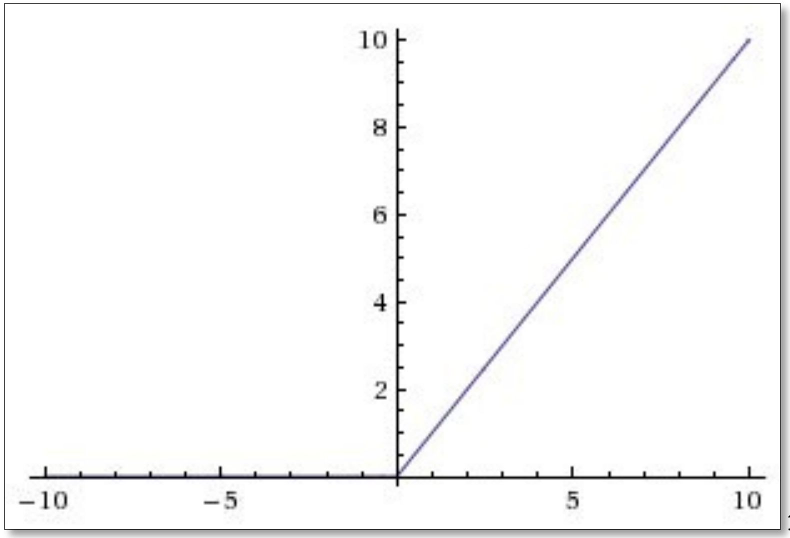
- Iterative process used in ML to find local minimum in our loss function measuring errors
- Moves in direction of steepest descent
- Step size decreases as steepness lessens to avoid missing minima
- Custom variants for NNs include [adam optimization]



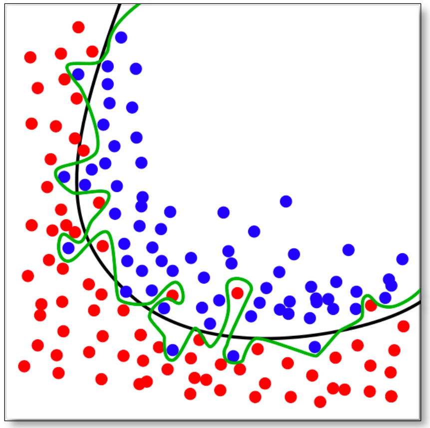
Activation Function

- Determines a node's output given its inputs
- The ReLu (rectified linear unit) is simple and a good choice for most networks
- Returns zero for negative values and its input for positive ones
  - $f(x) = \max(0,x)$





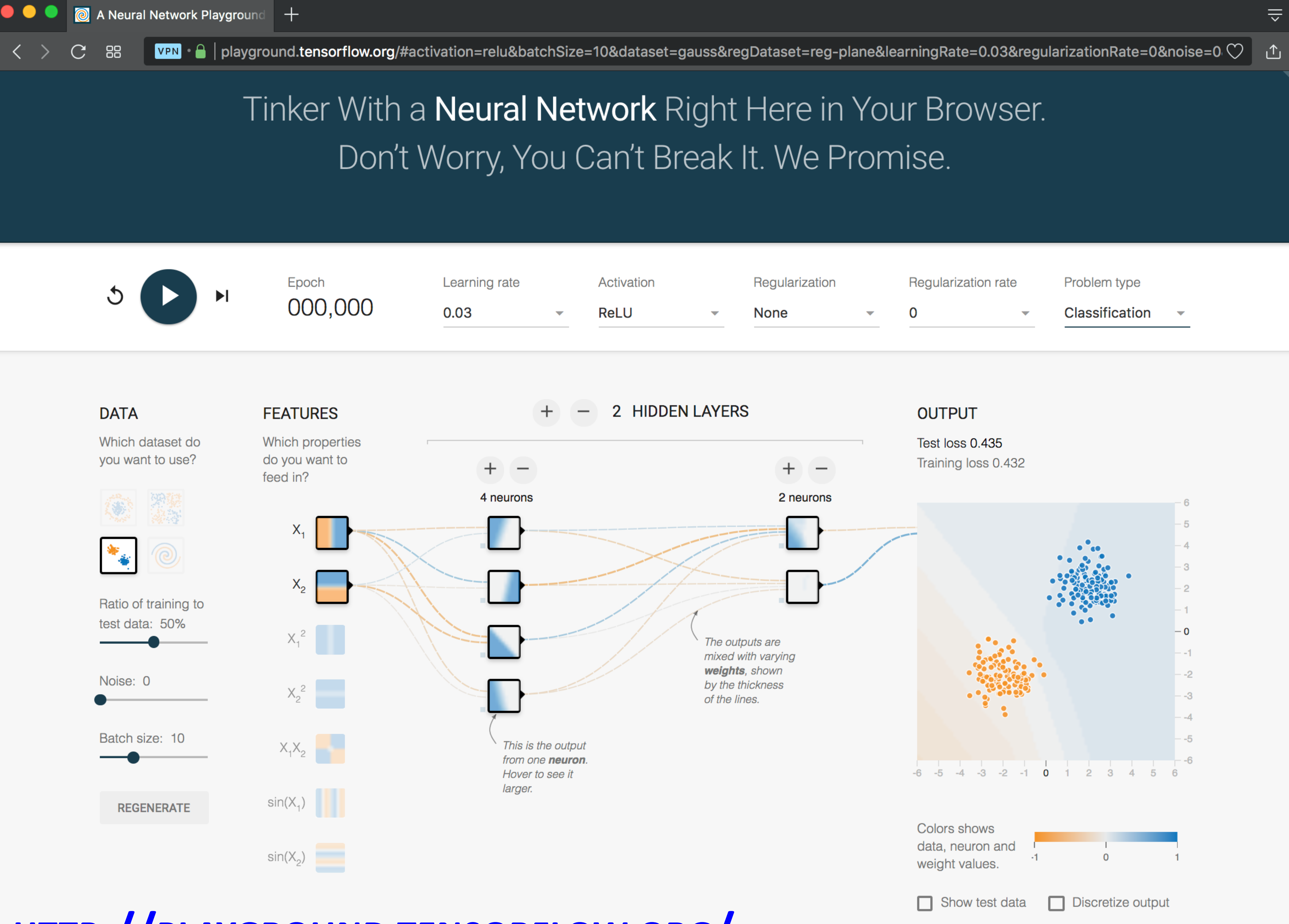
Regularization



- Parameter to control overfitting, i.e. when the model does well on training data but poorly on new, unseen data
- L2 regularization is the most common
- Using dropout is another common way of controlling overfitting in neural networks
  - At each training stage, some hidden nodes temporarily removed (dropped out)

Hyperparameter optimization

- How do we find the best settings for these hyperparameters?
- Experimentation
  - Experiment with a range of different settings (e.g., for learning rate) via multiple runs
  - Use a grid search tool, e.g., scikit learn's
- Experience
  - Similar problems with similar data will probably benefit from similar settings



[HTTP://PLAYGROUND.TENSORFLOW.ORG/](http://playground.tensorflow.org/)