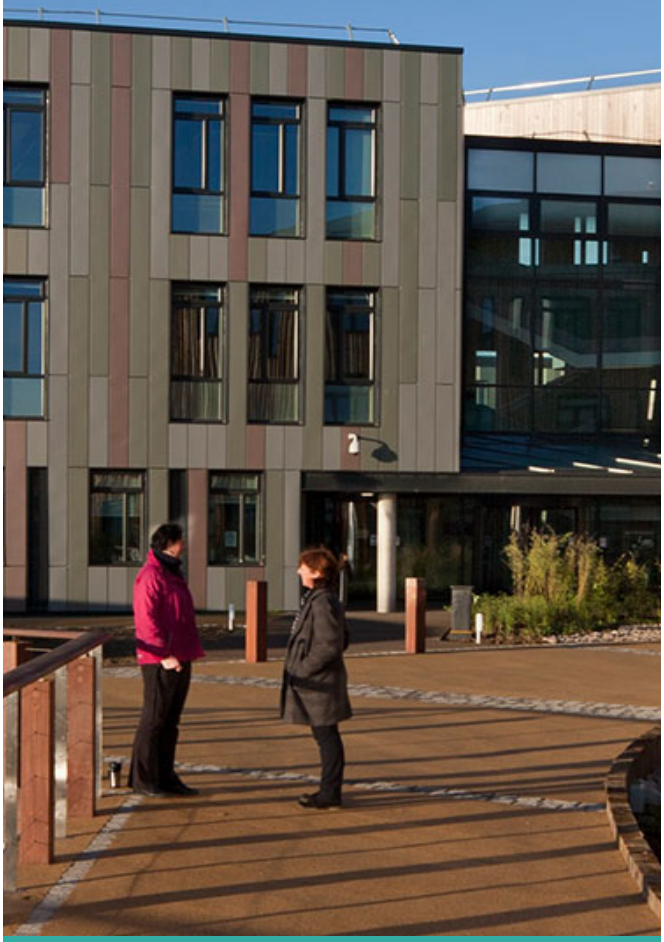
Headstart [2019]

I can see what you see

Looking through the eyes of a computer



11 JULY 2019

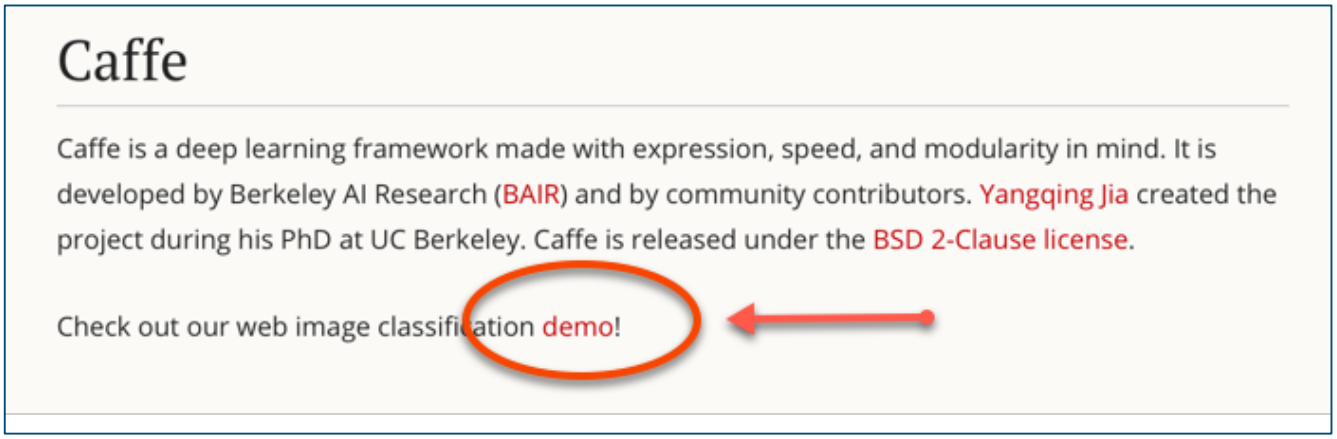
University of York
Colin Paterson

Exercise 1: Classifying Images

Opening the image classifier

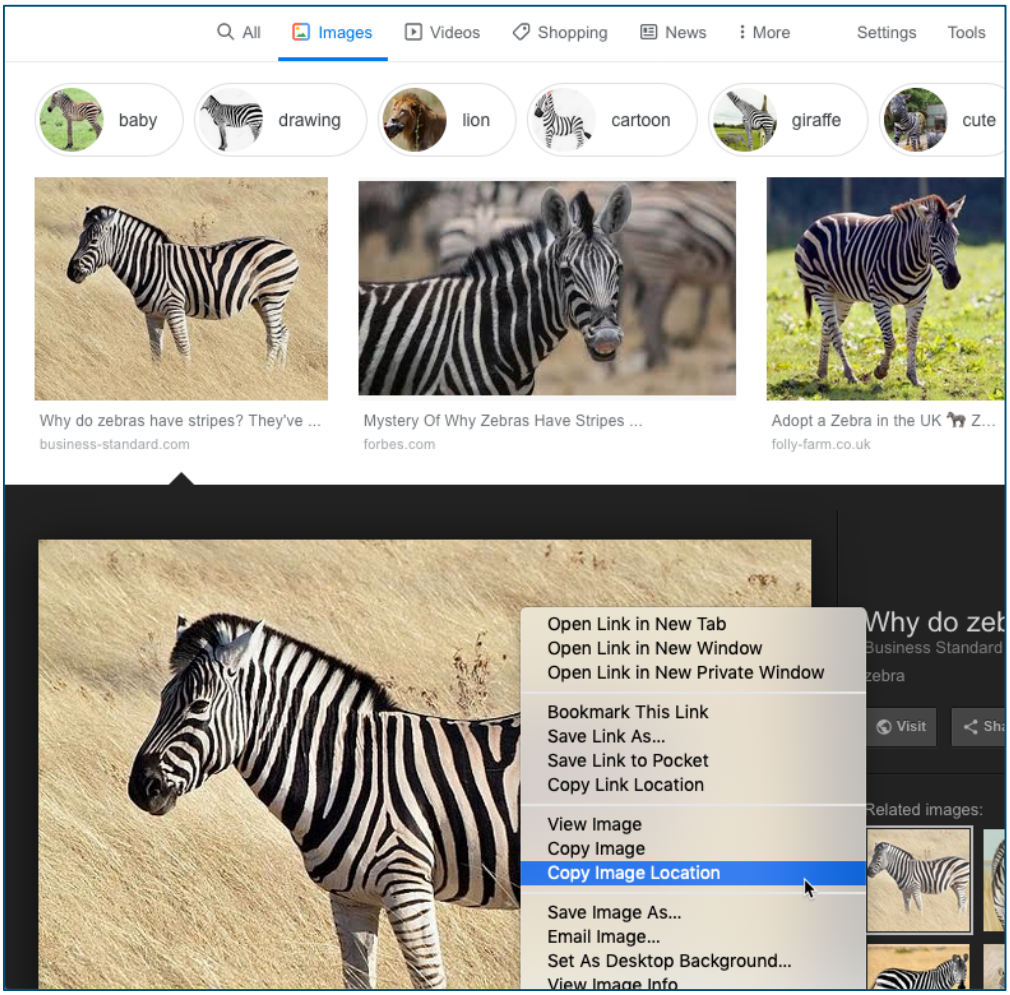
Start by opening the Firefox web browser and navigate to

http://caffee.berkeleyvision.org/



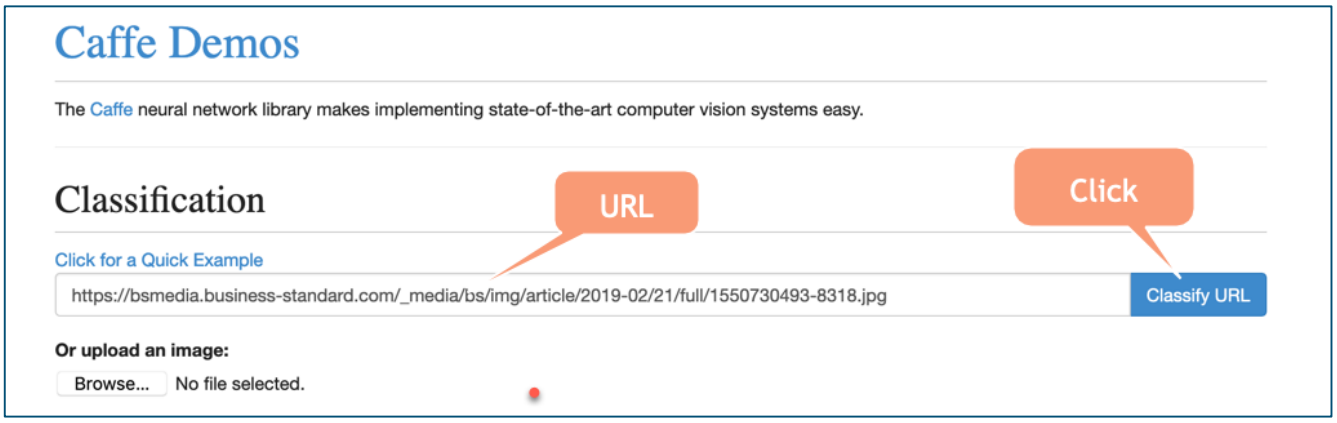
Find an image to classify

In a new browser tab, or window, open Google and search for an image of anything you wish to classify. I will search for a zebra. Click on the image then right click the larger image and select “Copy image Location”.

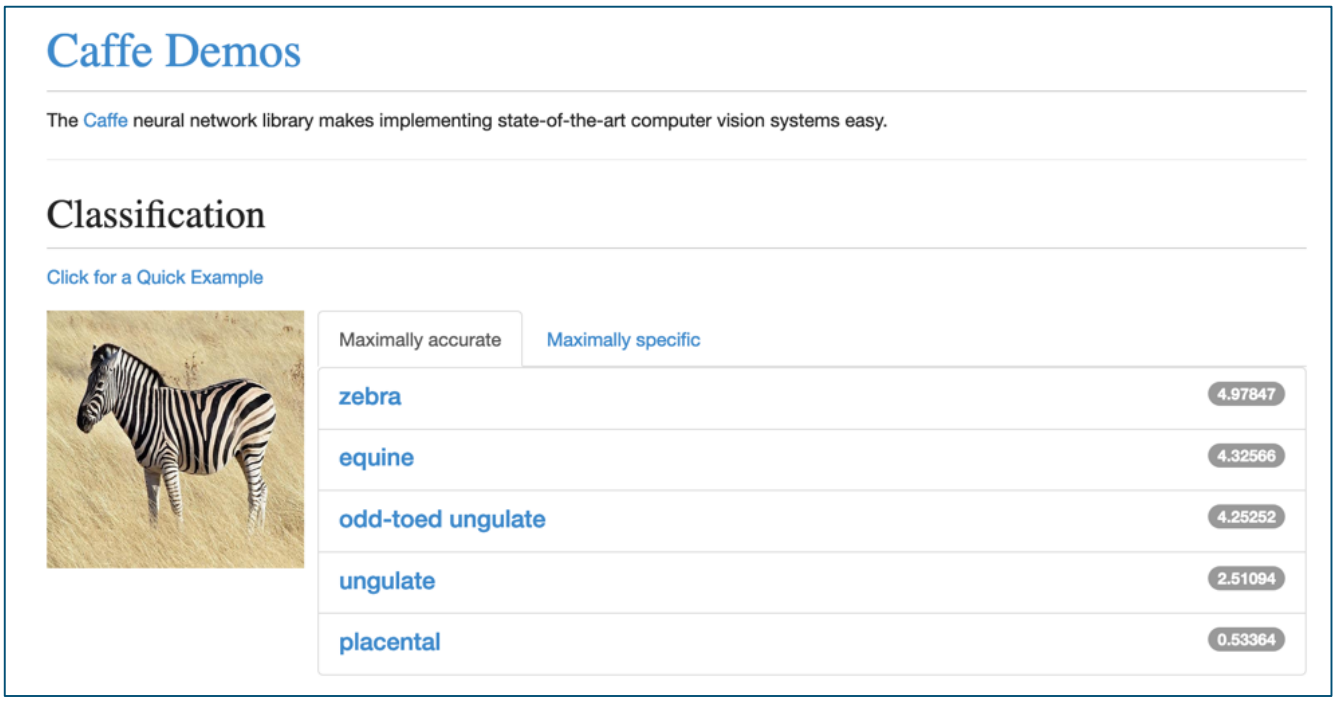


Classifying your image

Finally paste the image URL into the image classifier and press the Classify URL button:



Your results will look something like this:

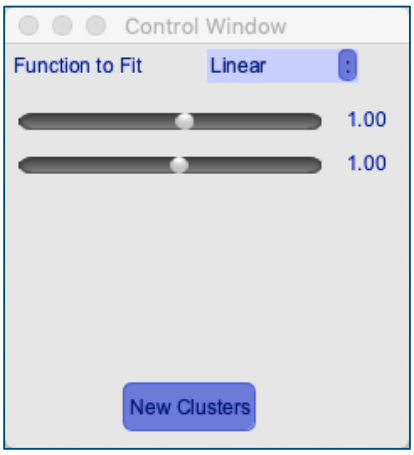


Exercise 2: Fitting functions

Open the application

On your PC navigate to the *"Fitting"* folder and double click *"Fitting.exe"* to start the program.

Two windows will open. A main display window and a control window.



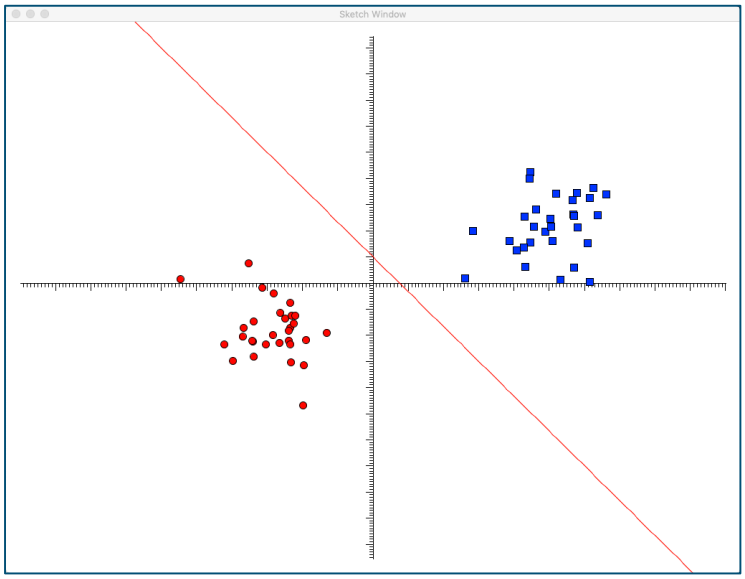
The control window allows you to select one of three functions to fit:
A Linear, quadratic and a 3rd order polynomial (or cubic).

The problem

Find the values of the function that best represent the boundary between two classes of data.

The aim is to change the values of the parameters of the function, using the sliders, such that the function represents the best possible boundary between the two classes (red and blue).

For the linear problem the solution may look as follows:



Once you have found a set of parameters that work then press the new clusters button a few times and see if your function still fits the data well.

Questions

Question: *What is happening when you press the "New clusters" button?*

Question: *How might you decide which function to use for data you have never seen before?*

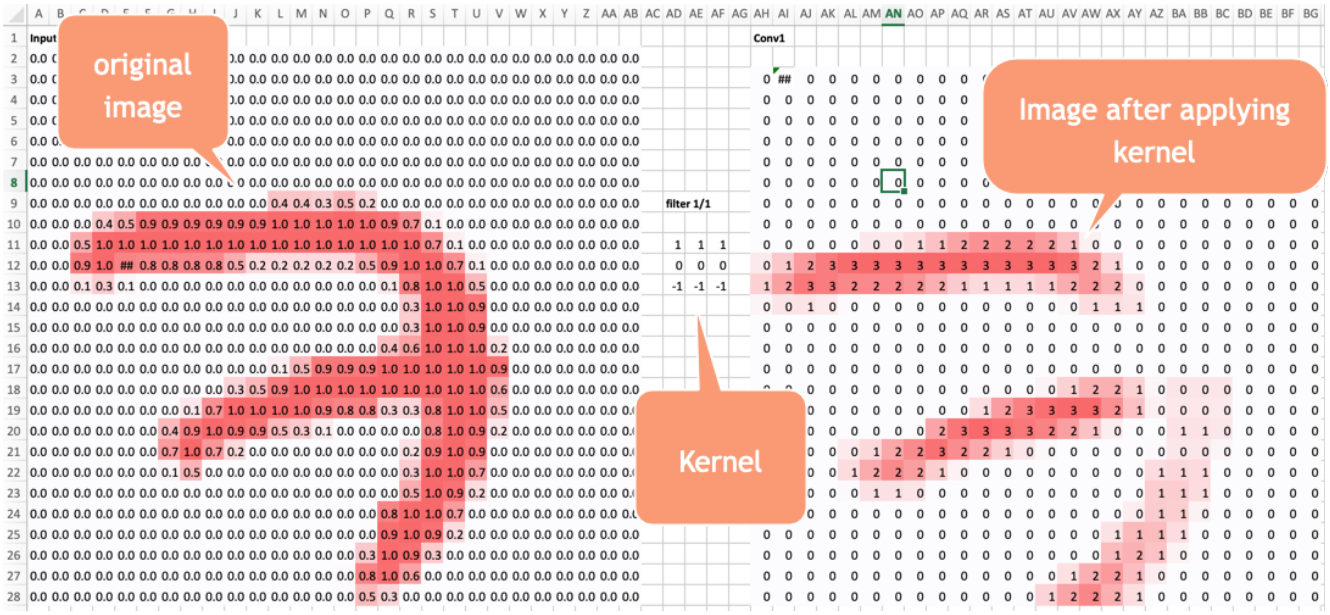
Question: *How might you be able to prove that the values you have chosen are the best possible values?*

Exercise 3: Convolutional Layers

Now that we understand what convolutional layers in a neural network are for we will now spend some time examining how they work.

Convolutional layers with Excel

On your local PC open Convolution.xlsx using Excel.



On the left of the spreadsheet is an example of an image from the MNIST database of hand drawn digits (a 7). In the center is the convolutional kernel to be applied and on the right is the result of applying the image.

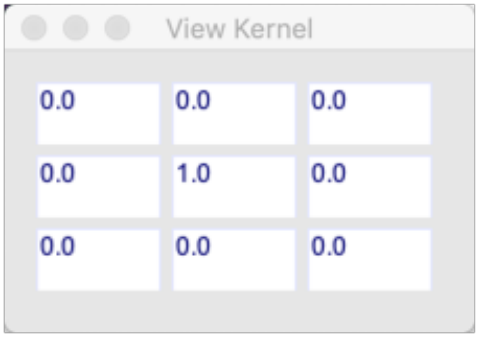
Question: *How would you describe what has happened to the image?*

Applying convolutional kernels to real images

On your local PC open the “Conv” folder and double click the “conv.exe” program to start it.

A file dialog will open. Click cancel to load the default image as shown. As in the

filtering example you have two windows the image and the control window which displays the kernel.



As you move the mouse over the image the kernel will be applied to the image. (Trust me it is working).

Examining kernels

The Kernels.xlsx file contains a number of “known” kernels can you describe what each of the kernels does to the image?

0	0	0
0	1	0
0	0	0

0.0625	0.125	0.0625
0.125	0.25	0.125
0.0625	0.125	0.0625

-1	-2	-1
0	0	0
1	2	1

-2	-1	0
-1	1	1
0	1	2

1	0	-1
2	0	-2
1	0	-1

-1	-1	-1
-1	8	-1
-1	-1	-1

-1	0	1
-2	0	2
-1	0	1

0	-1	0
-1	5	-1
0	-1	0

Note: *You can load a new image into the main window by clicking the main window and then pressing the 'L' key on the keyboard. There are four images in the "Conv/images/" directory and using these might help you to see what the kernel is doing.*

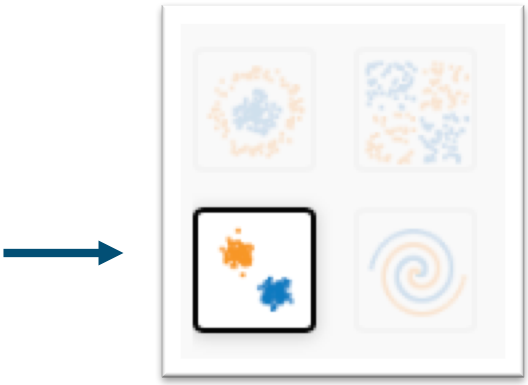
Exercise 4: Neural Networks

In this worksheet we will make use of the Tensorflow playground developed by Google. The playground demonstrates Google's Tensorflow library which is used widely for the development of neural networks.

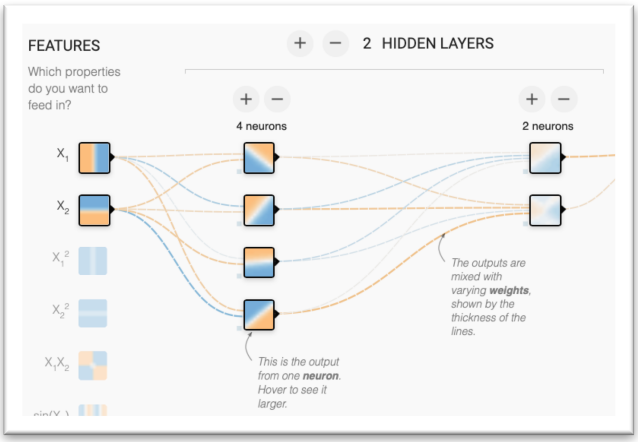
Task 1: Getting Started

Open the playground by visiting the website:
https://playground.tensorflow.org/

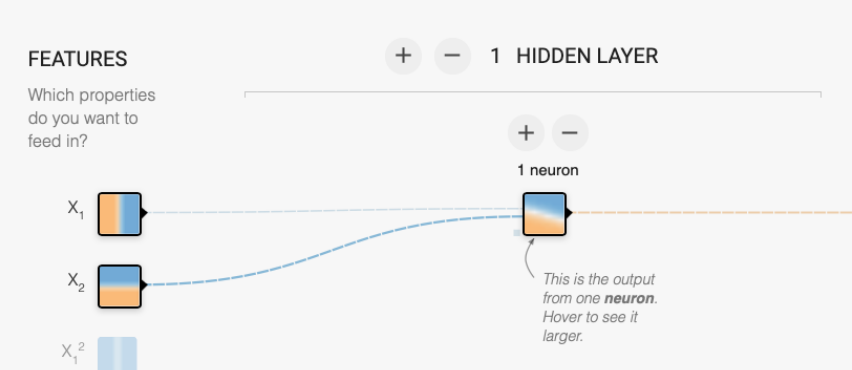
We are going to start by ***choosing a dataset*** to work with. On the left had side of the screen There are four datasets to choose from. We will start by selecting the one on the bottom left. This is two well separated classes and is similar to the XOR data set.



Now we are going to ***modify the structure*** of the neural network.



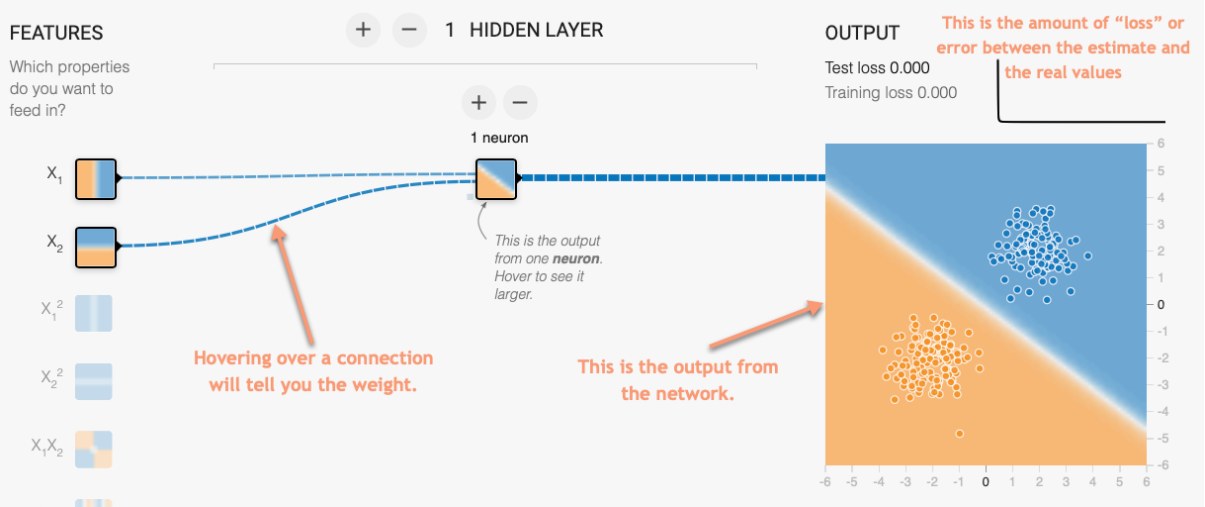
Using the **+** and **–** button change your network until it looks like the image below.



Next we will ***train the network*** using the play button.

When you are happy that the network has "stopped" training. i.e. the result is not improving press the play button again to stop.

The final step is to ***examine your network***

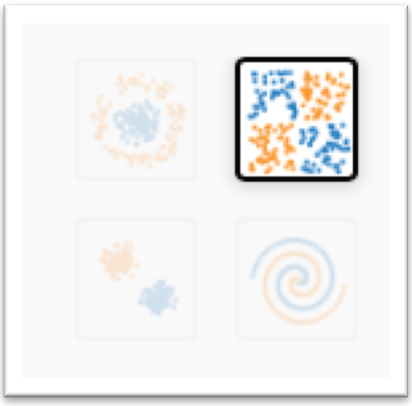


What we have learnt here is a simple linear function and we in fact we don't even need a hidden layer!

Question: Remove the hidden layer (by clicking the – button) and train again. Is the result any worse?

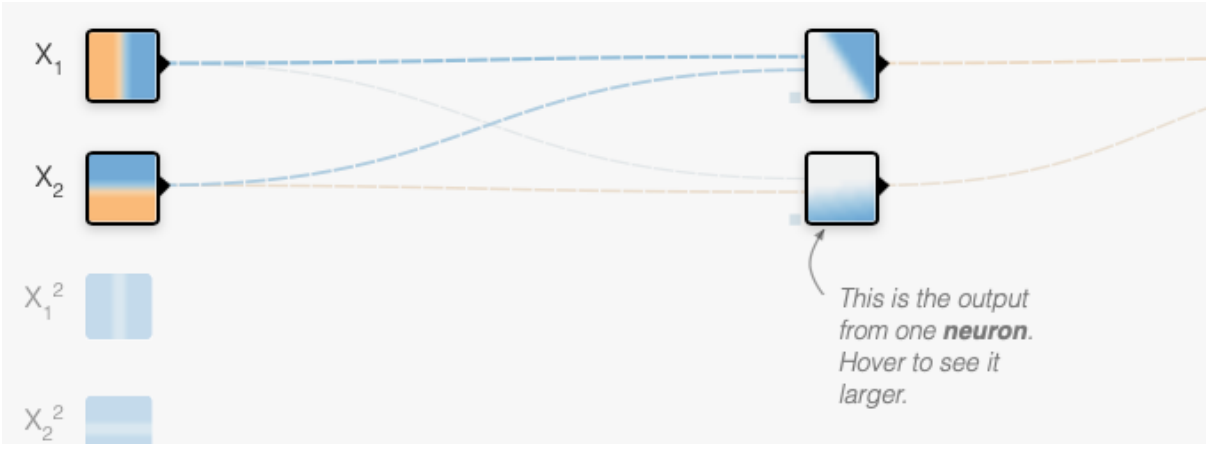
Task 2: Modelling non-linearities.

Let's start by choosing a new data set.



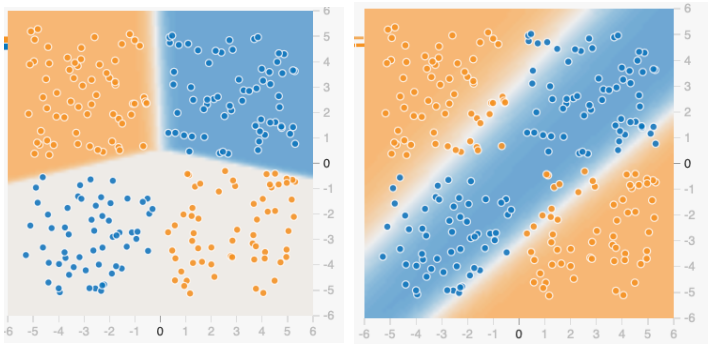
This data set cannot be separated using a single straight line. It requires a more complex, "non-linear", classifier.

Set your network to have the structure shown below (i.e. one hidden layer with two neurons):



Choose the ***ReLu activation function*** set the ***Learning Rate to 0.03*** and train the model (using the play button). Stop training when you reach ***1000 epochs*** .

Question: *What does your output look like?*



Question: *Both of these were generated from the same dataset with the same network structure. Why might this be?*

A network with one hidden layer of two neurons is not capable of learning this dataset. We need a more complex network.

Question: *How many hidden layers and neurons do we need to give us a Test loss of less than 0.01?*

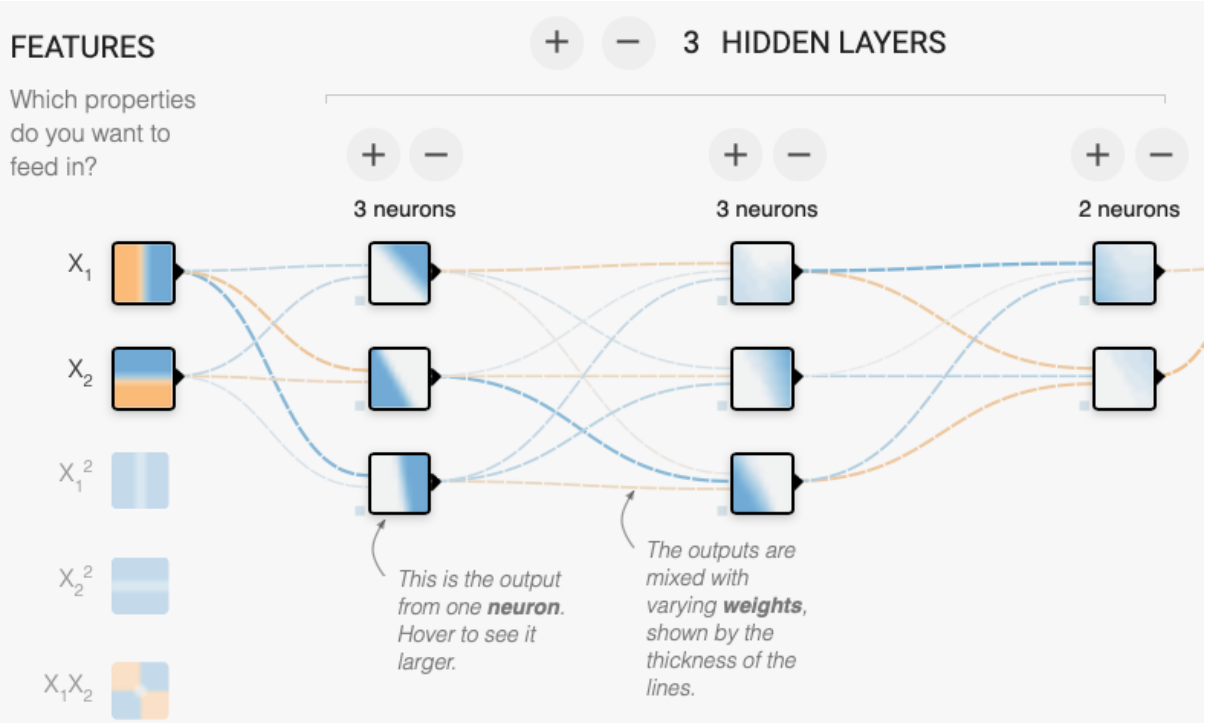
Question: *Does it make a difference which activation function you use?*

Question: *how does the learning rate affect the training?*

Task 3: Noise

So far we have only considered data sets where there is no noise. But in reality noise is almost always a concern. In this exercise we are going to continue looking at the data set used in the previously.

Start by constructing a neural network with the structure shown below.



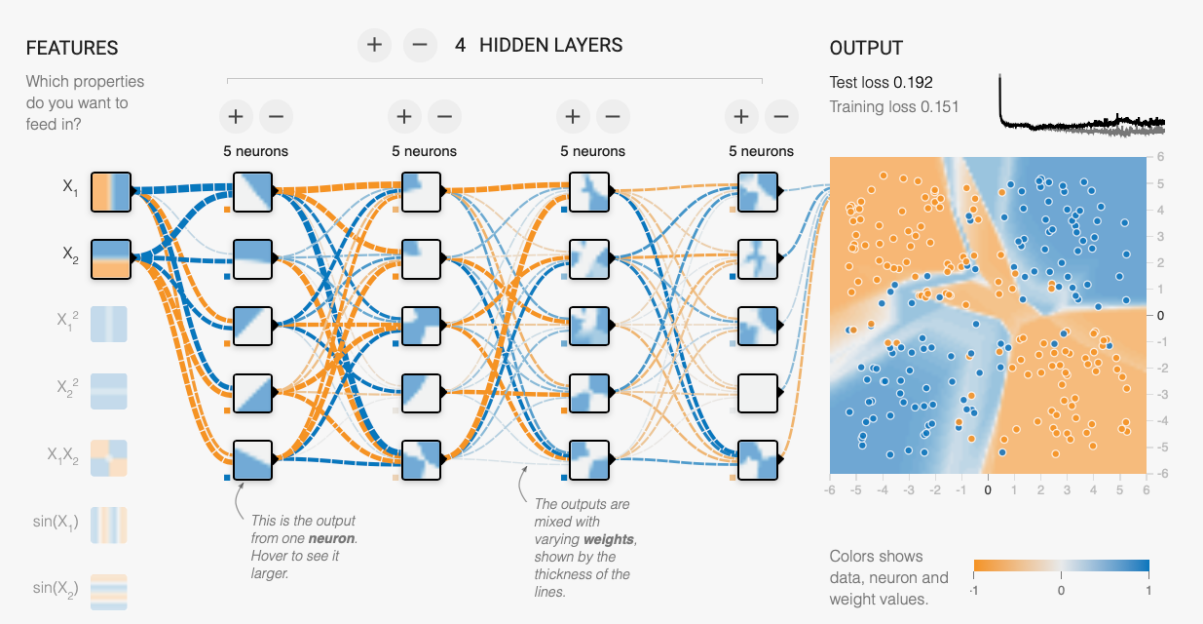
Train the network for the ReLU activation function with a Learning rate of 0.03.

Question: Can you get a test loss of zero with this network?

Introduce noise into the data set by changing the slider on the left had side of the screen.

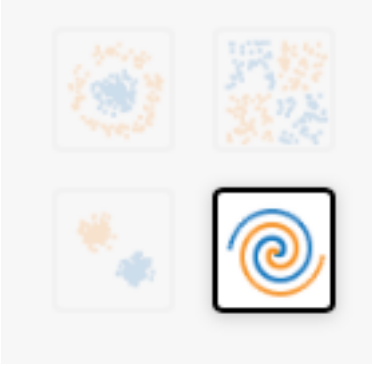
Question: Try resetting the system and re-training a few times … does the network ever fail to fit the data?

A network with 4 hidden layers with 5 neurons per layer was constructed and trained for 1000 epochs. The data set was the XOR set with Noise set to 30. ***Comment on the results*** shown below:



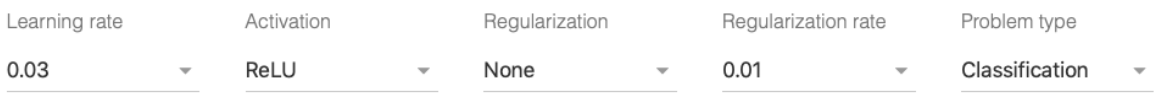
Task 4: Complex data sets

Start by selecting the Spiral data set. This data set is highly non-linear and is not easy to learn.



Start with a neural network with 2 hidden layers and 3 neurons per layer. Set the noise to zero.

Set the training parameters as shown below:



Run the training for 1000 epochs.
Question: How well does you model fit the data? What was your test loss?
Now change the structure and learning parameters to get a better model., feel free to change any of the options on screen and investigate their impact of training times and the final fit.

Question: *What is the best fit you can achieve?*

If you are interested in hearing more about the spiral problem and its solution there is a good video from Google showing how this can be approached.

https://www.youtube.com/watch?v=sxHjS1PbO8M