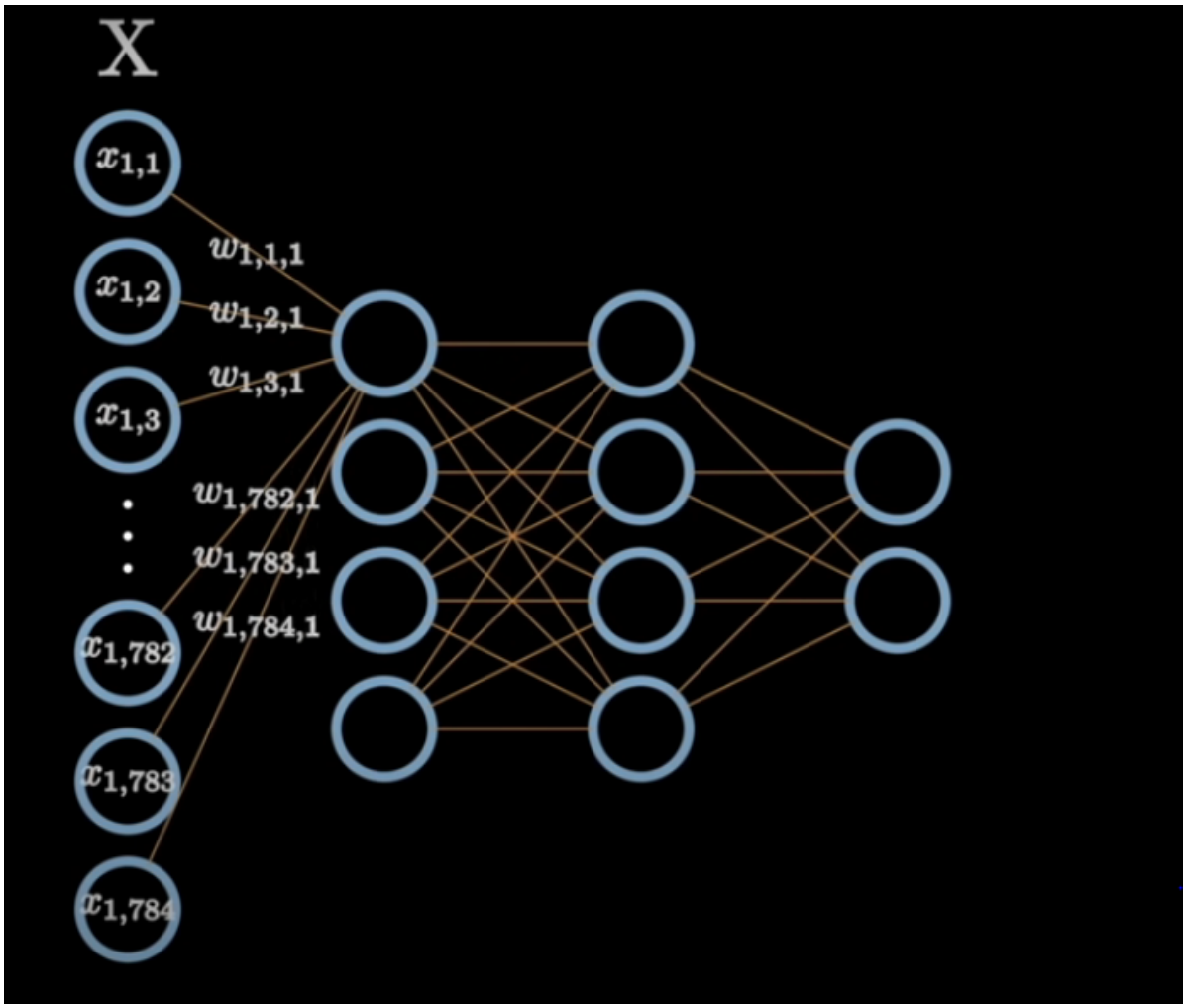
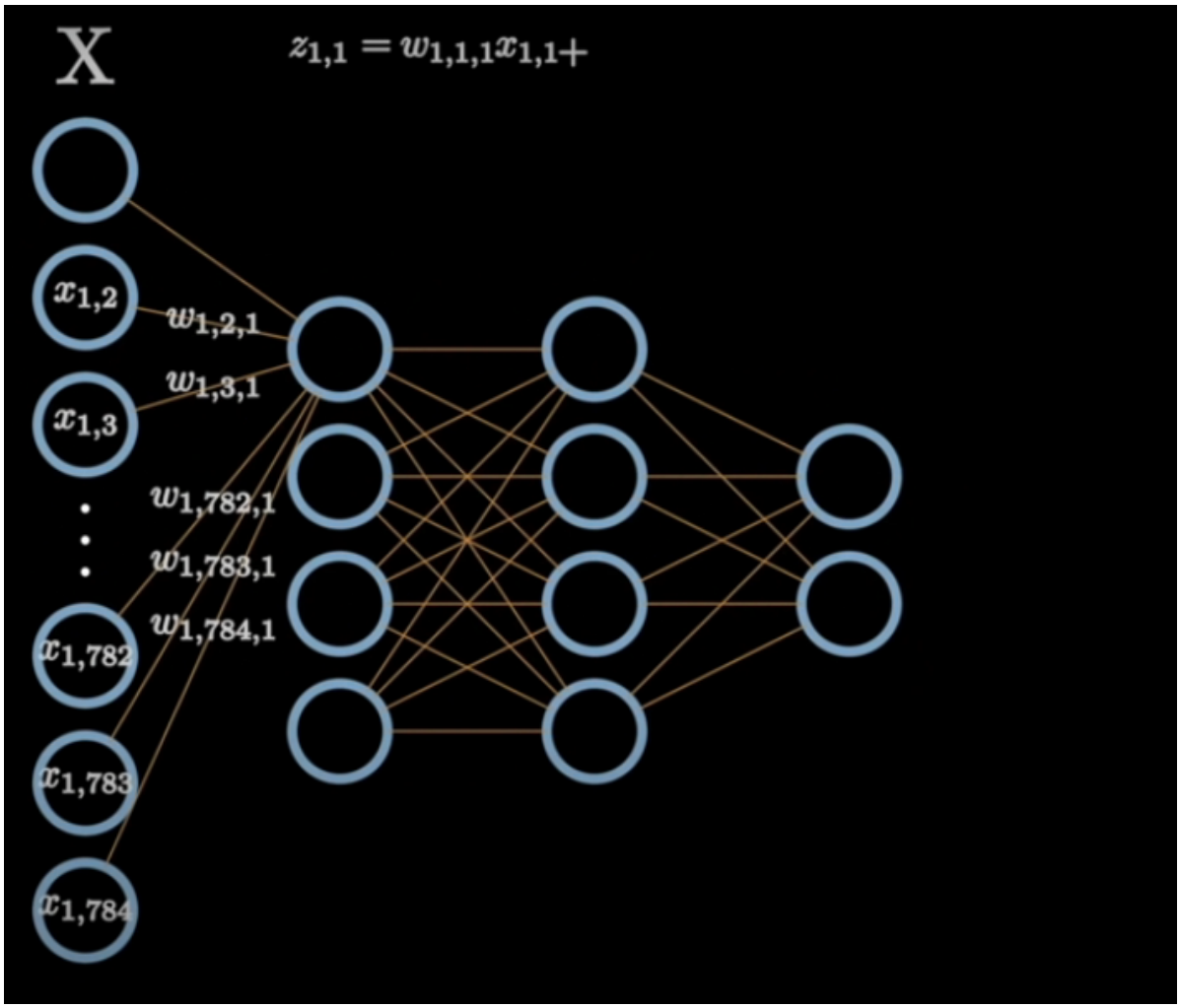
Sentdex Python Deep Learning

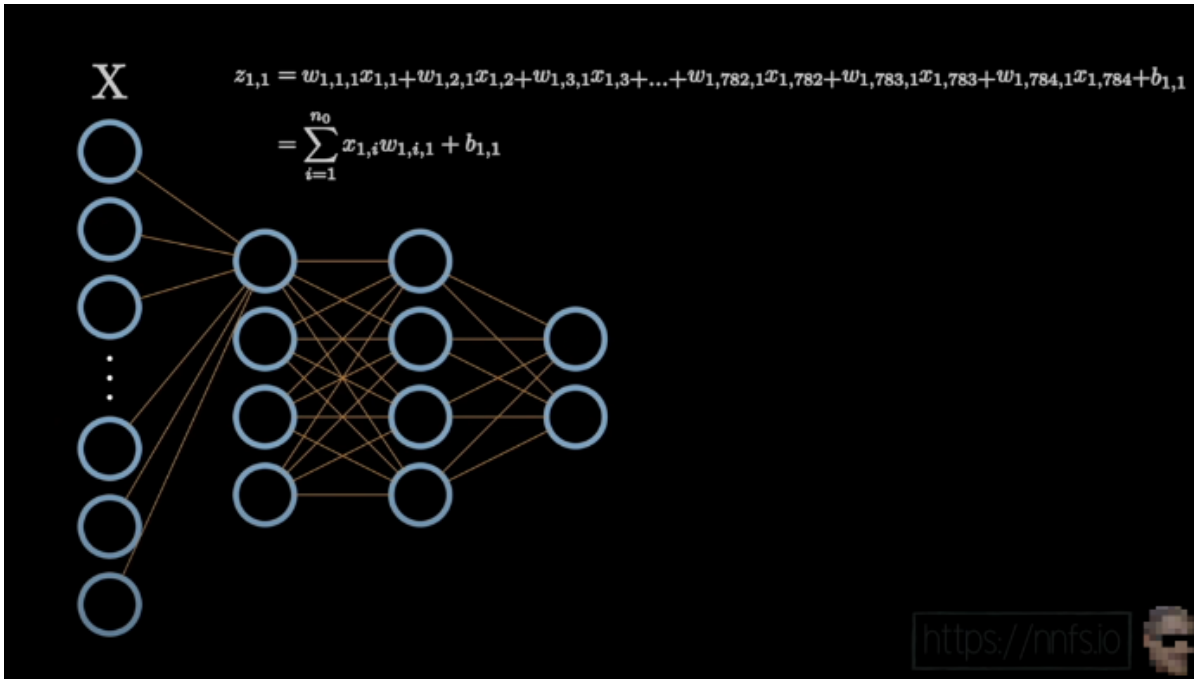
Quick Recap

▼ How does a basic Back propagation network work?



1.First we have all the weight and input values present

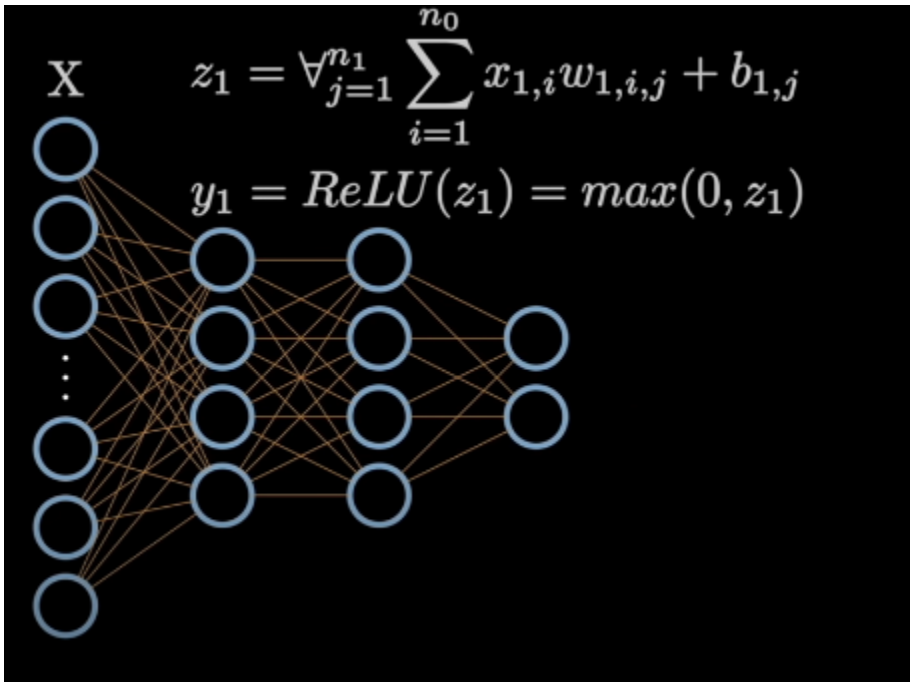

X
$z_{1,1} = w_{1,1,1}x_{1,1}+$
$x_{1,2}$
$x_{1,3}$
$x_{1,782}$
$x_{1,783}$
$x_{1,784}$
$w_{1,2,1}$
$w_{1,3,1}$
$w_{1,782,1}$
$w_{1,783,1}$
$w_{1,784,1}$



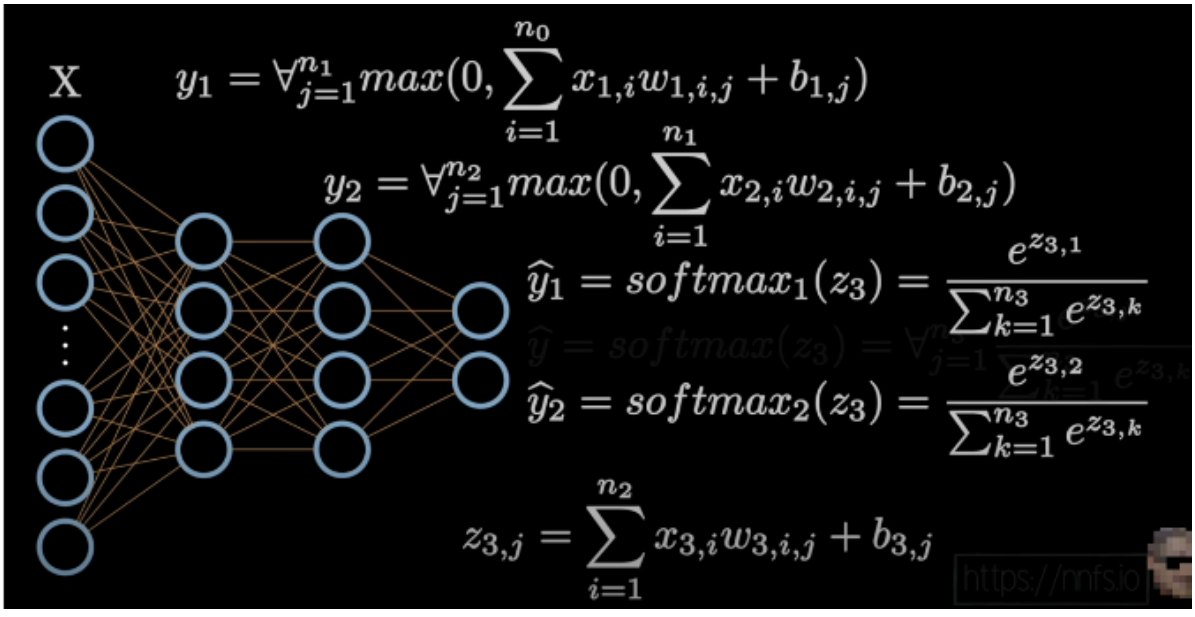
Then we calculated the weighted ouput .

$$z_1 = [\sum_{i=1}^{n_0} x_{1,i}w_{1,i,1} + b_{1,1}, \sum_{i=1}^{n_0} x_{1,i}w_{1,i,2} + b_{1,2}, \sum_{i=1}^{n_0} x_{1,i}w_{1,i,3} + b_{1,3}, \sum_{i=1}^{n_0} x_{1,i}w_{1,i,4} + b_{1,4}]$$

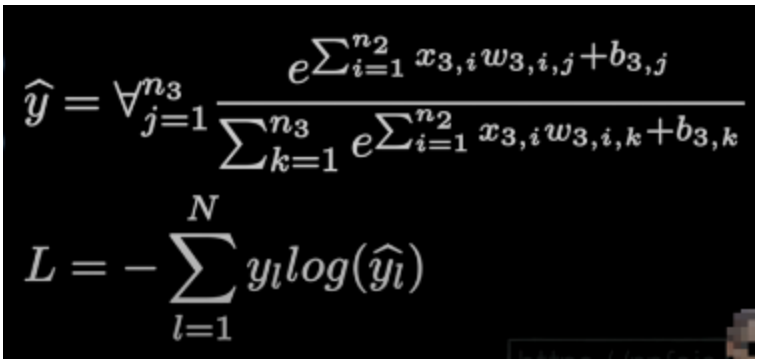
We proceed to this for all cells , not just one cell .



And then we calculate the RELU activation function .For second layer(over all) OR the first hidden layer

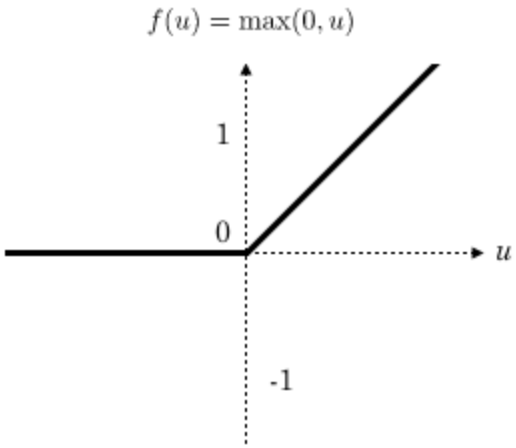


And then we perform softmax algortithm for the next layer.



And then finally we calculate loss function

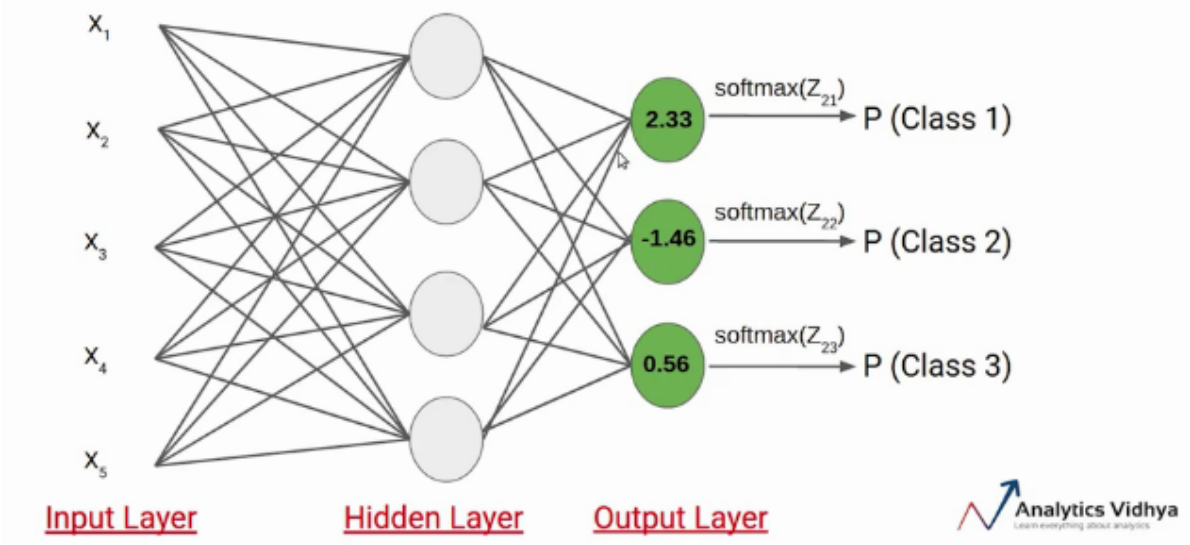
▼ What is a ReLU activatation function?



coding wise there are two ways to achieve this

```
if input > 0:
  return input
else:
  return 0
#OR
g(z) = max{0, z}
```

▼ Why do we need to find softmax activation function for each cell?



Any time we wish to represent a **probability distribution over a discrete variable with n possible values**, we may use the softmax function.

This can be seen as a generalization of the sigmoid function which was used to represent a probability distribution over a binary variable.

In other words this is similar to multiple logistic regression model .Keep in mind that For simple logistic regression we only found the outcome for binary problem.

▼ How can we get the results of the first layer? (dot product of all weights and input neurons together) ?

```
import numpy as np

inputs = [1, 2, 3, 2.5]
weights = [[0.2, 0.8, -0.5, 1.0],
           [0.5, -0.91, 0.26, -0.5],
           [-0.26, -0.27, 0.17, 0.87]]

biases = [2, 3, 0.5]

output = np.dot(weights, inputs) + biases
print(output)
```

▼ Why do we need to run batches?

We need to have batches because we can run mutltiple multiplications in parallel if we are ever dealing with GPU(which have 100 of cores) unlike cpu (which have just 4 cores)

▼ What does batch represent?

Batch represent the total data samples in a batch that will be used to train one data model of an epoch

For example , if the batch size is 5 for a given linear regression model(best fit line model) then it only takes one 5 data points to create the model.

▼ Why do we need the concept of batches, why is it not fine to use just one data sample per batch

We require multiple samples within a batch because its reduces the fidgetting or variation in our model .It will keep it consistent.

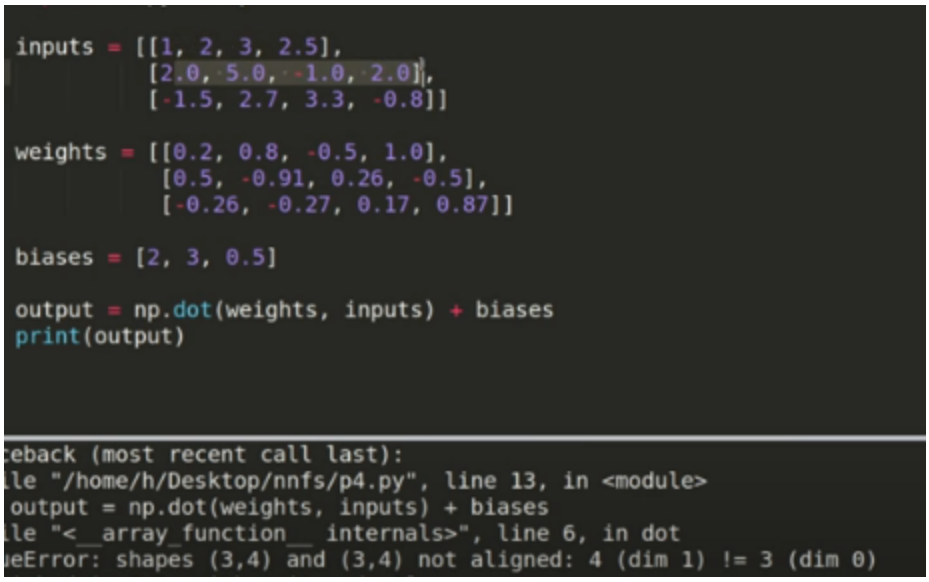
https://www.youtube.com/watch?v=TEWy9vZcxW4

Refer to this video from 5:32 to 7:32

▼ Why is bad to have too many data samples per batch as well? What is ideal batch size to have?

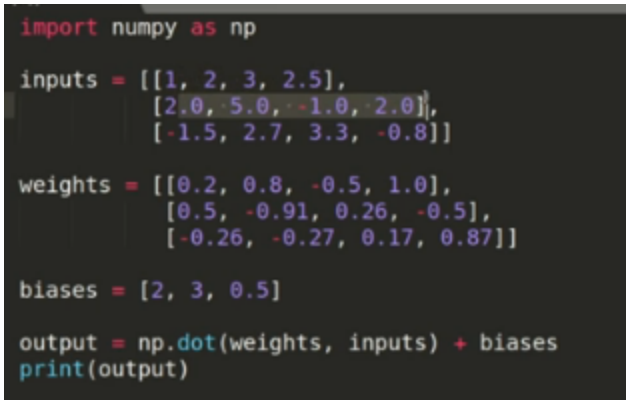
Because that would cause overfitting of the data .It depends on the dataset (How much the model is fidgetting), but in general , Its generally preferred to have 16 or 32 .That is ideal.

▼ What does this error mean in the following image and what does dim 0 and dim 1 mean in the error?

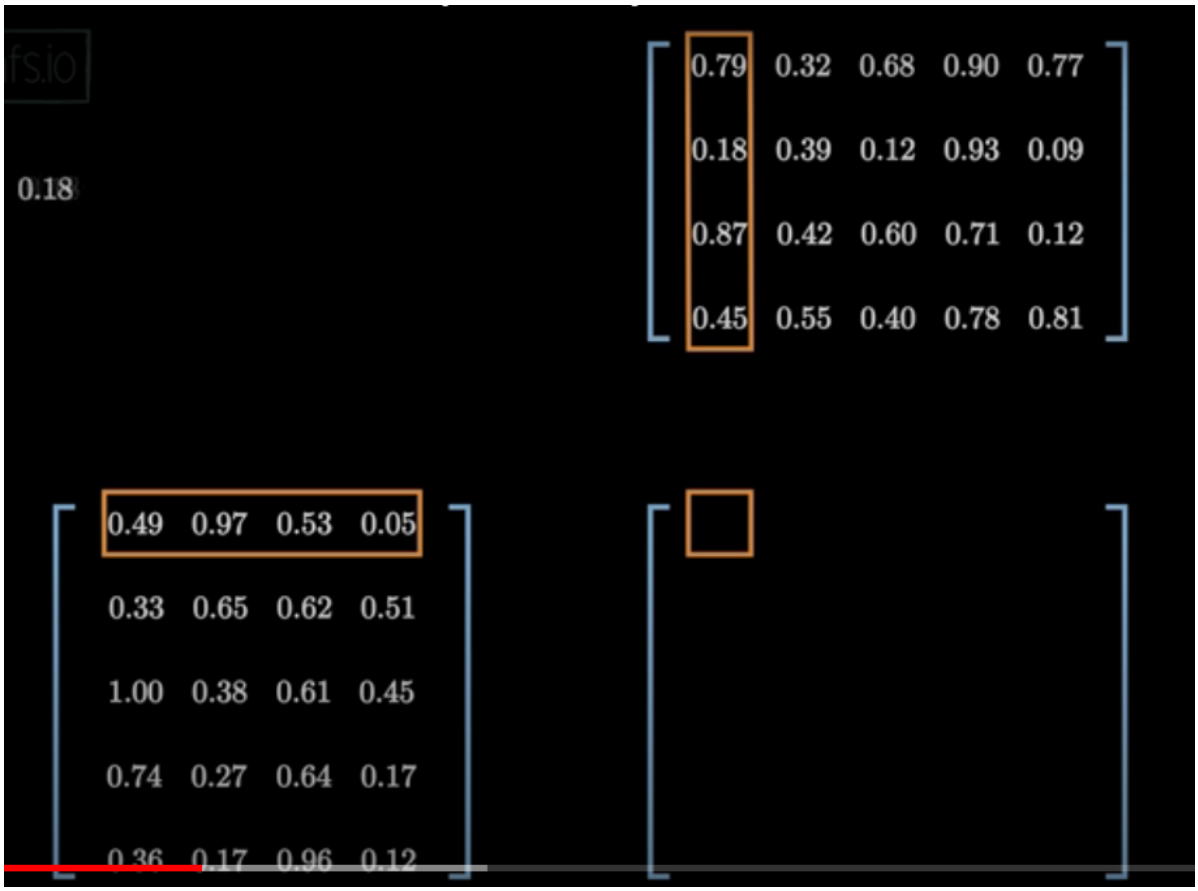


dim1 means the array within the array .So in weights the array within array has 4 elements [2.0,5.0,-1.0,2.0]

And dim0 means the first layer of the array .So in inputs we have 3 elements in the first array layer . And the error is basically telling us that in order to perform a dot product we need to have **same rows of first column and columns of second column.**



▼ In the above image what does the dot product actually do?



It performs matrix multiplication .The rows of the first matrix is multiplied with the column of the second matrix

▼ So since the matrix sizes arent matching for both matrices , how can we fix this issue?

```
biases = [2, 3, 0.5]

output = np.dot(inputs, np.array(weights).T) + biases
print(output)
```

We can transpose the second matrix and then perform dot product.

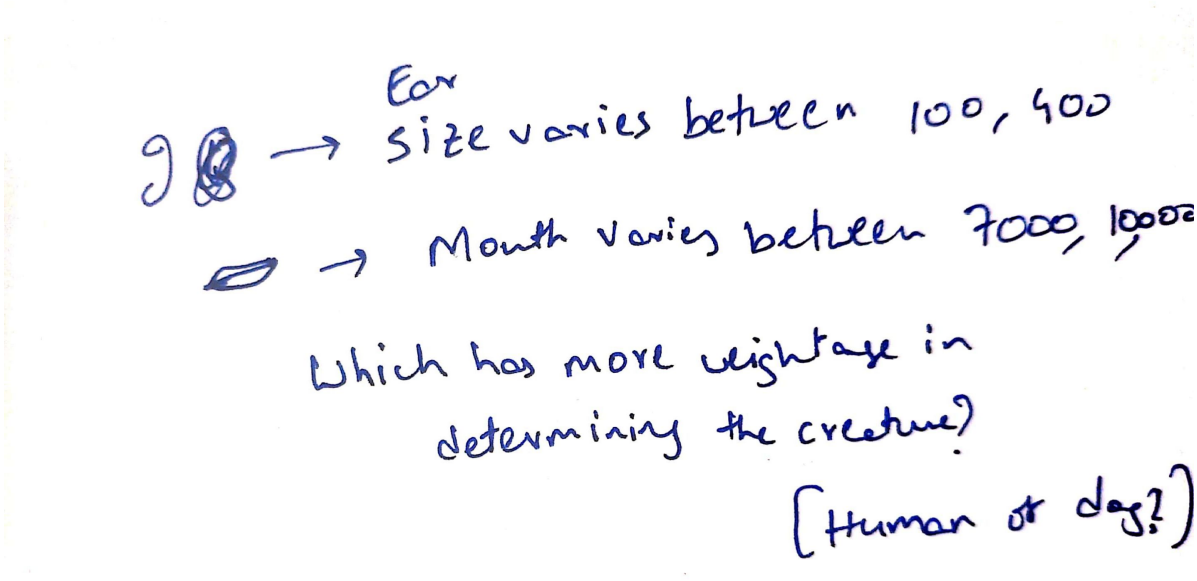
▼ What are biases connected to?

Biases are connected to each neuron of the next layer .So whenever you are mutliplying and coming up with the second layer , you are automatically adding the bias as well.(while generating the next layer) .

In this case we will have 4 biases , one for each neuron in the second layer.

▼ Why is it important to scale the input data in between 0 to 1?

We scale the data to make sure that all parameters are standardized .It also makes sure that all parameters are given equal weightage.



For example, let us take a problem statement .There are two input parameters in the input data set .We are taking the size of the ear and size of the mouth of a creature and based on that we are trying to determine whether its a human or a dog .

If we dont standardize the values for Ear and Mouth we will not be able to accurately estimate which has more wieghtage.

Hence its important to standardize

▼ Why is it important to have weights lying between -1 and 1?

We need to have weights between -1 to 1 because otherwise , let us say we have input = 5 ,and weight = 100

The output would be

5*100 = 500 .

The weights following that will keep increasing the total output of each layer .

And before you know it , the **output number is a super huge number**.

This is not desirable whatsoever .Hence its preferred to have -1 to 1 for weights.

▼ For the intial neural network why is it important to have non-zero values as part of the network?

It is important because , otherwise if you propagate through the entire network you will have 0 (Because both the weights and the biases all of them are 0). So the neural net needs to have **some** non zero values and then correct itself towards the correct model .

▼ What is the equivalent of batch size in a neural network ?[Not confirmed answer]

Its nothing but the input layer . The size of it determines the size of each batch

▼ What are the preferred initial values for weights and biases?

For weights its preferred to have -0.1 to 0.1

And for biases , we can just set it to 0

▼ After complete model training , what is the preferred loss that we are expecting

Less than 2 is preferred

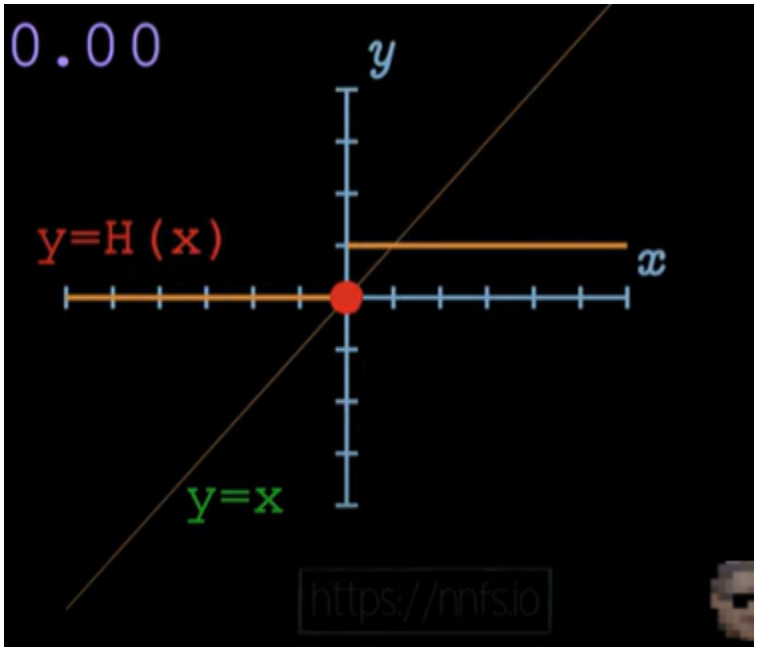
▼ When you are using this as an input , what does the array signify?

```
X = [[1, 2, 3, 2.5],
     [2.0, 5.0, -1.0, 2.0],
     [-1.5, 2.7, 3.3, -0.8]]
```

It signifies that there is a batch of 3 inputs and each input has 4 features .

Activation functions

▼ What is a step function?



Less than 0 its 0

More than 0 its 1

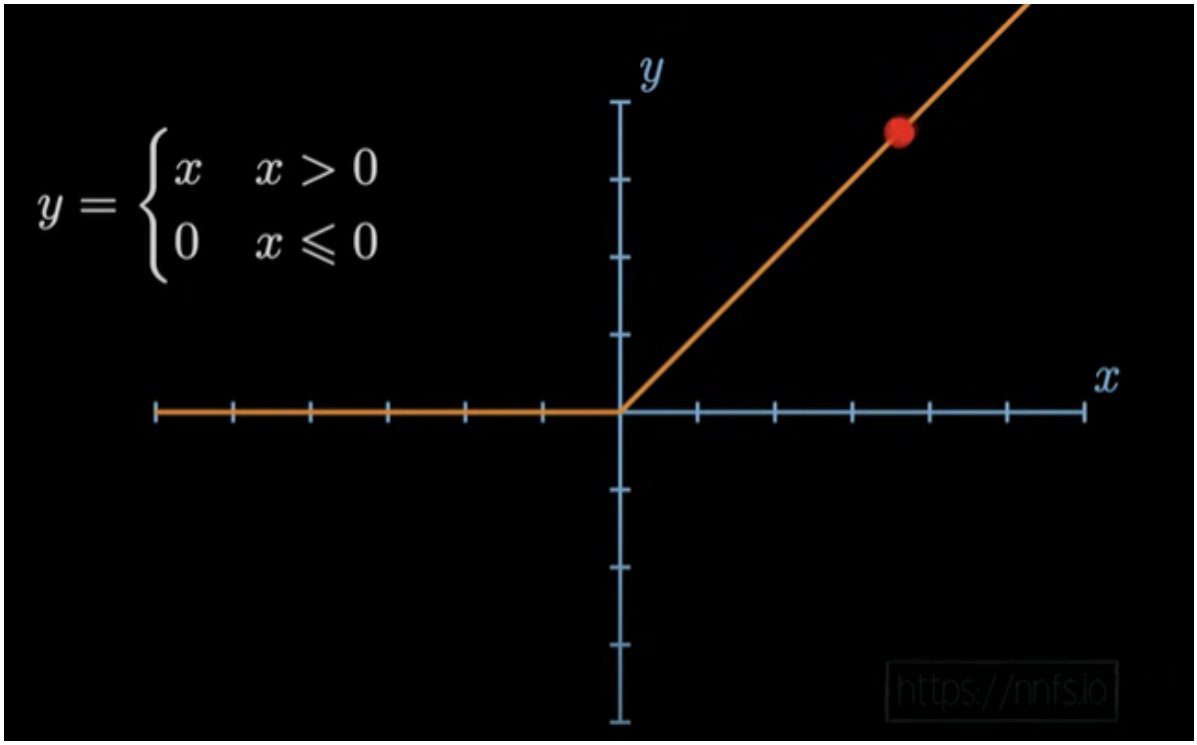
▼ Why is sigmoid function preferred

ITS Granular:

Its preferred because we dont get just a simple 0 or 1 , we get a value in between 0 and 1 .And we can change the threshold as to which is considered activated and

which is NOT activated

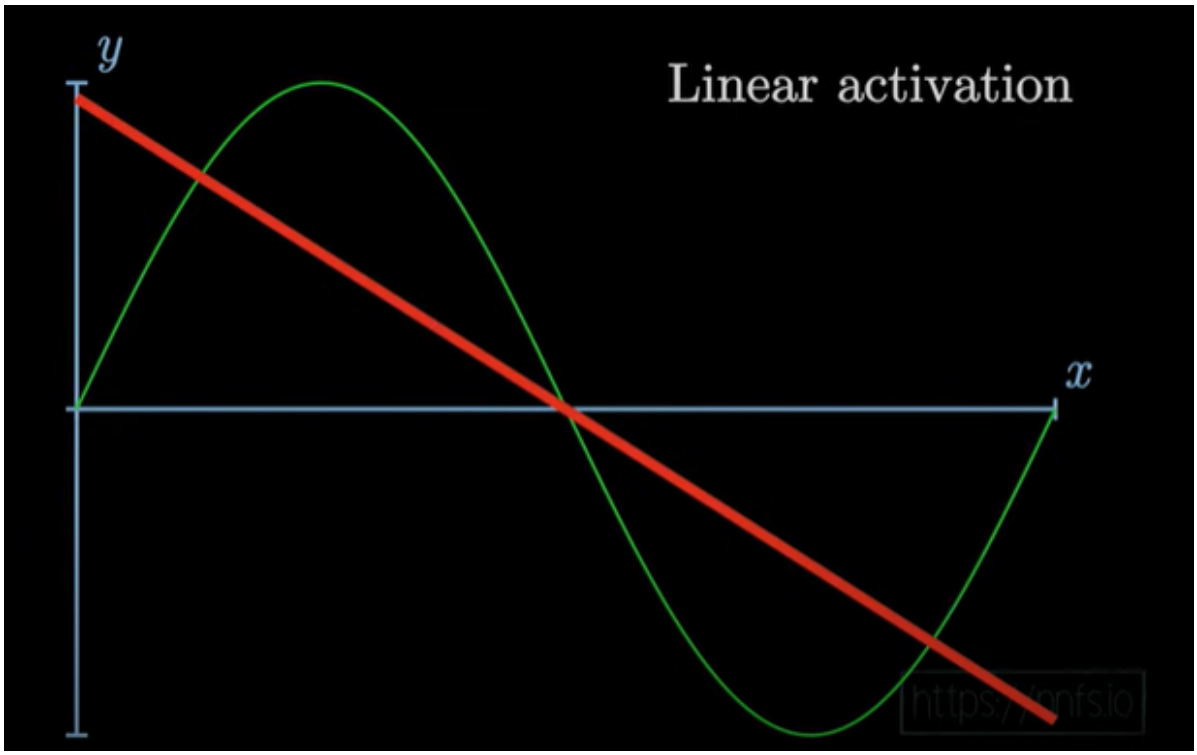
▼ What is a ReLU function?



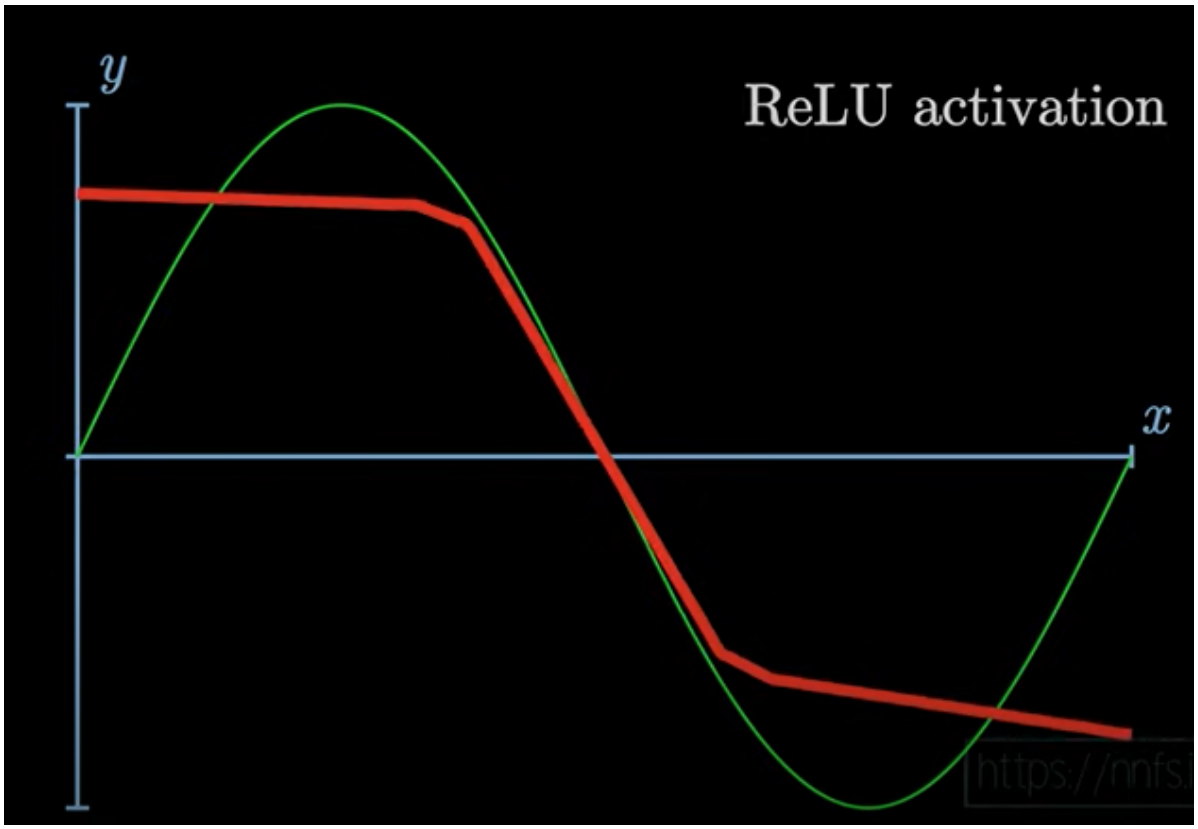
▼ Why would we want to use Activation functions at all? What advantage do we get from this?

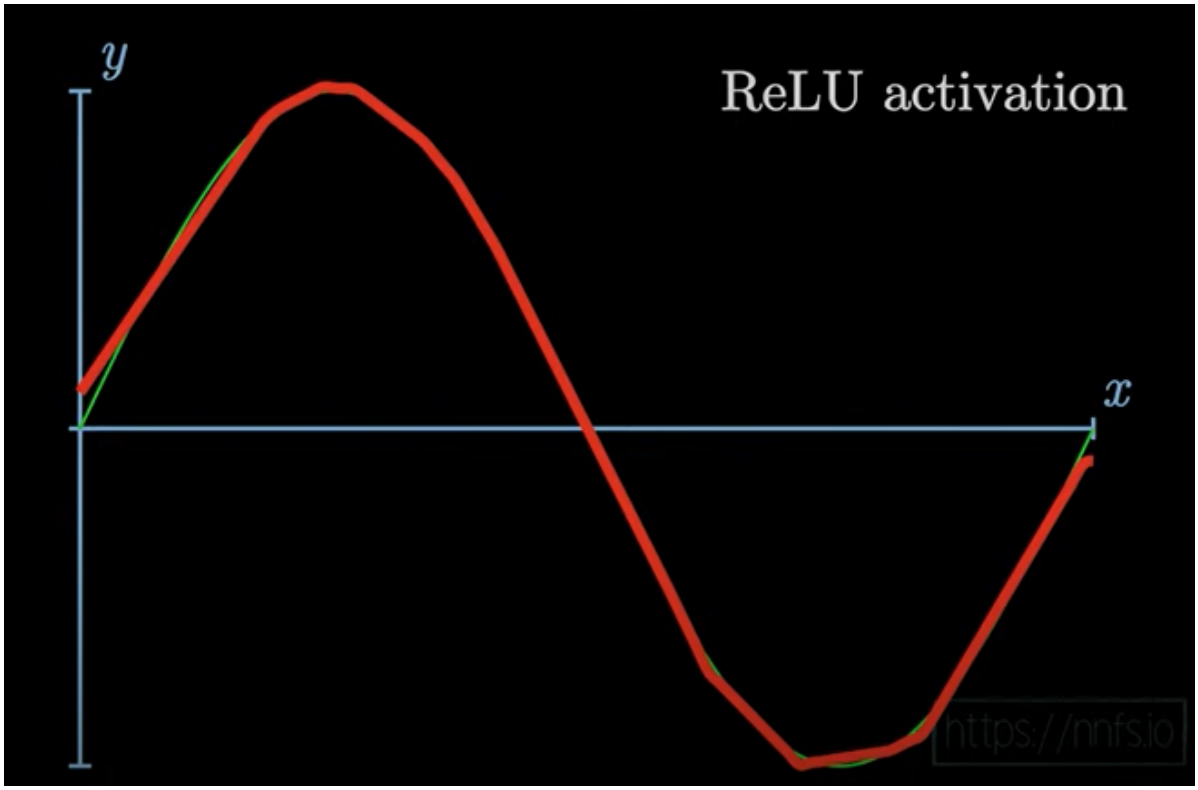
Let us take a sine wave for example , we create a neural net that can mimick the sine wave .

If we use just simple linear functions through the neural net , then this is the result that we are going to get .



As we can see there is not much we can map. But lets take an activation function LIKE ReLU activation function .Lets see how this will give us results

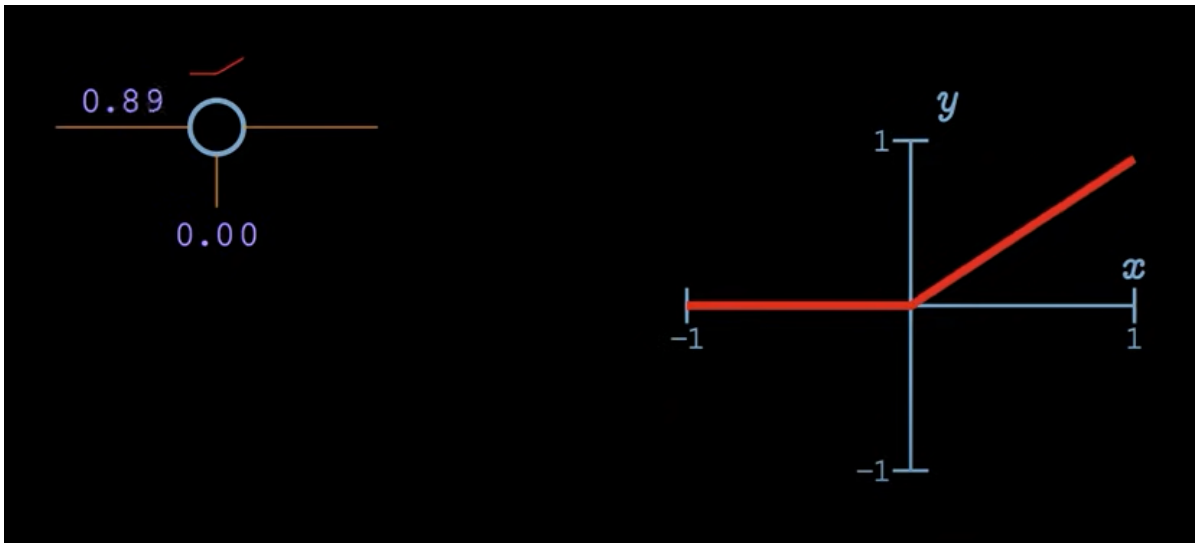

y
ReLU activation
x


y
ReLU activation
x
https://nnfs.io

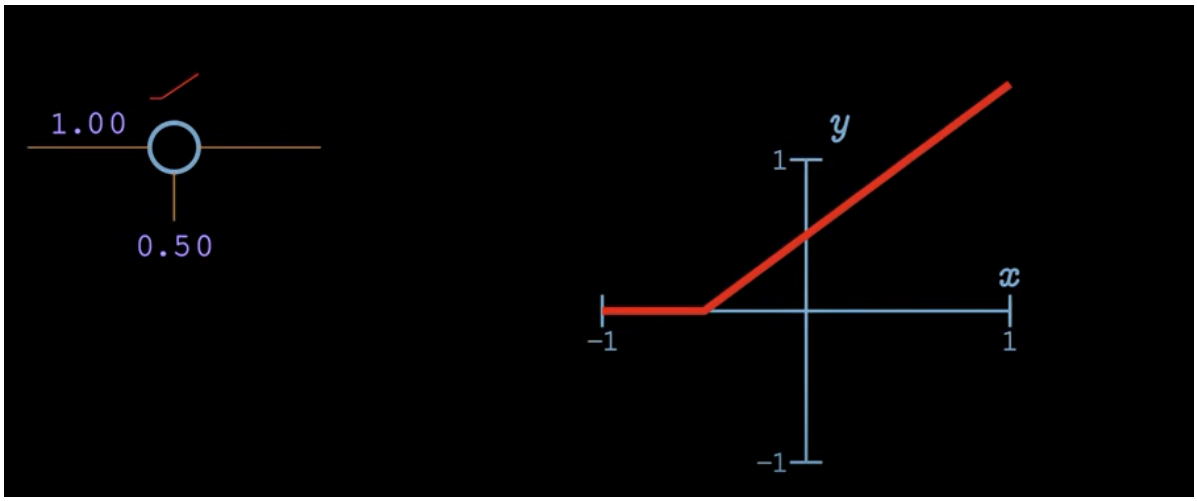
As we can see after a few iterations , we are getting this as the result. And while it may not be perfectly mimicking , this sure as hell still a huge improvement from linear functions.

▼ Why does ReLU activation work? How is it able to mimick while linear function isnt able to.WHATS SO SPECIAL ABOUT RELU that its just WORKS?!

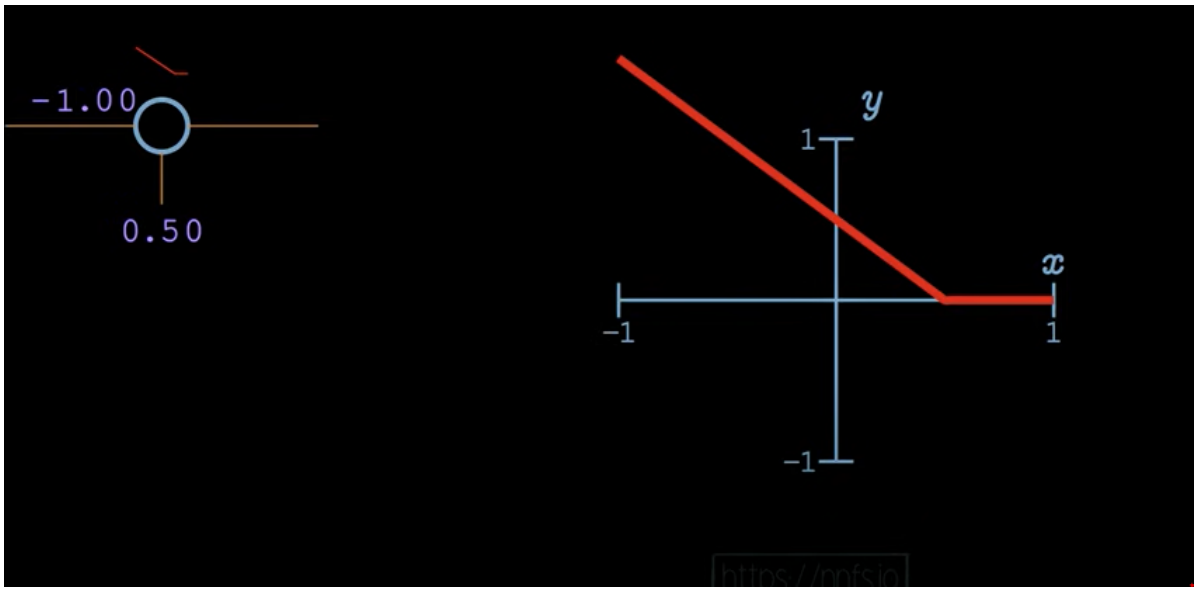
This is the result wihth 1 neuron



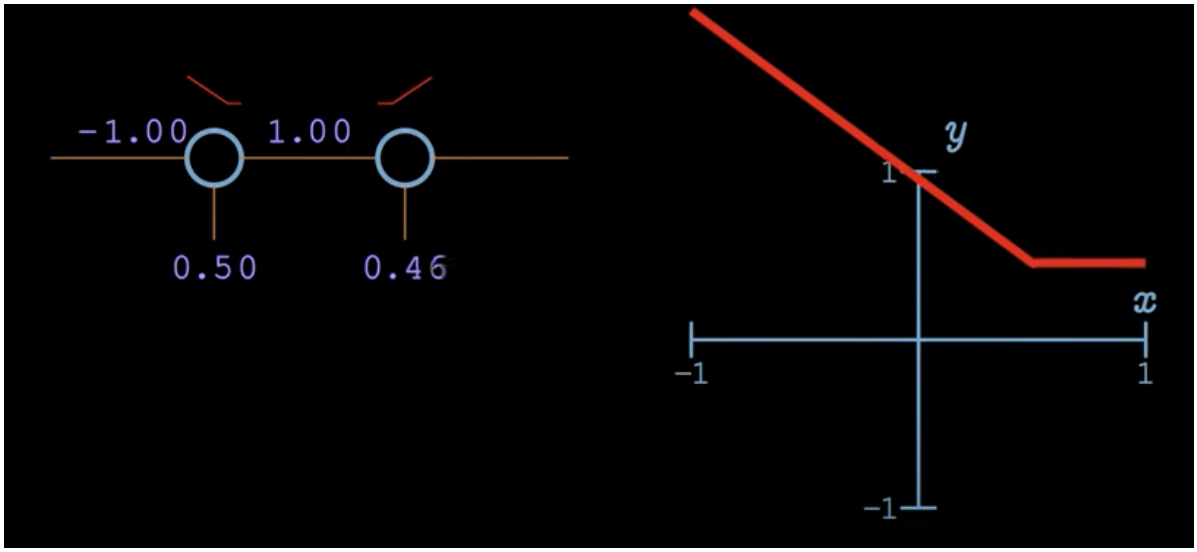
This is the result with 1 neuron and a bias≠0



This is the result with negative input weight

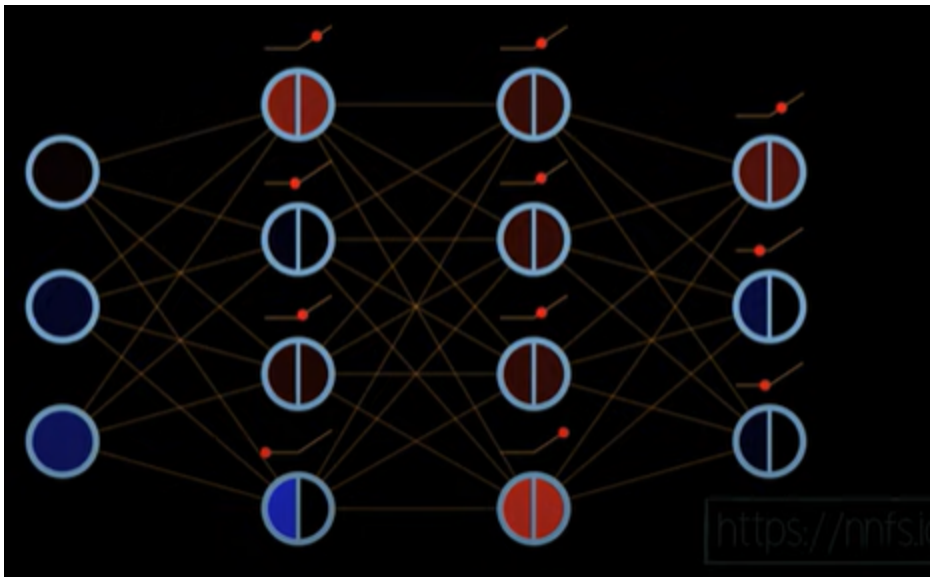


This is the result with 2 neurons



So as you can see , you have wide range of options , its almost as if you can draw a ploygon if you could .Its as if you are bending a methal rod to your will . There is no limit to what shape you would want.

▼ What is a standard template of a generic neural network modele?



As we can see here.The first layer is just your standard input layer with an array of features

Then ., starting from the second layer , its all sigmoid functions one after another. All the way till the ouput