Neural Networks

ATML Track1

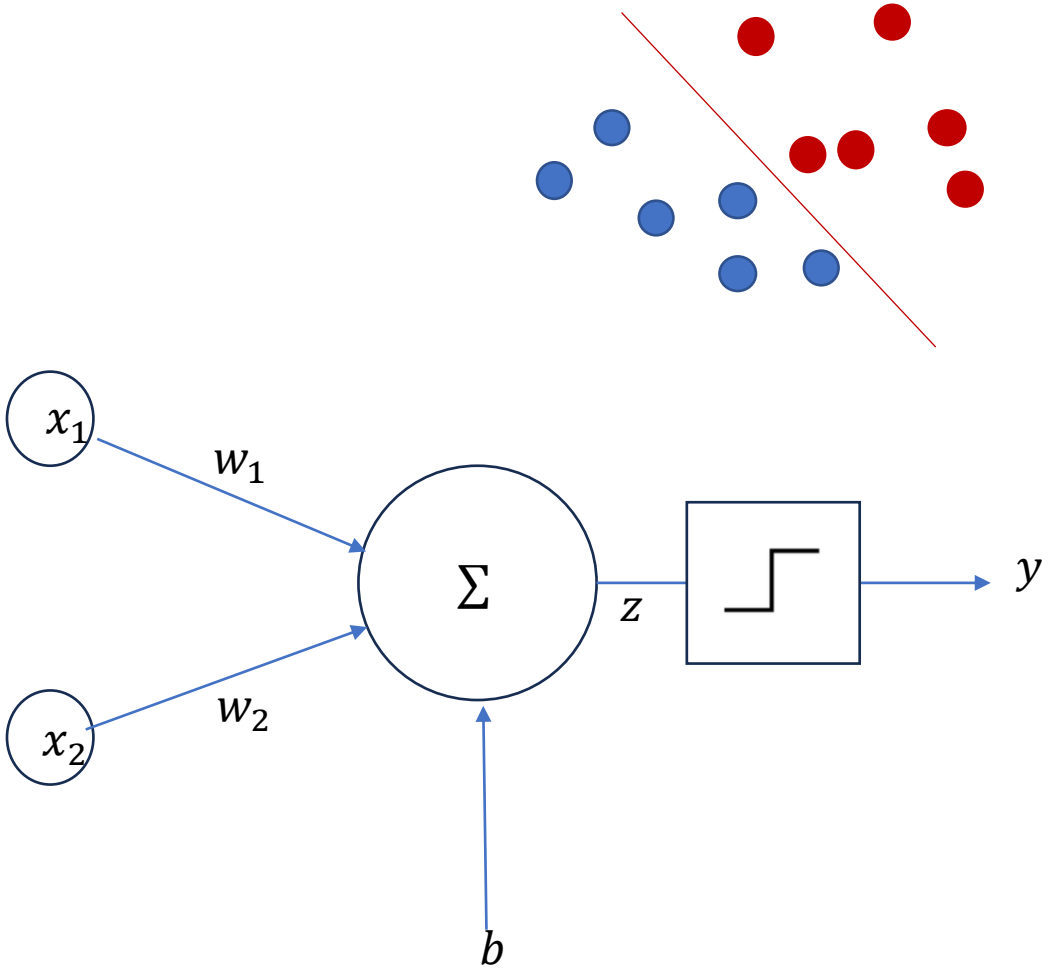
Rik Sarkar

Today

- Neural networks
- Logic with neural networks
- Activation functions
- Classifiers and cross entropy loss
- Overfitting

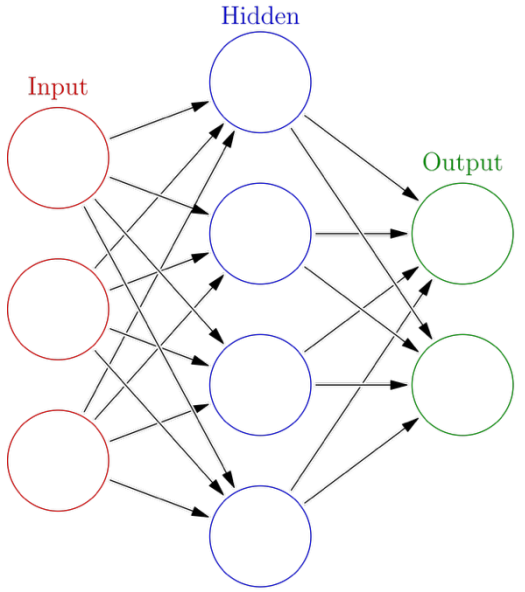
Single neuron

- Perceptron with threshold activation
  - $w_1, w_2, b \in \mathbb{R}$
- $y = (w_1 x_1 + w_2 x_2 + b \geq 0)$
  - Truth value 0/1
- We often write $z = Wx$

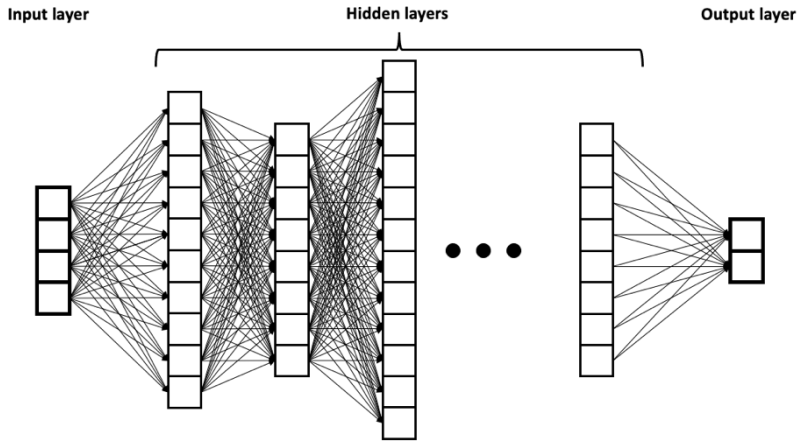


Neural network layers

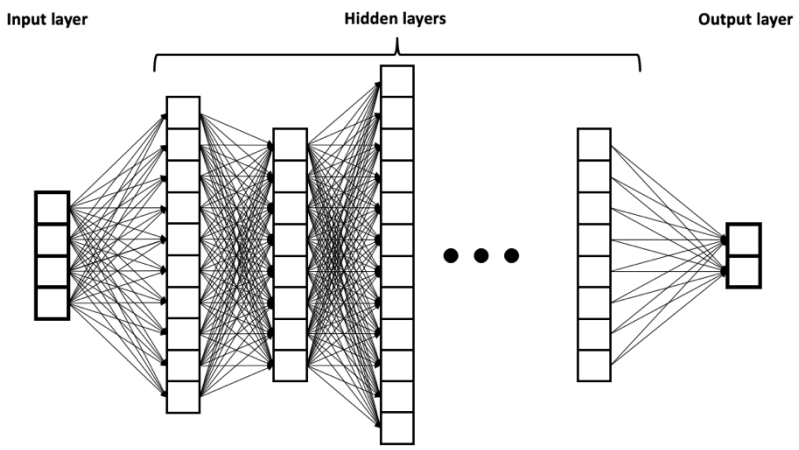
- Input, output and hidden layers
  - Input layer: just the individual input variables
  - Hidden and output layers: Activation functions



- Deep Neural networks
  - Multiple hidden layers
  - Output of a layer is the input to the next layer
  - So we can get more complicated functions as output



Multilayer perceptron



Usually

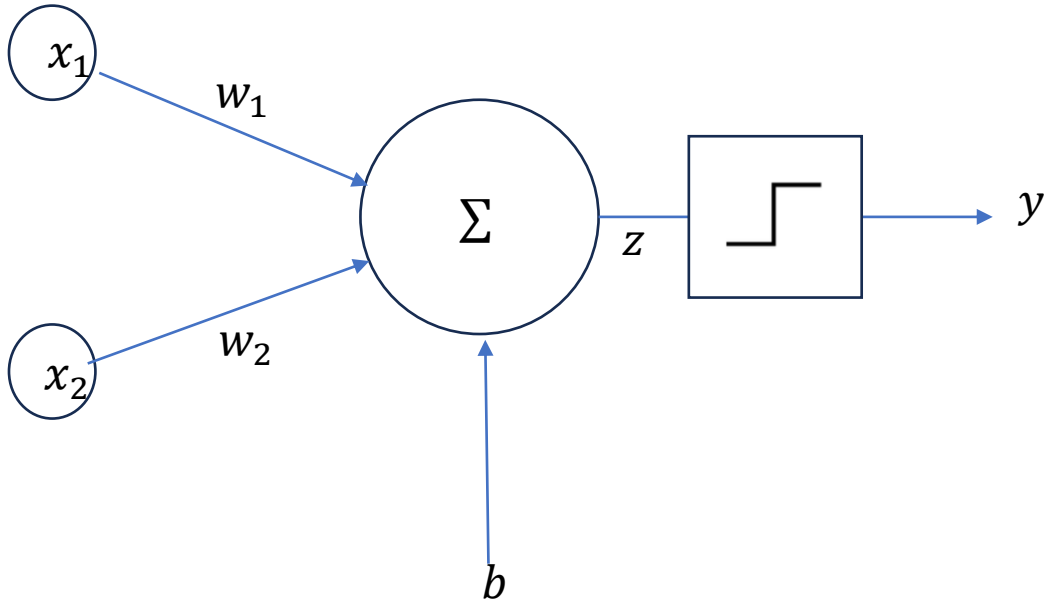
- Multiple hidden layers
- Feedforward (no arrows going backward)
- Fully connected (all outputs of layer x go into all neurons of layer x+1)
- Non-linear activation functions
  - Threshold, sigmoid, ReLU etc

Other types of deep networks

- Can have backward and skip connections, not fully connected etc

Logic using perceptron

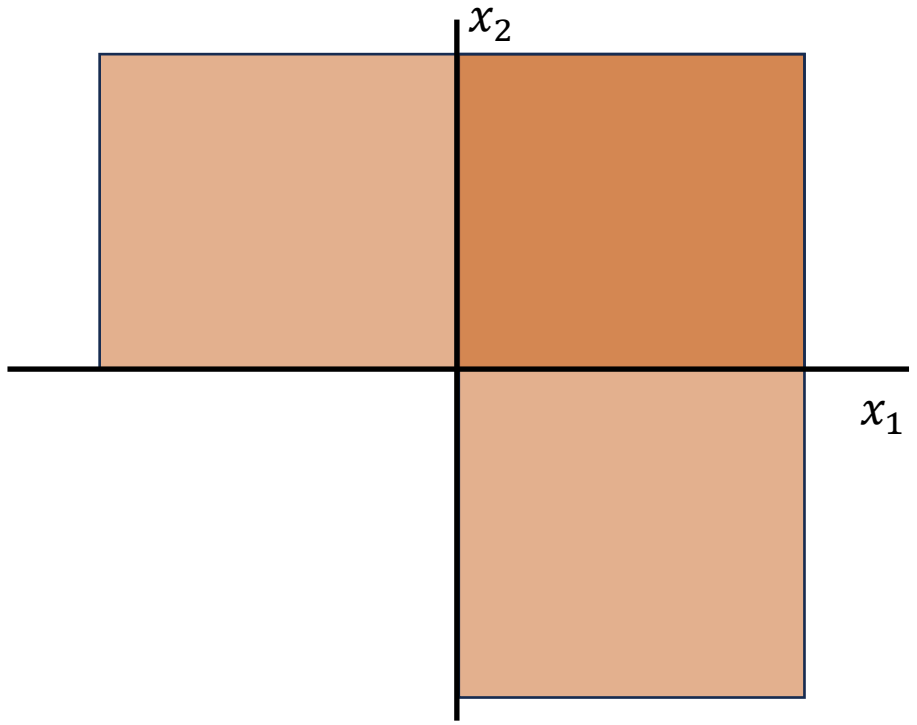
- Let us Implement Boolean expressions using perceptrons
- Perceptron with threshold activation
  - Suppose $y, x_1, x_2 \in \{0,1\}$
  - $w_1, w_2, b \in \mathbb{R}$
- $y = (w_1 x_1 + w_2 x_2 + b \geq 0)$
  - Truth value 0/1
- How would you implement AND?
  - Select values of w_1, w_2, b



Logic gates and Boolean expressions

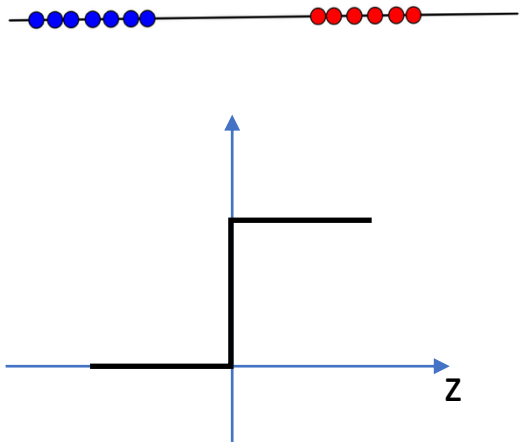
- Logic gates Can be implemented using perceptrons and threshold activation

- Deep neural network can implement arbitrary logic functions/expressions
  - With suitable choice of weights
  - Try out other basic logic functions
    - May need more layers
- Similarly they can be used to create shapes in space

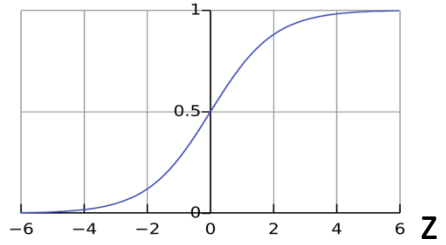


Activation functions

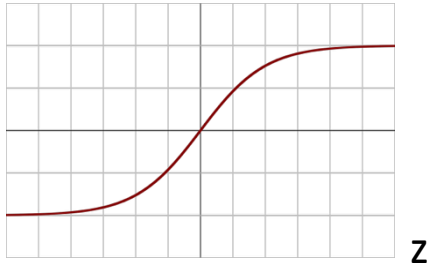
- With $z = Wx = \sum w_i x_i$
- Threshold or step function is $(z \geq 0)$
  - $y \in \{0, 1\}$, or $y \in \{-1, 1\}$



- Sigmoid, logistic or soft step
  - $\sigma(z) = \frac{1}{1+e^{-z}} = \frac{e^z}{1+e^z}$
  - Like step, but a but smoother and changing everywhere

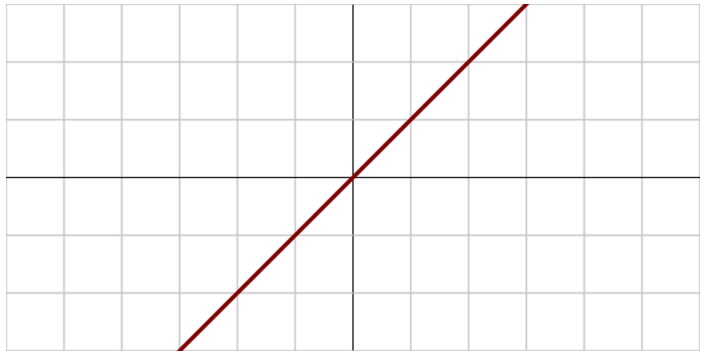


- $\tanh$: hyperbolic tangent
  - $\tanh z = \frac{e^z - e^{-z}}{e^z + e^{-z}}$



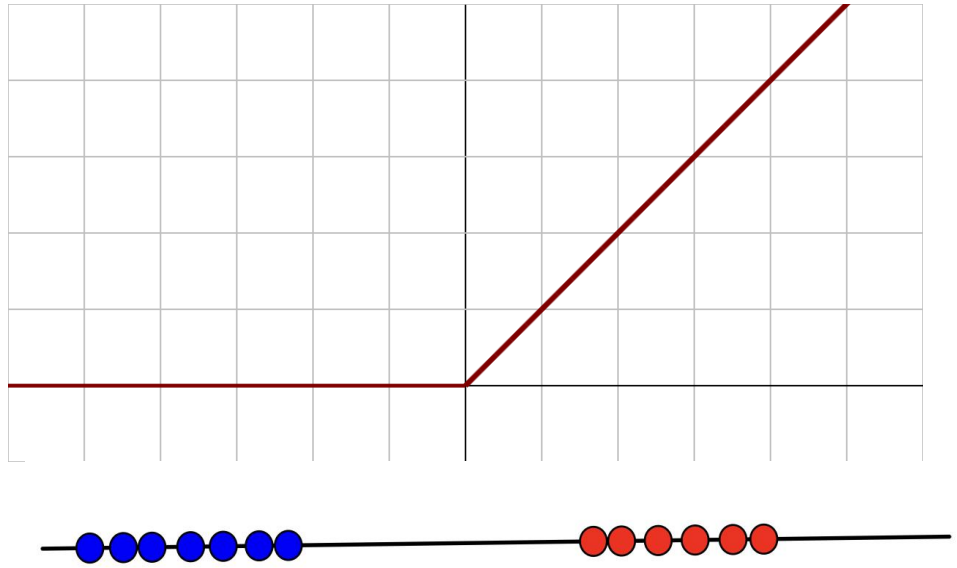
Linear activation

- $y = cz$



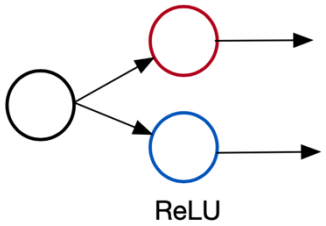
ReLU

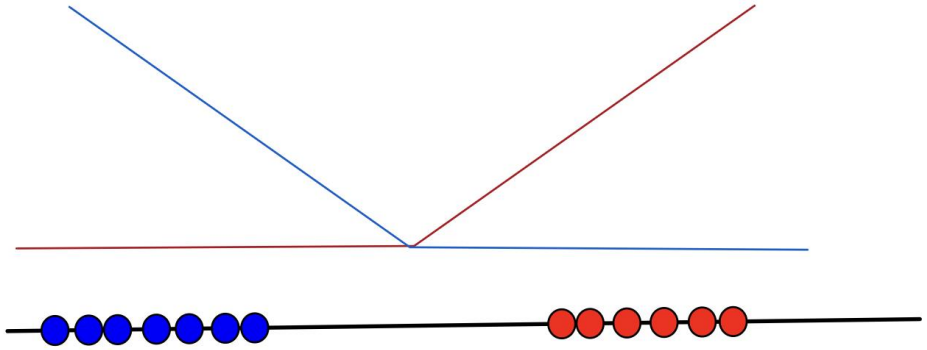
- Rectified linear unit most common type of activation today
- $y = \max(0, z)$
- Variants exist with smoother curve, leaky ReLU etc
- For many examples of activation, see https://en.wikipedia.org/wiki/Activation_function



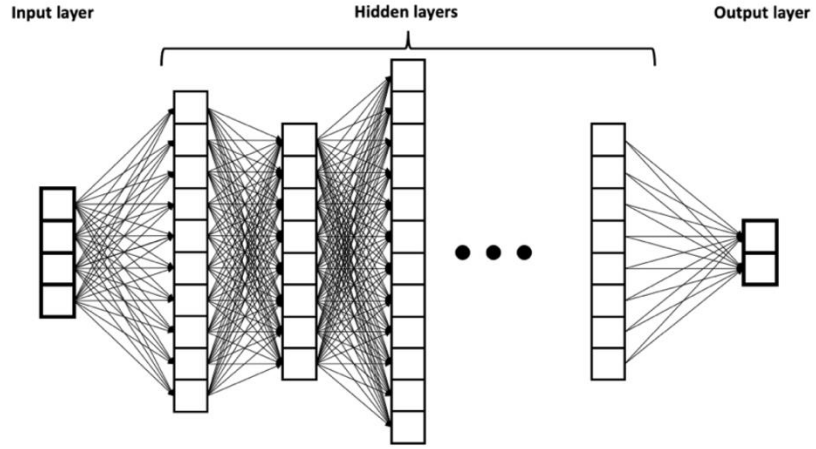
2-class 2-output with ReLU

- Using this network





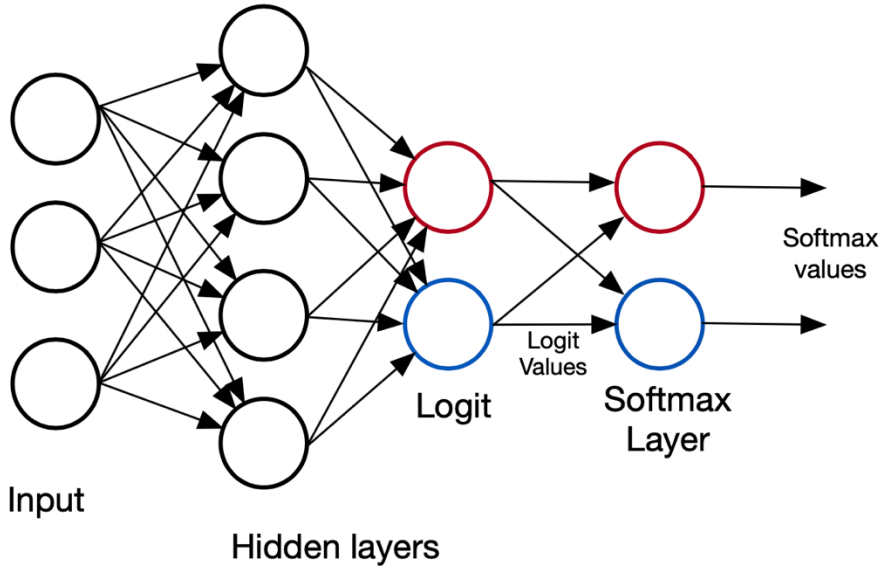
Optimisation in NNs

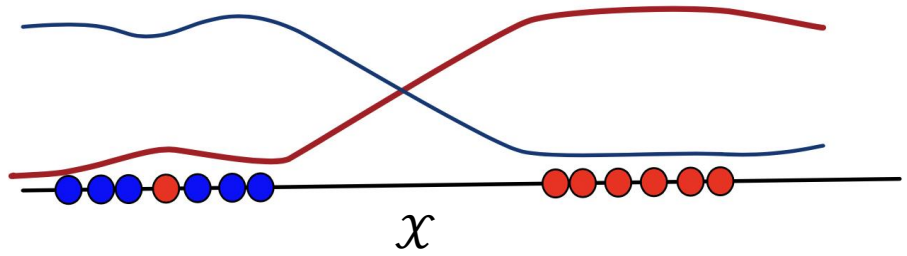


- Usually via SGD and its variants
- Computing gradient:
  - Via chain rule of differentiation
  - Can be done for any sequence of differentiable operations
  - In NNs, each layer computes a function of output of the previous layer
  - Chain rule is applied via *backpropagation algorithm*
- ReLU is the most popular activation

Common NN classifier structure

- Hidden layers use ReLU
- The output for classifiers often uses a different function called softmax
- Suppose the classification problem has n classes
- The Logit layer has n outputs
  - $s_1, \ldots, s_n$
  - scores for each class
  - Largest score is the most likely class
- We output softmax for each class
  - $sm_i = \frac{e^{s_i}}{\sum e^{s_i}}$
- The maximum among these represents the true class
- So, output of NN: A vector with score for each class
  - Normalised by the softmax layer





Question: Why softmax?

Hard max or exact max

- Take a vector of values eg. [2,4,2,9,3,1]
- Make a max vector indicating the position of the max eg. [0,0,0,1,0,0]
  - One hot encoding of position of max

- Or, for 2 classes [red, blue] with values [3, 11]
  - Max vector [0, 1]

Softmax

- Due to exponentiation
  - A value that is larger than the others will become much larger
  - Normalisation will cause this to be close to 1
  - Other values to be close to zero
  - E.g. vector [3,6]
    - Simple normalized vector: [0.33, 0.66]
    - But taking exponentials: $e^3 = 20.08, e^6 = 403.42$
    - Softmax vector = [0.05, 0.95]
- Substitute for hard-max, but differentiable and changing continuously
- Normalized, can be treated as probability p_i for each class

Cross entropy loss

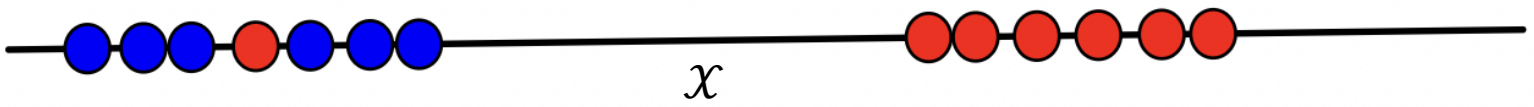
- Neural networks are usually trained on the cross entropy loss of their output p
- Given:
  - Data point x
  - Probability estimate vector p computed by NN through softmax
  - Truth label vector t: indicator vector or one-hot encoding where only the true class has value 1.

p=[0.1, 0.5, 0.2, 0.2]
t= [0.0, 1.0, 0.0, 0.0]

- Cross entropy loss: $\ell_{CE} = -\sum t_i \ln p_i$
  - Measures difference between the two probability distributions
- When is ℓ_{CE} large? How large can it be?

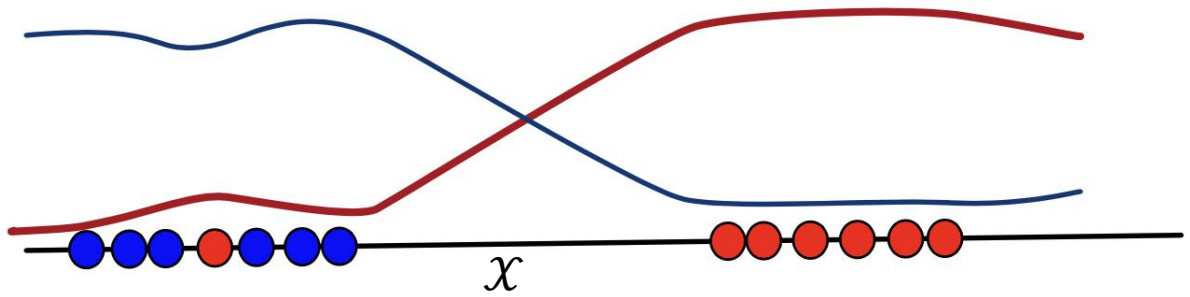
Logit and probability curves

- Consider probability outputs for this data and this data space
  - One curve for each class
- The optimizer (e.g. SGD) tries to create weights so that scores are high at the corresponding colors



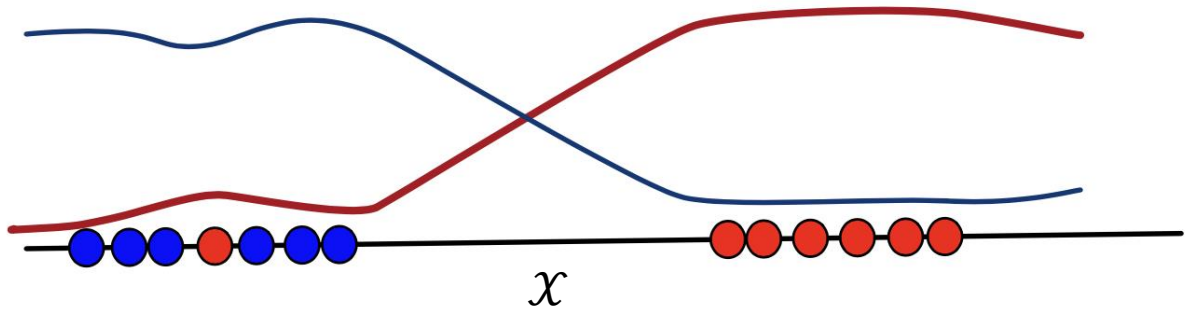
Logit and probability curves

- Consider probability outputs for this data and this data space
  - One curve for each class
- The optimizer (e.g. SGD) tries to create weights so that scores are high at the corresponding colors



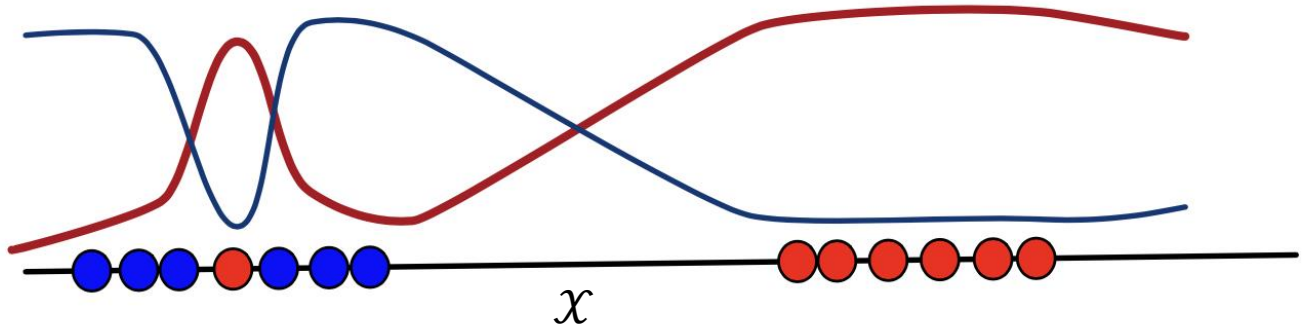
Logit and probability curves

- A reasonable model sacrifices the outlier for better generalization
- But what is the cross entropy loss at the outlier?
- What happens if we continue to run the optimizer?



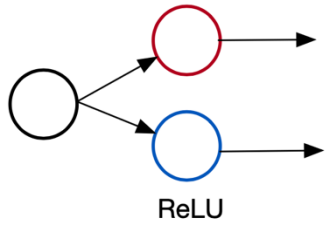
Overfitting

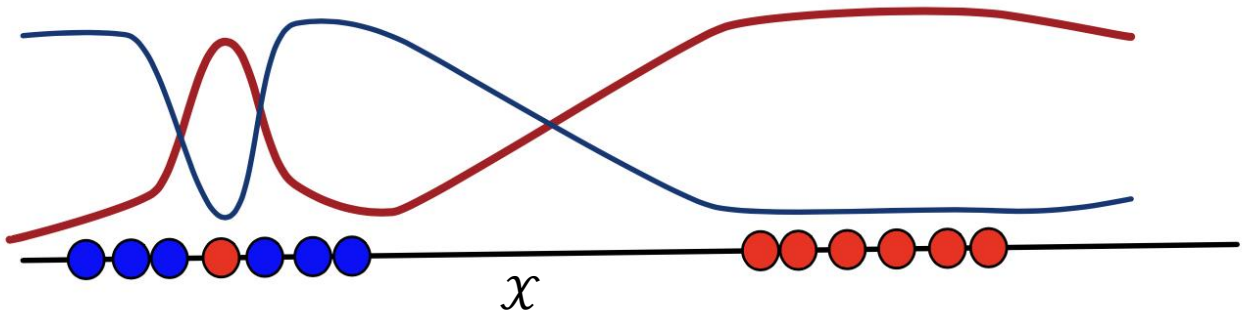
- Optimiser tries to modify red and blue curves
- Such that large CE losses become smaller



Can this happen with simple NNs?

- E.g. with a single perceptron?
- Or with this simple NN:

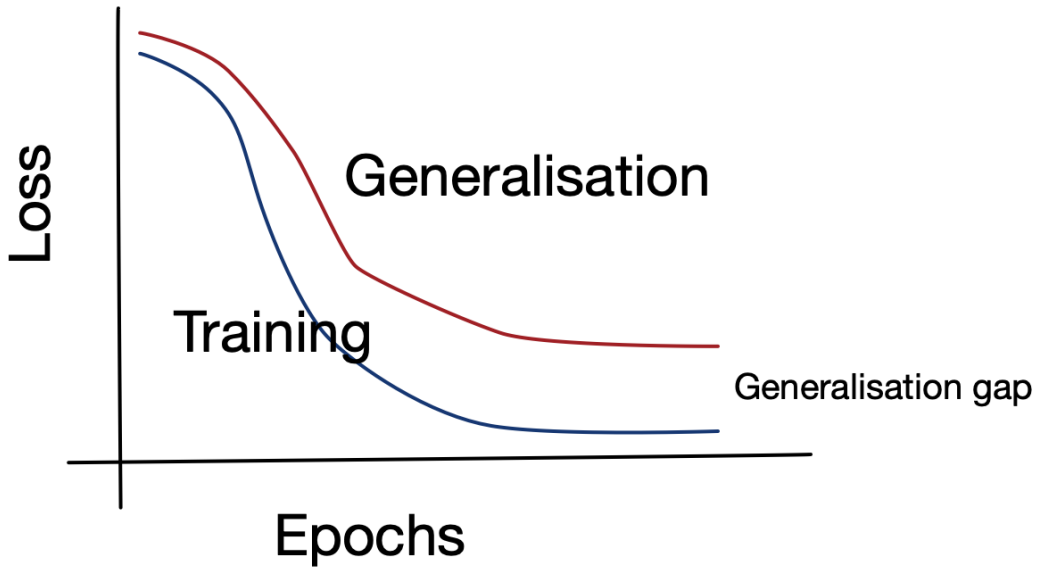




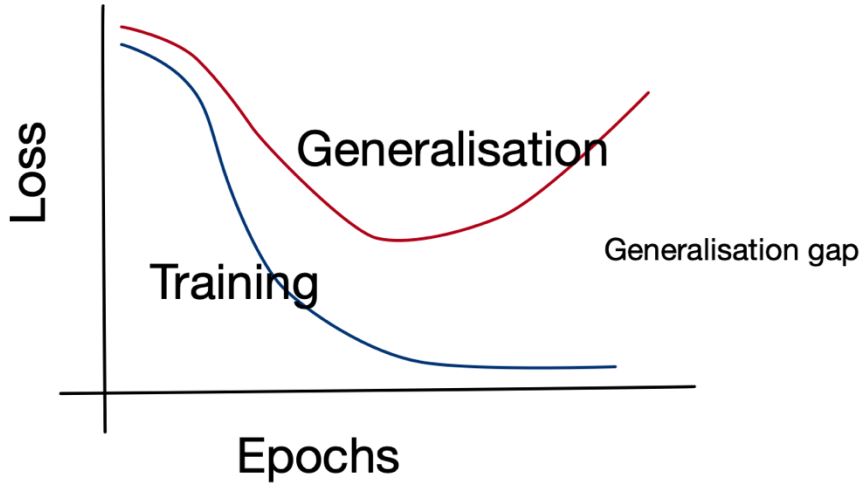
Generalisation gap for neural network classifiers

- With large neural networks

What we might expect

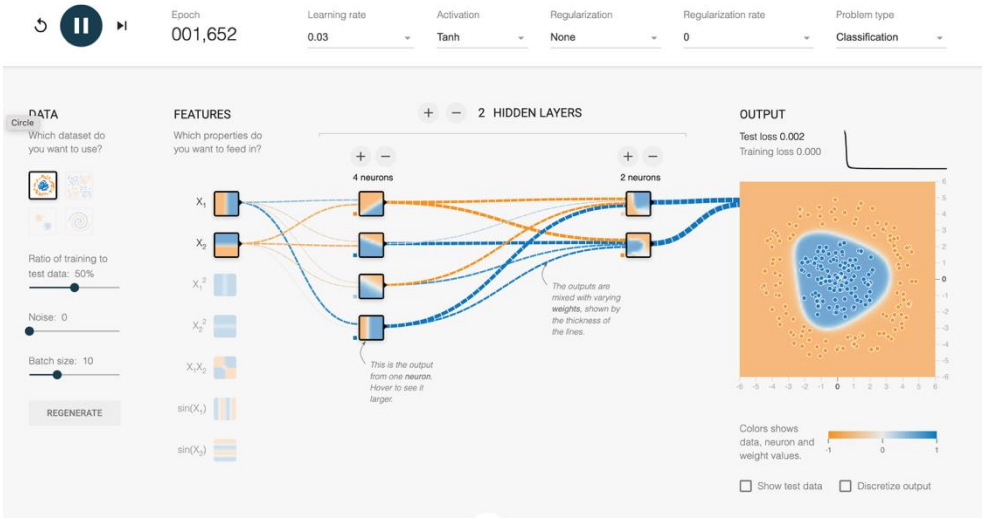


What we find



Now that we have seen Neural networks:

- Try out:
  - https://playground.tensorflow.org/



- For one of the datasets (e.g. the circular one or the diametrically opposite quadrants one) modify the hidden layer so that tanh activation does not find a solution, but ReLU does!
  - Post on Piazza!