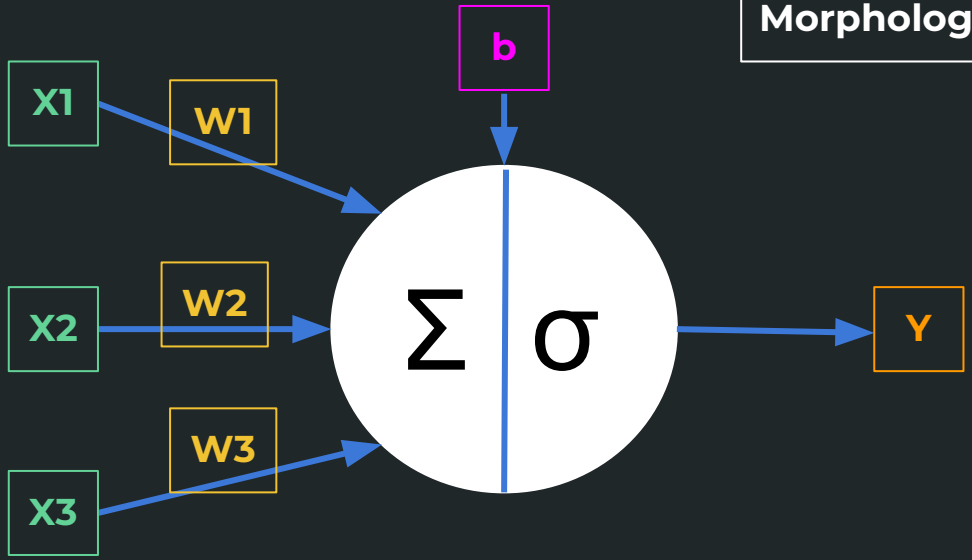
Artificial Neural Network

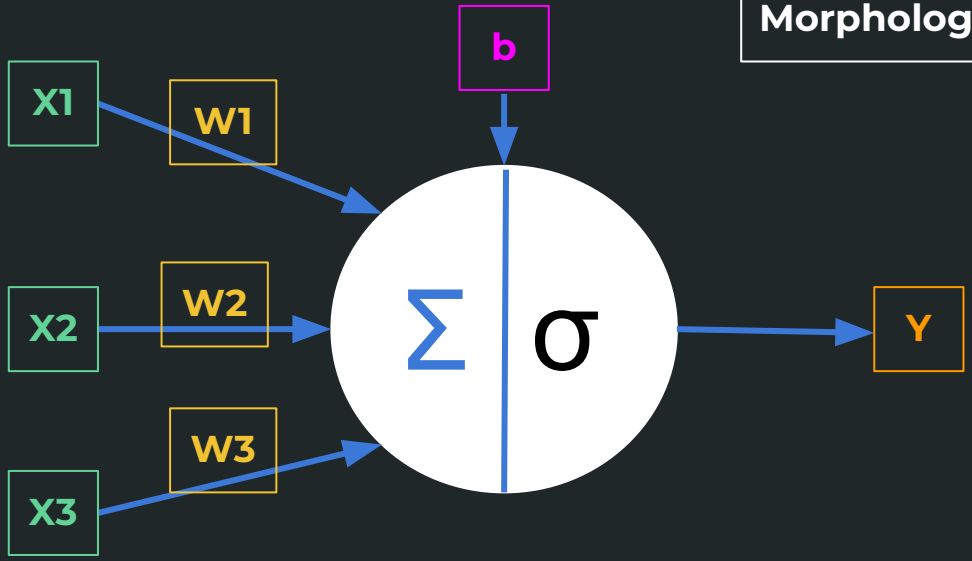
Morphology of an Artificial Neuron
Activation Function
A simple Artificial Neural Network
Forward Propagation
Gradient Descent
Backpropagation
Regularisation (if possible)

Morphology of an Artificial Neuron



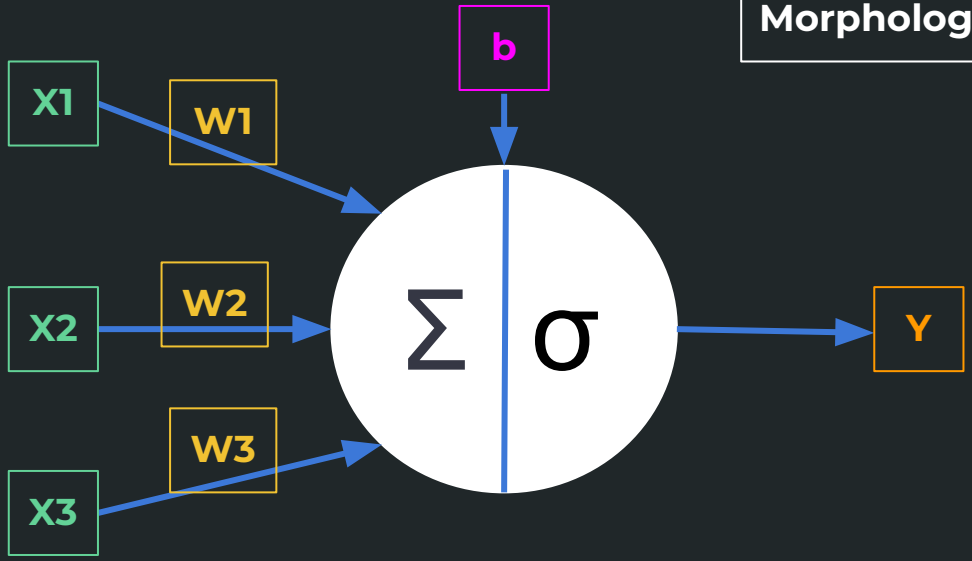
- An Artificial Neuron can have more than one input to the neuron but it should have only one output
- Each input edge will have a weight associated with it. Weights can be ‘-ve’, ‘+ve’, or ‘0’.
- Generally, -1 < weights < 1.
- Each neuron has a bias component associated with it.
- Neuron computes output by two sequential process: Summation of (weighted) Input and Activation

Morphology of an Artificial Neuron



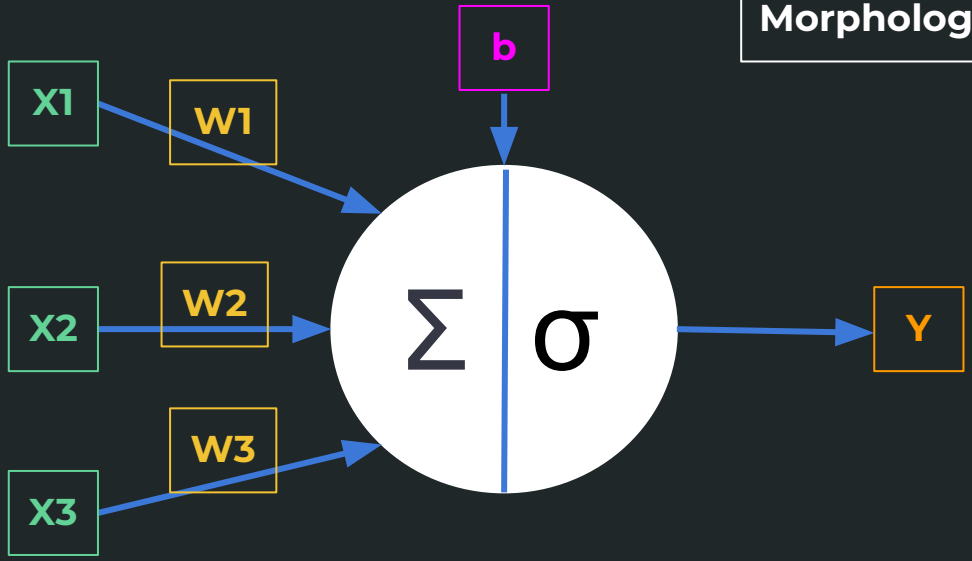
$$\begin{bmatrix} X1 & X2 & X3 \end{bmatrix} \times \begin{bmatrix} W1 \\ W2 \\ W3 \end{bmatrix} = X1W1 + X2W2 + X3W3$$

Morphology of an Artificial Neuron



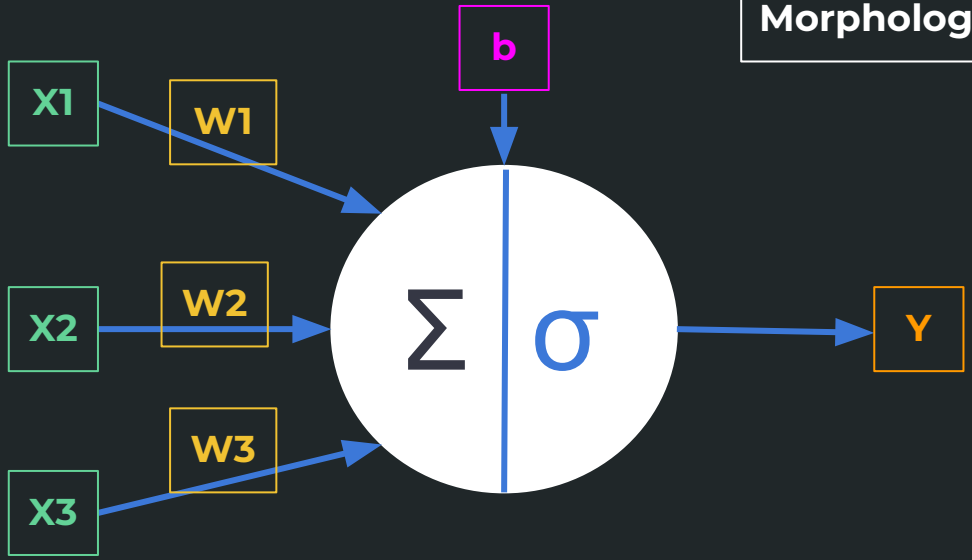
$$\begin{bmatrix} X1 & X2 & X3 \end{bmatrix} \times \begin{bmatrix} W1 \\ W2 \\ W3 \end{bmatrix} = X1W1 + X2W2 + X3W3 + b$$

Morphology of an Artificial Neuron



$$\begin{bmatrix} X1 & X2 & X3 \end{bmatrix} \times \begin{bmatrix} W1 \\ W2 \\ W3 \end{bmatrix} = X1W1 + X2W2 + X3W3 + b$$

Morphology of an Artificial Neuron

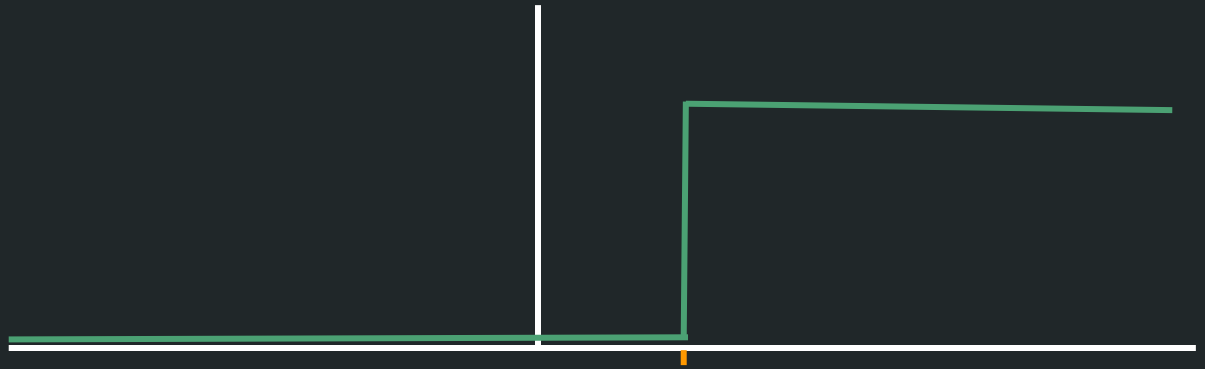


$$\sigma\left(X1W1 + X2W2 + X3W3 + b \right) = Y$$

Artificial Neural Network

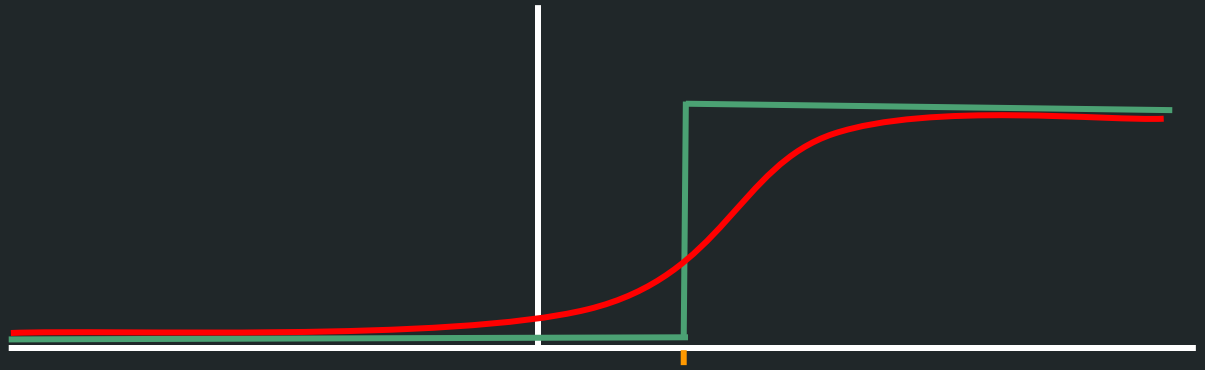
~~# Morphology of an Artificial Neuron~~
Activation Function
A simple Artificial Neural Network
Forward Propagation
Gradient Descent
Backpropagation
Regularisation (if possible)

Activation Function



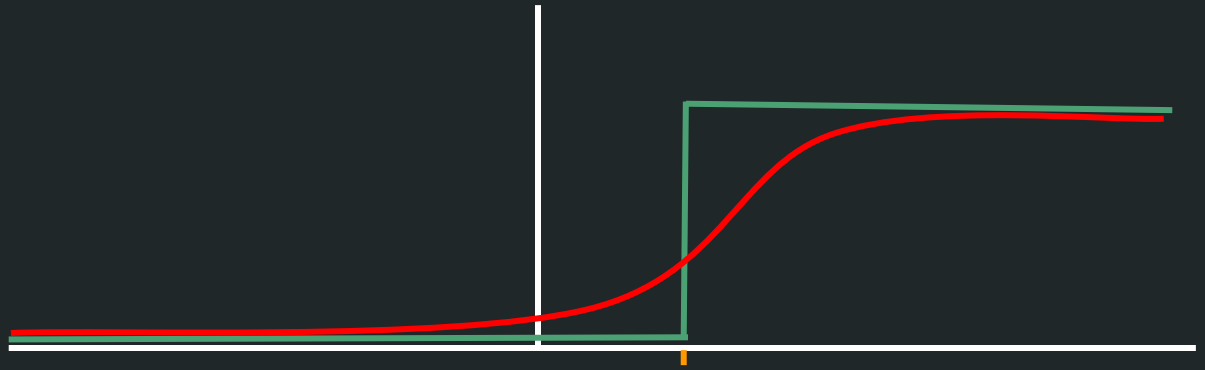
- Inspired from Biological Neurons

Activation Function



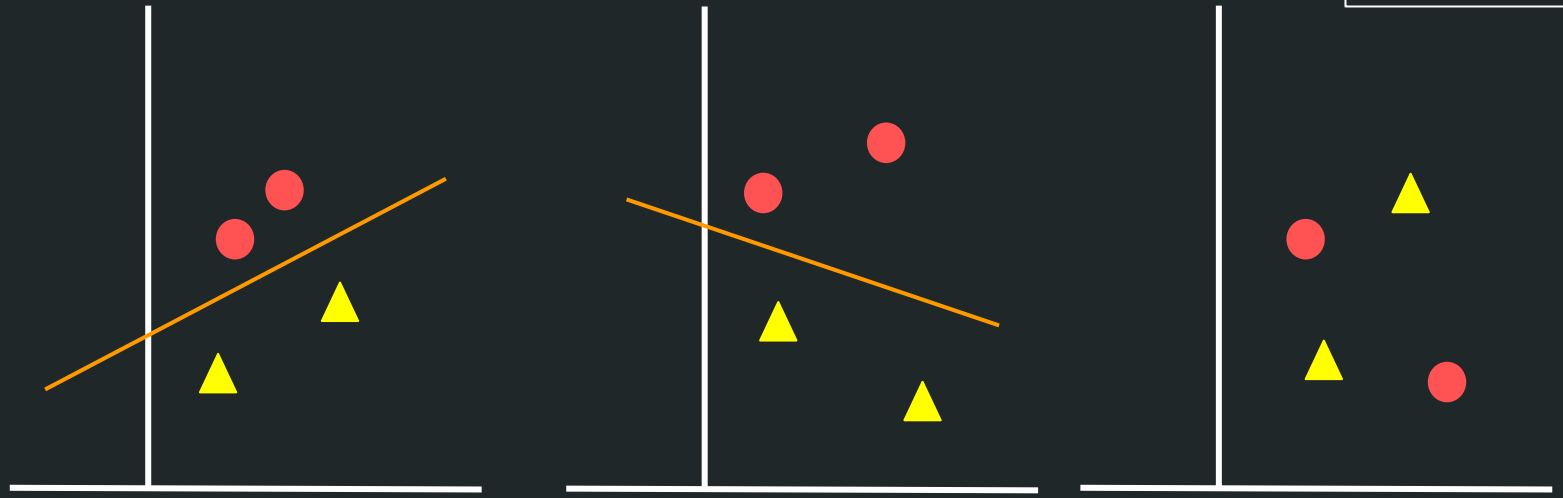
- Inspired from Biological Neurons

Activation Function



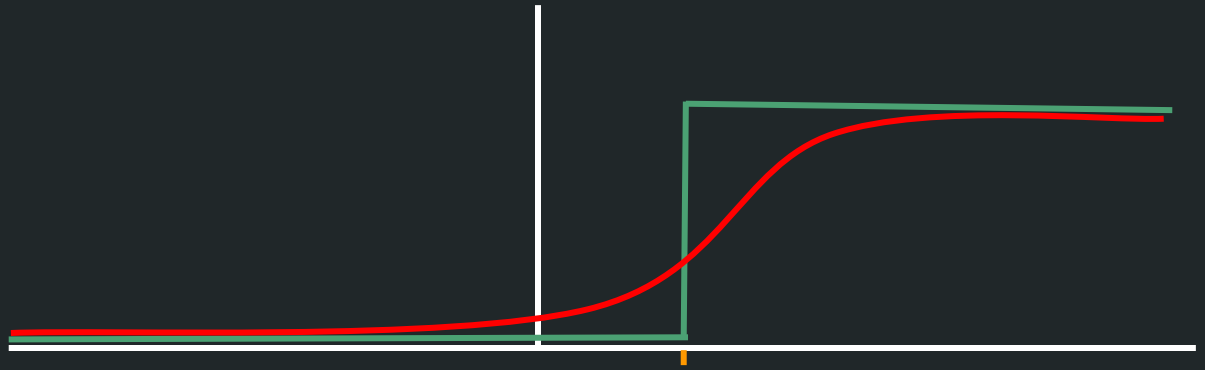
- Inspired from Biological Neurons
- Activation Function is a nonlinear mathematical Function

Activation Function



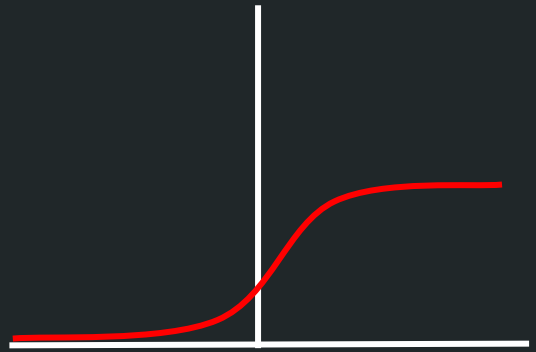
- Inspired from Biological Neurons
- Activation Function is a nonlinear mathematical Function

Activation Function

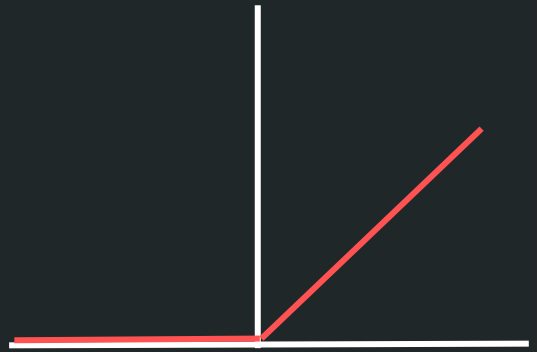


- Inspired from Biological Neurons
- Activation Function is a nonlinear mathematical Function
- Activation Function provides a upper bound to the output

Activation Function



$$\text{Sigmoid}(x) = \frac{1}{1+e^{-x}}$$



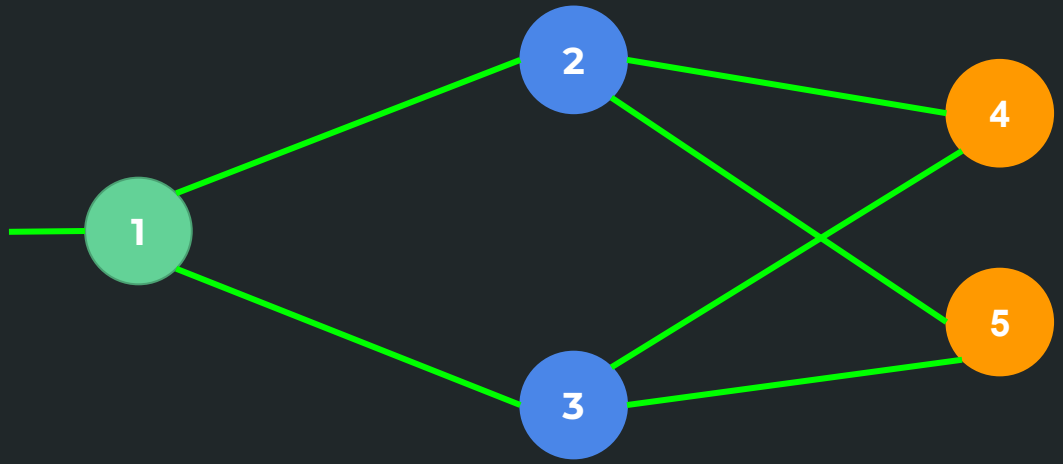
$$\text{ReLu}(x) = \begin{cases} x & \text{if } x>0 \\ 0 & \text{if } x<=0 \end{cases}$$

$$\text{Softmax} = \frac{e^{zi}}{\sum_{j=1}^{k} 1+e^{zj}}$$

- Inspired from Biological Neurons
- Activation Function is a nonlinear mathematical Function
- Activation Function provides a upper bound to the output

Artificial Neural Network

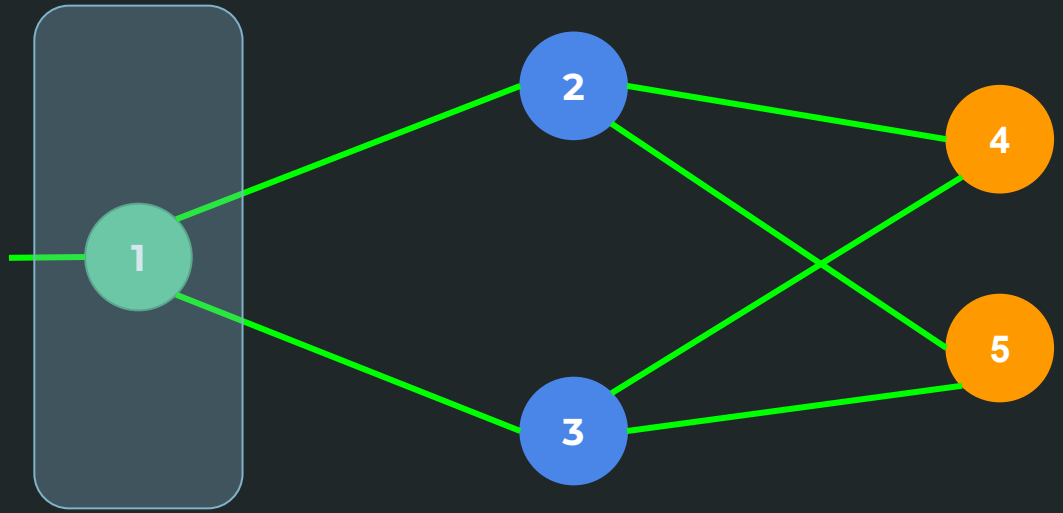
~~# Morphology of an Artificial Neuron~~
~~# Activation Function~~
A simple Artificial Neural Network
Forward Propagation
Gradient Descent
Backpropagation
Regularisation (if possible)



Simple Artificial Neural Network

- Output of each neuron in a layer is connected to the every other neurons in the next layer

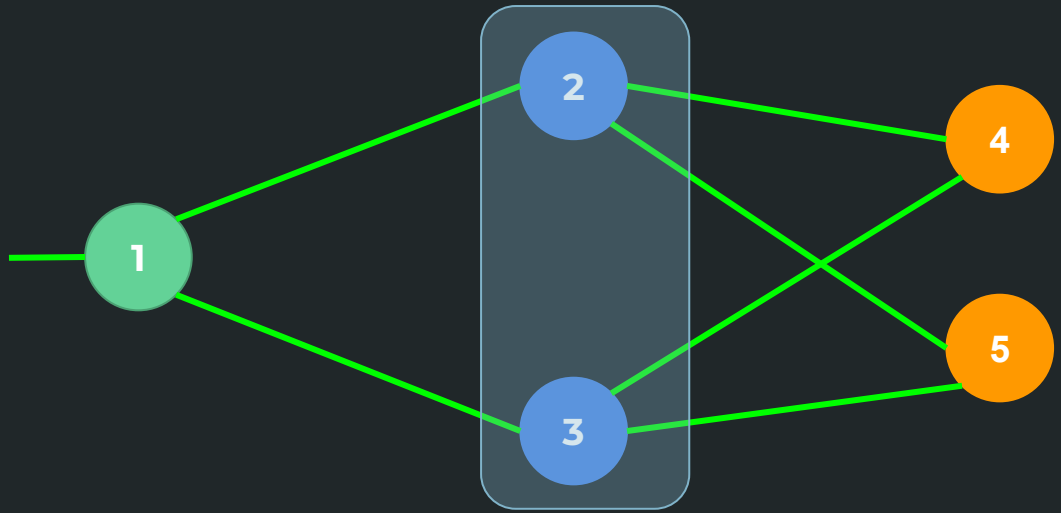
Artificial Neural Net.



Input Layer:

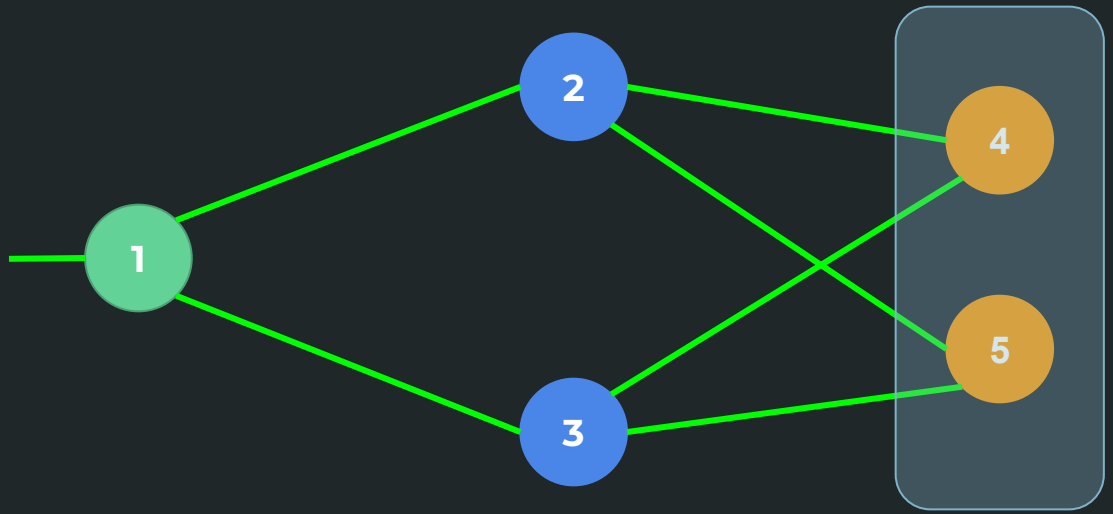
- Neurons in the input layer takes the original input fed into the model/system
- There can be more than one neurons in the Input Layer
- Input Neurons doesn't the alters the original input by any means, it simply passes the input to the next layers neuron

Artificial Neural Net.



Hidden Layer:

- Neurons in the hidden layer, works like an artificial neuron (ie. it sums the weighted input and applies the activation function)
- There is no upper limit to number of neurons in a hidden layer, and same goes to the number of hidden layer. This number is set as per the requirement.
- But at least one neuron per hidden layer, and one hidden layer in the system



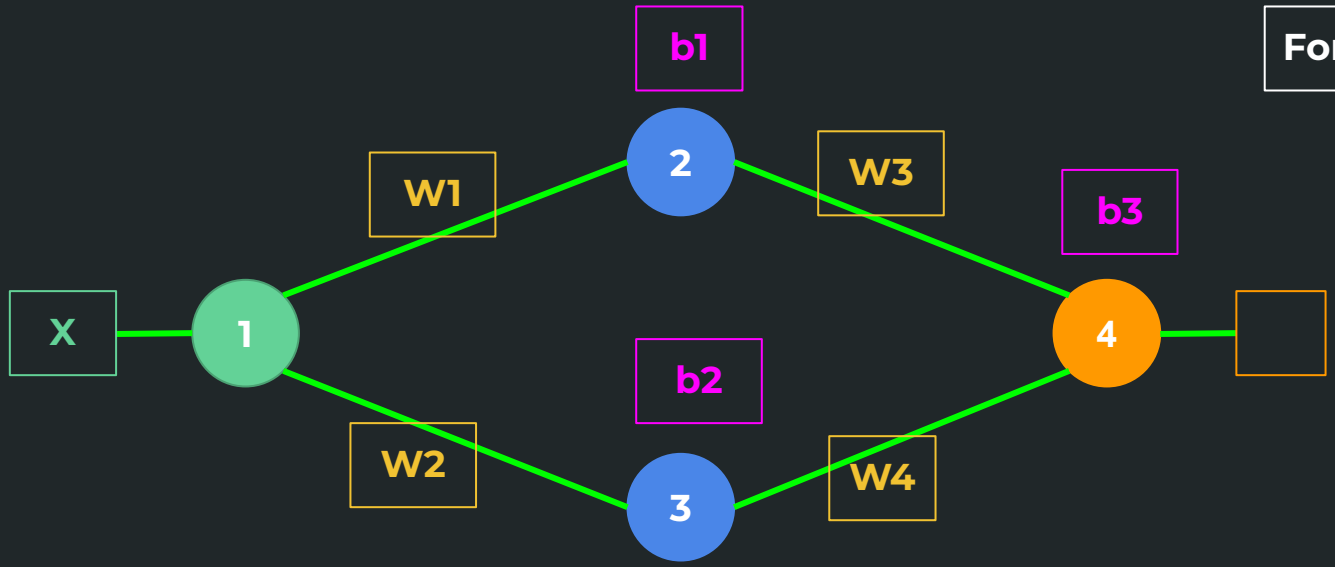
Output Layer:

- Number of neurons in the output layer is set according to the requirement
- This is the final output of the model

Artificial Neural Network

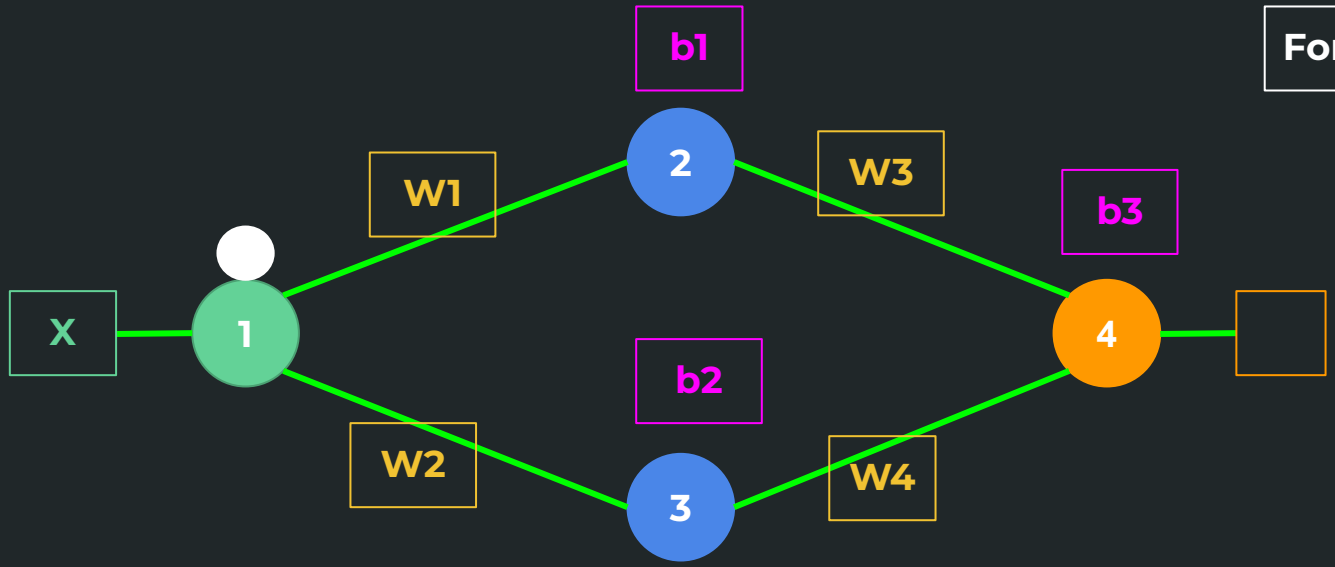
~~# Morphology of an Artificial Neuron~~
~~# Activation Function~~
~~# A simple Artificial Neural Network~~
Forward Propagation
Gradient Descent
Backpropagation
Regularisation (if possible)

Forward Propagation



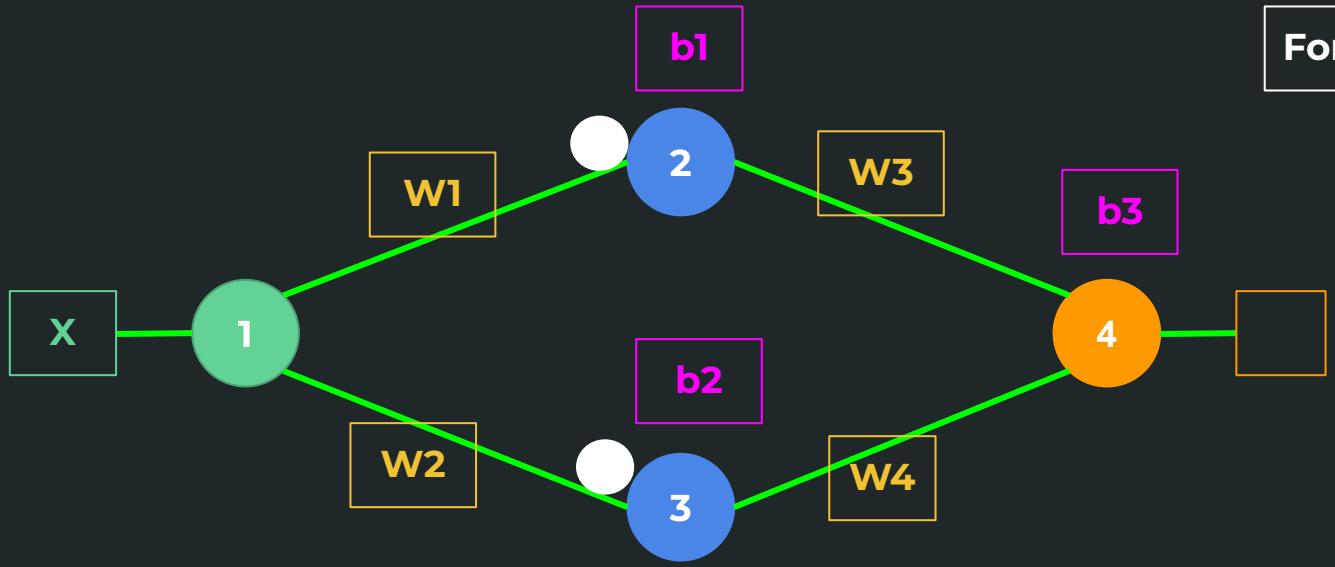
- 1 I/P layer, 1 Hidden Layer, 1 O/P layer
- 1 I/P neuron, 2 Hidden layer neuron, 1 O/P neuron
- It is consider that weights and bias are optimized
- Activation Function: σ

Forward Propagation



1 X

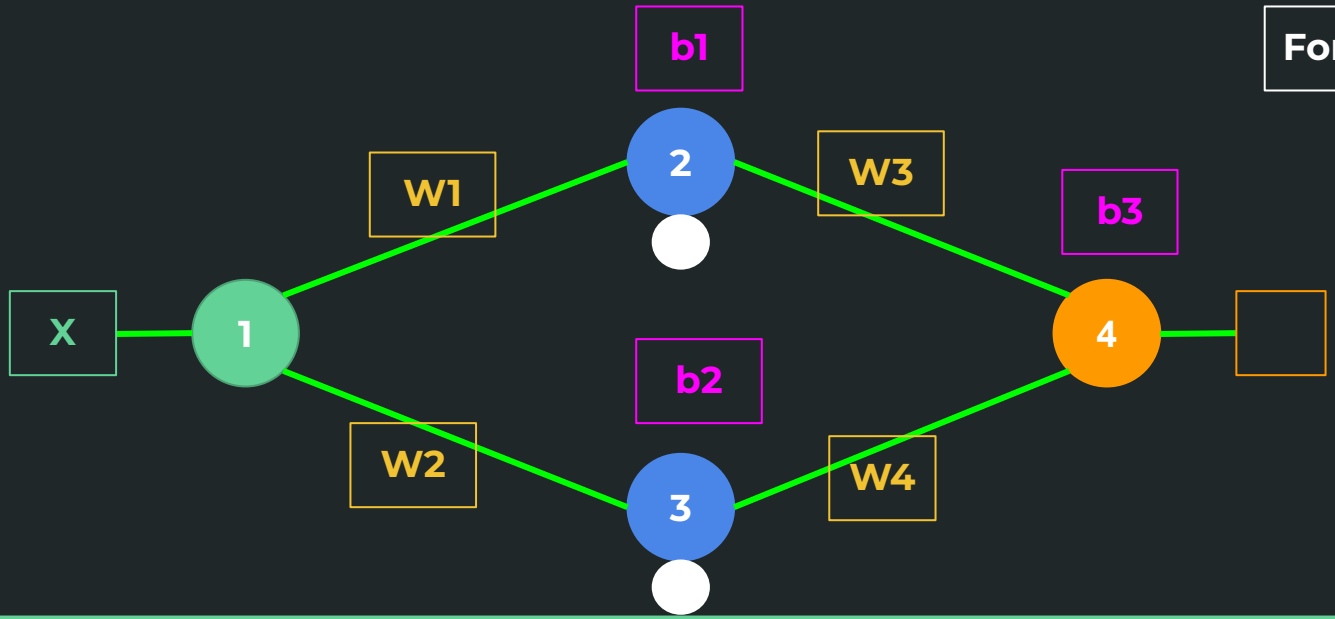
Forward Propagation



XW1

XW2

Forward Propagation



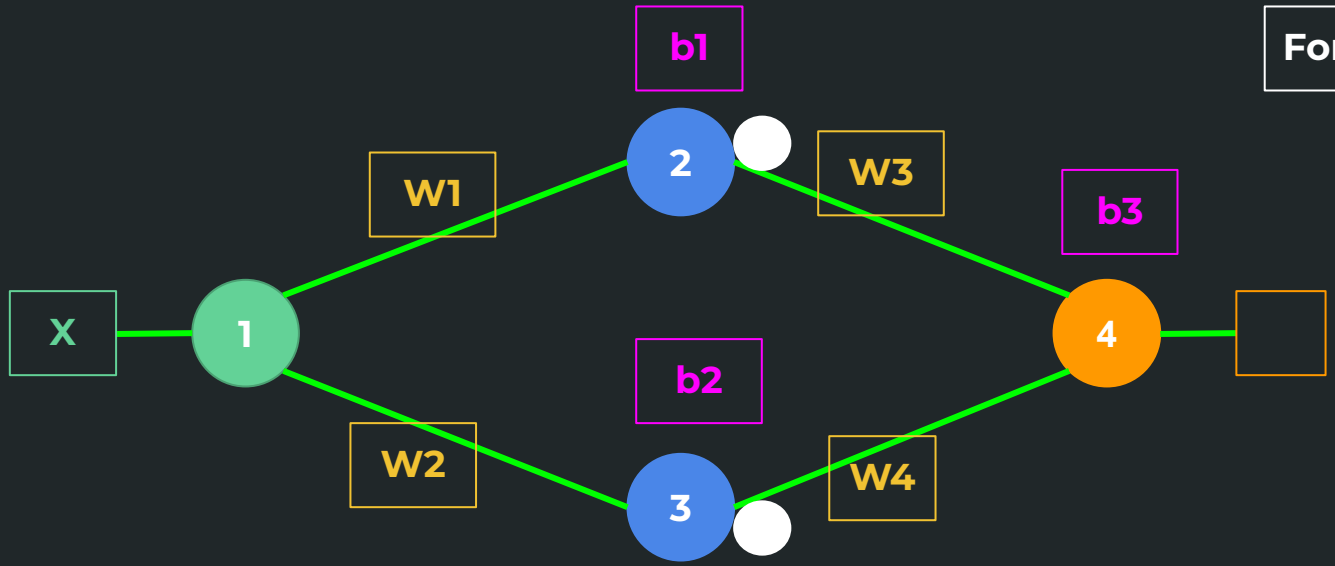
1 — 2 : XW1

1 — 3 : XW2

2 : XW1 + b1

3 : XW2 + b2

Forward Propagation



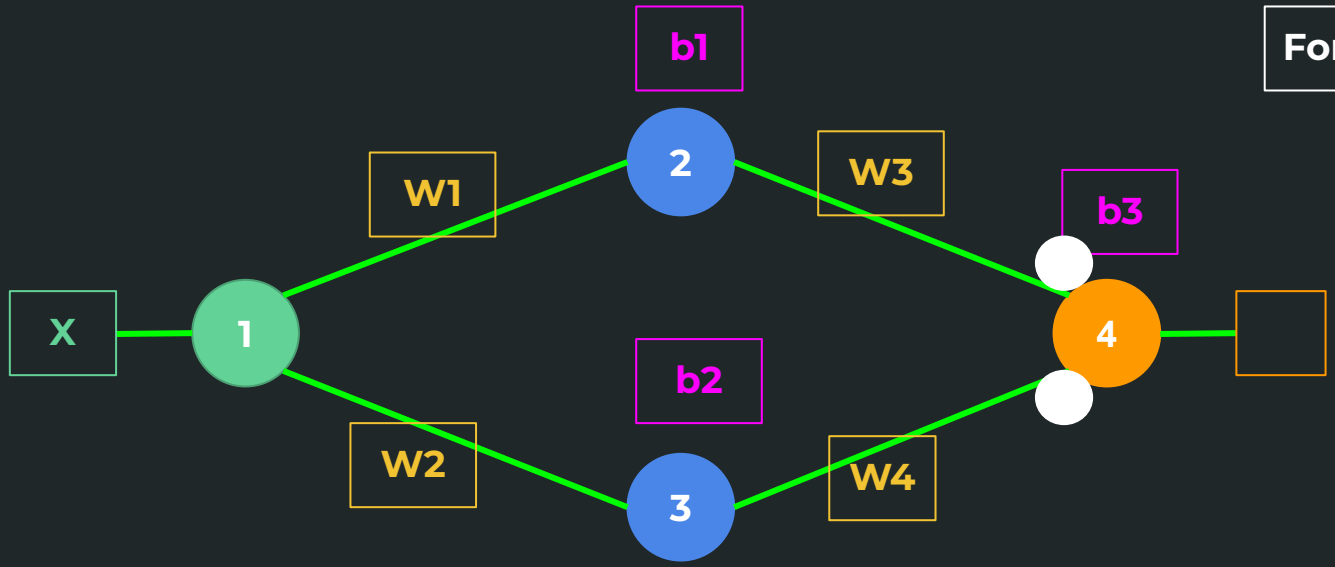
1 — 2 : XW1

1 — 3 : XW2

2 : σ(XW1 + b1)

3 : σ(XW2 + b2)

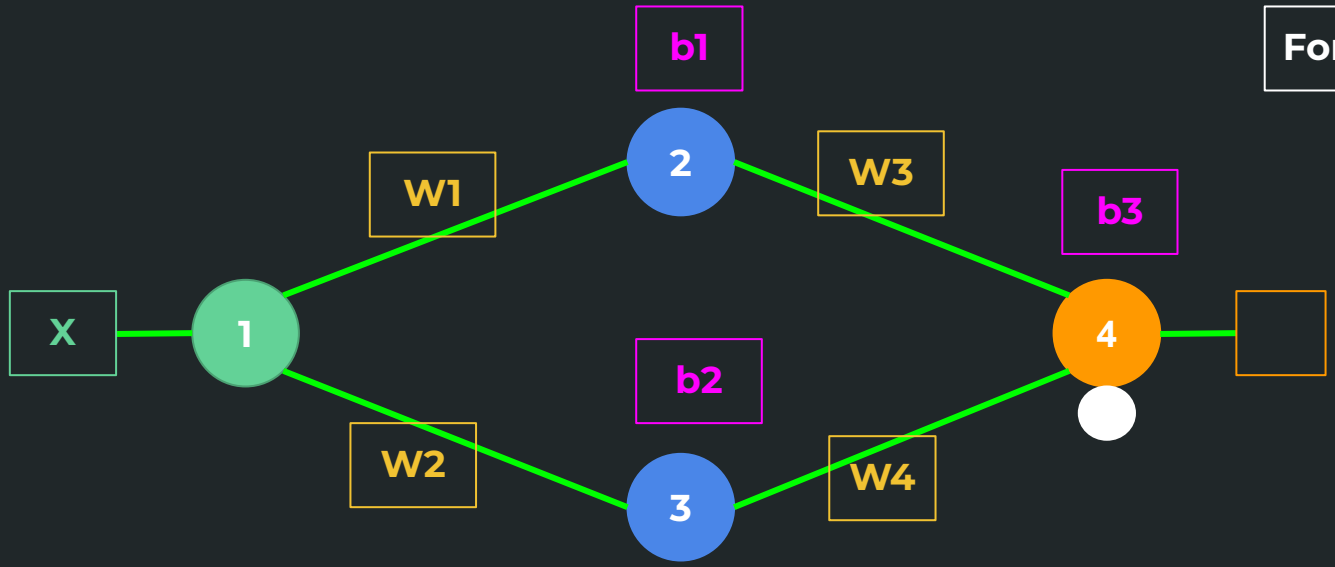
Forward Propagation



$W3.\sigma(XW1 + b1)$

$W4.\sigma(XW2 + b2)$

Forward Propagation

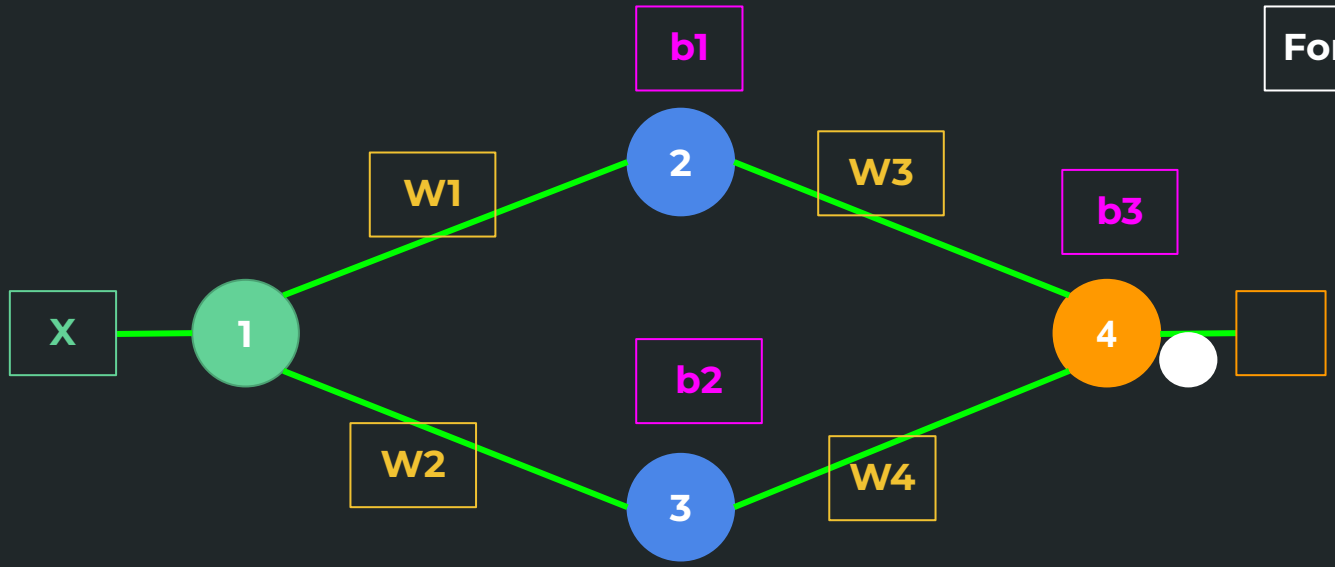


2 — 4 : $W3.\sigma(XW1 + b1)$

3 — 4 : $W4.\sigma(XW2 + b2)$

4 : $W3.\sigma(XW1 + b1) + W4.\sigma(XW2 + b2) + b3$

Forward Propagation



2 — 4 : $W3.\sigma(XW1 + b1)$

3 — 4 : $W4.\sigma(XW2 + b2)$

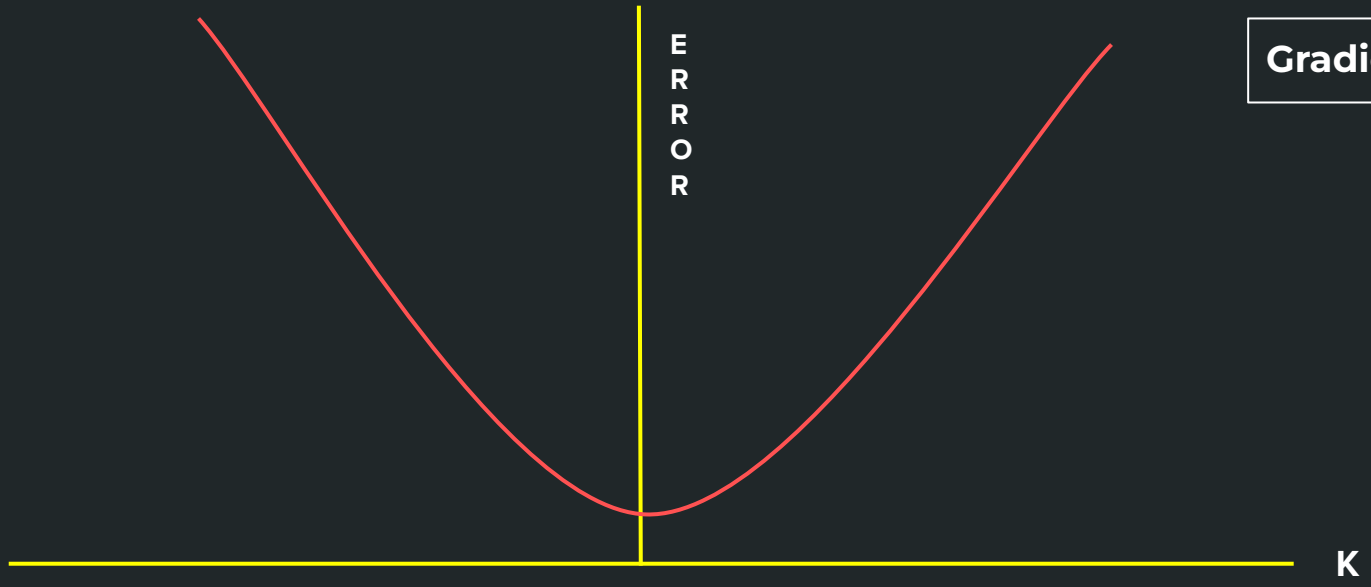
4 : $W3.\sigma(XW1 + b1) + W4.\sigma(XW2 + b2) + b3$

4 : $\sigma(W3.\sigma(XW1 + b1) + W4.\sigma(XW2 + b2) + b3)$

Artificial Neural Network

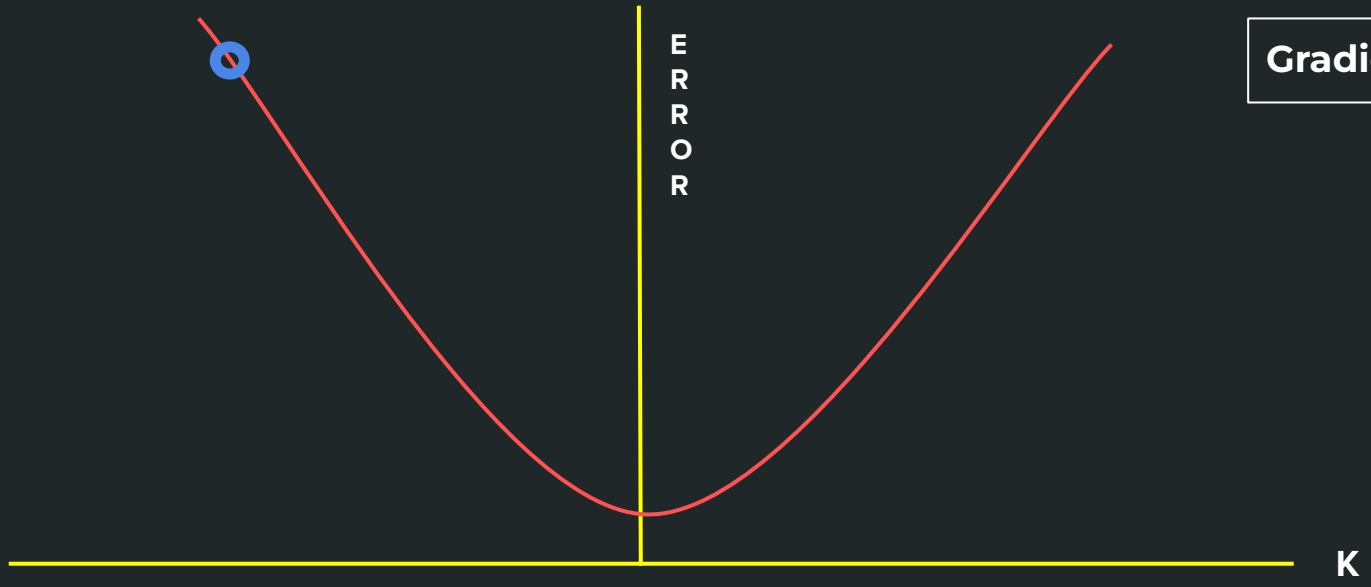
~~# Morphology of an Artificial Neuron~~
~~# Activation Function~~
~~# A simple Artificial Neural Network~~
~~# Forward Propagation~~
Gradient Descent
Backpropagation
Regularisation (if possible)

Gradient Descent



- Error Vs K (Some Optimizable Parameter)

Gradient Descent

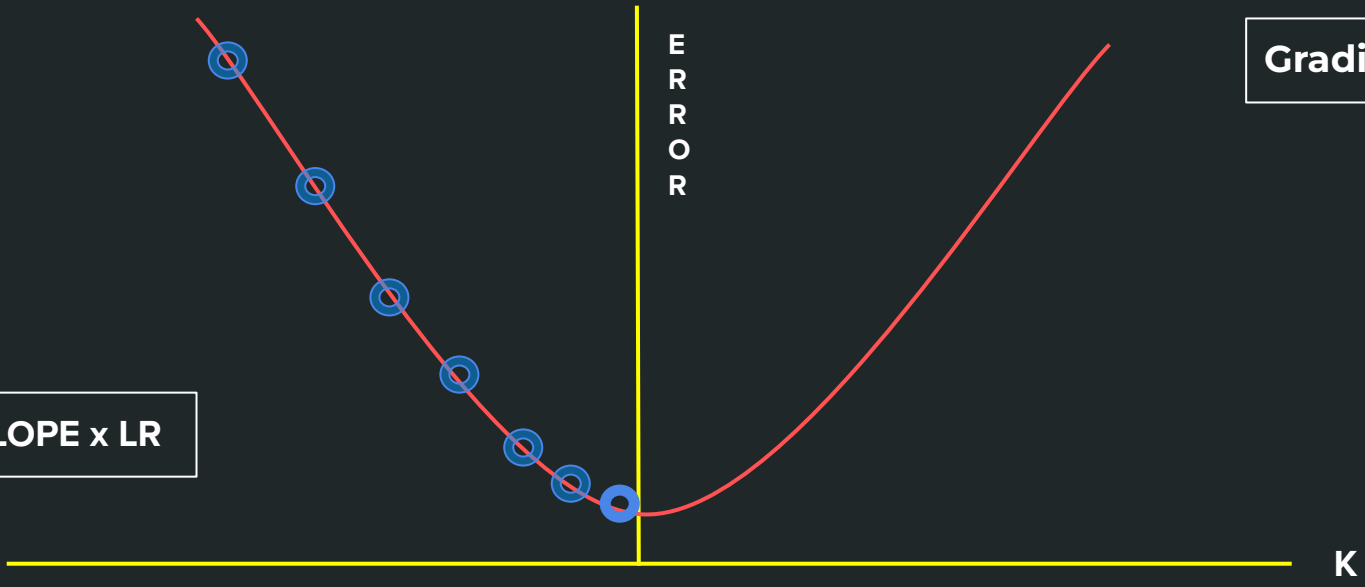


- Error Vs K (Some Optimizable Parameter)
- Randomly chosen a K value

Gradient Descent

- Error Vs K (Some Optimizable Parameter)
- Randomly chosen a K value
- Find the slope, and proceed acc. to the slope

Gradient Descent

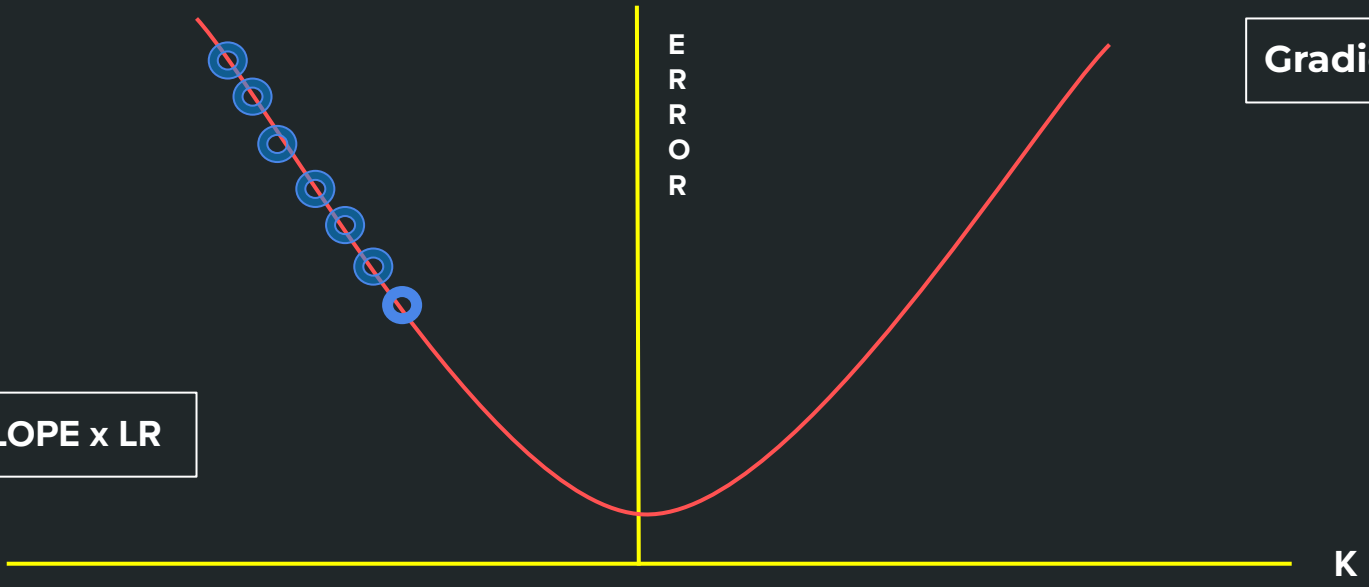


SLOPE

K = K - SLOPE x LR

- Error Vs K (Some Optimizable Parameter)
- Randomly chosen a K value
- Find the slope, and proceed acc. to the slope
- Update the K value
- When minimum is further away it takes larger step, and when minimum is closer it takes smaller steps

Gradient Descent

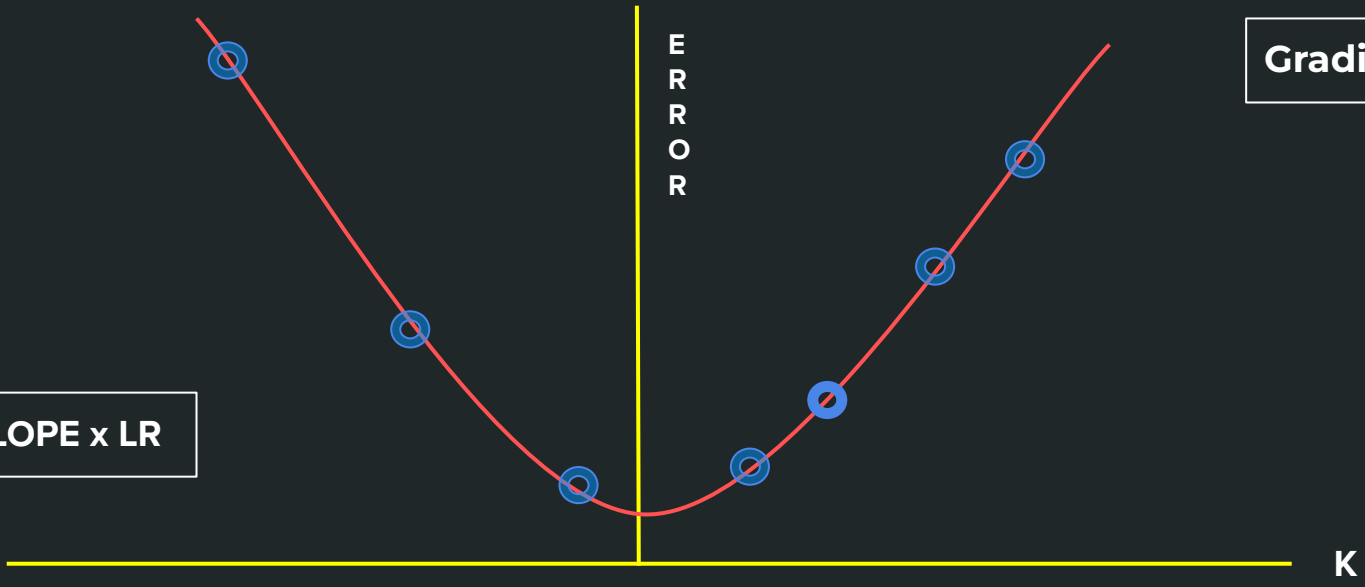


SLOPE

K = K - SLOPE x LR

- Very Small LR, increase number of computation

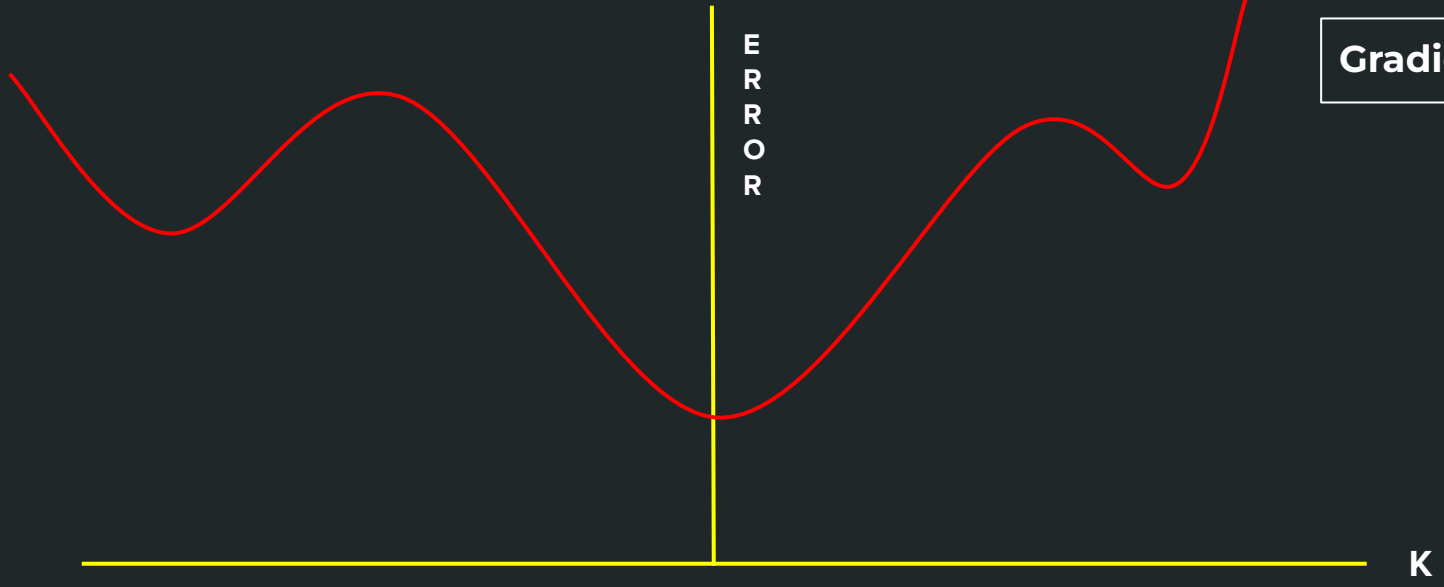
Gradient Descent



SLOPE

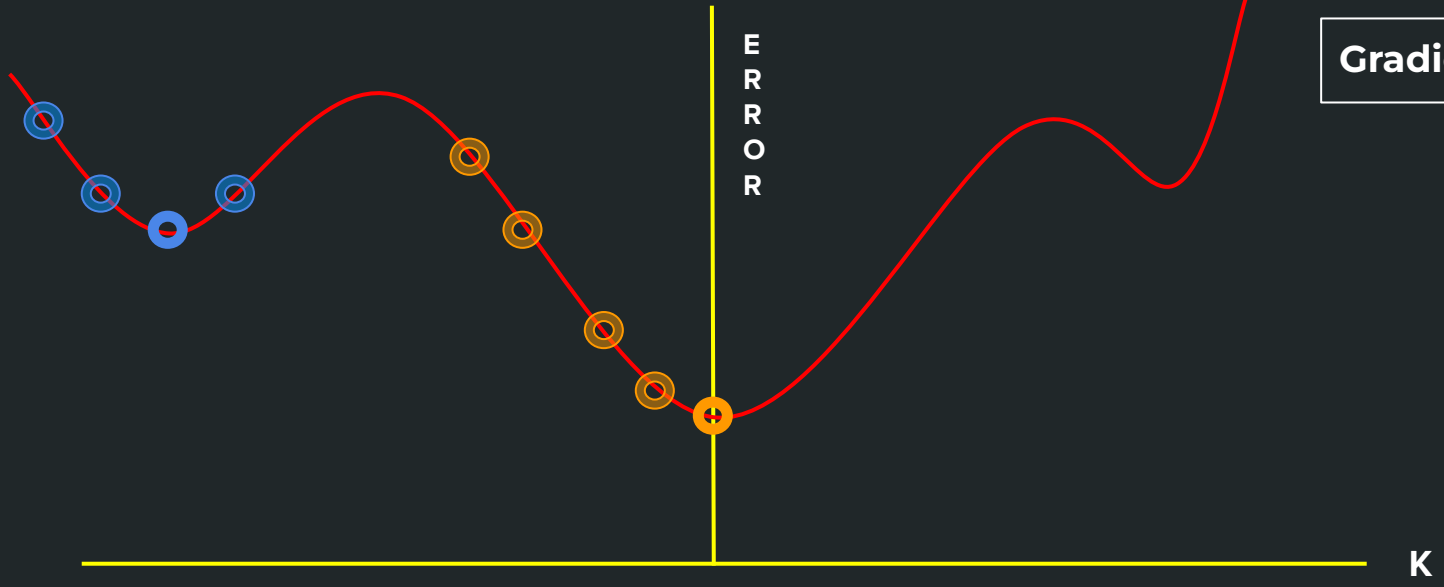
K = K - SLOPE x LR

- Very large LR, decrease the number of computation but will overshoot



Gradient Descent

- We need to choose the initial K value randomly.

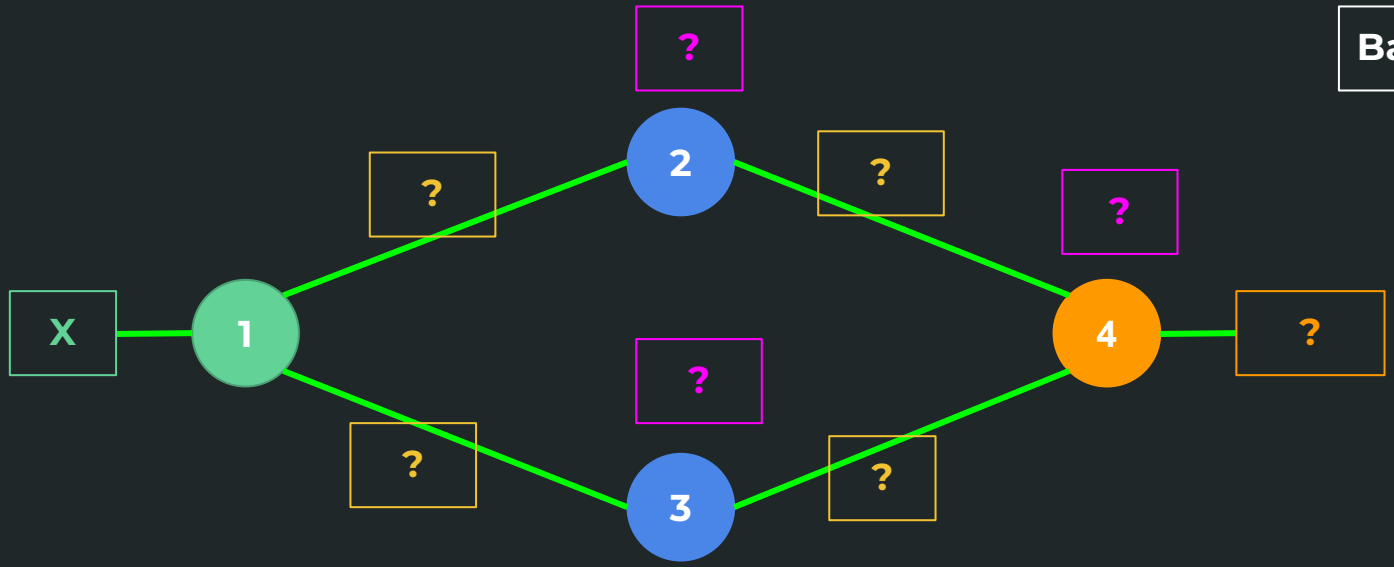


- We need to choose the initial K value randomly.

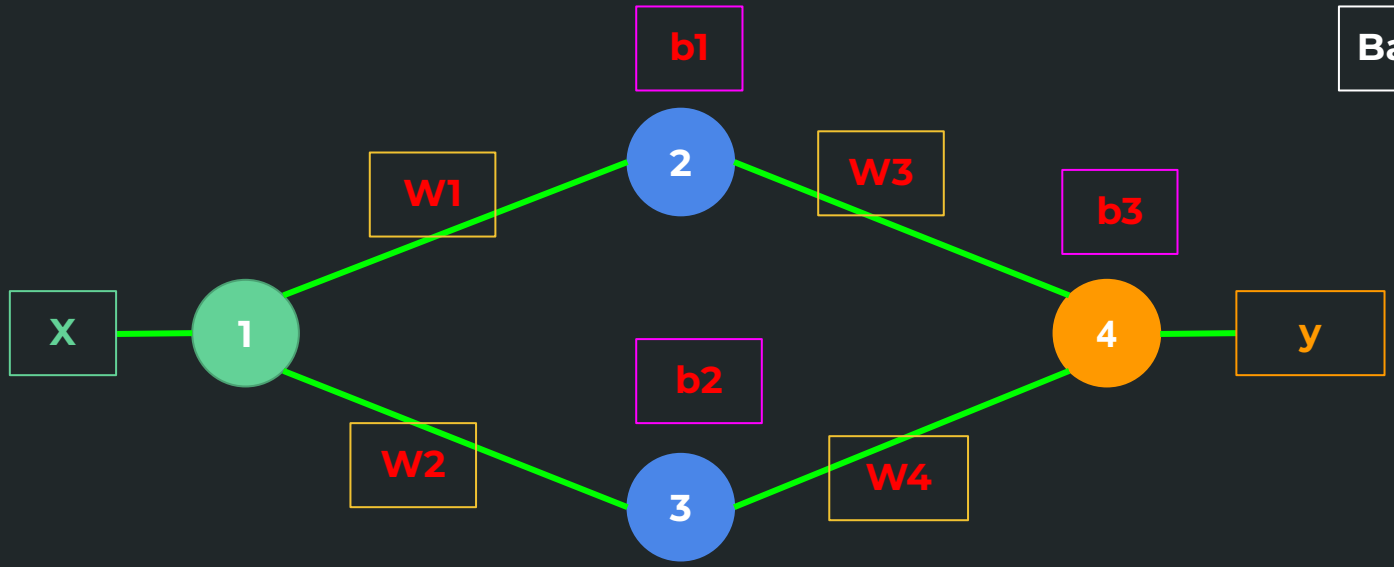
Artificial Neural Network

~~# Morphology of an Artificial Neuron~~
~~# Activation Function~~
~~# A simple Artificial Neural Network~~
~~# Forward Propagation~~
~~# Gradient Descent~~
Backpropagation
Regularisation (if possible)

Backpropagation

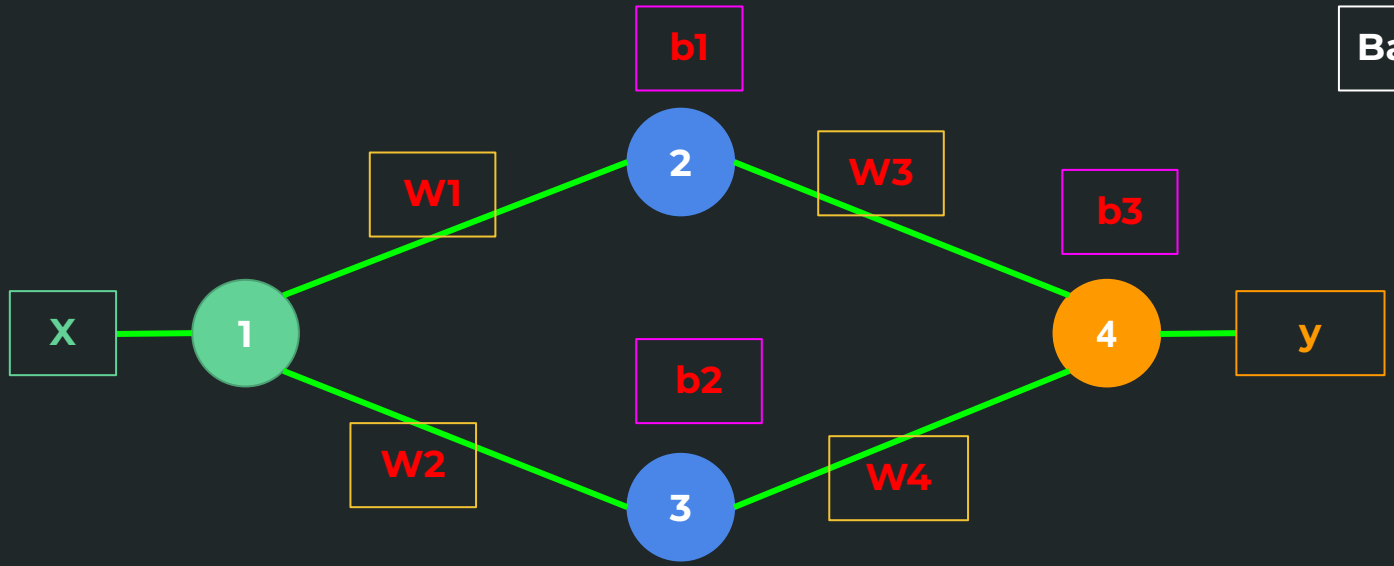


Backpropagation



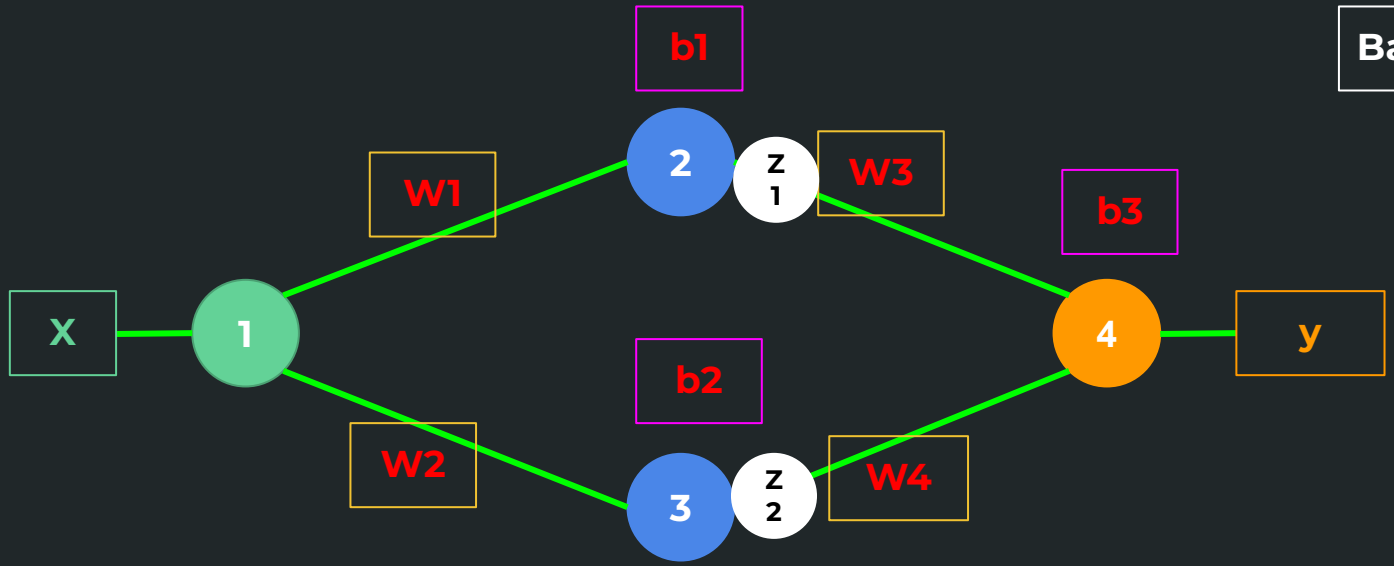
- Choose the weights and biases randomly
- Forward propagate the input and get the output (irrespective of accuracy)

Backpropagation



$$y(X , W1 , W2 , W3 , W4 , W5 , W6 , b1 , b2 , b3)$$

Backpropagation



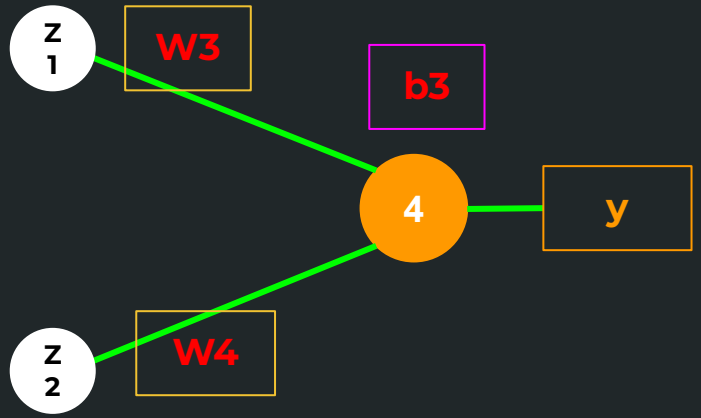
$$y(X , W1 , W2 , W3 , W4 , W5 , W6 , b1 , b2 , b3)$$

$$y(W3 , W4 , b3 , Z1 , Z2)$$

$$Z1 = \sigma(XW1 + b1)$$

$$Z2 = \sigma(XW2 + b2)$$

Backpropagation

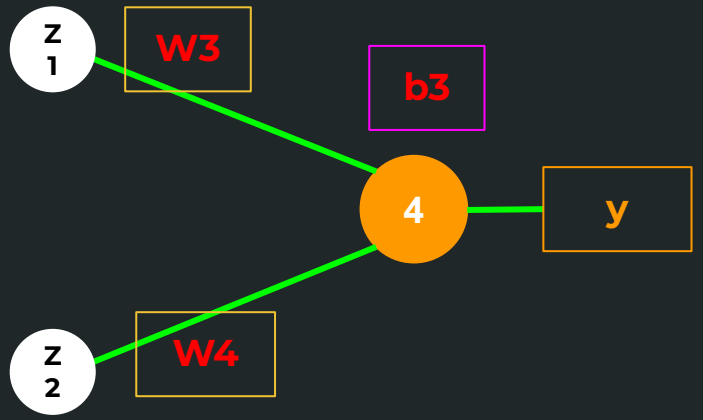


$$y(W3 , W4 , b3 , Z1 , Z2)$$

$$Z1 = \sigma(XW1 + b1)$$

$$Z2 = \sigma(XW2 + b2)$$

Backpropagation



$$y(W3 , W4 , b3 , Z1 , Z2)$$

$$ERROR = (T - y)$$

$$SSR = (T - y)^2$$

$$SSR = (T - \sigma(Z1W3 + Z2W4 + b3))^2$$

$$Z1 = \sigma(XW1 + b1)$$

$$Z2 = \sigma(XW2 + b2)$$

Backpropagation

$$SSR = (T - \sigma(Z1W3 + Z2W4 + b3))^2$$

$$\frac{\partial(SSR)}{\partial W3} = 2.(T - \sigma(Z1W3 + Z2W4 + b3)).(-\sigma'(Z1W3 + Z2W4 + b3)).Z1$$

$$\frac{\partial(SSR)}{\partial W3} = -2.(ERROR).(\sigma'(Z1W3 + Z2W4 + b3)).Z1$$

$$SSR = (T - \sigma(Z1W3 + Z2W4 + b3))^2$$

Backpropagation

$$\frac{\partial(SSR)}{\partial W3} = -2.(ERROR).(\sigma'(Z1W3 + Z2W4 + b3)).Z1$$

$$\frac{\partial(SSR)}{\partial W4} = -2.(ERROR).(\sigma'(Z1W3 + Z2W4 + b3)).Z2$$

$$\frac{\partial(SSR)}{\partial b3} = -2.(ERROR).(\sigma'(Z1W3 + Z2W4 + b3)).1$$

$$W3 = W3 - LR . \frac{\partial(SSR)}{\partial W3}$$

$$W4 = W4 - LR . \frac{\partial(SSR)}{\partial W4}$$

$$b3 = b3 - LR . \frac{\partial(SSR)}{\partial b3}$$

Backpropagation

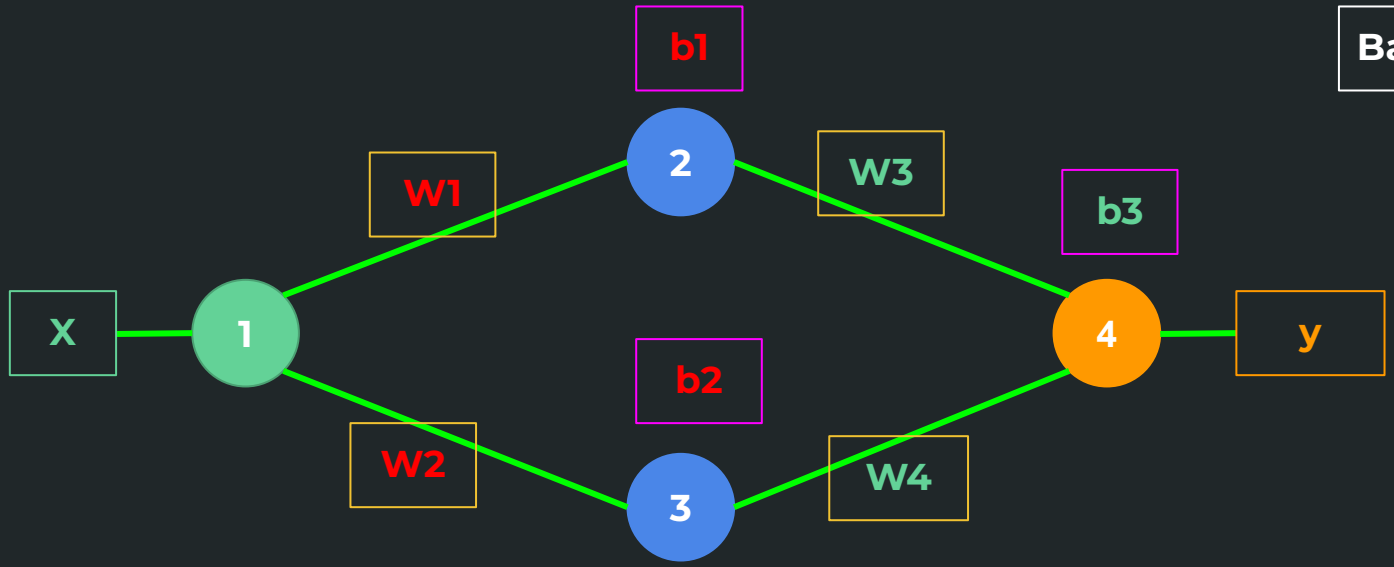
$$\frac{\partial(SSR)}{\partial W} = -2 \cdot (\text{ERROR}) \cdot \sigma'(\dots) \cdot (\text{I/P Corros. to the weight})$$

$$\frac{\partial(SSR)}{\partial b} = -2 \cdot (\text{ERROR}) \cdot \sigma'(\dots) \cdot 1$$

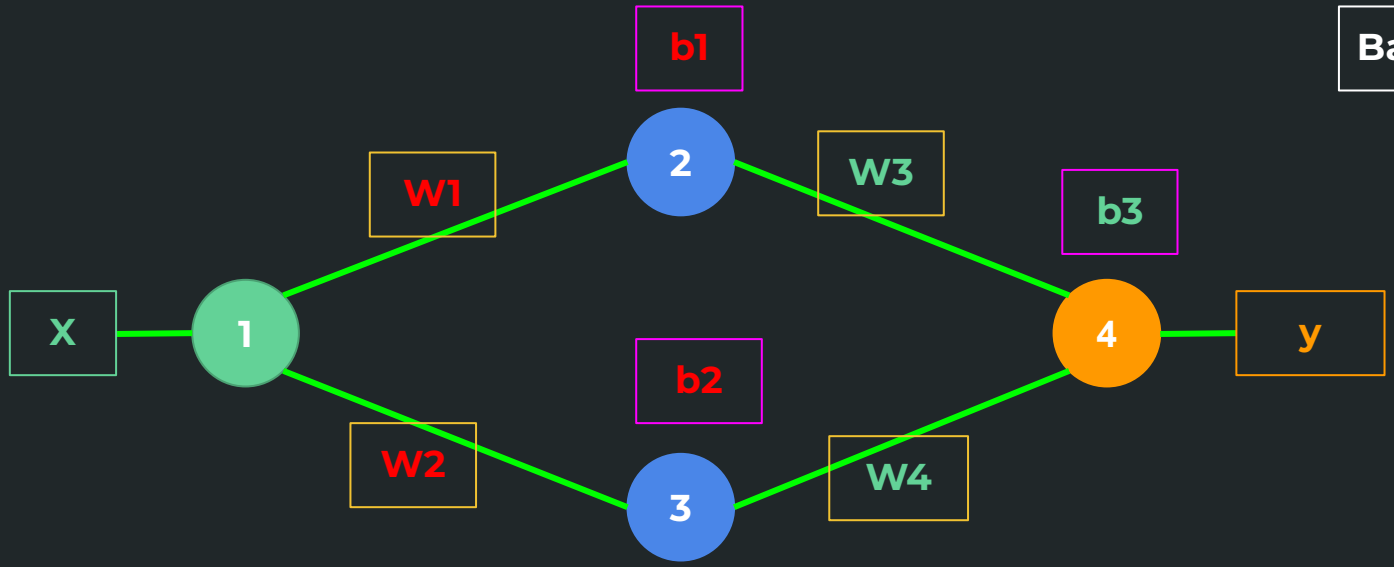
$$W = W - LR \cdot \frac{\partial(SSR)}{\partial W}$$

$$b = b - LR \cdot \frac{\partial(SSR)}{\partial b}$$

Backpropagation

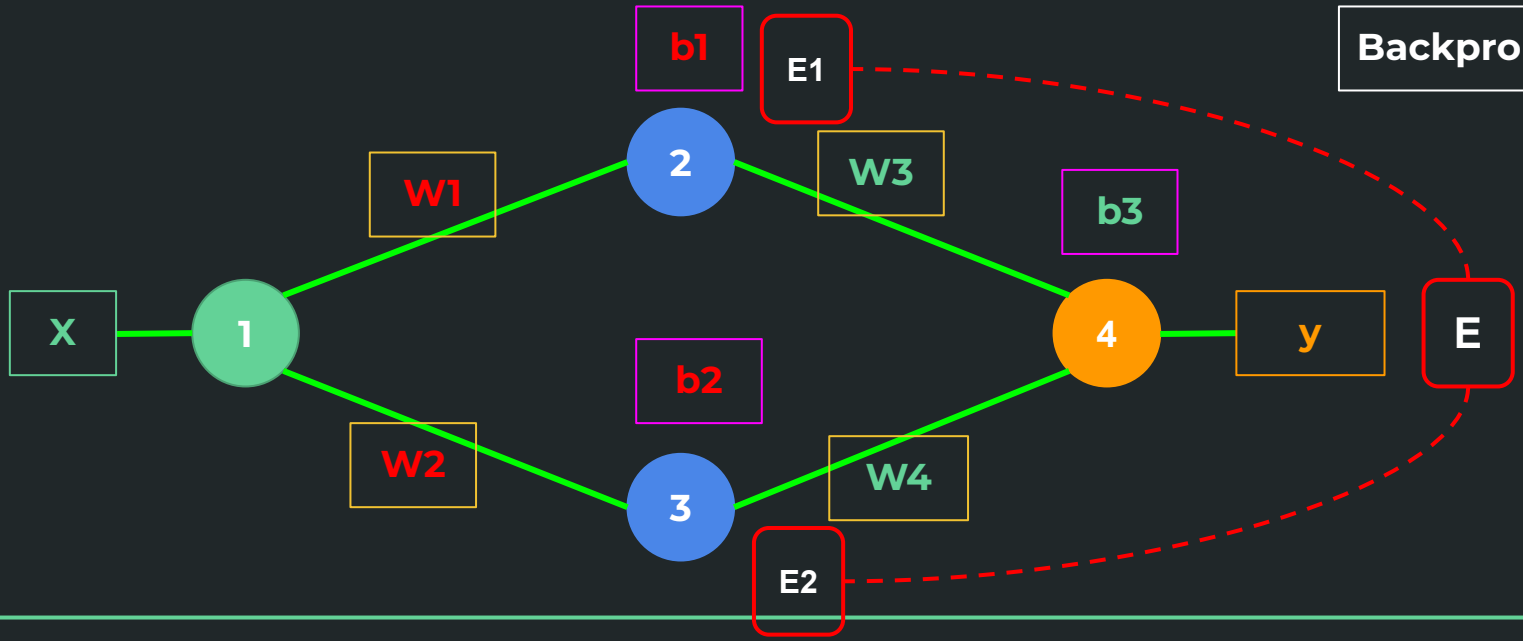


Backpropagation



ERROR = ?

Backpropagation



$$E1 = \frac{W3}{W3 + W4} E \qquad E2 = \frac{W4}{W3 + W4} E$$