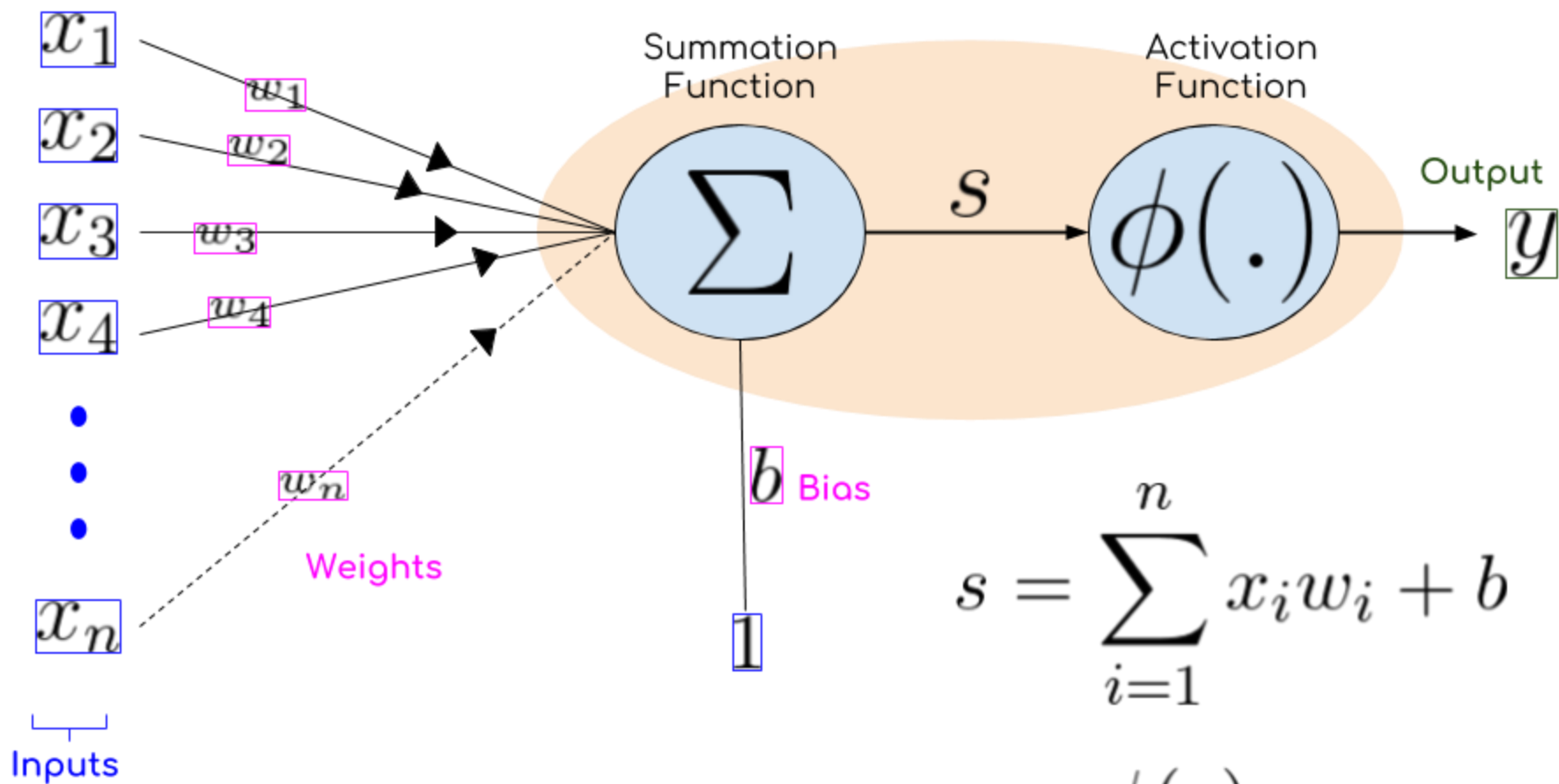


NEURAL NETWORKS

BACKPROPAGATION, GENERAL CONSIDERATIONS

Ravi Kiran
CVIT, IIIT Hyderabad

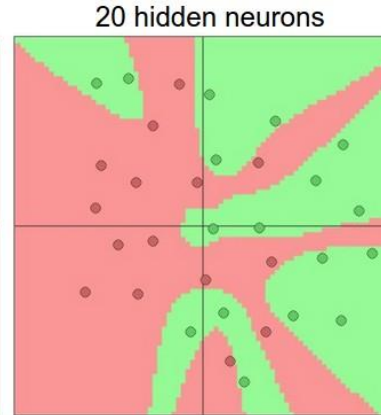
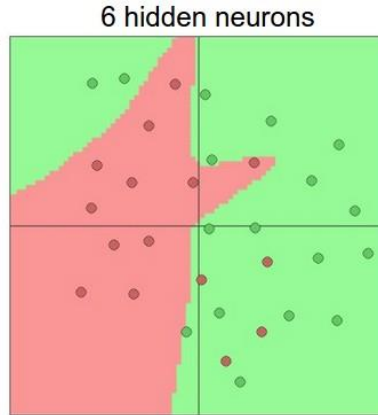
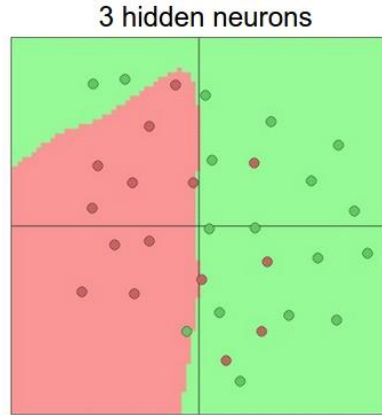
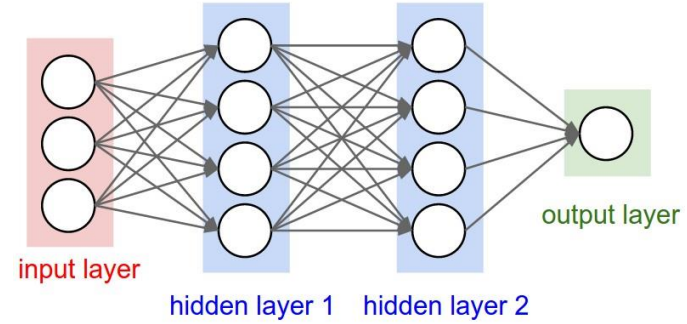
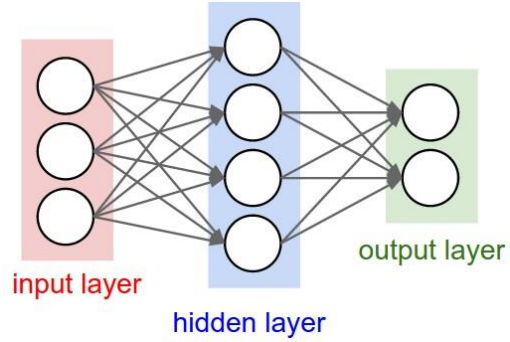
19.02.2019



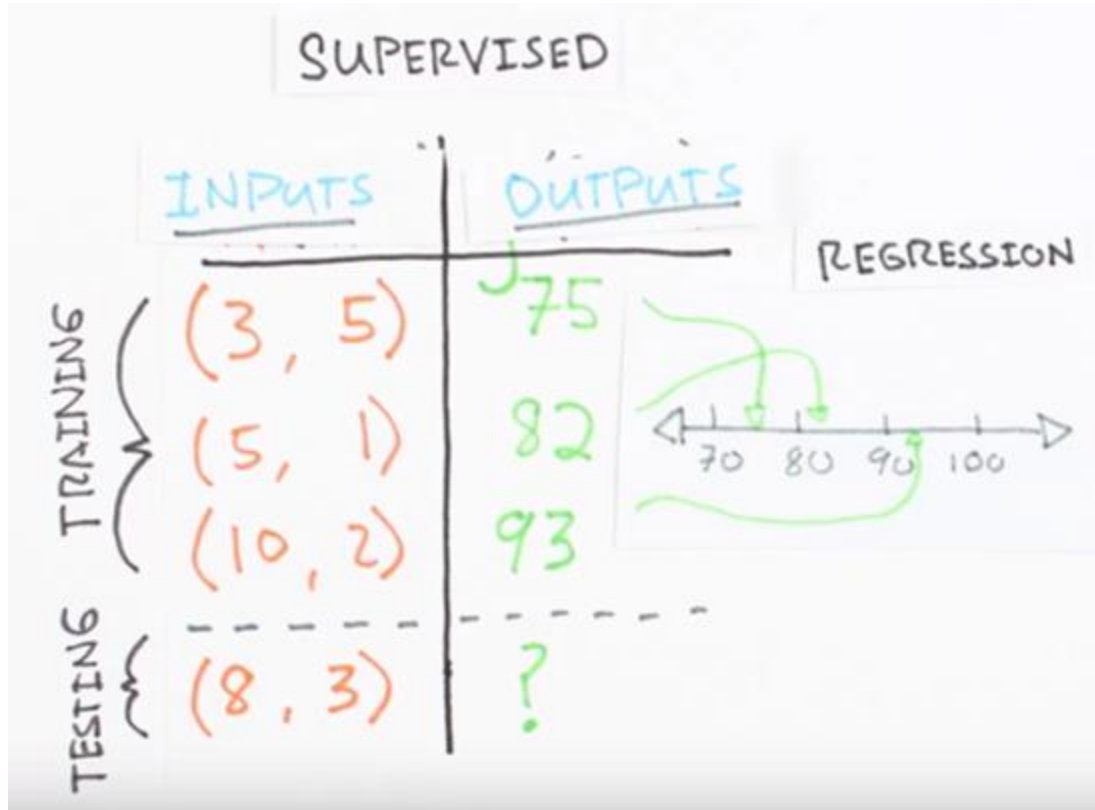
$$s = \sum_{i=1}^n x_i w_i + b$$

$$y = \phi(s)$$

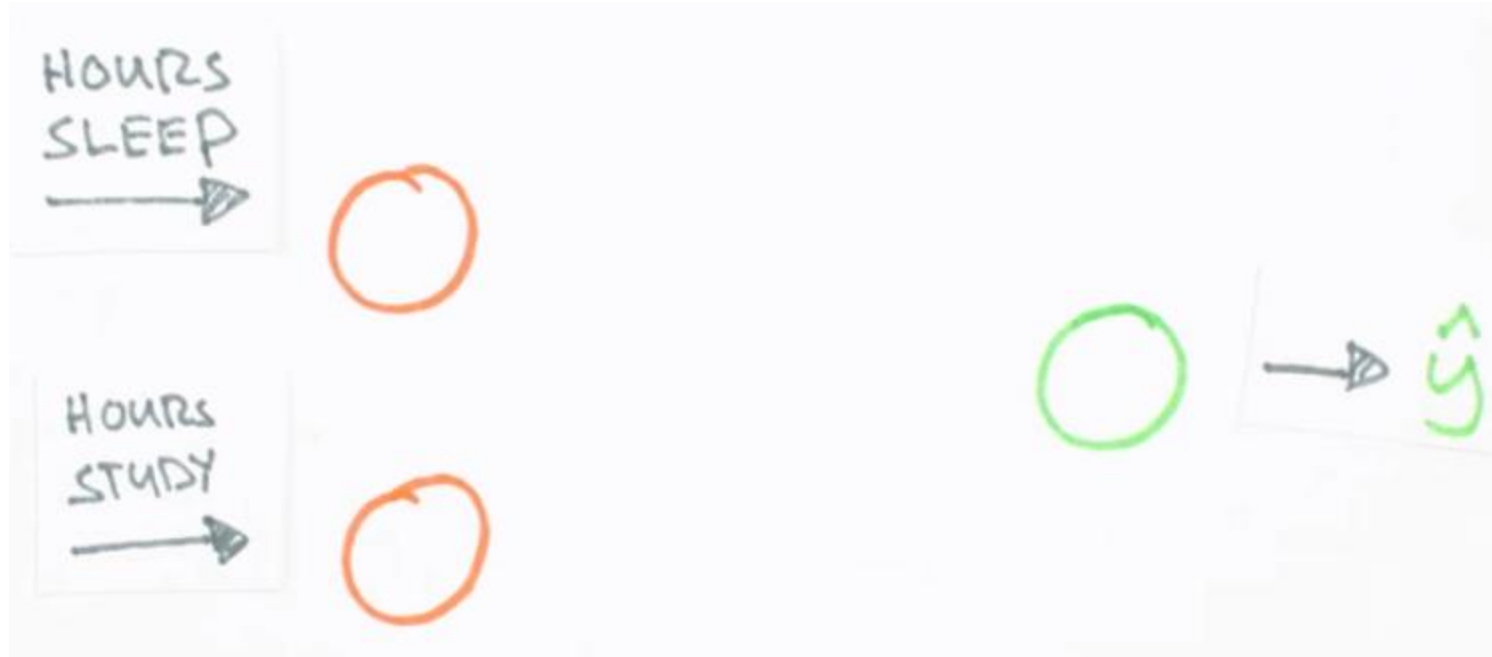
WHY USE ONLY ONE NEURON ?



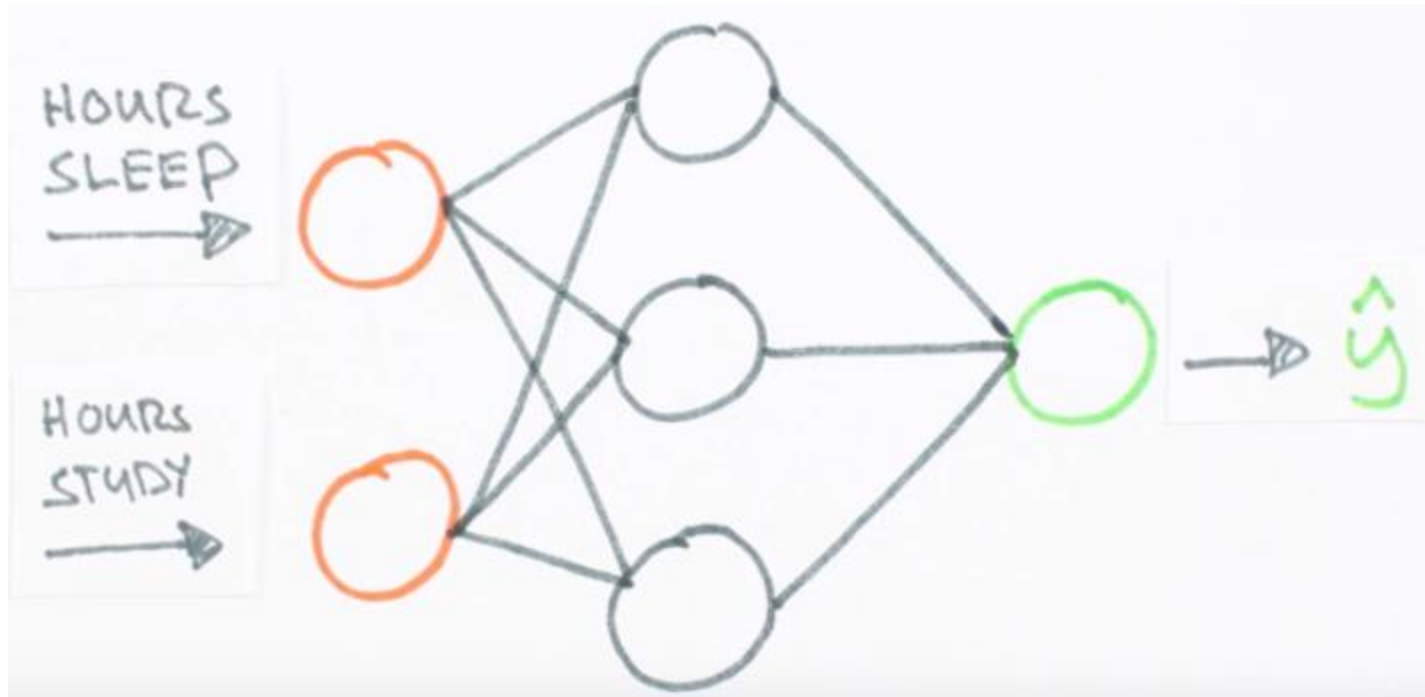
MULTI-NEURON NETWORKS



MULTI-NEURON NETWORKS :: ARCHITECTURE

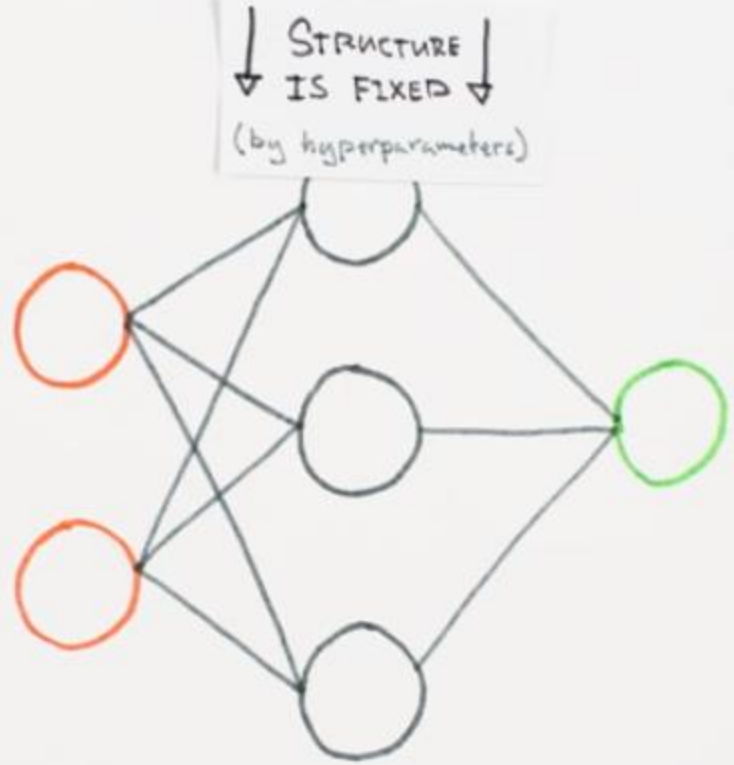


MULTI-NEURON NETWORKS :: ARCHITECTURE



MULTI-NEURON NETWORKS :: ARCHITECTURE

```
In [ ]: class Neural_Network(object):  
        def __init__(self):  
            #Define HyperParameters  
            self.inputLayerSize = 2  
            self.outputLayerSize = 1  
            self.hiddenLayerSize = 3
```



Gradient Descent

1. Initialize the parameters \mathbf{w} to some guess (usually all zeros, or random values)
2. Update the parameters:
$$\mathbf{w} = \mathbf{w} - \eta \nabla L(\mathbf{w})$$
3. Update the learning rate η
4. Repeat steps 2-3 until $\nabla L(\mathbf{w})$ is close to zero.

Stopping Criteria

Stop when the norm of the gradient is below some threshold, θ :

$$\|\nabla L(\mathbf{w})\| < \theta$$

Common values of θ are around .01, but if it is taking too long, you can make the threshold larger.

MULTI-NEURON NETWORKS :: TRAINING

INITIALIZE NETWORK WITH RANDOM WEIGHTS

Initialize weights w ; // Random or 0

WHILE [NOT CONVERGED]

DO FORWARD PROP

DO BACKPROP AND DETERMINE CHANGE IN WEIGHTS

UPDATE ALL WEIGHTS IN ALL LAYERS

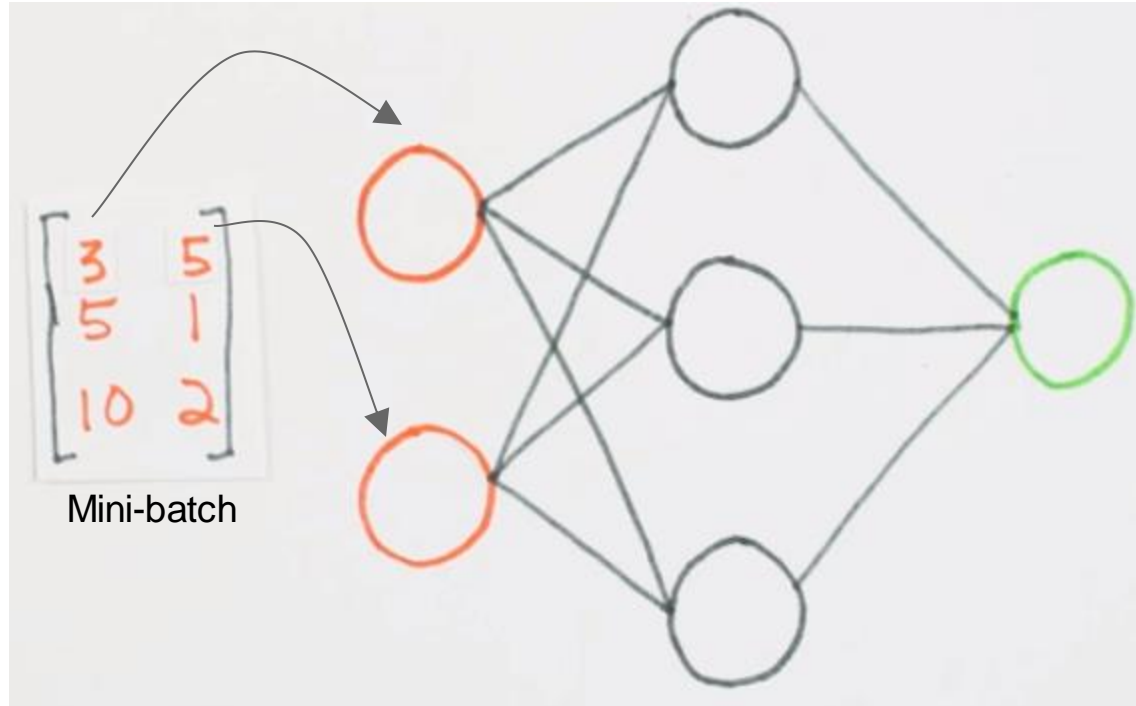
```
Until [all examples correctly classified]
  For each training sample ( $x, y$ )
    Compute  $yt := x^T w$ 
    if  $y == yt$  // Correctly classified
      continue ;
    else // Update weights
       $dw = (y - yt) * x$  ;
       $w = w + dw$  ;
    EndIf
  EndFor
EndUntil
```

MULTI-NEURON NETWORKS :: FORWARD PROPAGATION

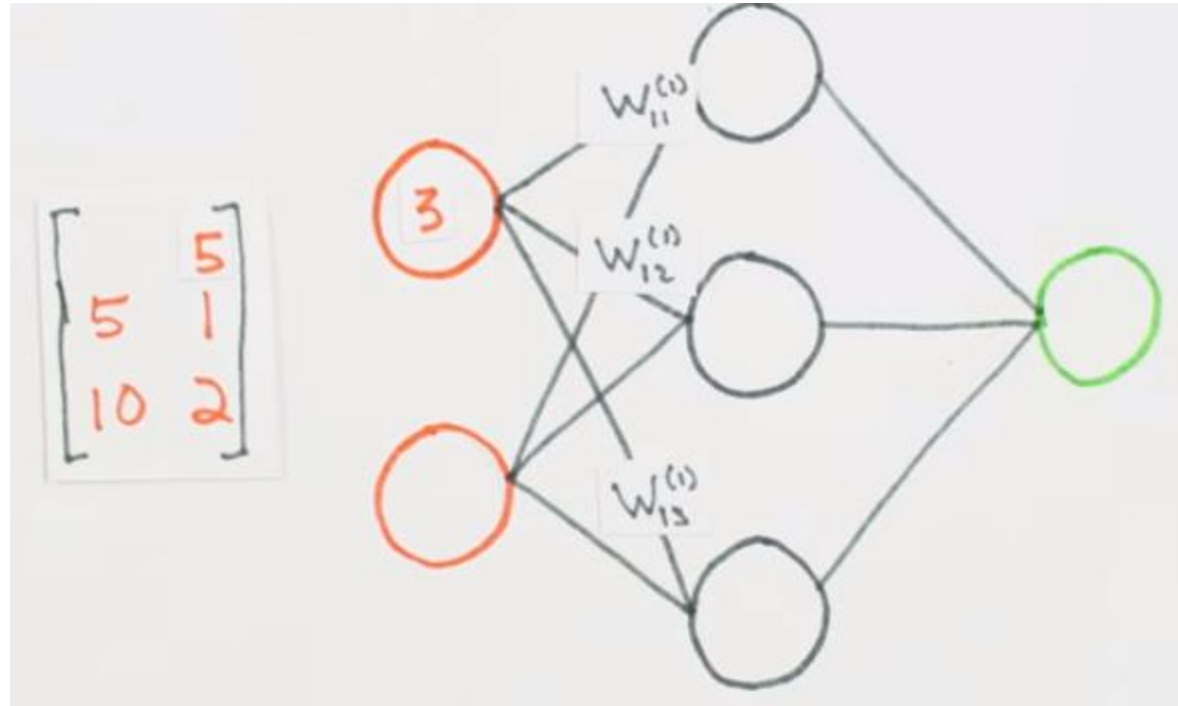
```
In [1]: class Neural_Network(object):  
        def __init__(self):  
            #Define HyperParameters  
            self.inputLayerSize = 2  
            self.outputLayerSize = 1  
            self.hiddenLayerSize = 3  
  
        def forward(self, X):  
            #Propagate inputs through network
```



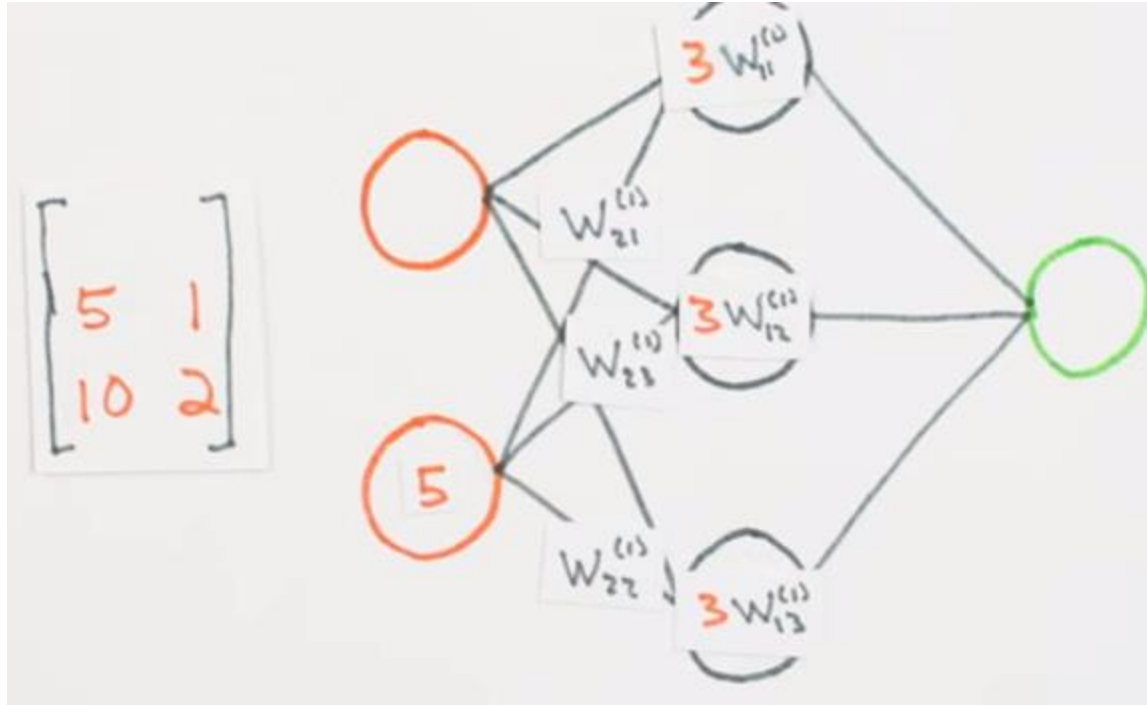
MULTI-NEURON NETWORKS :: FORWARD PROPAGATION



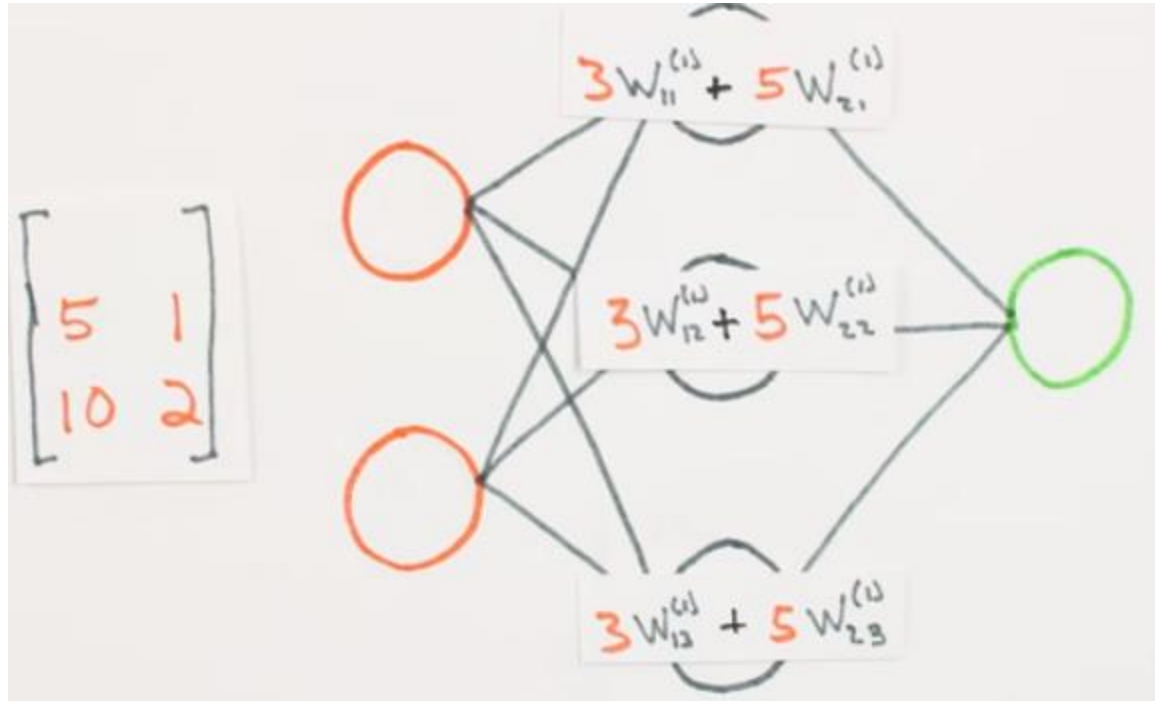
MULTI-NEURON NETWORKS :: FORWARD PROPAGATION



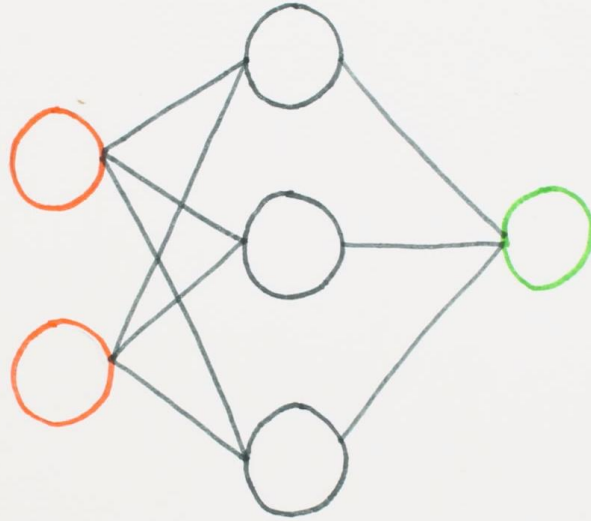
MULTI-NEURON NETWORKS :: FORWARD PROPAGATION



MULTI-NEURON NETWORKS :: FORWARD PROPAGATION

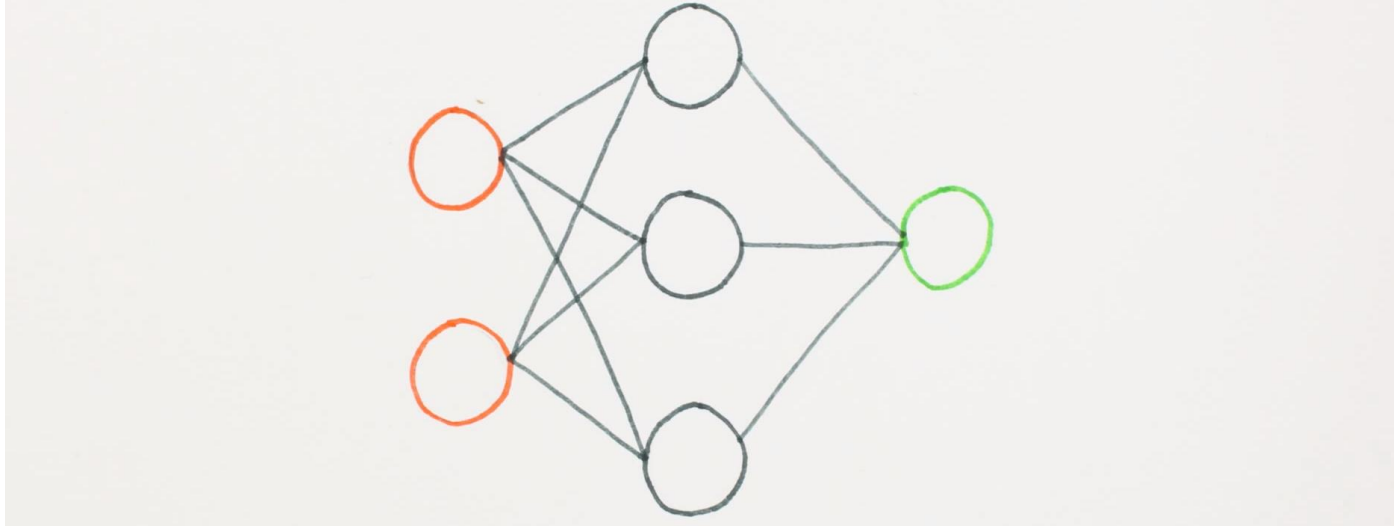


MULTI-NEURON NETWORKS :: FORWARD PROPAGATION



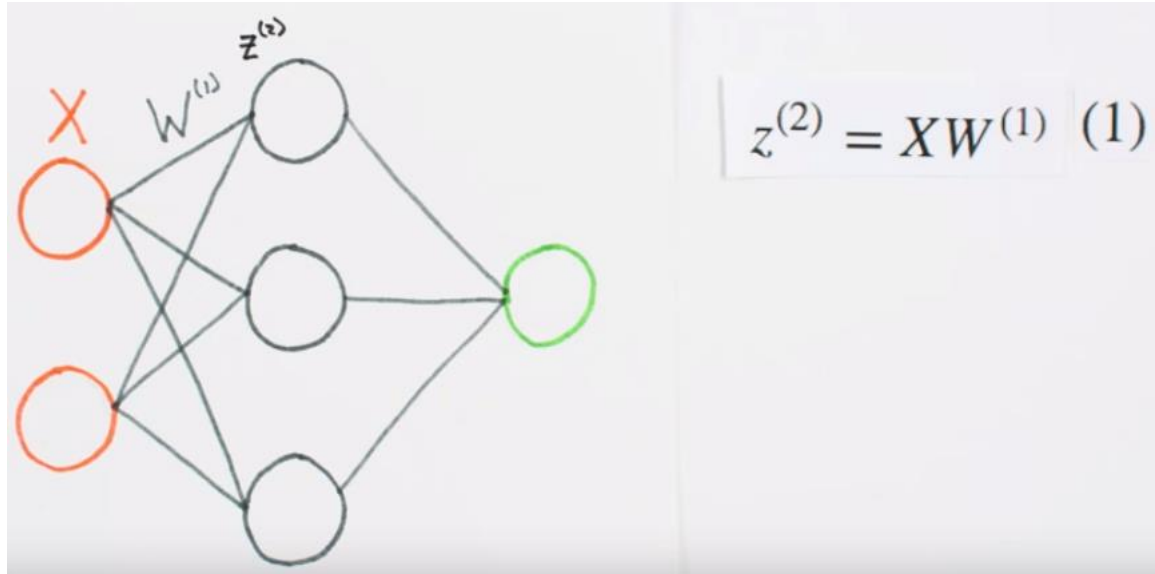
$$\begin{bmatrix} 3 & 5 \\ 5 & 1 \\ 10 & 2 \end{bmatrix} \begin{bmatrix} W_{11}^{(1)} & W_{12}^{(1)} & W_{13}^{(1)} \\ W_{21}^{(1)} & W_{22}^{(1)} & W_{23}^{(1)} \end{bmatrix} = \begin{bmatrix} 3W_{11}^{(1)} + 5W_{21}^{(1)} & 3W_{12}^{(1)} + 5W_{22}^{(1)} & 3W_{13}^{(1)} + 5W_{23}^{(1)} \\ 5W_{11}^{(1)} + 1W_{21}^{(1)} & 5W_{12}^{(1)} + 1W_{22}^{(1)} & 5W_{13}^{(1)} + 1W_{23}^{(1)} \\ 10W_{11}^{(1)} + 2W_{21}^{(1)} & 10W_{12}^{(1)} + 2W_{22}^{(1)} & 10W_{13}^{(1)} + 2W_{23}^{(1)} \end{bmatrix}$$

MULTI-NEURON NETWORKS :: FORWARD PROPAGATION

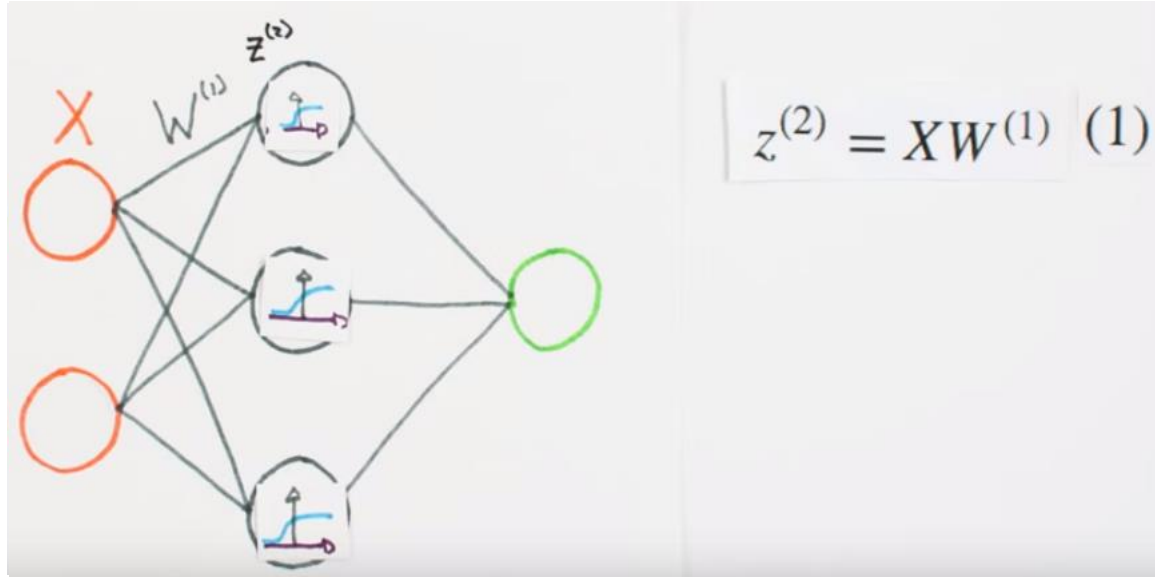


X	$W^{(1)}$	=	$Z^{(2)}$
---	-----------	---	-----------

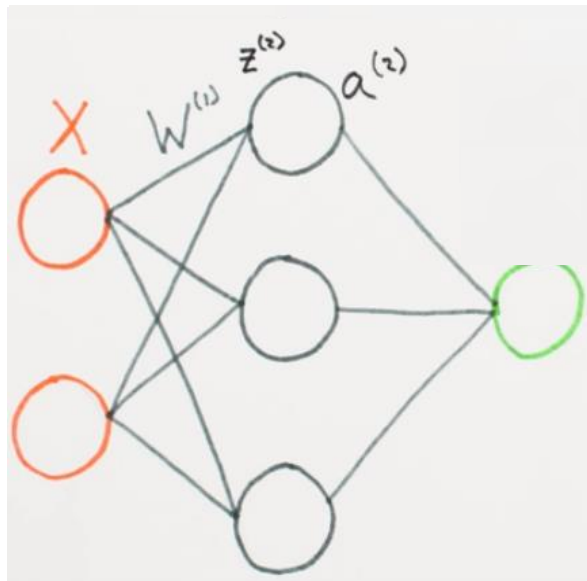
MULTI-NEURON NETWORKS :: FORWARD PROPAGATION



MULTI-NEURON NETWORKS :: FORWARD PROPAGATION

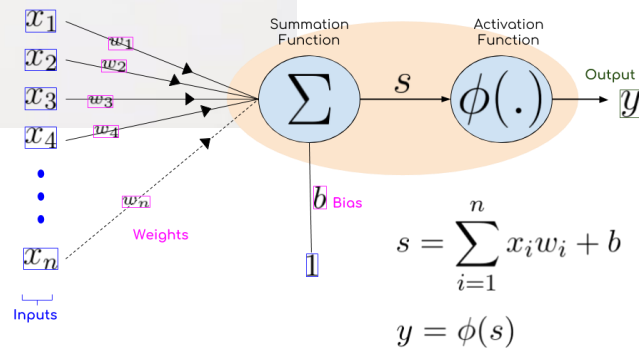


MULTI-NEURON NETWORKS :: FORWARD PROPAGATION

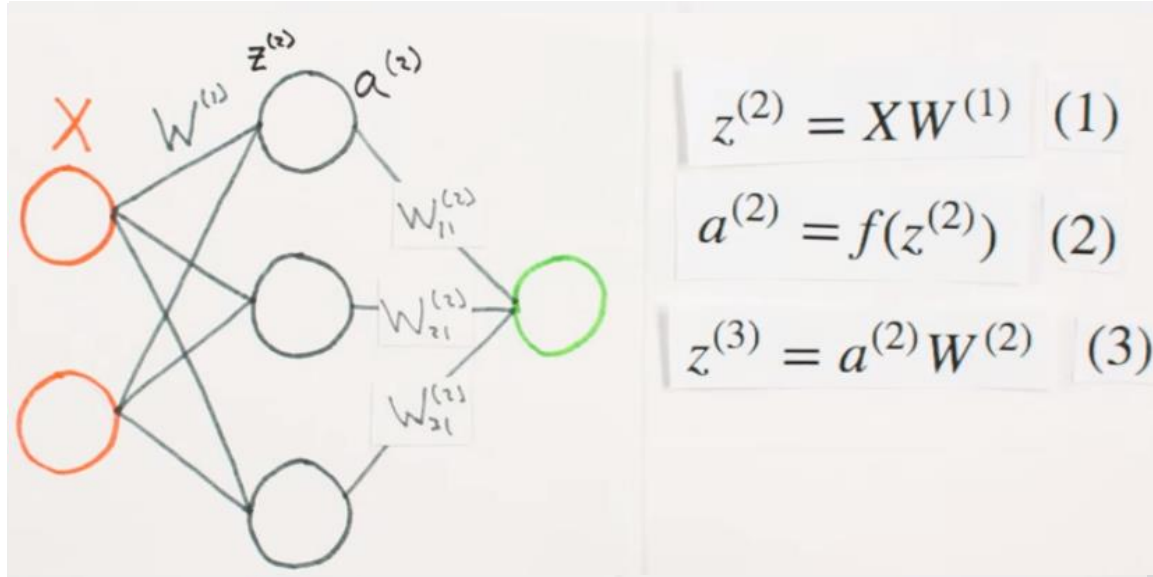


$$z^{(2)} = XW^{(1)} \quad (1)$$

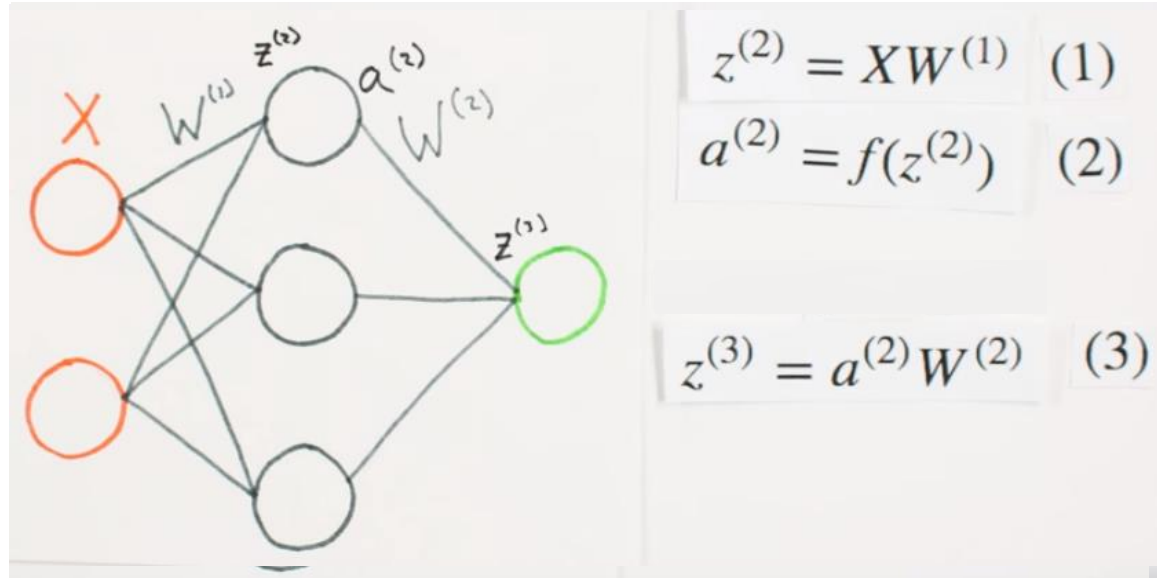
$$a^{(2)} = f(z^{(2)}) \quad (2)$$



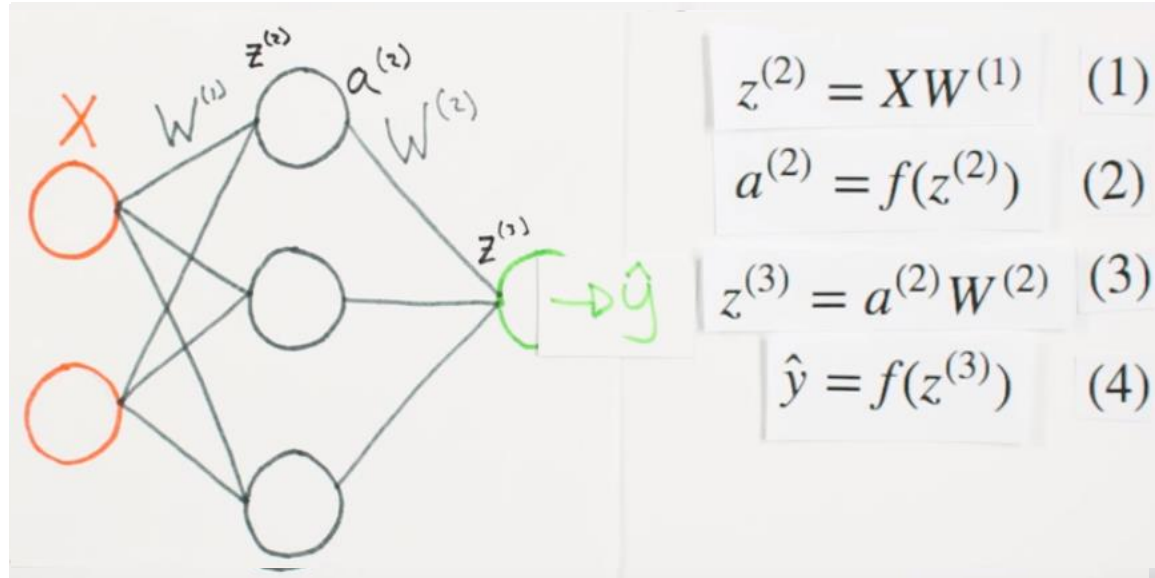
MULTI-NEURON NETWORKS :: FORWARD PROPAGATION



MULTI-NEURON NETWORKS :: FORWARD PROPAGATION



MULTI-NEURON NETWORKS :: FORWARD PROPAGATION



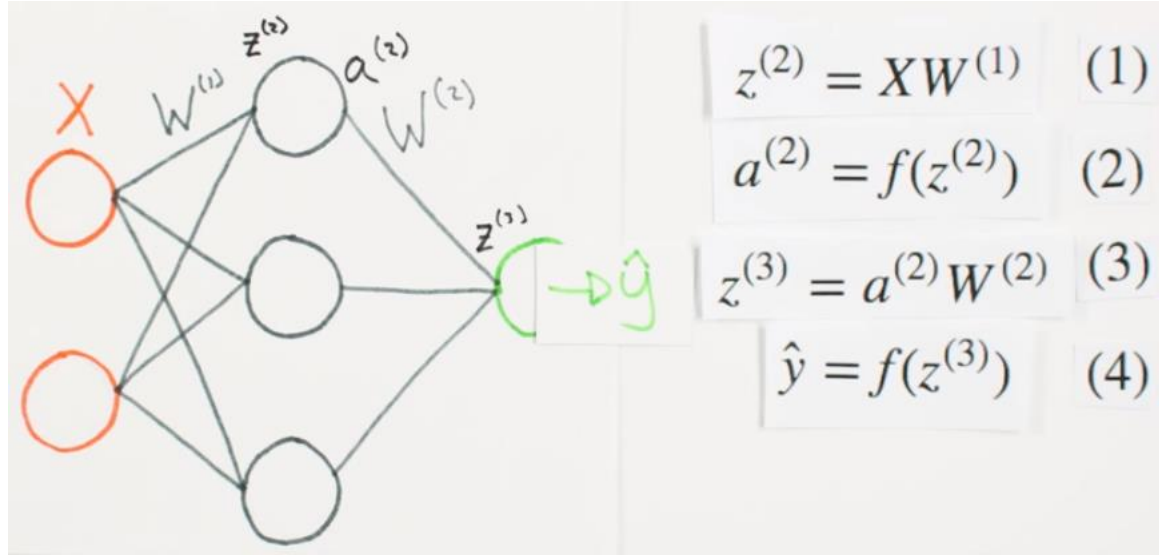
MULTI-NEURON NETWORKS :: FORWARD PROPAGATION

```
In [1]: class Neural_Network(object):
def __init__(self):
    #Define Hyperparameters
    self.inputLayerSize = 2
    self.outputLayerSize = 1
    self.hiddenLayerSize = 3

    #Weights (Parameters)
    self.W1 = np.random.randn(self.inputLayerSize, \
                                self.hiddenLayerSize)
    self.W2 = np.random.randn(self.hiddenLayerSize, \
                                self.outputLayerSize)

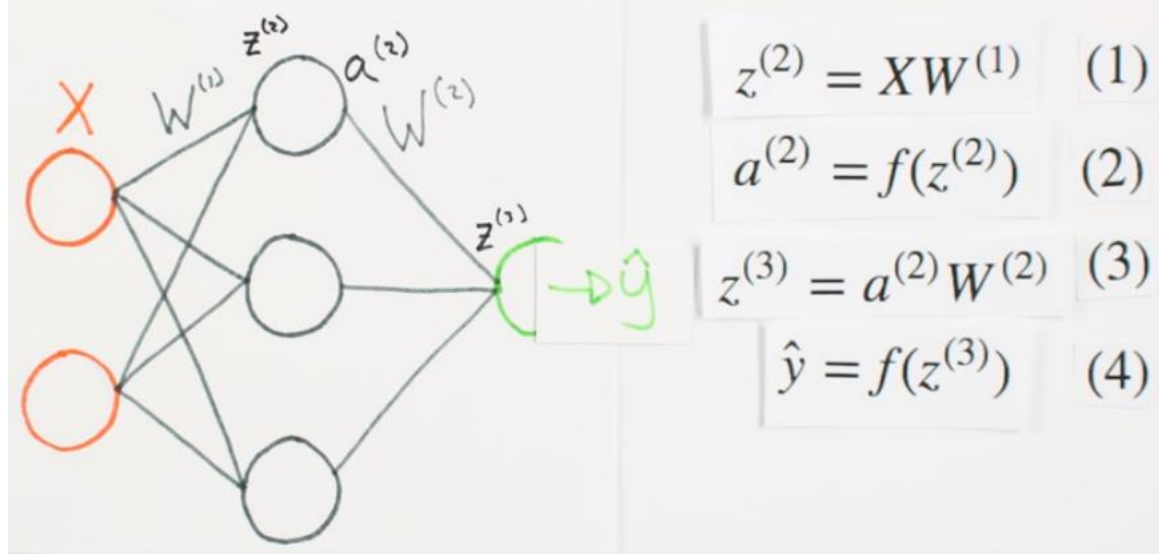
def forward(self, X):
    #Propagate inputs though network
    self.z2 = np.dot(X, self.W1)
    self.a2 = self.sigmoid(self.z2)
    self.z3 = np.dot(self.a2, self.W2)
    yHat = self.sigmoid(self.z3)
    return yHat

def sigmoid(self, z):
    #Apply sigmoid activation function to scalar, vector, or
    return 1/(1+np.exp(-z))
```

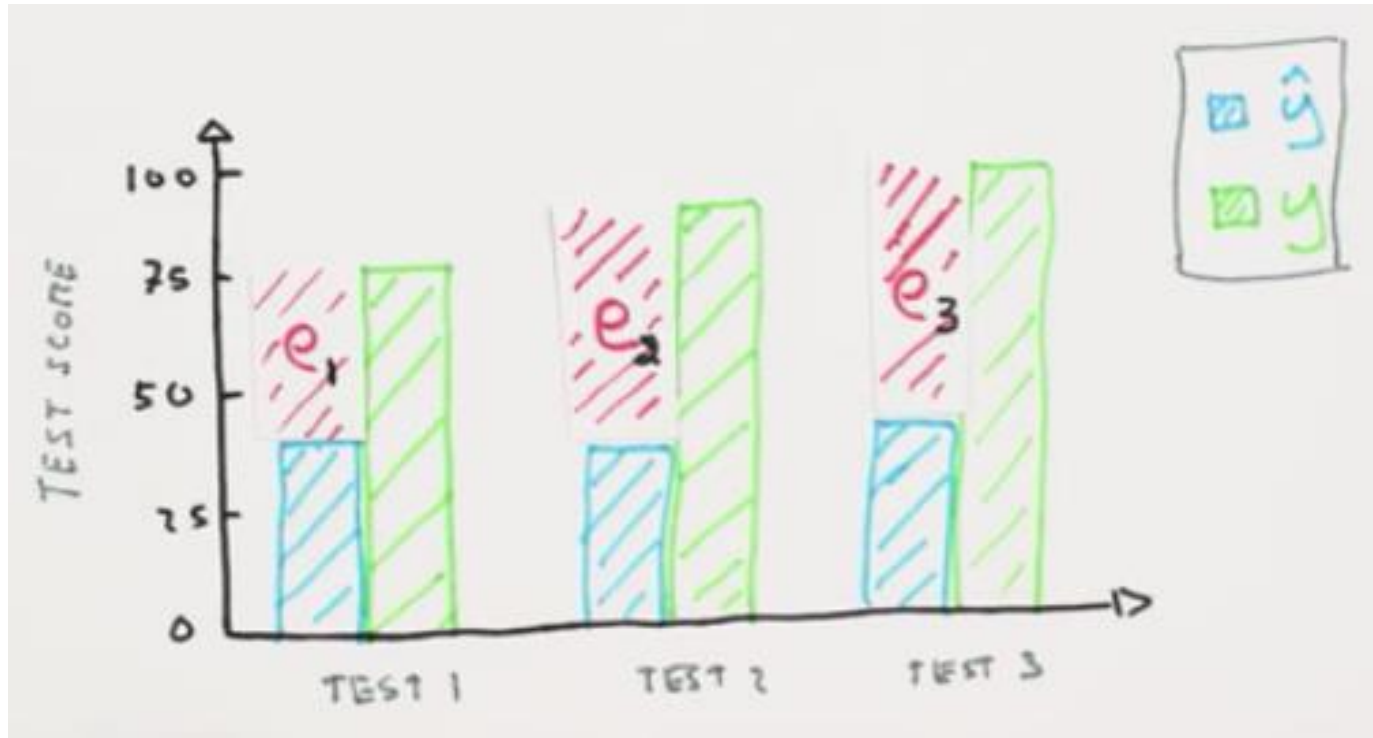


MULTI-NEURON NETWORKS :: FORWARD PROPAGATION

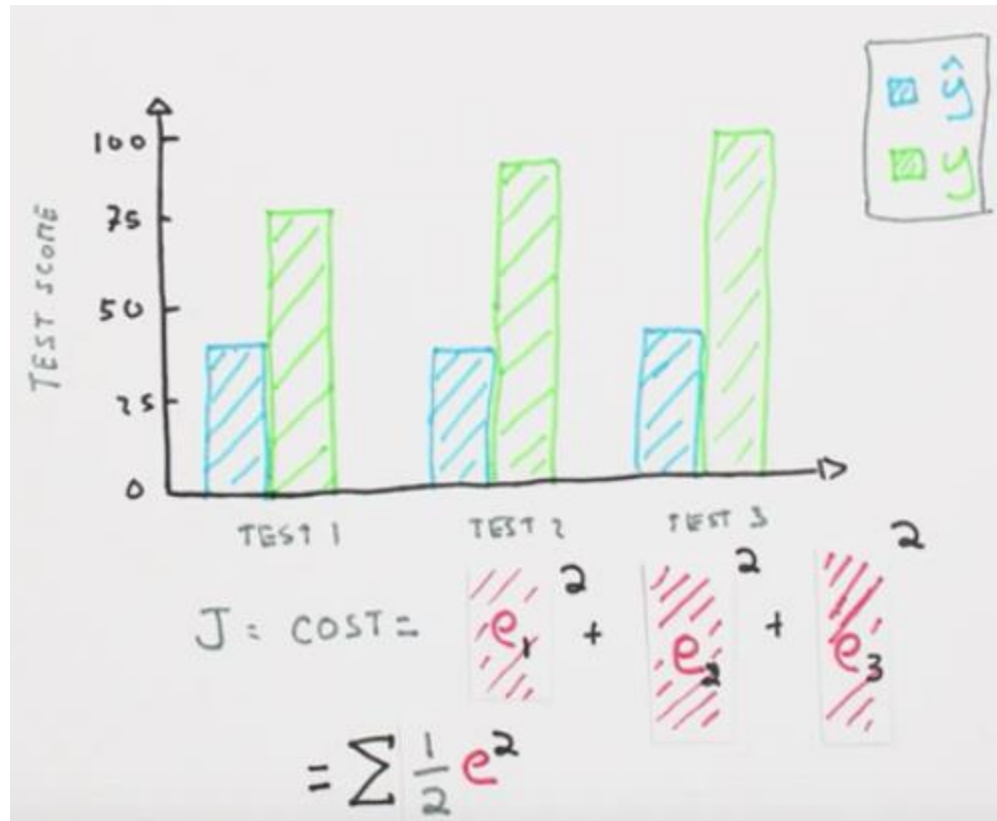
```
In [7]: NN = Neural_Network()
In [8]: yHat = NN.forward(X)
In [9]: yHat
Out[9]: array([[ 0.59470263],
               [ 0.58177822],
               [ 0.50641742]])
In [10]: y
Out[10]: array([[ 0.75],
                [ 0.82],
                [ 0.93]])
```



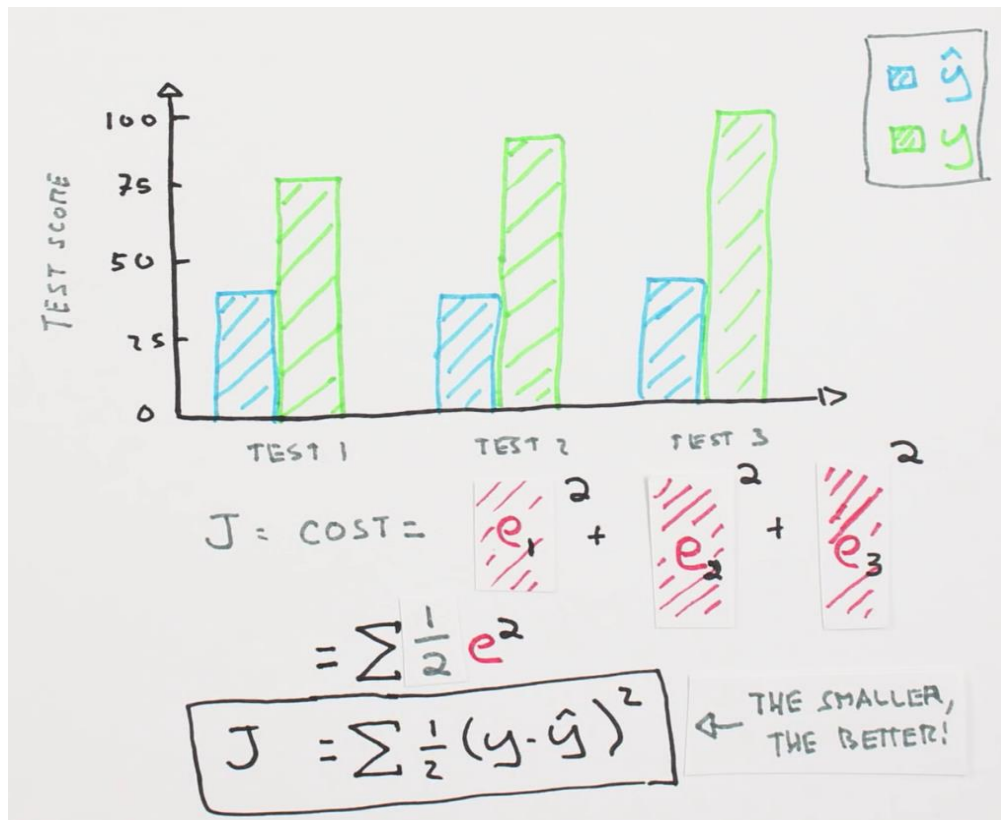
MULTI-NEURON NETWORKS :: GRADIENT DESCENT



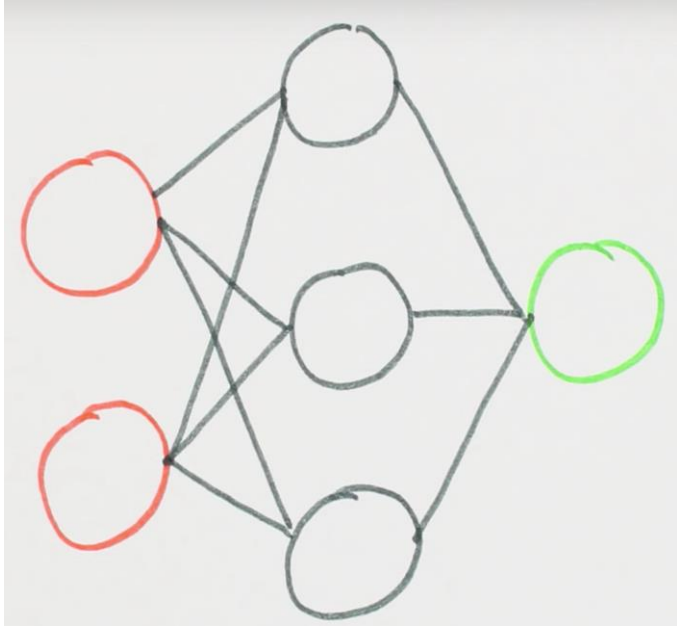
MULTI-NEURON NETWORKS :: GRADIENT DESCENT



MULTI-NEURON NETWORKS :: GRADIENT DESCENT

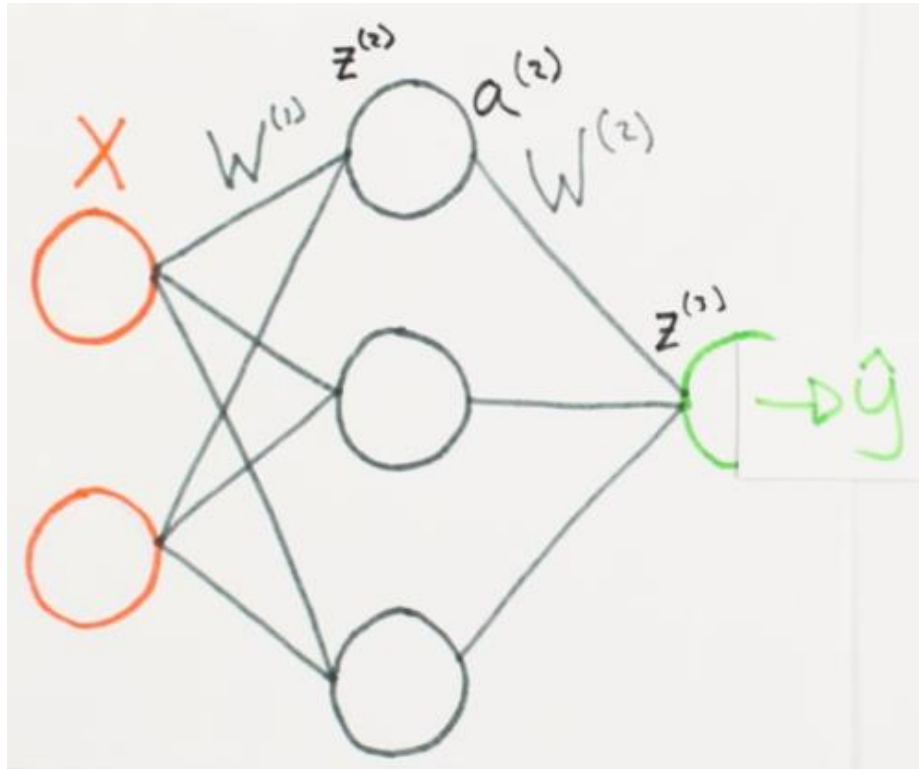


MULTI-NEURON NETWORKS :: GRADIENT DESCENT



Training a Network
=
Minimizing a Cost
Function

MULTI-NEURON NETWORKS :: GRADIENT DESCENT



$$z^{(2)} = XW^{(1)} \quad (1)$$

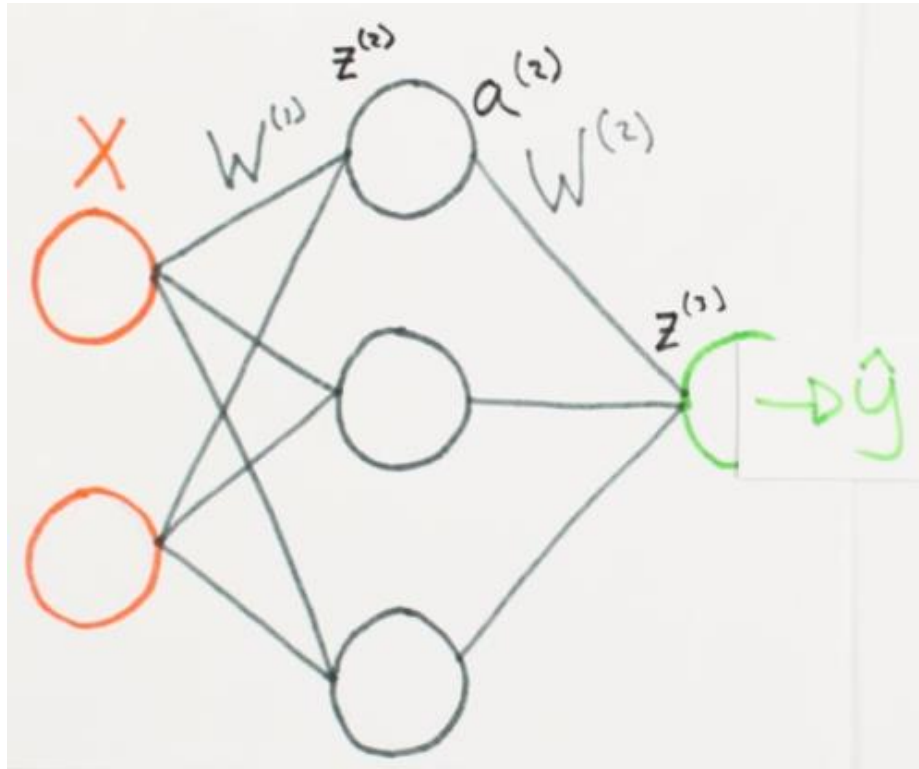
$$a^{(2)} = f(z^{(2)}) \quad (2)$$

$$z^{(3)} = a^{(2)}W^{(2)} \quad (3)$$

$$\hat{y} = f(z^{(3)}) \quad (4)$$

$$J = \sum \frac{1}{2} (y - \hat{y})^2 \quad (5)$$

MULTI-NEURON NETWORKS :: GRADIENT DESCENT



$$z^{(2)} = XW^{(1)} \quad (1)$$

$$a^{(2)} = f(z^{(2)}) \quad (2)$$

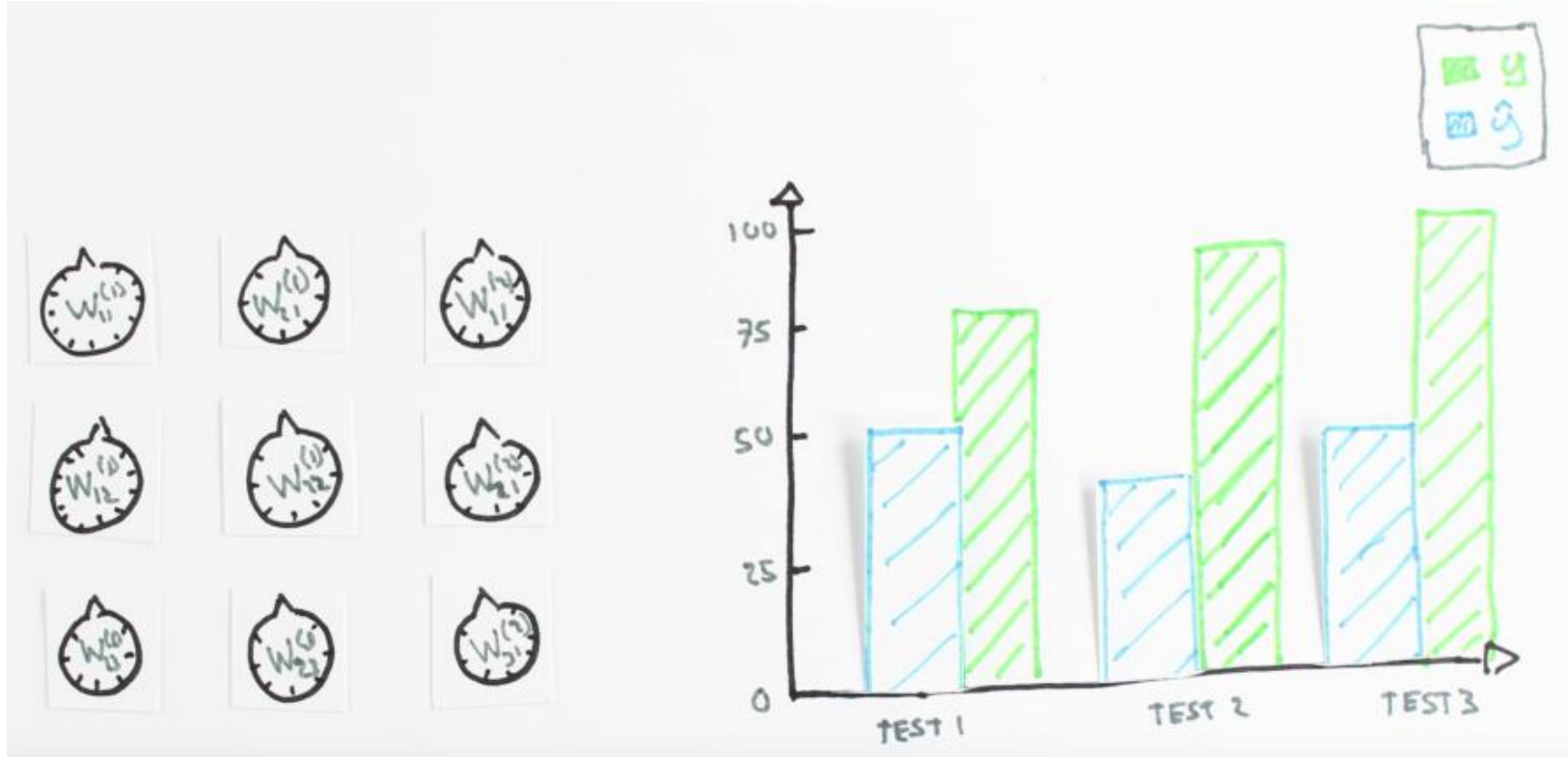
$$z^{(3)} = a^{(2)}W^{(2)} \quad (3)$$

$$\hat{y} = f(z^{(3)}) \quad (4)$$

$$J = \sum \frac{1}{2} (y - \hat{y})^2 \quad (5)$$

$$J = \sum \frac{1}{2} (y - f(f(XW^{(1)})W^{(2)}))^2$$

MULTI-NEURON NETWORKS :: GRADIENT DESCENT



MULTI-NEURON NETWORKS :: TRAINING

INITIALIZE NETWORK WITH RANDOM WEIGHTS

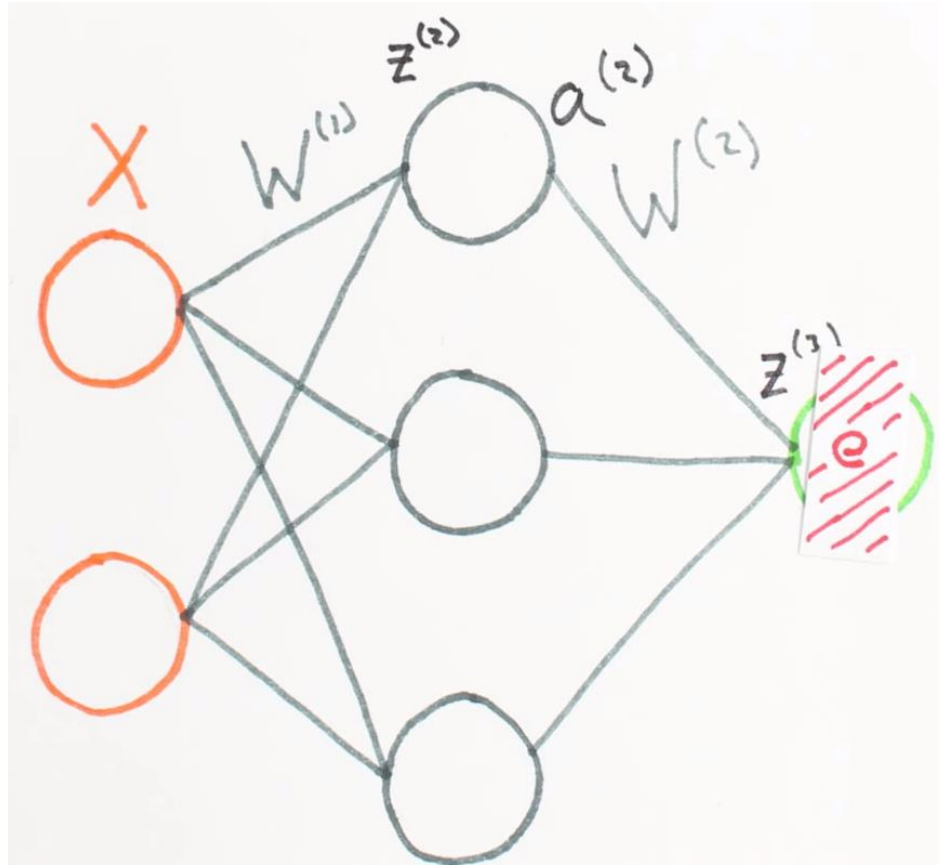
WHILE [NOT CONVERGED]

DO FORWARD PROP

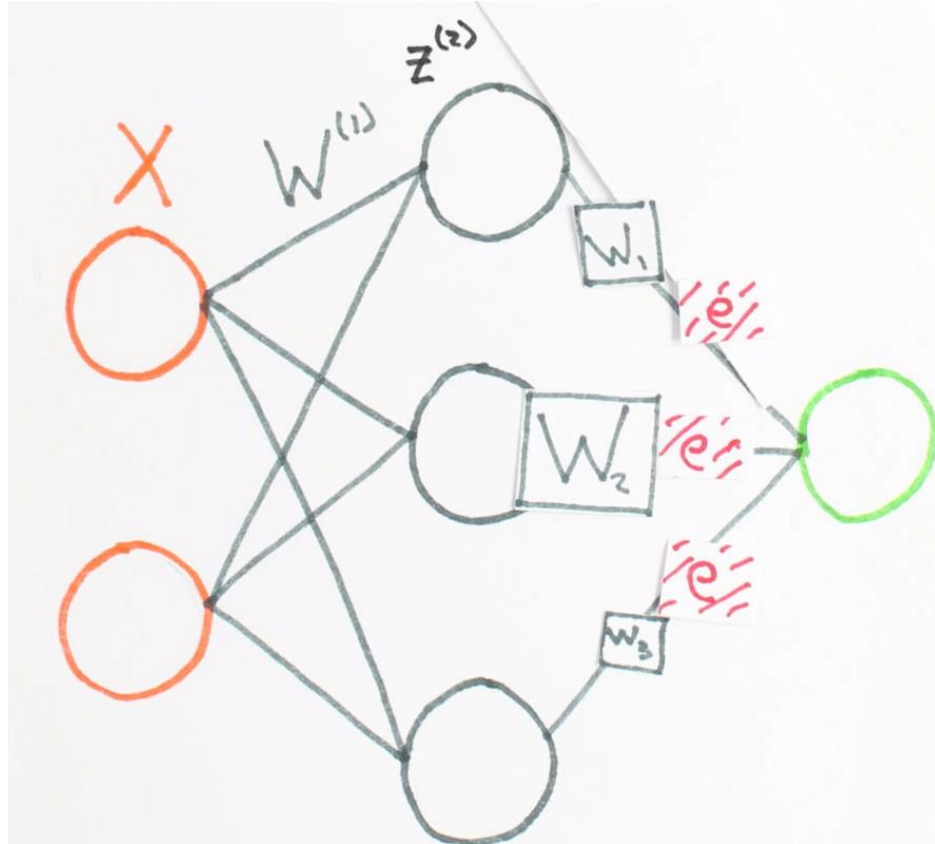
DO BACKPROP AND DETERMINE CHANGE IN WEIGHTS

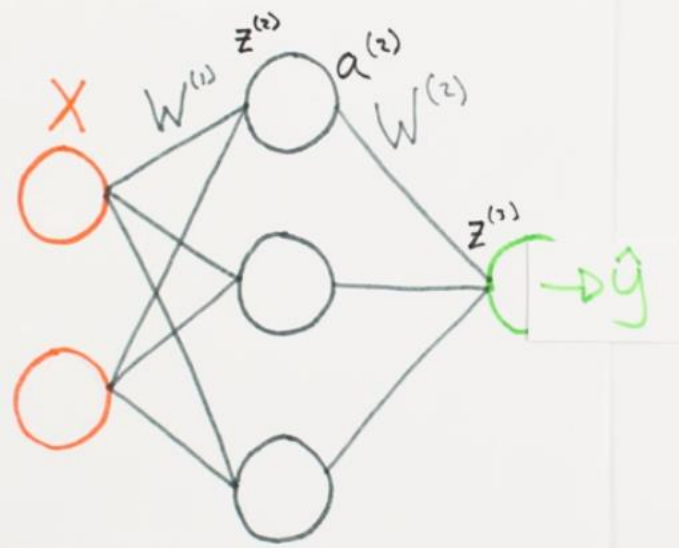
UPDATE ALL WEIGHTS IN ALL LAYERS

MULTI-NEURON NETWORKS :: BACKPROPAGATION



MULTI-NEURON NETWORKS :: BACKPROPAGATION





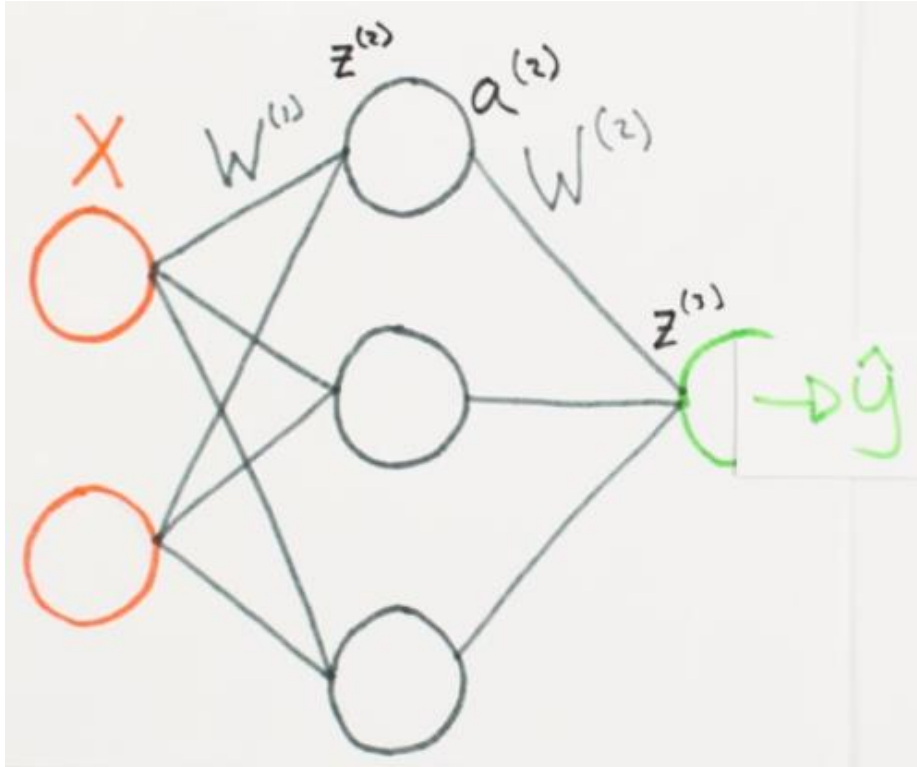
$$z^{(2)} = XW^{(1)} \quad (1)$$

$$a^{(2)} = f(z^{(2)}) \quad (2)$$

$$z^{(3)} = a^{(2)}W^{(2)} \quad (3)$$

$$\hat{y} = f(z^{(3)}) \quad (4)$$

MULTI-NEURON NETWORKS :: BACKPROPAGATION



$$J = \sum \frac{1}{2} (y - f(f(XW^{(1)})W^{(2)}))^2$$

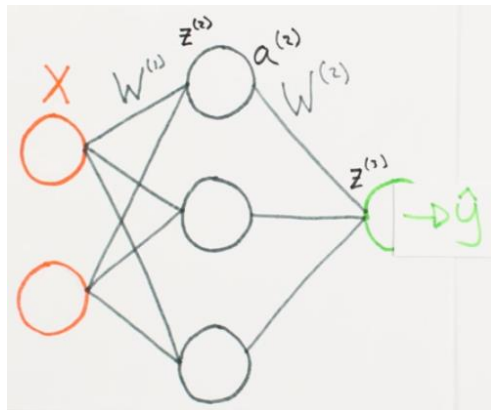
↑ HOW DOES THIS CHANGE IF I CHANGE THESE? ↑

$$\frac{\partial J}{\partial W}$$

$$W^{(1)} = \begin{bmatrix} W_{11}^{(1)} & W_{12}^{(1)} & W_{13}^{(1)} \\ W_{21}^{(1)} & W_{22}^{(1)} & W_{23}^{(1)} \end{bmatrix}$$

$$W^{(2)} = \begin{bmatrix} W_{11}^{(2)} \\ W_{21}^{(2)} \\ W_{31}^{(2)} \end{bmatrix}$$

MULTI-NEURON NETWORKS :: BACKPROPAGATION



$$J = \sum \frac{1}{2} (y - f(f(XW^{(1)})W^{(2)}))^2$$

↑ HOW DOES THIS CHANGE IF I CHANGE THESE? ↑

$$\frac{\partial J}{\partial W}$$

$$W^{(1)} = \begin{bmatrix} W_{11}^{(1)} & W_{12}^{(1)} & W_{13}^{(1)} \\ W_{21}^{(1)} & W_{22}^{(1)} & W_{23}^{(1)} \end{bmatrix}$$

$$W^{(2)} = \begin{bmatrix} W_{11}^{(2)} \\ W_{21}^{(2)} \\ W_{31}^{(2)} \end{bmatrix}$$

$$\frac{\partial J}{\partial W^{(1)}} = \begin{bmatrix} \frac{\partial J}{\partial W_{11}^{(1)}} & \frac{\partial J}{\partial W_{12}^{(1)}} & \frac{\partial J}{\partial W_{13}^{(1)}} \\ \frac{\partial J}{\partial W_{21}^{(1)}} & \frac{\partial J}{\partial W_{22}^{(1)}} & \frac{\partial J}{\partial W_{23}^{(1)}} \end{bmatrix}$$

$$\frac{\partial J}{\partial W^{(2)}} = \begin{bmatrix} \frac{\partial J}{\partial W_{11}^{(2)}} \\ \frac{\partial J}{\partial W_{21}^{(2)}} \\ \frac{\partial J}{\partial W_{31}^{(2)}} \end{bmatrix}$$

MULTI-NEURON NETWORKS :: BACKPROPAGATION

$$J = \sum \frac{1}{2} (y - \hat{y})^2$$

ADDS COST FROM EACH EXAMPLE
↓

$$\frac{\partial J}{\partial W^{(2)}} = \frac{\partial \sum \frac{1}{2} (y - \hat{y})^2}{\partial W^{(2)}}$$

$$z^{(2)} = XW^{(1)} \quad (1)$$

$$a^{(2)} = f(z^{(2)}) \quad (2)$$

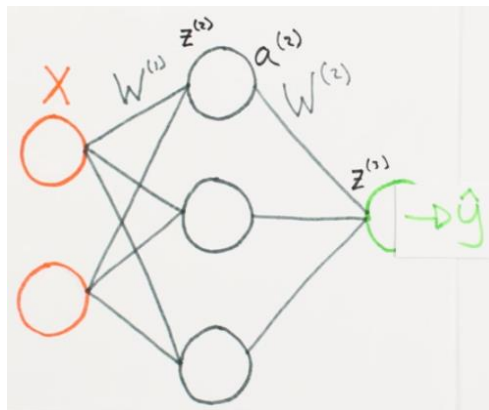
$$z^{(3)} = a^{(2)}W^{(2)} \quad (3)$$

$$\hat{y} = f(z^{(3)}) \quad (4)$$

$$J = \sum \frac{1}{2} (y - f(f(XW^{(1)})W^{(2)}))^2$$

↑ HOW DOES THIS CHANGE IF I CHANGE THESE? ↑

$$\frac{\partial J}{\partial W}$$



$$W^{(1)} = \begin{bmatrix} W_{11}^{(1)} & W_{12}^{(1)} & W_{13}^{(1)} \\ W_{21}^{(1)} & W_{22}^{(1)} & W_{23}^{(1)} \end{bmatrix}$$

$$W^{(2)} = \begin{bmatrix} W_{11}^{(2)} \\ W_{21}^{(2)} \\ W_{31}^{(2)} \end{bmatrix}$$

$$\frac{\partial J}{\partial W^{(1)}} = \begin{bmatrix} \frac{\partial J}{\partial W_{11}^{(1)}} & \frac{\partial J}{\partial W_{12}^{(1)}} & \frac{\partial J}{\partial W_{13}^{(1)}} \\ \frac{\partial J}{\partial W_{21}^{(1)}} & \frac{\partial J}{\partial W_{22}^{(1)}} & \frac{\partial J}{\partial W_{23}^{(1)}} \end{bmatrix}$$

$$\frac{\partial J}{\partial W^{(2)}} = \begin{bmatrix} \frac{\partial J}{\partial W_{11}^{(2)}} \\ \frac{\partial J}{\partial W_{21}^{(2)}} \\ \frac{\partial J}{\partial W_{31}^{(2)}} \end{bmatrix}$$

MULTI-NEURON NETWORKS :: BACKPROPAGATION

$$J = \sum \frac{1}{2} (y - \hat{y})^2$$

ADDS COST FROM EACH EXAMPLE
↓

$$\frac{\partial J}{\partial W^{(2)}} = \frac{\partial \sum \frac{1}{2} (y - \hat{y})^2}{\partial W^{(2)}}$$

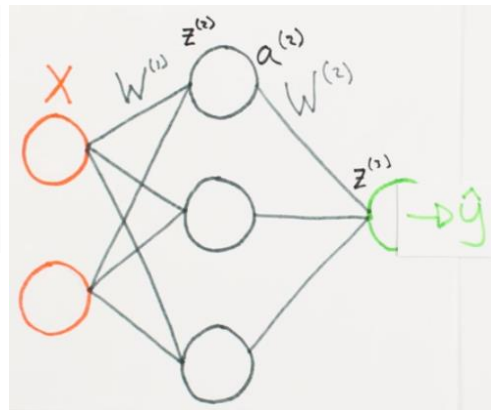
$$\frac{\partial J}{\partial W^{(2)}} = \sum \frac{\partial \frac{1}{2} (y - \hat{y})^2}{\partial W^{(2)}}$$

$$\begin{aligned} z^{(2)} &= XW^{(1)} & (1) \\ a^{(2)} &= f(z^{(2)}) & (2) \\ z^{(3)} &= a^{(2)}W^{(2)} & (3) \\ \hat{y} &= f(z^{(3)}) & (4) \end{aligned}$$

$$J = \sum \frac{1}{2} (y - f(f(XW^{(1)})W^{(2)}))^2$$

↑ HOW DOES THIS CHANGE IF I CHANGE THESE? ↑

$$\frac{\partial J}{\partial W}$$



$$W^{(1)} = \begin{bmatrix} W_{11}^{(1)} & W_{12}^{(1)} & W_{13}^{(1)} \\ W_{21}^{(1)} & W_{22}^{(1)} & W_{23}^{(1)} \end{bmatrix}$$

$$W^{(2)} = \begin{bmatrix} W_{11}^{(2)} \\ W_{21}^{(2)} \\ W_{31}^{(2)} \end{bmatrix}$$

$$\frac{\partial J}{\partial W^{(1)}} = \begin{bmatrix} \frac{\partial J}{\partial W_{11}^{(1)}} & \frac{\partial J}{\partial W_{12}^{(1)}} & \frac{\partial J}{\partial W_{13}^{(1)}} \\ \frac{\partial J}{\partial W_{21}^{(1)}} & \frac{\partial J}{\partial W_{22}^{(1)}} & \frac{\partial J}{\partial W_{23}^{(1)}} \end{bmatrix}$$

$$\frac{\partial J}{\partial W^{(2)}} = \begin{bmatrix} \frac{\partial J}{\partial W_{11}^{(2)}} \\ \frac{\partial J}{\partial W_{21}^{(2)}} \\ \frac{\partial J}{\partial W_{31}^{(2)}} \end{bmatrix}$$

MULTI-NEURON NETWORKS :: BACKPROPAGATION

$$J = \sum \frac{1}{2} (y - \hat{y})^2$$

ADDS COST FROM EACH EXAMPLE
↓

$$\frac{\partial J}{\partial W^{(2)}} = \frac{\partial \sum \frac{1}{2} (y - \hat{y})^2}{\partial W^{(2)}}$$

$$\frac{\partial J}{\partial W^{(2)}} = \sum \frac{\partial \frac{1}{2} (y - \hat{y})^2}{\partial W^{(2)}}$$

$$\frac{\partial \frac{1}{2} (y - \hat{y})^2}{\partial W^{(2)}}$$

$$z^{(2)} = XW^{(1)} \quad (1)$$

$$a^{(2)} = f(z^{(2)}) \quad (2)$$

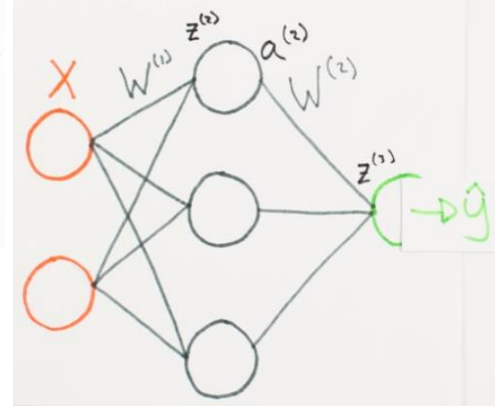
$$z^{(3)} = a^{(2)}W^{(2)} \quad (3)$$

$$\hat{y} = f(z^{(3)}) \quad (4)$$

$$J = \sum \frac{1}{2} (y - f(f(XW^{(1)})W^{(2)}))^2$$

↑ HOW DOES THIS CHANGE IF I CHANGE THESE? ↑

$$\frac{\partial J}{\partial W}$$



$$W^{(1)} = \begin{bmatrix} W_{11}^{(1)} & W_{12}^{(1)} & W_{13}^{(1)} \\ W_{21}^{(1)} & W_{22}^{(1)} & W_{23}^{(1)} \end{bmatrix}$$

$$W^{(2)} = \begin{bmatrix} W_{11}^{(2)} \\ W_{21}^{(2)} \\ W_{31}^{(2)} \end{bmatrix}$$

$$\frac{\partial J}{\partial W^{(1)}} = \begin{bmatrix} \frac{\partial J}{\partial W_{11}^{(1)}} & \frac{\partial J}{\partial W_{12}^{(1)}} & \frac{\partial J}{\partial W_{13}^{(1)}} \\ \frac{\partial J}{\partial W_{21}^{(1)}} & \frac{\partial J}{\partial W_{22}^{(1)}} & \frac{\partial J}{\partial W_{23}^{(1)}} \end{bmatrix}$$

$$\frac{\partial J}{\partial W^{(2)}} = \begin{bmatrix} \frac{\partial J}{\partial W_{11}^{(2)}} \\ \frac{\partial J}{\partial W_{21}^{(2)}} \\ \frac{\partial J}{\partial W_{31}^{(2)}} \end{bmatrix}$$

MULTI-NEURON NETWORKS :: BACKPROPAGATION

$$J = \sum \frac{1}{2} (y - \hat{y})^2$$

ADDS COST FROM EACH EXAMPLE
↓

$$\frac{\partial J}{\partial W^{(2)}} = \frac{\partial \sum \frac{1}{2} (y - \hat{y})^2}{\partial W^{(2)}}$$

$$\frac{\partial J}{\partial W^{(2)}} = \sum \frac{\partial \frac{1}{2} (y - \hat{y})^2}{\partial W^{(2)}}$$

$$\frac{\partial \frac{1}{2} (y - \hat{y})^2}{\partial W^{(2)}}$$

$$\frac{\partial J}{\partial W^{(2)}} = -(y - \hat{y}) \frac{\partial \hat{y}}{\partial W^{(2)}}$$

$$\begin{aligned} z^{(2)} &= XW^{(1)} & (1) \\ a^{(2)} &= f(z^{(2)}) & (2) \\ z^{(3)} &= a^{(2)}W^{(2)} & (3) \\ \hat{y} &= f(z^{(3)}) & (4) \end{aligned}$$

$$J = \sum \frac{1}{2} (y - f(f(XW^{(1)})W^{(2)}))^2$$

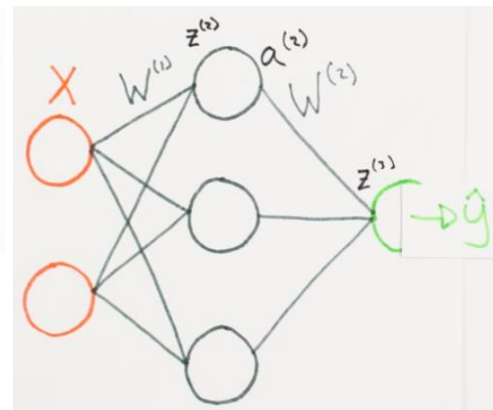
↑ HOW DOES THIS CHANGE IF I CHANGE THESE? ↑

$$\frac{\partial J}{\partial W}$$

CHAIN RULE

ex $\frac{d}{dx} (3x + 2x^2)^2 = 2(3x + 2x^2)(3 + 4x)$

$$\frac{dz}{dx} = \frac{dz}{dy} \cdot \frac{dy}{dx}$$



$$W^{(1)} = \begin{bmatrix} W_{11}^{(1)} & W_{12}^{(1)} & W_{13}^{(1)} \\ W_{21}^{(1)} & W_{22}^{(1)} & W_{23}^{(1)} \end{bmatrix}$$

$$\frac{\partial J}{\partial W^{(1)}} = \begin{bmatrix} \frac{\partial J}{\partial W_{11}^{(1)}} & \frac{\partial J}{\partial W_{12}^{(1)}} & \frac{\partial J}{\partial W_{13}^{(1)}} \\ \frac{\partial J}{\partial W_{21}^{(1)}} & \frac{\partial J}{\partial W_{22}^{(1)}} & \frac{\partial J}{\partial W_{23}^{(1)}} \end{bmatrix}$$

$$W^{(2)} = \begin{bmatrix} W_{11}^{(2)} \\ W_{21}^{(2)} \\ W_{31}^{(2)} \end{bmatrix}$$

$$\frac{\partial J}{\partial W^{(2)}} = \begin{bmatrix} \frac{\partial J}{\partial W_{11}^{(2)}} \\ \frac{\partial J}{\partial W_{21}^{(2)}} \\ \frac{\partial J}{\partial W_{31}^{(2)}} \end{bmatrix}$$

MULTI-NEURON NETWORKS :: BACKPROPAGATION

$$J = \sum \frac{1}{2} (y - \hat{y})^2$$

$$\hat{y} = f(z^{(3)})$$

$$z^{(2)} = XW^{(1)} \quad (1)$$

$$a^{(2)} = f(z^{(2)}) \quad (2)$$

$$z^{(3)} = a^{(2)}W^{(2)} \quad (3)$$

$$\hat{y} = f(z^{(3)}) \quad (4)$$

$$\frac{\partial J}{\partial W^{(2)}} = -(y - \hat{y}) \frac{\partial \hat{y}}{\partial z^{(3)}} \frac{\partial z^{(3)}}{\partial W^{(2)}}$$

ADDS COST FROM EACH EXAMPLE
↓

$$\frac{\partial J}{\partial W^{(2)}} = \frac{\partial \sum \frac{1}{2} (y - \hat{y})^2}{\partial W^{(2)}}$$

$$\frac{\partial J}{\partial W^{(2)}} = \sum \frac{\partial \frac{1}{2} (y - \hat{y})^2}{\partial W^{(2)}}$$

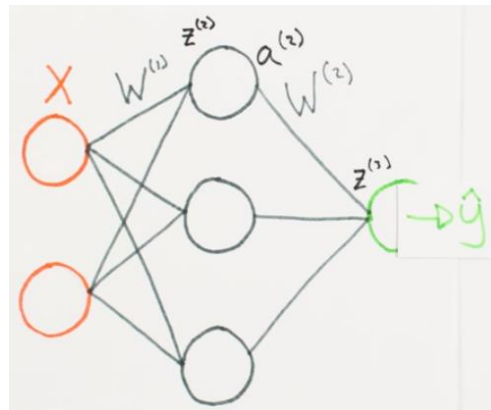
$$\frac{\partial \frac{1}{2} (y - \hat{y})^2}{\partial W^{(2)}}$$

$$\frac{\partial J}{\partial W^{(2)}} = -(y - \hat{y}) \frac{\partial \hat{y}}{\partial W^{(2)}}$$

$$J = \sum \frac{1}{2} (y - f(f(XW^{(1)})W^{(2)}))^2$$

↑ HOW DOES THIS CHANGE IF I CHANGE THESE? ↗ ↖

$$\frac{\partial J}{\partial W}$$



CHAIN RULE

ex $\frac{d}{dx} (3x + 2x^2)^2 = 2(3x + 2x^2)(3 + 4x)$

$$\frac{dz}{dx} = \frac{dz}{dy} \cdot \frac{dy}{dx}$$

MULTI-NEURON NETWORKS :: BACKPROPAGATION

$$J = \sum \frac{1}{2} (y - \hat{y})^2$$

ADDS COST FROM EACH EXAMPLE
↓

$$\frac{\partial J}{\partial W^{(2)}} = \frac{\partial \sum \frac{1}{2} (y - \hat{y})^2}{\partial W^{(2)}}$$

$$\frac{\partial J}{\partial W^{(2)}} = \sum \frac{\partial \frac{1}{2} (y - \hat{y})^2}{\partial W^{(2)}}$$

$$\frac{\partial \frac{1}{2} (y - \hat{y})^2}{\partial W^{(2)}}$$

$$\frac{\partial J}{\partial W^{(2)}} = -(y - \hat{y}) \frac{\partial \hat{y}}{\partial W^{(2)}}$$

$$\hat{y} = f(z^{(3)})$$

$$\frac{\partial J}{\partial W^{(2)}} = -(y - \hat{y}) \frac{\partial \hat{y}}{\partial z^{(3)}} \frac{\partial z^{(3)}}{\partial W^{(2)}}$$

$$\frac{\partial J}{\partial W^{(2)}} = -(y - \hat{y}) f'(z^{(3)}) \frac{\partial z^{(3)}}{\partial W^{(2)}}$$

$$z^{(2)} = XW^{(1)} \quad (1)$$

$$a^{(2)} = f(z^{(2)}) \quad (2)$$

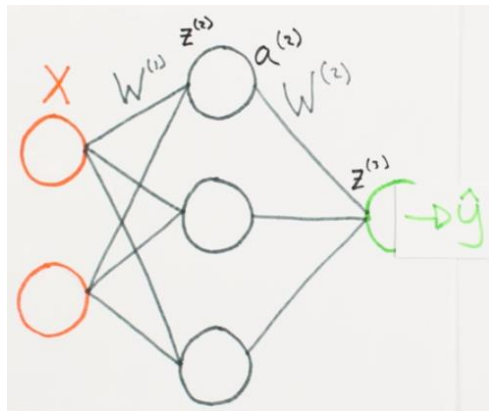
$$z^{(3)} = a^{(2)}W^{(2)} \quad (3)$$

$$\hat{y} = f(z^{(3)}) \quad (4)$$

$$J = \sum \frac{1}{2} (y - f(f(XW^{(1)})W^{(2)}))^2$$

↑ HOW DOES THIS CHANGE IF I CHANGE THESE? ↗ ↖

$$\frac{\partial J}{\partial W}$$



$$W^{(1)} = \begin{bmatrix} W_{11}^{(1)} & W_{12}^{(1)} & W_{13}^{(1)} \\ W_{21}^{(1)} & W_{22}^{(1)} & W_{23}^{(1)} \end{bmatrix}$$

$$\frac{\partial J}{\partial W^{(1)}} = \begin{bmatrix} \frac{\partial J}{\partial W_{11}^{(1)}} & \frac{\partial J}{\partial W_{12}^{(1)}} & \frac{\partial J}{\partial W_{13}^{(1)}} \\ \frac{\partial J}{\partial W_{21}^{(1)}} & \frac{\partial J}{\partial W_{22}^{(1)}} & \frac{\partial J}{\partial W_{23}^{(1)}} \end{bmatrix}$$

$$W^{(2)} = \begin{bmatrix} W_{11}^{(2)} \\ W_{21}^{(2)} \\ W_{31}^{(2)} \end{bmatrix}$$

$$\frac{\partial J}{\partial W^{(2)}} = \begin{bmatrix} \frac{\partial J}{\partial W_{11}^{(2)}} \\ \frac{\partial J}{\partial W_{21}^{(2)}} \\ \frac{\partial J}{\partial W_{31}^{(2)}} \end{bmatrix}$$

MULTI-NEURON NETWORKS :: BACKPROPAGATION

$$J = \sum \frac{1}{2} (y - \hat{y})^2$$

ADDS COST FROM EACH EXAMPLE
↓

$$\frac{\partial J}{\partial W^{(2)}} = \frac{\partial \sum \frac{1}{2} (y - \hat{y})^2}{\partial W^{(2)}}$$

$$\frac{\partial J}{\partial W^{(2)}} = \sum \frac{\partial \frac{1}{2} (y - \hat{y})^2}{\partial W^{(2)}}$$

$$\frac{\partial J}{\partial W^{(2)}} = -(y - \hat{y}) \frac{\partial \hat{y}}{\partial W^{(2)}}$$

$$\hat{y} = f(z^{(3)})$$

$$\frac{\partial J}{\partial W^{(2)}} = -(y - \hat{y}) \frac{\partial \hat{y}}{\partial z^{(3)}} \frac{\partial z^{(3)}}{\partial W^{(2)}}$$

$$\frac{\partial J}{\partial W^{(2)}} = -(y - \hat{y}) f'(z^{(3)}) \frac{\partial z^{(3)}}{\partial W^{(2)}}$$

$$\frac{\partial J}{\partial W^{(2)}} = (a^{(2)})^T \delta^{(3)} \quad (6)$$

$$\delta^{(3)} = -(y - \hat{y}) f'(z^{(3)})$$



Backprop error

$$z^{(2)} = XW^{(1)} \quad (1)$$

$$a^{(2)} = f(z^{(2)}) \quad (2)$$

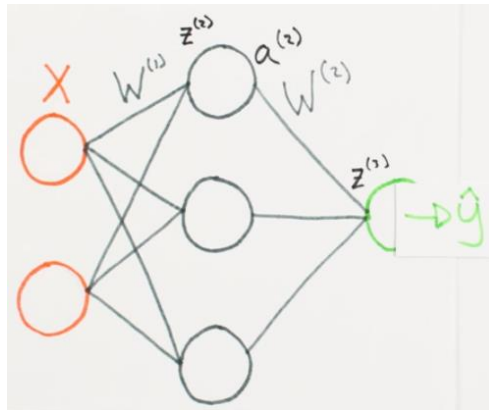
$$z^{(3)} = a^{(2)}W^{(2)} \quad (3)$$

$$\hat{y} = f(z^{(3)}) \quad (4)$$

$$J = \sum \frac{1}{2} (y - f(f(XW^{(1)})W^{(2)}))^2$$

↑ HOW DOES THIS CHANGE IF I CHANGE THESE? ↗ ↖

$$\frac{\partial J}{\partial W}$$



$$W^{(1)} = \begin{bmatrix} w_{11}^{(1)} & w_{12}^{(1)} & w_{13}^{(1)} \\ w_{21}^{(1)} & w_{22}^{(1)} & w_{23}^{(1)} \end{bmatrix} \quad \frac{\partial J}{\partial W^{(1)}} = \begin{bmatrix} \frac{\partial J}{\partial w_{11}^{(1)}} & \frac{\partial J}{\partial w_{12}^{(1)}} & \frac{\partial J}{\partial w_{13}^{(1)}} \\ \frac{\partial J}{\partial w_{21}^{(1)}} & \frac{\partial J}{\partial w_{22}^{(1)}} & \frac{\partial J}{\partial w_{23}^{(1)}} \end{bmatrix}$$

$$W^{(2)} = \begin{bmatrix} w_{11}^{(2)} \\ w_{21}^{(2)} \\ w_{31}^{(2)} \end{bmatrix} \quad \frac{\partial J}{\partial W^{(2)}} = \begin{bmatrix} \frac{\partial J}{\partial w_{11}^{(2)}} \\ \frac{\partial J}{\partial w_{21}^{(2)}} \\ \frac{\partial J}{\partial w_{31}^{(2)}} \end{bmatrix}$$

MULTI-NEURON NETWORKS :: BACKPROPAGATION

$$J = \sum \frac{1}{2} (y - \hat{y})^2$$

$$\frac{\partial J}{\partial W^{(1)}} = \frac{\partial \left[\frac{1}{2} (y - \hat{y})^2 \right]}{\partial W^{(1)}}$$

$$z^{(2)} = XW^{(1)} \quad (1)$$

$$a^{(2)} = f(z^{(2)}) \quad (2)$$

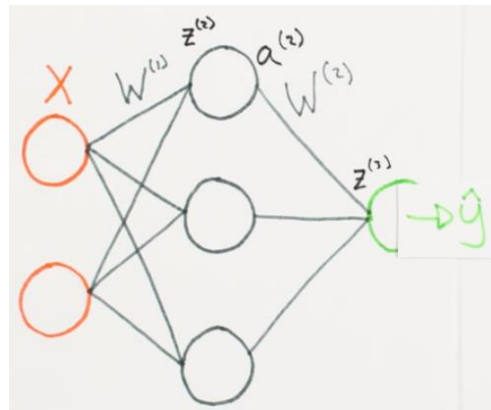
$$z^{(3)} = a^{(2)}W^{(2)} \quad (3)$$

$$\hat{y} = f(z^{(3)}) \quad (4)$$

$$J = \sum \frac{1}{2} (y - f(f(XW^{(1)})W^{(2)}))^2$$

↑ HOW DOES THIS CHANGE IF I CHANGE THESE? ↑

$$\frac{\partial J}{\partial W}$$



$$W^{(1)} = \begin{bmatrix} W_{11}^{(1)} & W_{12}^{(1)} & W_{13}^{(1)} \\ W_{21}^{(1)} & W_{22}^{(1)} & W_{23}^{(1)} \end{bmatrix}$$

$$W^{(2)} = \begin{bmatrix} W_{11}^{(2)} \\ W_{21}^{(2)} \\ W_{31}^{(2)} \end{bmatrix}$$

$$\frac{\partial J}{\partial W^{(1)}} = \begin{bmatrix} \frac{\partial J}{\partial W_{11}^{(1)}} & \frac{\partial J}{\partial W_{12}^{(1)}} & \frac{\partial J}{\partial W_{13}^{(1)}} \\ \frac{\partial J}{\partial W_{21}^{(1)}} & \frac{\partial J}{\partial W_{22}^{(1)}} & \frac{\partial J}{\partial W_{23}^{(1)}} \end{bmatrix}$$

$$\frac{\partial J}{\partial W^{(2)}} = \begin{bmatrix} \frac{\partial J}{\partial W_{11}^{(2)}} \\ \frac{\partial J}{\partial W_{21}^{(2)}} \\ \frac{\partial J}{\partial W_{31}^{(2)}} \end{bmatrix}$$

MULTI-NEURON NETWORKS :: BACKPROPAGATION

$$J = \sum \frac{1}{2} (y - \hat{y})^2$$

$$\frac{\partial J}{\partial W^{(1)}} = \frac{\partial}{\partial W^{(1)}} \left[\frac{1}{2} (y - \hat{y})^2 \right]$$

$$\frac{\partial J}{\partial W^{(1)}} = -(y - \hat{y}) \frac{\partial \hat{y}}{\partial z^{(3)}} \frac{\partial z^{(3)}}{\partial W^{(1)}}$$

$$\frac{\partial J}{\partial W^{(1)}} = -(y - \hat{y}) f'(z^{(3)}) \frac{\partial z^{(3)}}{\partial W^{(1)}}$$

$$z^{(2)} = XW^{(1)} \quad (1)$$

$$a^{(2)} = f(z^{(2)}) \quad (2)$$

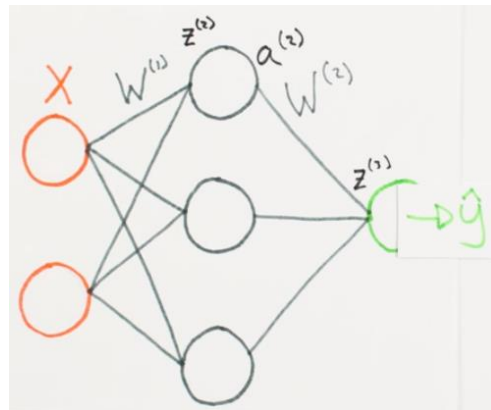
$$z^{(3)} = a^{(2)}W^{(2)} \quad (3)$$

$$\hat{y} = f(z^{(3)}) \quad (4)$$

$$J = \sum \frac{1}{2} (y - f(f(XW^{(1)})W^{(2)}))^2$$

↑ HOW DOES THIS CHANGE IF I CHANGE THESE? ↗ ↖

$$\frac{\partial J}{\partial W}$$



$$W^{(1)} = \begin{bmatrix} w_{11}^{(1)} & w_{12}^{(1)} & w_{13}^{(1)} \\ w_{21}^{(1)} & w_{22}^{(1)} & w_{23}^{(1)} \end{bmatrix}$$

$$W^{(2)} = \begin{bmatrix} w_{11}^{(2)} \\ w_{21}^{(2)} \\ w_{31}^{(2)} \end{bmatrix}$$

$$\frac{\partial J}{\partial W^{(1)}} = \begin{bmatrix} \frac{\partial J}{\partial w_{11}^{(1)}} & \frac{\partial J}{\partial w_{12}^{(1)}} & \frac{\partial J}{\partial w_{13}^{(1)}} \\ \frac{\partial J}{\partial w_{21}^{(1)}} & \frac{\partial J}{\partial w_{22}^{(1)}} & \frac{\partial J}{\partial w_{23}^{(1)}} \end{bmatrix}$$

$$\frac{\partial J}{\partial W^{(2)}} = \begin{bmatrix} \frac{\partial J}{\partial w_{11}^{(2)}} \\ \frac{\partial J}{\partial w_{21}^{(2)}} \\ \frac{\partial J}{\partial w_{31}^{(2)}} \end{bmatrix}$$

MULTI-NEURON NETWORKS :: BACKPROPAGATION

$$J = \sum \frac{1}{2} (y - \hat{y})^2$$

$$\frac{\partial J}{\partial W^{(1)}} = \frac{\partial}{\partial W^{(1)}} \left[\frac{1}{2} (y - \hat{y})^2 \right]$$

$$\frac{\partial J}{\partial W^{(1)}} = -(y - \hat{y}) \frac{\partial \hat{y}}{\partial z^{(3)}} \frac{\partial z^{(3)}}{\partial W^{(1)}}$$

$$\frac{\partial J}{\partial W^{(1)}} = -(y - \hat{y}) f'(z^{(3)}) \frac{\partial z^{(3)}}{\partial W^{(1)}}$$

$$\delta^{(3)} = -(y - \hat{y}) f'(z^{(3)})$$

$$\frac{\partial J}{\partial W^{(1)}} = \delta^{(3)} \frac{\partial z^{(3)}}{\partial W^{(1)}}$$

$$z^{(2)} = XW^{(1)} \quad (1)$$

$$a^{(2)} = f(z^{(2)}) \quad (2)$$

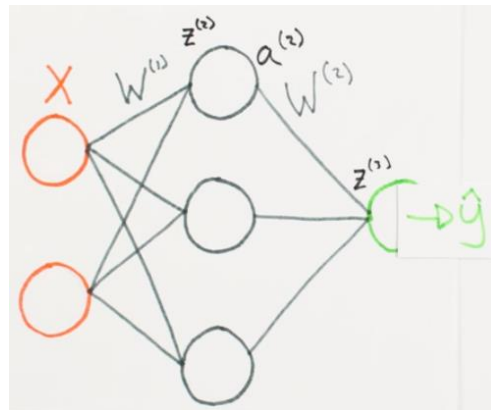
$$z^{(3)} = a^{(2)}W^{(2)} \quad (3)$$

$$\hat{y} = f(z^{(3)}) \quad (4)$$

$$J = \sum \frac{1}{2} (y - f(f(XW^{(1)})W^{(2)}))^2$$

How does this change if I change these?

$$\frac{\partial J}{\partial W}$$



$$W^{(1)} = \begin{bmatrix} w_{11}^{(1)} & w_{12}^{(1)} & w_{13}^{(1)} \\ w_{21}^{(1)} & w_{22}^{(1)} & w_{23}^{(1)} \end{bmatrix}$$

$$W^{(2)} = \begin{bmatrix} w_{11}^{(2)} \\ w_{21}^{(2)} \\ w_{31}^{(2)} \end{bmatrix}$$

$$\frac{\partial J}{\partial W^{(1)}} = \begin{bmatrix} \frac{\partial J}{\partial w_{11}^{(1)}} & \frac{\partial J}{\partial w_{12}^{(1)}} & \frac{\partial J}{\partial w_{13}^{(1)}} \\ \frac{\partial J}{\partial w_{21}^{(1)}} & \frac{\partial J}{\partial w_{22}^{(1)}} & \frac{\partial J}{\partial w_{23}^{(1)}} \end{bmatrix}$$

$$\frac{\partial J}{\partial W^{(2)}} = \begin{bmatrix} \frac{\partial J}{\partial w_{11}^{(2)}} \\ \frac{\partial J}{\partial w_{21}^{(2)}} \\ \frac{\partial J}{\partial w_{31}^{(2)}} \end{bmatrix}$$

MULTI-NEURON NETWORKS :: BACKPROPAGATION

$$J = \sum \frac{1}{2} (y - \hat{y})^2$$

$$\frac{\partial J}{\partial W^{(1)}} = \frac{\partial}{\partial W^{(1)}} \left[\frac{1}{2} (y - \hat{y})^2 \right]$$

$$\frac{\partial J}{\partial W^{(1)}} = -(y - \hat{y}) \frac{\partial \hat{y}}{\partial z^{(3)}} \frac{\partial z^{(3)}}{\partial W^{(1)}}$$

$$\frac{\partial J}{\partial W^{(1)}} = -(y - \hat{y}) f'(z^{(3)}) \frac{\partial z^{(3)}}{\partial W^{(1)}}$$

$$\delta^{(3)} = -(y - \hat{y}) f'(z^{(3)})$$

$$\frac{\partial J}{\partial W^{(1)}} = \delta^{(3)} \frac{\partial z^{(3)}}{\partial W^{(1)}}$$

$$\frac{\partial J}{\partial W^{(1)}} = \delta^{(3)} \frac{\partial z^{(3)}}{\partial a^{(2)}} \frac{\partial a^{(2)}}{\partial W^{(1)}}$$

$$z^{(2)} = XW^{(1)} \quad (1)$$

$$a^{(2)} = f(z^{(2)}) \quad (2)$$

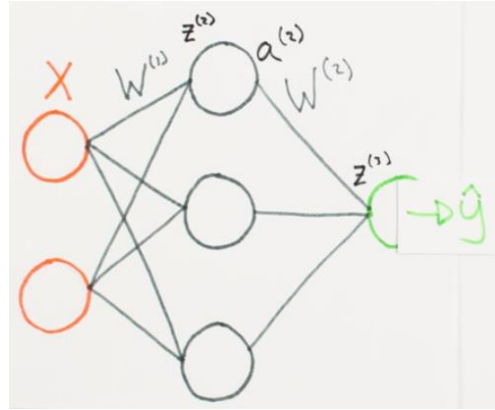
$$z^{(3)} = a^{(2)}W^{(2)} \quad (3)$$

$$\hat{y} = f(z^{(3)}) \quad (4)$$

$$J = \sum \frac{1}{2} (y - f(f(XW^{(1)})W^{(2)}))^2$$

How does this change if I change these?

$$\frac{\partial J}{\partial W}$$



$$W^{(1)} = \begin{bmatrix} w_{11}^{(1)} & w_{12}^{(1)} & w_{13}^{(1)} \\ w_{21}^{(1)} & w_{22}^{(1)} & w_{23}^{(1)} \end{bmatrix}$$

$$\frac{\partial J}{\partial W^{(1)}} = \begin{bmatrix} \frac{\partial J}{\partial w_{11}^{(1)}} & \frac{\partial J}{\partial w_{12}^{(1)}} & \frac{\partial J}{\partial w_{13}^{(1)}} \\ \frac{\partial J}{\partial w_{21}^{(1)}} & \frac{\partial J}{\partial w_{22}^{(1)}} & \frac{\partial J}{\partial w_{23}^{(1)}} \end{bmatrix}$$

$$W^{(2)} = \begin{bmatrix} w_{11}^{(2)} \\ w_{21}^{(2)} \\ w_{31}^{(2)} \end{bmatrix}$$

$$\frac{\partial J}{\partial W^{(2)}} = \begin{bmatrix} \frac{\partial J}{\partial w_{11}^{(2)}} \\ \frac{\partial J}{\partial w_{21}^{(2)}} \\ \frac{\partial J}{\partial w_{31}^{(2)}} \end{bmatrix}$$

MULTI-NEURON NETWORKS :: BACKPROPAGATION

$$J = \sum \frac{1}{2} (y - \hat{y})^2$$

$$\frac{\partial J}{\partial W^{(1)}} = \frac{\partial}{\partial W^{(1)}} \left[\frac{1}{2} (y - \hat{y})^2 \right]$$

$$\frac{\partial J}{\partial W^{(1)}} = -(y - \hat{y}) \frac{\partial \hat{y}}{\partial z^{(3)}} \frac{\partial z^{(3)}}{\partial W^{(1)}}$$

$$\frac{\partial J}{\partial W^{(1)}} = -(y - \hat{y}) f'(z^{(3)}) \frac{\partial z^{(3)}}{\partial W^{(1)}}$$

$$\frac{\partial J}{\partial W^{(1)}} = \delta^{(3)} \frac{\partial z^{(3)}}{\partial W^{(1)}}$$

$$\frac{\partial J}{\partial W^{(1)}} = \delta^{(3)} \frac{\partial z^{(3)}}{\partial a^{(2)}} \frac{\partial a^{(2)}}{\partial W^{(1)}}$$

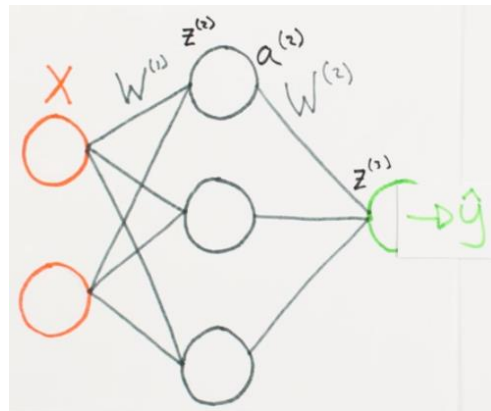
$$\frac{\partial J}{\partial W^{(1)}} = \delta^{(3)} (W^{(2)})^T \frac{\partial a^{(2)}}{\partial W^{(1)}}$$

$$\begin{aligned} z^{(2)} &= XW^{(1)} & (1) \\ a^{(2)} &= f(z^{(2)}) & (2) \\ z^{(3)} &= a^{(2)}W^{(2)} & (3) \\ \hat{y} &= f(z^{(3)}) & (4) \end{aligned}$$

$$J = \sum \frac{1}{2} (y - f(f(XW^{(1)})W^{(2)}))^2$$

How does this change if I change these?

$$\frac{\partial J}{\partial W}$$



$$W^{(1)} = \begin{bmatrix} w_{11}^{(1)} & w_{12}^{(1)} & w_{13}^{(1)} \\ w_{21}^{(1)} & w_{22}^{(1)} & w_{23}^{(1)} \end{bmatrix}$$

$$\frac{\partial J}{\partial W^{(1)}} = \begin{bmatrix} \frac{\partial J}{\partial w_{11}^{(1)}} & \frac{\partial J}{\partial w_{12}^{(1)}} & \frac{\partial J}{\partial w_{13}^{(1)}} \\ \frac{\partial J}{\partial w_{21}^{(1)}} & \frac{\partial J}{\partial w_{22}^{(1)}} & \frac{\partial J}{\partial w_{23}^{(1)}} \end{bmatrix}$$

$$W^{(2)} = \begin{bmatrix} w_{11}^{(2)} \\ w_{21}^{(2)} \\ w_{31}^{(2)} \end{bmatrix}$$

$$\frac{\partial J}{\partial W^{(2)}} = \begin{bmatrix} \frac{\partial J}{\partial w_{11}^{(2)}} \\ \frac{\partial J}{\partial w_{21}^{(2)}} \\ \frac{\partial J}{\partial w_{31}^{(2)}} \end{bmatrix}$$

MULTI-NEURON NETWORKS :: BACKPROPAGATION

$$J = \sum \frac{1}{2} (y - \hat{y})^2$$

$$\frac{\partial J}{\partial W^{(1)}} = \frac{\partial}{\partial W^{(1)}} \left[\frac{1}{2} (y - \hat{y})^2 \right]$$

$$\frac{\partial J}{\partial W^{(1)}} = -(y - \hat{y}) \frac{\partial \hat{y}}{\partial z^{(3)}} \frac{\partial z^{(3)}}{\partial W^{(1)}}$$

$$\frac{\partial J}{\partial W^{(1)}} = -(y - \hat{y}) f'(z^{(3)}) \frac{\partial z^{(3)}}{\partial W^{(1)}}$$

$$\frac{\partial J}{\partial W^{(1)}} = \delta^{(3)} \frac{\partial z^{(3)}}{\partial W^{(1)}}$$

$$\frac{\partial J}{\partial W^{(1)}} = \delta^{(3)} \frac{\partial z^{(3)}}{\partial a^{(2)}} \frac{\partial a^{(2)}}{\partial W^{(1)}}$$

$$\frac{\partial J}{\partial W^{(1)}} = \delta^{(3)} (W^{(2)})^T \frac{\partial a^{(2)}}{\partial W^{(1)}}$$

$$\frac{\partial J}{\partial W^{(1)}} = \delta^{(3)} (W^{(2)})^T \frac{\partial a^{(2)}}{\partial z^{(2)}} \frac{\partial z^{(2)}}{\partial W^{(1)}}$$

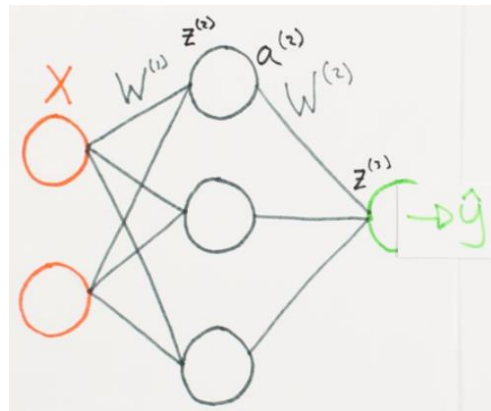
$$\frac{\partial J}{\partial W^{(1)}} = \delta^{(3)} (W^{(2)})^T f'(z^{(2)}) \frac{\partial z^{(2)}}{\partial W^{(1)}}$$

$$\begin{aligned} z^{(2)} &= XW^{(1)} & (1) \\ a^{(2)} &= f(z^{(2)}) & (2) \\ z^{(3)} &= a^{(2)}W^{(2)} & (3) \\ \hat{y} &= f(z^{(3)}) & (4) \end{aligned}$$

$$J = \sum \frac{1}{2} (y - f(f(XW^{(1)})W^{(2)}))^2$$

↑ HOW DOES THIS CHANGE IF I CHANGE THESE? ↑

$$\frac{\partial J}{\partial W}$$



$$W^{(1)} = \begin{bmatrix} w_{11}^{(1)} & w_{12}^{(1)} & w_{13}^{(1)} \\ w_{21}^{(1)} & w_{22}^{(1)} & w_{23}^{(1)} \end{bmatrix}$$

$$W^{(2)} = \begin{bmatrix} w_{11}^{(2)} \\ w_{21}^{(2)} \\ w_{31}^{(2)} \end{bmatrix}$$

$$\frac{\partial J}{\partial W^{(1)}} = \begin{bmatrix} \frac{\partial J}{\partial w_{11}^{(1)}} & \frac{\partial J}{\partial w_{12}^{(1)}} & \frac{\partial J}{\partial w_{13}^{(1)}} \\ \frac{\partial J}{\partial w_{21}^{(1)}} & \frac{\partial J}{\partial w_{22}^{(1)}} & \frac{\partial J}{\partial w_{23}^{(1)}} \end{bmatrix}$$

$$\frac{\partial J}{\partial W^{(2)}} = \begin{bmatrix} \frac{\partial J}{\partial w_{11}^{(2)}} \\ \frac{\partial J}{\partial w_{21}^{(2)}} \\ \frac{\partial J}{\partial w_{31}^{(2)}} \end{bmatrix}$$

MULTI-NEURON NETWORKS :: BACKPROPAGATION

$$J = \sum \frac{1}{2} (y - \hat{y})^2$$

$$\frac{\partial J}{\partial W^{(1)}} = \frac{\partial}{\partial W^{(1)}} \left[\frac{1}{2} (y - \hat{y})^2 \right]$$

$$\frac{\partial J}{\partial W^{(1)}} = -(y - \hat{y}) \frac{\partial \hat{y}}{\partial z^{(3)}} \frac{\partial z^{(3)}}{\partial W^{(1)}}$$

$$\frac{\partial J}{\partial W^{(1)}} = -(y - \hat{y}) f'(z^{(3)}) \frac{\partial z^{(3)}}{\partial W^{(1)}}$$

$$\frac{\partial J}{\partial W^{(1)}} = \delta^{(3)} \frac{\partial z^{(3)}}{\partial W^{(1)}}$$

$$\frac{\partial J}{\partial W^{(1)}} = \delta^{(3)} \frac{\partial z^{(3)}}{\partial a^{(2)}} \frac{\partial a^{(2)}}{\partial W^{(1)}}$$

$$\frac{\partial J}{\partial W^{(1)}} = \delta^{(3)} (W^{(2)})^T \frac{\partial a^{(2)}}{\partial W^{(1)}}$$

$$\frac{\partial J}{\partial W^{(1)}} = \delta^{(3)} (W^{(2)})^T \frac{\partial a^{(2)}}{\partial z^{(2)}} \frac{\partial z^{(2)}}{\partial W^{(1)}}$$

$$\frac{\partial J}{\partial W^{(1)}} = \delta^{(3)} (W^{(2)})^T f'(z^{(2)}) \frac{\partial z^{(2)}}{\partial W^{(1)}}$$

$$\frac{\partial J}{\partial W^{(1)}} = X^T \delta^{(3)} (W^{(2)})^T f'(z^{(2)})$$

$$z^{(2)} = XW^{(1)} \quad (1)$$

$$a^{(2)} = f(z^{(2)}) \quad (2)$$

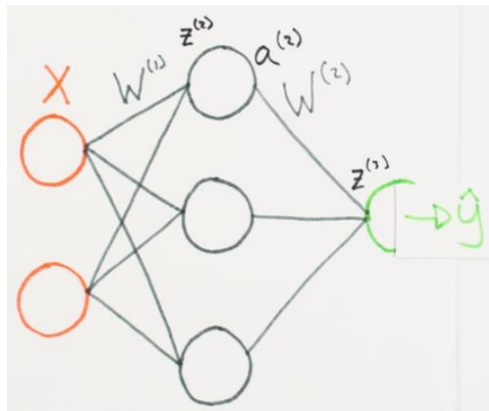
$$z^{(3)} = a^{(2)}W^{(2)} \quad (3)$$

$$\hat{y} = f(z^{(3)}) \quad (4)$$

$$J = \sum \frac{1}{2} (y - f(f(XW^{(1)})W^{(2)}))^2$$

↑ HOW DOES THIS CHANGE IF I CHANGE THESE? ↗ ↖

$$\frac{\partial J}{\partial W}$$



$$W^{(1)} = \begin{bmatrix} w_{11}^{(1)} & w_{12}^{(1)} & w_{13}^{(1)} \\ w_{21}^{(1)} & w_{22}^{(1)} & w_{23}^{(1)} \end{bmatrix}$$

$$W^{(2)} = \begin{bmatrix} w_{11}^{(2)} \\ w_{21}^{(2)} \\ w_{31}^{(2)} \end{bmatrix}$$

$$\frac{\partial J}{\partial W^{(1)}} = \begin{bmatrix} \frac{\partial J}{\partial w_{11}^{(1)}} & \frac{\partial J}{\partial w_{12}^{(1)}} & \frac{\partial J}{\partial w_{13}^{(1)}} \\ \frac{\partial J}{\partial w_{21}^{(1)}} & \frac{\partial J}{\partial w_{22}^{(1)}} & \frac{\partial J}{\partial w_{23}^{(1)}} \end{bmatrix}$$

$$\frac{\partial J}{\partial W^{(2)}} = \begin{bmatrix} \frac{\partial J}{\partial w_{11}^{(2)}} \\ \frac{\partial J}{\partial w_{21}^{(2)}} \\ \frac{\partial J}{\partial w_{31}^{(2)}} \end{bmatrix}$$

MULTI-NEURON NETWORKS :: BACKPROPAGATION

$$J = \sum \frac{1}{2} (y - \hat{y})^2$$

$$\frac{\partial J}{\partial W^{(2)}} = (a^{(2)})^T \delta^{(3)}$$

$$\delta^{(3)} = -(y - \hat{y}) f'(z^{(3)})$$

$$\frac{\partial J}{\partial W^{(1)}} = X^T \delta^{(2)}$$

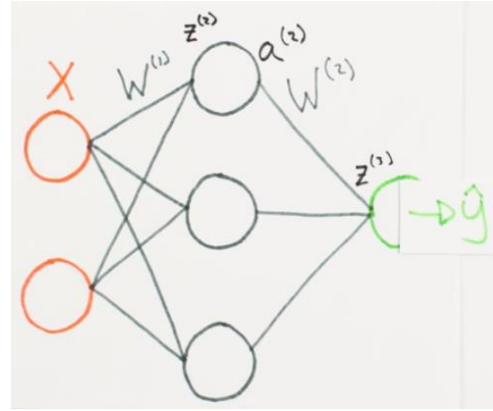
$$\delta^{(2)} = \delta^{(3)} (W^{(2)})^T f'(z^{(2)})$$

$$\begin{aligned} z^{(2)} &= XW^{(1)} & (1) \\ a^{(2)} &= f(z^{(2)}) & (2) \\ z^{(3)} &= a^{(2)}W^{(2)} & (3) \\ \hat{y} &= f(z^{(3)}) & (4) \end{aligned}$$

$$J = \sum \frac{1}{2} (y - f(f(XW^{(1)})W^{(2)}))^2$$

How does this change if I change these?

$$\frac{\partial J}{\partial W}$$



$$\begin{aligned} W^{(1)} &= \begin{bmatrix} W_{11}^{(1)} & W_{12}^{(1)} & W_{13}^{(1)} \\ W_{21}^{(1)} & W_{22}^{(1)} & W_{23}^{(1)} \end{bmatrix} & \frac{\partial J}{\partial W^{(1)}} &= \begin{bmatrix} \frac{\partial J}{\partial W_{11}^{(1)}} & \frac{\partial J}{\partial W_{12}^{(1)}} & \frac{\partial J}{\partial W_{13}^{(1)}} \\ \frac{\partial J}{\partial W_{21}^{(1)}} & \frac{\partial J}{\partial W_{22}^{(1)}} & \frac{\partial J}{\partial W_{23}^{(1)}} \end{bmatrix} \\ W^{(2)} &= \begin{bmatrix} W_{11}^{(2)} \\ W_{21}^{(2)} \\ W_{31}^{(2)} \end{bmatrix} & \frac{\partial J}{\partial W^{(2)}} &= \begin{bmatrix} \frac{\partial J}{\partial W_{11}^{(2)}} \\ \frac{\partial J}{\partial W_{21}^{(2)}} \\ \frac{\partial J}{\partial W_{31}^{(2)}} \end{bmatrix} \end{aligned}$$

MULTI-NEURON NETWORKS :: BACKPROPAGATION

$$J = \sum \frac{1}{2} (y - \hat{y})^2$$

$$\frac{\partial J}{\partial W^{(2)}} = (a^{(2)})^T \delta^{(3)}$$

$$\delta^{(3)} = -(y - \hat{y})f'(z^{(3)})$$

$$\frac{\partial J}{\partial W^{(1)}} = X^T \delta^{(2)}$$

$$\delta^{(2)} = \delta^{(3)} (W^{(2)})^T f'(z^{(2)})$$

```
In [2]: NN = Neural_Network()

In [3]: cost1 = NN.costFunction(X,y)

In [4]: dJdW1, dJdW2 = NN.costFunctionPrime(X,y)

In [5]: dJdW1
Out[5]: array([[ -0.0071096 , -0.01059837, -0.00094283],
               [ -0.00172302, -0.00234379, -0.00019984]])

In [6]: dJdW2
Out[6]: array([[ -0.0229961 ],
               [-0.01631712],
               [-0.02079302]])

In [7]: scalar = 3
NN.W1 = NN.W1 + scalar*dJdW1
NN.W2 = NN.W2 + scalar*dJdW2
cost2 = NN.costFunction(X,y)

In [8]: print cost1, cost2
[ 0.01906658] [ 0.02396064]

In [9]: dJdW1, dJdW2 = NN.costFunctionPrime(X,y)
NN.W1 = NN.W1 - scalar*dJdW1
NN.W2 = NN.W2 - scalar*dJdW2
cost3 = NN.costFunction(X,y)

In [10]: print cost2, cost3
[ 0.02396064] [ 0.01773225]
```

$$z^{(2)} = XW^{(1)} \quad (1)$$

$$a^{(2)} = f(z^{(2)}) \quad (2)$$

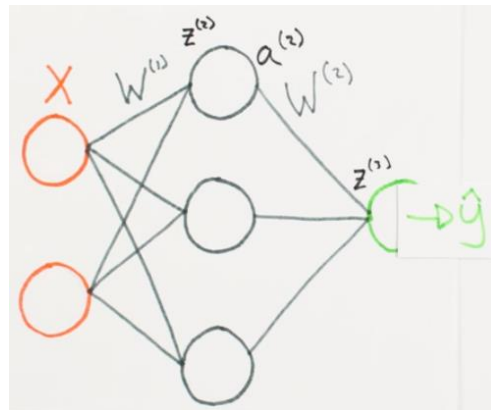
$$z^{(3)} = a^{(2)}W^{(2)} \quad (3)$$

$$\hat{y} = f(z^{(3)}) \quad (4)$$

$$J = \sum \frac{1}{2} (y - f(f(XW^{(1)})W^{(2)}))^2$$

How does this change if I change these?

$$\frac{\partial J}{\partial W}$$



$$W^{(1)} = \begin{bmatrix} W_{11}^{(1)} & W_{12}^{(1)} & W_{13}^{(1)} \\ W_{21}^{(1)} & W_{22}^{(1)} & W_{23}^{(1)} \end{bmatrix} \quad \frac{\partial J}{\partial W^{(1)}} = \begin{bmatrix} \frac{\partial J}{\partial W_{11}^{(1)}} & \frac{\partial J}{\partial W_{12}^{(1)}} & \frac{\partial J}{\partial W_{13}^{(1)}} \\ \frac{\partial J}{\partial W_{21}^{(1)}} & \frac{\partial J}{\partial W_{22}^{(1)}} & \frac{\partial J}{\partial W_{23}^{(1)}} \end{bmatrix}$$

$$W^{(2)} = \begin{bmatrix} W_{11}^{(2)} \\ W_{21}^{(2)} \\ W_{31}^{(2)} \end{bmatrix} \quad \frac{\partial J}{\partial W^{(2)}} = \begin{bmatrix} \frac{\partial J}{\partial W_{11}^{(2)}} \\ \frac{\partial J}{\partial W_{21}^{(2)}} \\ \frac{\partial J}{\partial W_{31}^{(2)}} \end{bmatrix}$$

MULTI-NEURON NETWORKS :: TRAINING

INITIALIZE NETWORK WITH RANDOM WEIGHTS

WHILE [NOT CONVERGED]

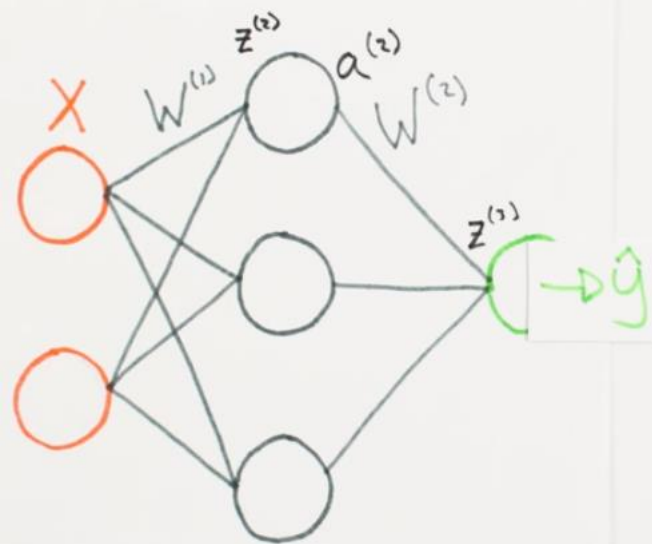
DO FORWARD PROP

DO BACKPROP AND DETERMINE CHANGE IN WEIGHTS

UPDATE ALL WEIGHTS IN ALL LAYERS



One
Iteration



$$z^{(2)} = XW^{(1)} \quad (1)$$

$$a^{(2)} = f(z^{(2)}) \quad (2)$$

$$z^{(3)} = a^{(2)}W^{(2)} \quad (3)$$

$$\hat{y} = f(z^{(3)}) \quad (4)$$

MULTI-NEURON NETWORKS :: TRAINING

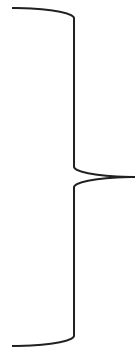
INITIALIZE NETWORK WITH RANDOM WEIGHTS

WHILE [NOT CONVERGED]

DO FORWARD PROP

DO BACKPROP AND DETERMINE CHANGE IN WEIGHTS

UPDATE ALL WEIGHTS IN ALL LAYERS



One
Iteration

MULTI-NEURON NETWORKS :: BACKPROPAGATION

$$J = \sum \frac{1}{2} (y - \hat{y})^2$$

$$\frac{\partial J}{\partial W^{(2)}} = (a^{(2)})^T \delta^{(3)}$$

$$\delta^{(3)} = -(y - \hat{y}) f'(z^{(3)})$$

$$\frac{\partial J}{\partial W^{(1)}} = X^T \delta^{(2)}$$

$$\delta^{(2)} = \delta^{(3)} (W^{(2)})^T f'(z^{(2)})$$

$$z^{(2)} = XW^{(1)} \quad (1)$$

$$a^{(2)} = f(z^{(2)}) \quad (2)$$

$$z^{(3)} = a^{(2)}W^{(2)} \quad (3)$$

$$\hat{y} = f(z^{(3)}) \quad (4)$$

$$W^{(1)} = \begin{bmatrix} w_{11}^{(1)} & w_{12}^{(1)} & w_{13}^{(1)} \\ w_{21}^{(1)} & w_{22}^{(1)} & w_{23}^{(1)} \end{bmatrix}$$

$$\frac{\partial J}{\partial W^{(1)}} = \begin{bmatrix} \frac{\partial J}{\partial w_{11}^{(1)}} & \frac{\partial J}{\partial w_{12}^{(1)}} & \frac{\partial J}{\partial w_{13}^{(1)}} \\ \frac{\partial J}{\partial w_{21}^{(1)}} & \frac{\partial J}{\partial w_{22}^{(1)}} & \frac{\partial J}{\partial w_{23}^{(1)}} \end{bmatrix}$$

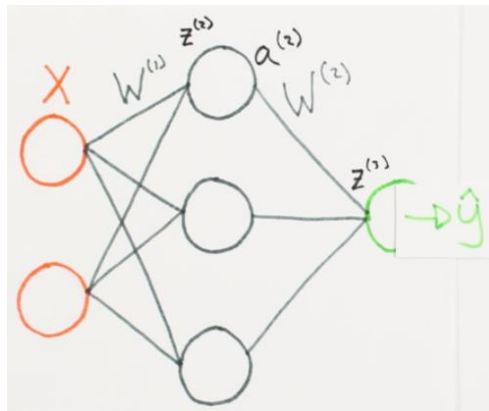
$$W^{(2)} = \begin{bmatrix} w_{11}^{(2)} \\ w_{21}^{(2)} \\ w_{31}^{(2)} \end{bmatrix}$$

$$\frac{\partial J}{\partial W^{(2)}} = \begin{bmatrix} \partial J / \partial w_{11}^{(2)} \\ \partial J / \partial w_{21}^{(2)} \\ \partial J / \partial w_{31}^{(2)} \end{bmatrix}$$

$$J = \sum \frac{1}{2} (y - f(f(XW^{(1)})W^{(2)}))^2$$

↑ HOW DOES THIS CHANGE IF I CHANGE THESE? ↑

$$\frac{\partial J}{\partial W}$$



MULTI-NEURON NETWORKS :: TRAINING

INITIALIZE NETWORK WITH RANDOM WEIGHTS

WHILE [NOT CONVERGED]

DO FORWARD PROP

DO BACKPROP AND DETERMINE CHANGE IN WEIGHTS

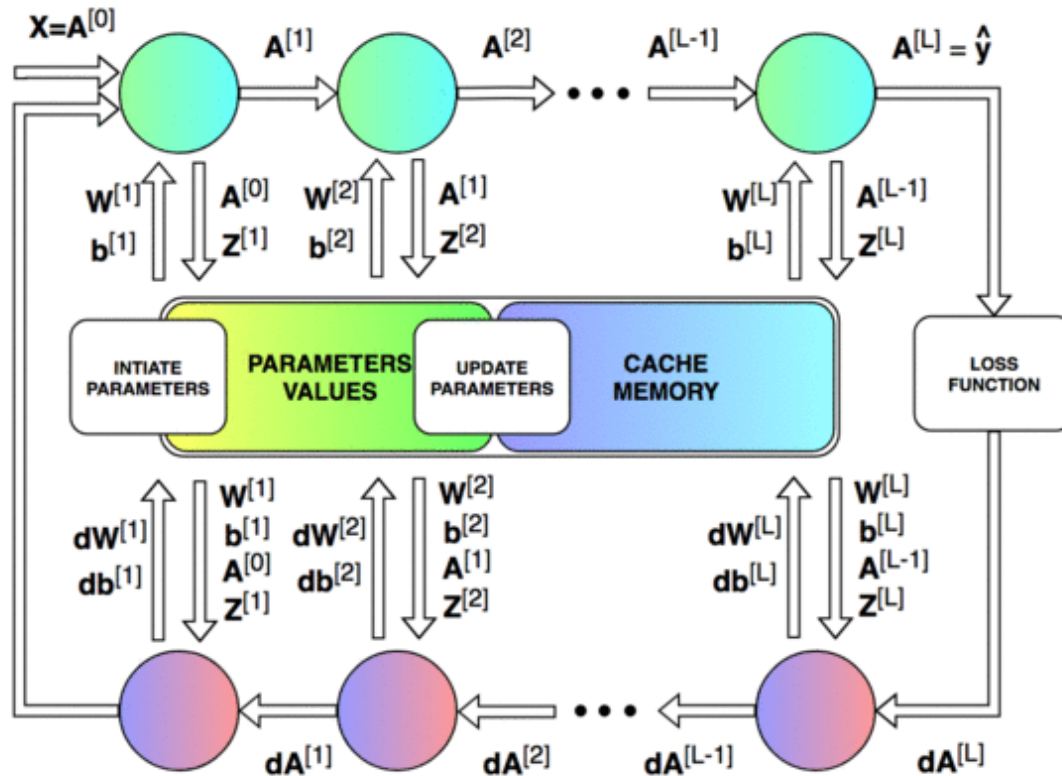
UPDATE ALL WEIGHTS IN ALL LAYERS

$$\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \eta^{(t)} \nabla_{\mathbf{w}} \mathbf{J}(\mathbf{w})$$



One
Iteration

FORWARD PROPAGATION



BACKWARD PROPAGATION

MULTI-NEURON NETWORK — NUMERICAL GRADIENT CHECKING

$$f'(x) = \lim_{\Delta x \rightarrow 0} \frac{f(x+\Delta x) - f(x)}{\Delta x}$$

$$f(x) = x^2$$

MULTI-NEURON NETWORK — NUMERICAL GRADIENT CHECKING

$$f'(x) = \lim_{\Delta x \rightarrow 0} \frac{f(x+\Delta x) - f(x)}{\Delta x}$$

$$f(x) = x^2$$

$$f'(x) = \lim_{\Delta x \rightarrow 0} \frac{(x+\Delta x)^2 - x^2}{\Delta x}$$

$$= \lim_{\Delta x \rightarrow 0} \frac{x^2 + 2x\Delta x + \Delta x^2 - x^2}{\Delta x}$$

$$= \lim_{\Delta x \rightarrow 0} \frac{\Delta x^2 + 2x\Delta x}{\Delta x}$$

$$= \lim_{\Delta x \rightarrow 0} \Delta x + 2x = \boxed{2x}$$

MULTI-NEURON NETWORK — NUMERICAL GRADIENT CHECKING

$$f'(x) = \lim_{\Delta x \rightarrow 0} \frac{f(x+\Delta x) - f(x)}{\Delta x}$$

$$f(x) = x^2$$

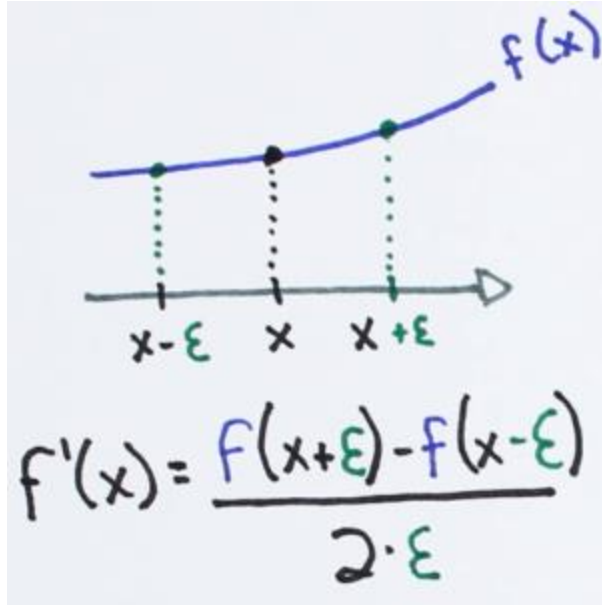
$$f'(x) = \lim_{\Delta x \rightarrow 0} \frac{(x+\Delta x)^2 - x^2}{\Delta x}$$

$$= \lim_{\Delta x \rightarrow 0} \frac{x^2 + 2x\Delta x + \Delta x^2 - x^2}{\Delta x}$$

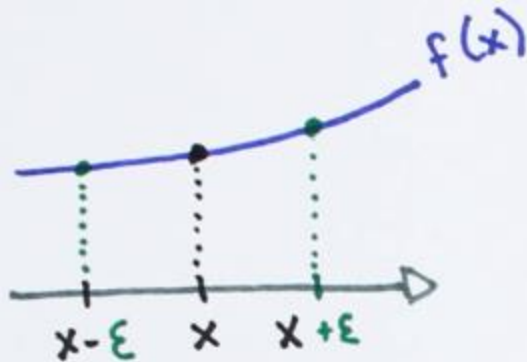
$$= \lim_{\Delta x \rightarrow 0} \frac{\Delta x^2 + 2x\Delta x}{\Delta x}$$

$$= \lim_{\Delta x \rightarrow 0} \Delta x + 2x = \boxed{2x}$$

MULTI-NEURON NETWORK — NUMERICAL GRADIENT CHECKING



MULTI-NEURON NETWORK — NUMERICAL GRADIENT CHECKING



$$f'(x) = \frac{f(x+\epsilon) - f(x-\epsilon)}{2 \cdot \epsilon}$$

```
In [4]: def f(x):  
        return x**2
```

```
In [5]: epsilon = 1e-4  
x = 1.5
```

```
In [6]: numericGradient = (f(x+epsilon) - f(x-epsilon)) / (2*epsilon)
```

```
In [7]: numericGradient, 2*x
```

```
Out[7]: (2.9999999999996696, 3.0)
```

```
In [ ]: |
```

MULTI-NEURON NETWORK — NUMERICAL GRADIENT CHECKING

Parameter vector θ

→ $\theta \in \mathbb{R}^n$ (E.g. θ is “unrolled” version of $\Theta^{(1)}$, $\Theta^{(2)}$, $\Theta^{(3)}$)

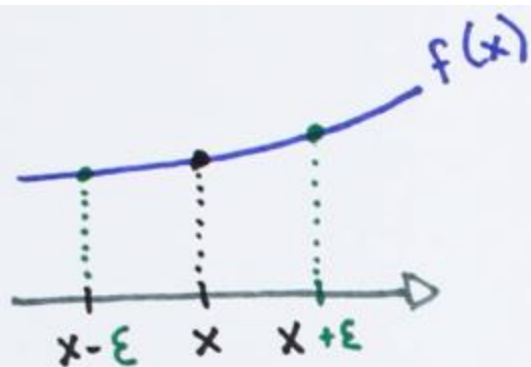
→ $\theta = [\theta_1, \theta_2, \theta_3, \dots, \theta_n]$

$$\rightarrow \frac{\partial}{\partial \theta_1} J(\theta) \approx \frac{J(\theta_1 + \epsilon, \theta_2, \theta_3, \dots, \theta_n) - J(\theta_1 - \epsilon, \theta_2, \theta_3, \dots, \theta_n)}{2\epsilon}$$

$$\frac{\partial}{\partial \theta_2} J(\theta) \approx \frac{J(\theta_1, \theta_2 + \epsilon, \theta_3, \dots, \theta_n) - J(\theta_1, \theta_2 - \epsilon, \theta_3, \dots, \theta_n)}{2\epsilon}$$

⋮

MULTI-NEURON NETWORK — NUMERICAL GRADIENT CHECKING



$$f'(x) = \frac{f(x+\epsilon) - f(x-\epsilon)}{2 \cdot \epsilon}$$

```
In [3]: numgrad = computeNumericalGradient(NN, X, y)
```

```
In [4]: grad = NN.computeGradients(X, y)
```

```
In [5]: numgrad
```

```
Out[5]: array([ -7.10752568e-03,  -6.30194392e-03,  -4.96392693e-03,
                -6.55946987e-04,   7.57595597e-05,  -1.11297012e-03,
                -8.81243102e-03,  -4.45550176e-03,  -1.93471143e-02])
```

```
In [6]: grad
```

```
Out[6]: array([ -7.10752569e-03,  -6.30194393e-03,  -4.96392693e-03,
                -6.55946985e-04,   7.57595604e-05,  -1.11297012e-03,
                -8.81243100e-03,  -4.45550176e-03,  -1.93471142e-02])
```

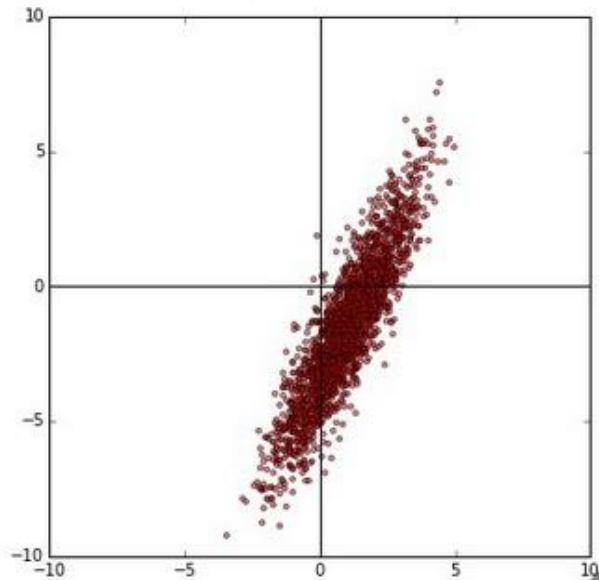
```
In [7]: norm(grad-numgrad)/norm(grad+numgrad)
```

```
Out[7]: 1.9824969610227768e-09
```

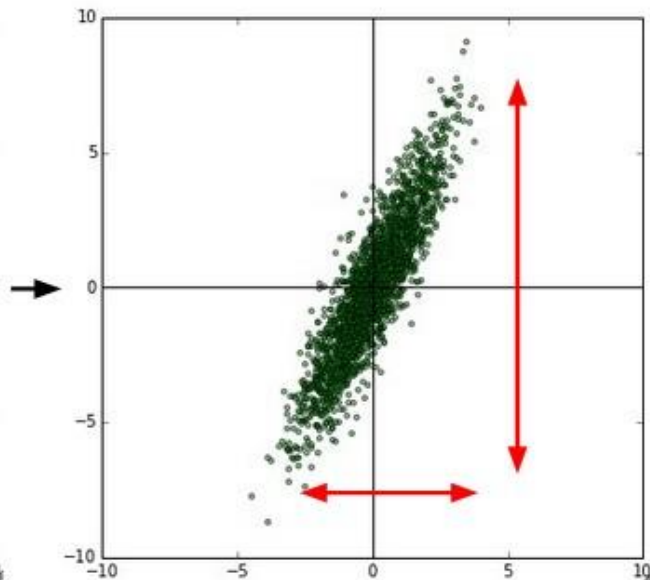
DATA SETUP

- Preprocessing:

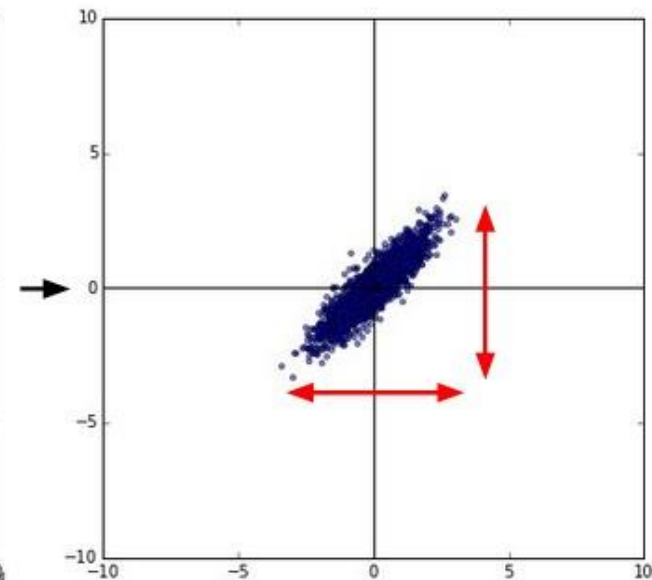
original data



zero-centered data



normalized data

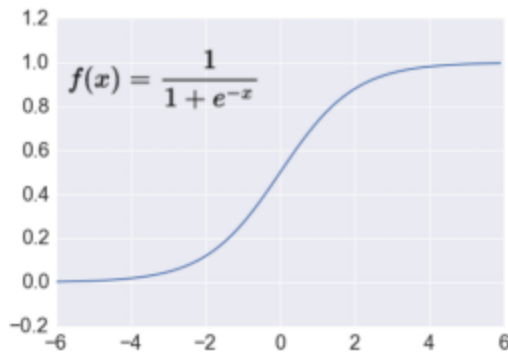


WEIGHT INITIALIZATION

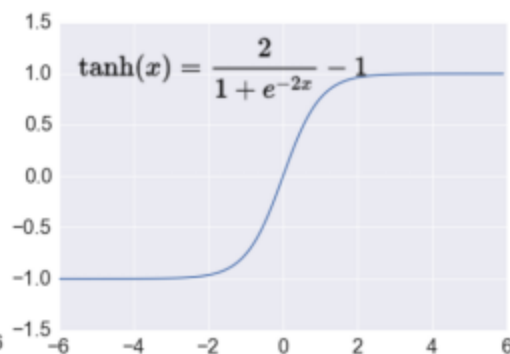
- ALL ZEROS
- `RANDOM[0,1]`
- `RANDOM[-1,1]`
- `w = np.random.randn(n) * sqrt(2.0/n)`, `n` = # of inputs to neuron

ACTIVATION FUNCTIONS

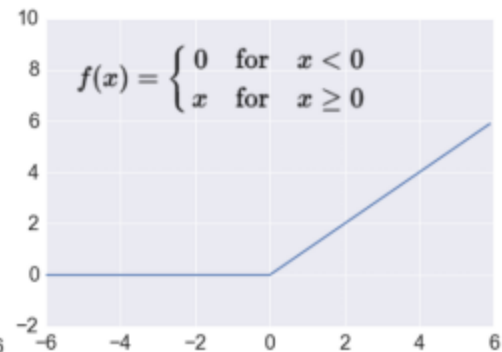
Sigmoid



TanH

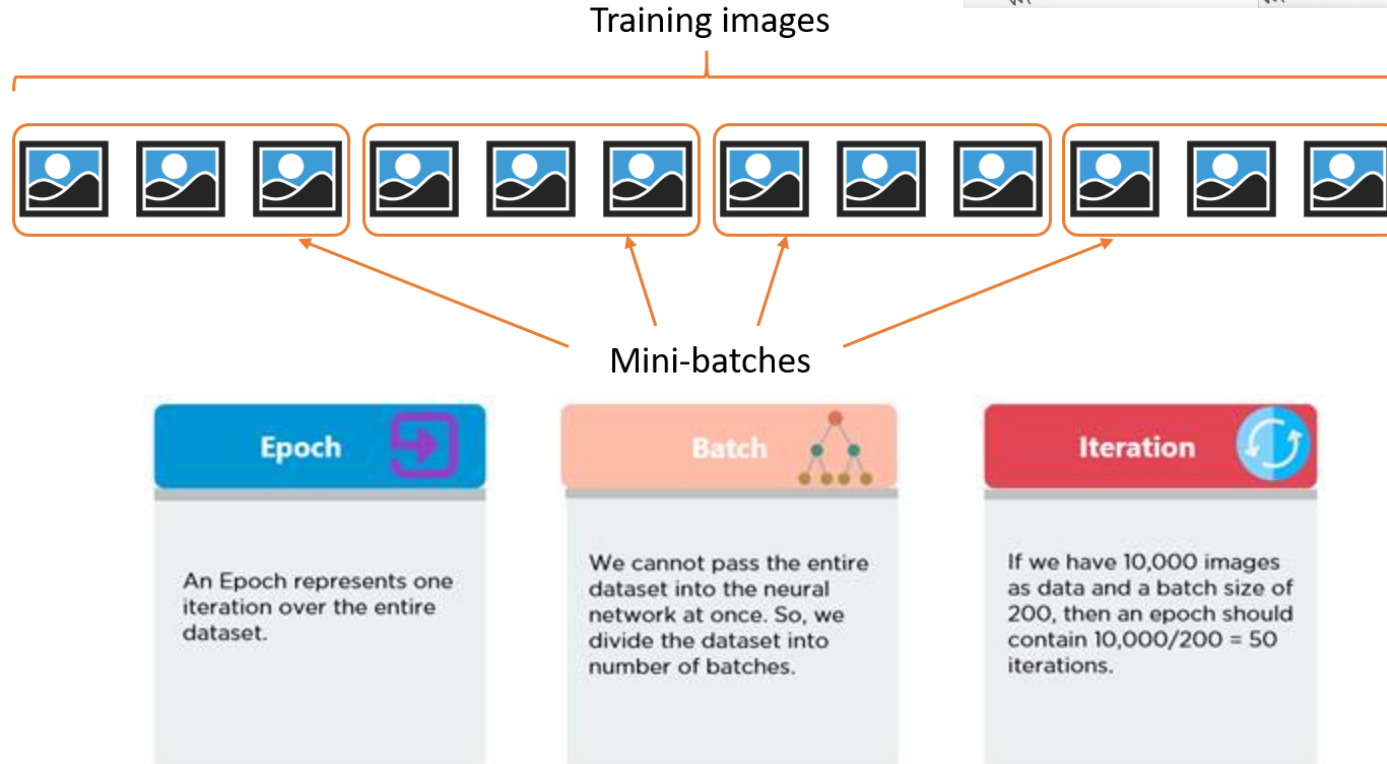
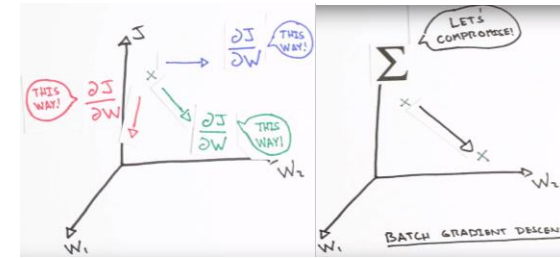


ReLU



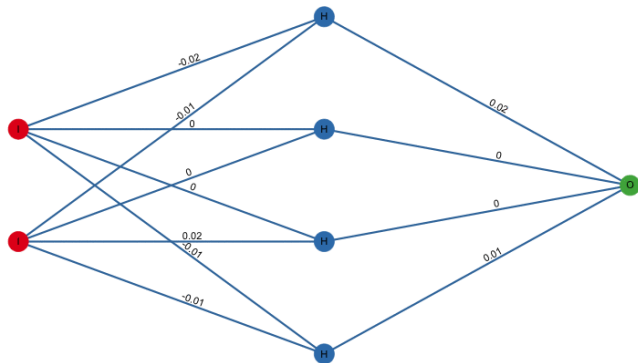
MINIBATCH VS SINGLE

- Average error, gradients



MULTI-NEURON NETWORKS :: TRAINING

Training a neural net at iteration 0



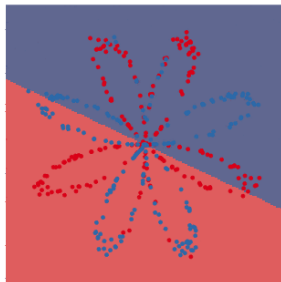
0.7

0.6

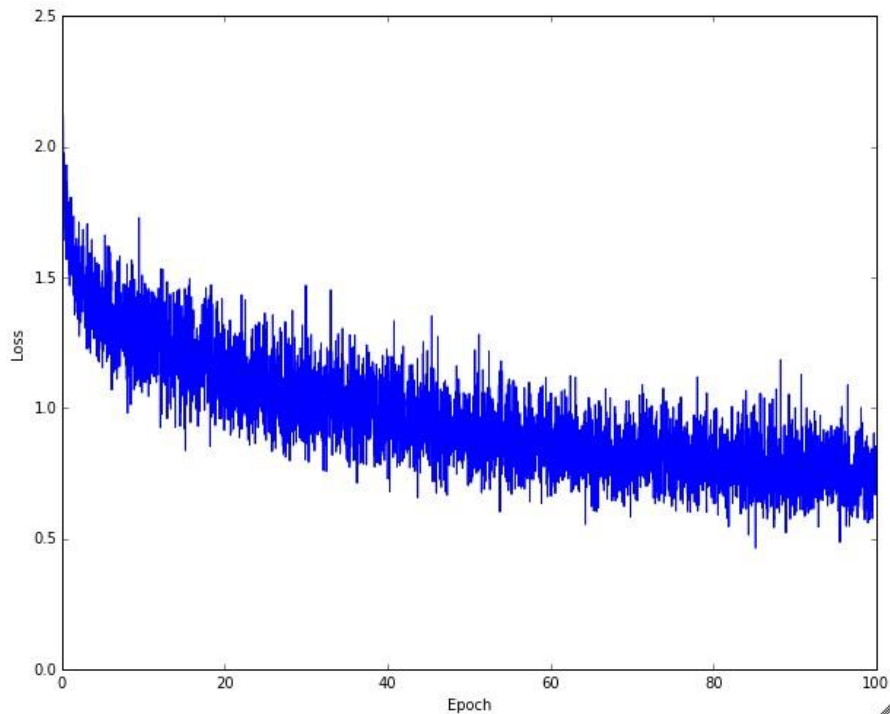
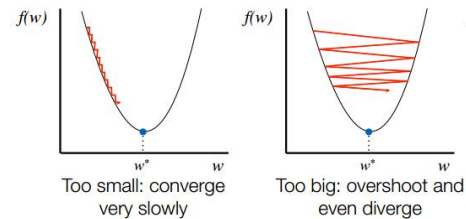
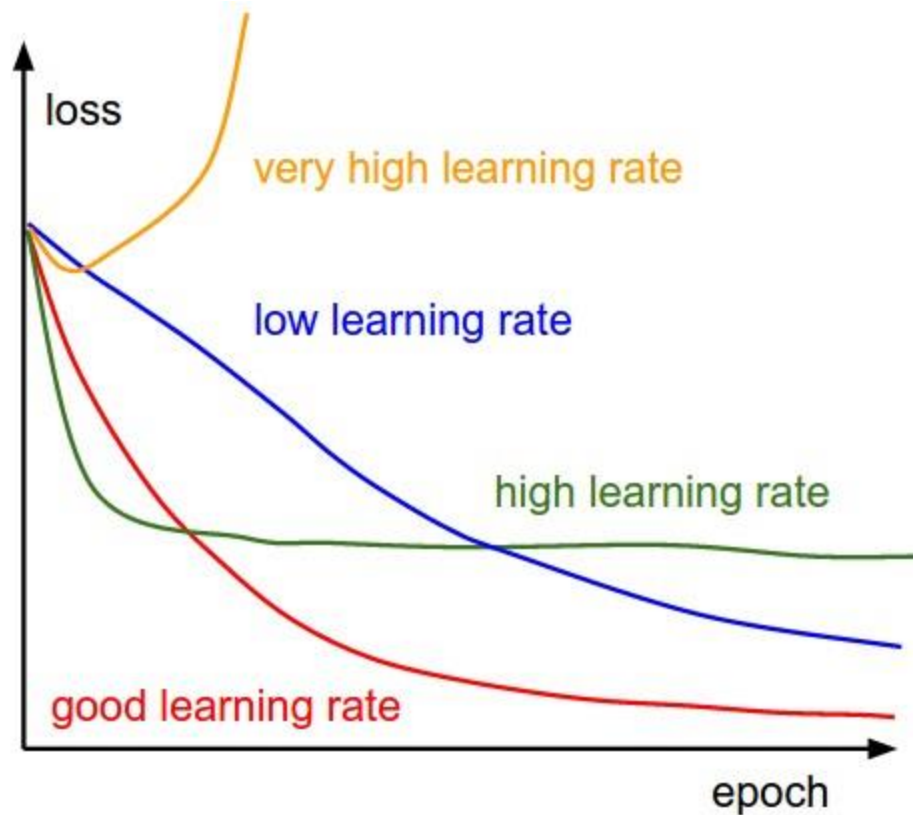
0.5

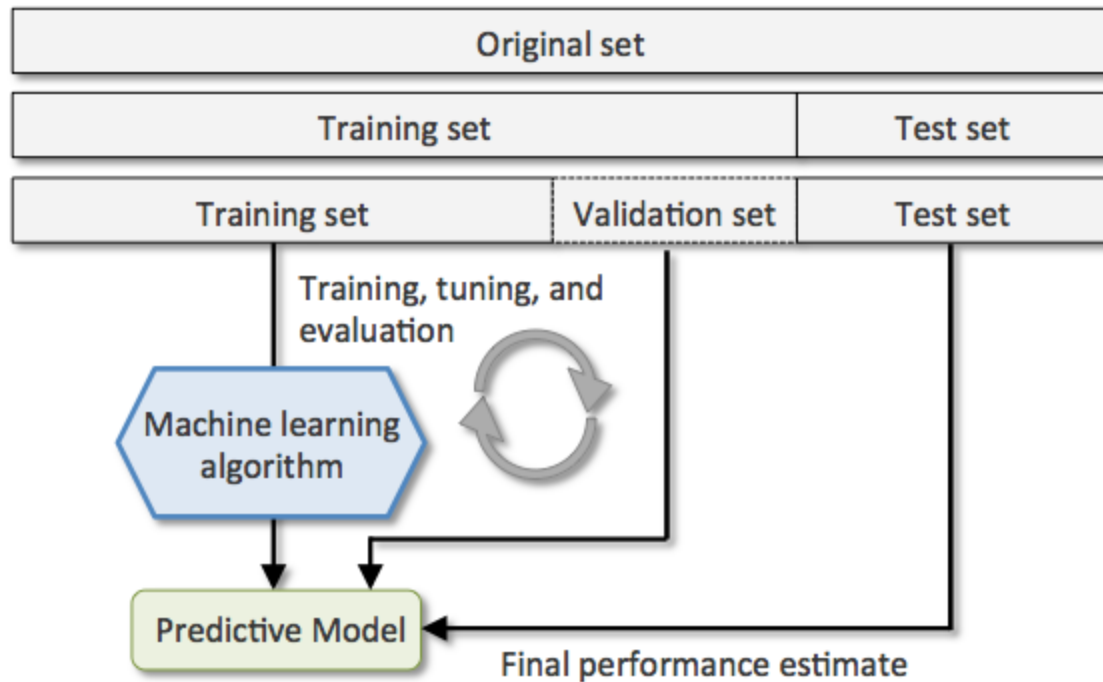
0.4

0.3

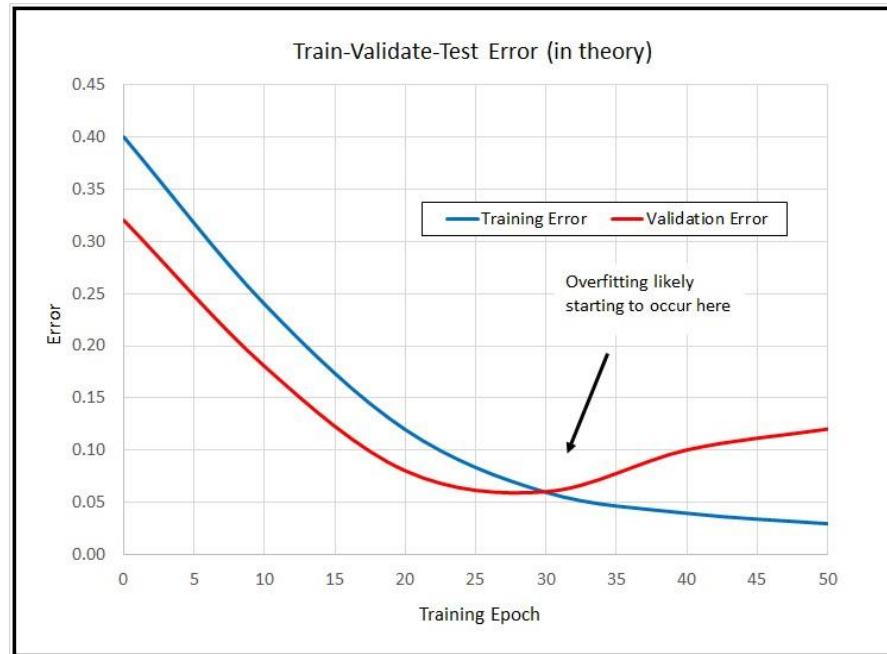
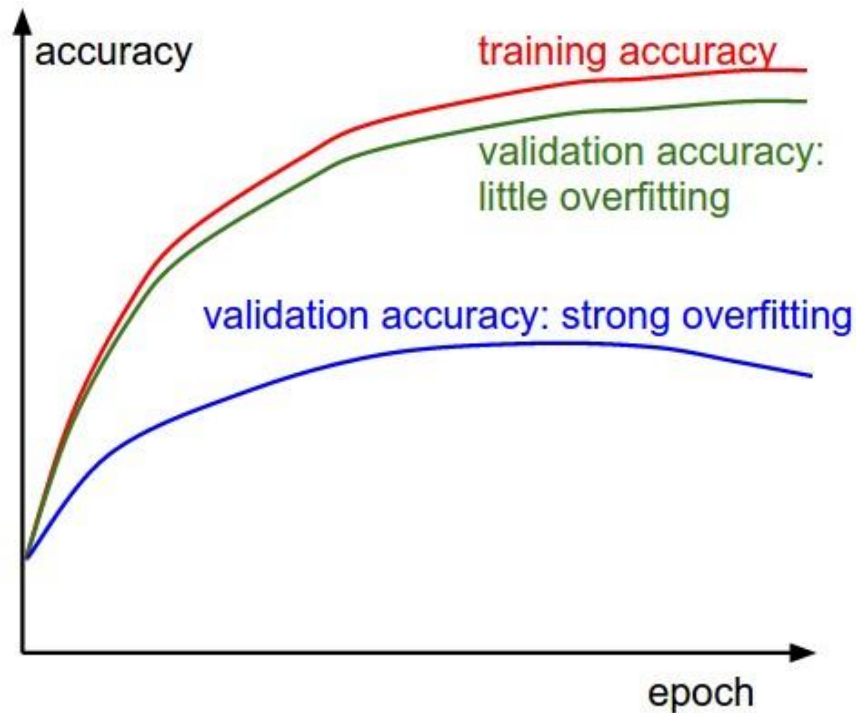


TRAINING — SETTING LEARNING RATE





WHEN TO STOP TRAINING



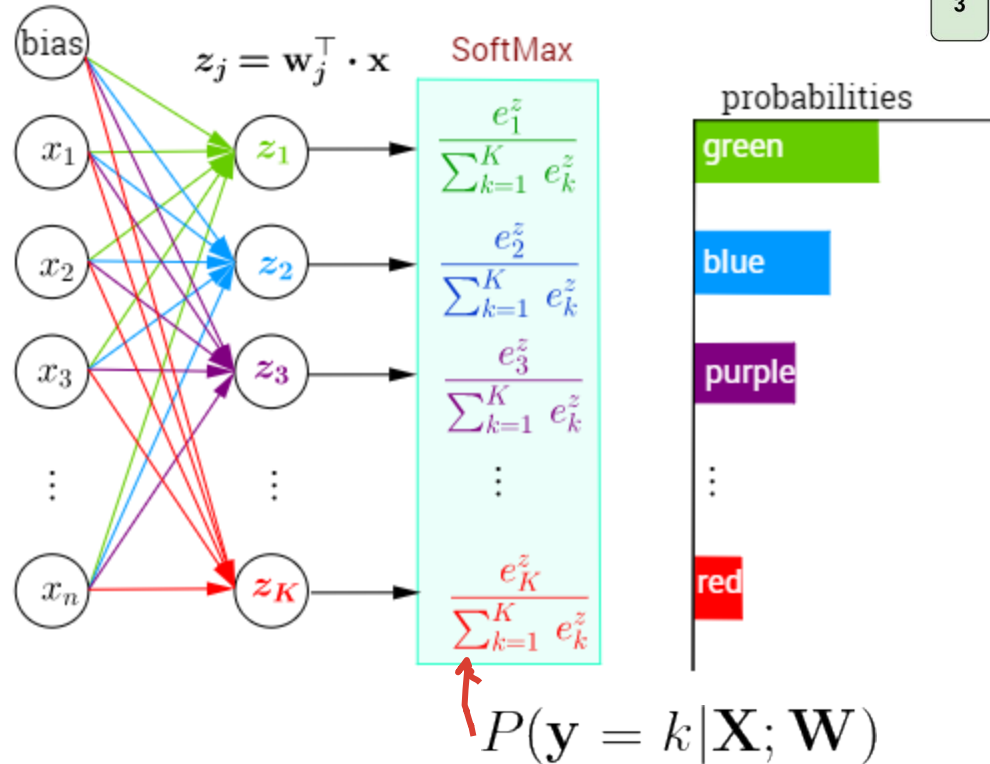
CLASSIFICATION

- How to represent class labels ?
- Diagram
- Loss Function

CLASSIFICATION LOSS

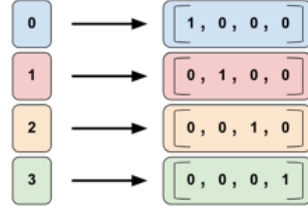
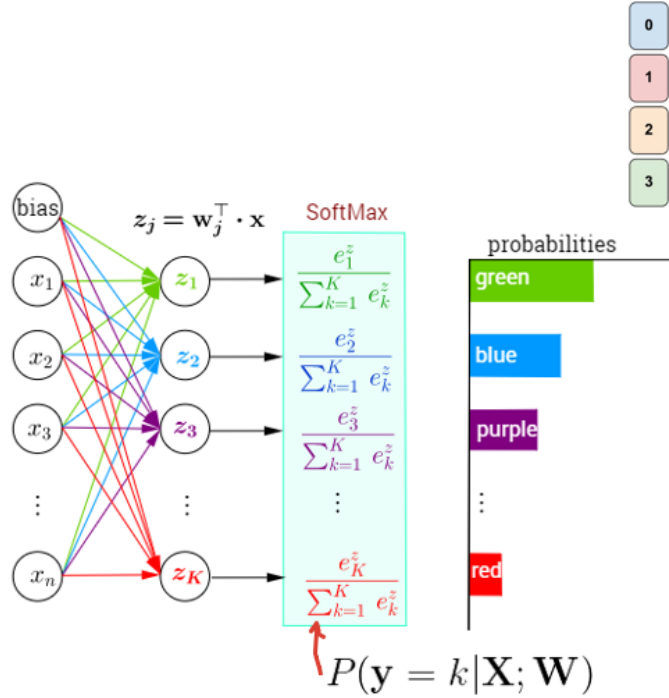
Multi-Class Classification with NN and SoftMax Function

$$\mathbf{z} = \begin{bmatrix} z_1 \\ z_2 \\ z_3 \\ \vdots \\ z_K \end{bmatrix} = \begin{bmatrix} \mathbf{w}_1^\top \\ \mathbf{w}_2^\top \\ \mathbf{w}_3^\top \\ \vdots \\ \mathbf{w}_K^\top \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix}$$



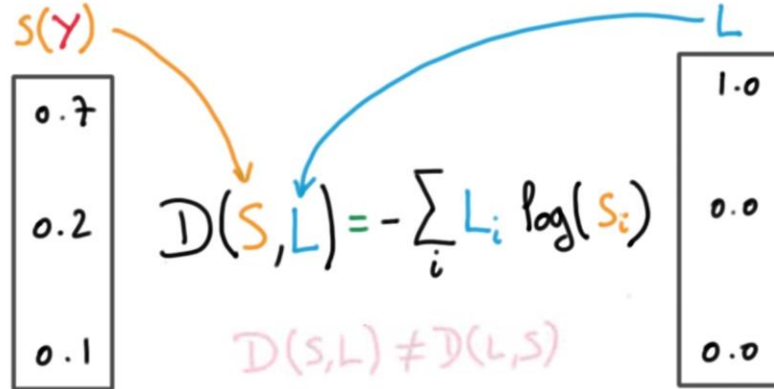
0	→	$\begin{bmatrix} 1, 0, 0, 0 \end{bmatrix}$
1	→	$\begin{bmatrix} 0, 1, 0, 0 \end{bmatrix}$
2	→	$\begin{bmatrix} 0, 0, 1, 0 \end{bmatrix}$
3	→	$\begin{bmatrix} 0, 0, 0, 1 \end{bmatrix}$

CLASSIFICATION LOSS



$$\begin{aligned} D_{\text{KL}}(p|q) &= \sum_i p_i \log \frac{p_i}{q_i} \\ &= \sum_i (-p_i \log q_i + p_i \log p_i) \\ &= -\sum_i p_i \log q_i + \sum_i p_i \log p_i \\ &= -\sum_i p_i \log q_i - \sum_i p_i \log \frac{1}{p_i} \\ &= -\sum_i p_i \log q_i - H(p) \\ &= \sum_i p_i \log \frac{1}{q_i} - H(p) \end{aligned}$$

CROSS-ENTROPY



CLASSIFICATION LOSS



computed				targets				correct?

0.3	0.3	0.4		0	0	1		yes
0.3	0.4	0.3		0	1	0		yes
0.1	0.2	0.7		1	0	0		no

Average Classification Error ?

CLASSIFICATION LOSS

A

computed				targets				correct?

0.3	0.3	0.4		0	0	1		yes
0.3	0.4	0.3		0	1	0		yes
0.1	0.2	0.7		1	0	0		no

Average Classification Error ?

B

computed				targets				correct?

0.1	0.2	0.7		0	0	1		yes
0.1	0.7	0.2		0	1	0		yes
0.3	0.4	0.3		1	0	0		no

Average Classification Error ?

CLASSIFICATION LOSS

A

computed				targets				correct?

0.3	0.3	0.4		0	0	1		yes
0.3	0.4	0.3		0	1	0		yes
0.1	0.2	0.7		1	0	0		no

Which classifier is better ?

B

computed				targets				correct?

0.1	0.2	0.7		0	0	1		yes
0.1	0.7	0.2		0	1	0		yes
0.3	0.4	0.3		1	0	0		no

CLASSIFICATION LOSS

A

computed	targets	correct?
0.3 0.3 0.4	0 0 1	yes
0.3 0.4 0.3	0 1 0	yes
0.1 0.2 0.7	1 0 0	no

Which classifier is better ?

B

computed	targets	correct?
0.1 0.2 0.7	0 0 1	yes
0.1 0.7 0.2	0 1 0	yes
0.3 0.4 0.3	1 0 0	no



Classification accuracy is a crude way to measure how well NN has been trained!

CLASSIFICATION LOSS

A

computed	targets	correct?
0.3 0.3 0.4	0 0 1	yes
0.3 0.4 0.3	0 1 0	yes
0.1 0.2 0.7	1 0 0	no

B

computed	targets	correct?
0.1 0.2 0.7	0 0 1	yes
0.1 0.7 0.2	0 1 0	yes
0.3 0.4 0.3	1 0 0	no



Classification accuracy is a crude way to measure how well NN has been trained!

CROSS-ENTROPY

$\mathcal{D}(S, L) = - \sum_i L_i \log(S_i)$

$\mathcal{D}(S, L) \neq \mathcal{D}(L, S)$

Diagram illustrating Cross-Entropy loss calculation. It shows two vectors: $S(Y)$ (0.7, 0.2, 0.1) and L (1.0, 0.0, 0.0). Arrows indicate the calculation of $\mathcal{D}(S, L)$ using the formula $\mathcal{D}(S, L) = - \sum_i L_i \log(S_i)$. The diagram also notes that $\mathcal{D}(S, L) \neq \mathcal{D}(L, S)$.

Cross-entropy error ?

CLASSIFICATION LOSS

A

MSE ?

computed		targets		correct?
0.3 0.3 0.4		0 0 1		yes
0.3 0.4 0.3		0 1 0		yes
0.1 0.2 0.7		1 0 0		no

B

computed		targets		correct?
0.1 0.2 0.7		0 0 1		yes
0.1 0.7 0.2		0 1 0		yes
0.3 0.4 0.3		1 0 0		no



$\ln()$ function in cross-entropy takes into account the closeness of a prediction and is a more granular way to compute error.

RESOURCES

- <https://playground.tensorflow.org/>
- <https://betterexplained.com/articles/derivatives-product-power-chain/>
- <http://www.3blue1brown.com/videos/2017/10/9/neural-network>
- <http://cs231n.github.io/neural-networks-2/>
- <http://cs231n.github.io/neural-networks-3>