BACKPROPAGATION

aniruddha kudalkar

THIS VIDEO COVERS

aniruddha kudalkar

- Training neural networks
- What is Backpropagation ?
- Gradient Descent

BACKPROPAGATION 🥹

aniruddha kudalkar

Backpropagation adjusts neural network weights by propagating errors backward through the network for learning.

Gradient descent optimizes functions by iteratively adjusting parameters in the direction of steepest descent.

DATASET

anirudha kudalkar

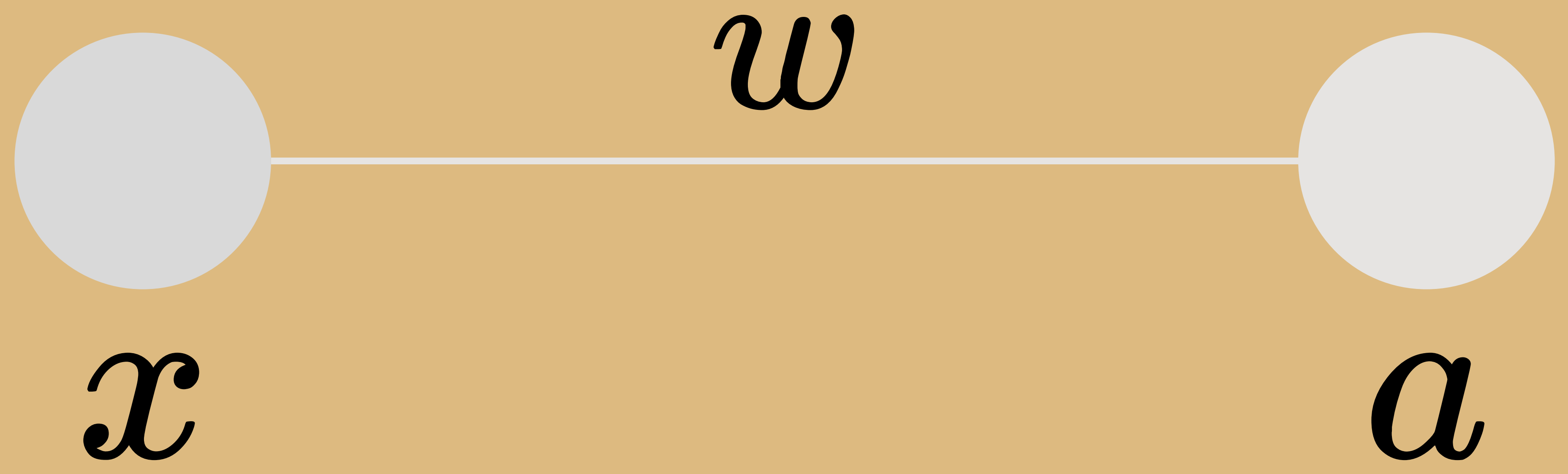
input (x)	desired output (y)
1.2	0.7

OUR NEURAL NETWORK

aniruddha kudalkar

- single input & output, one node; single weight
- no non linearity is considered
- no bias unit

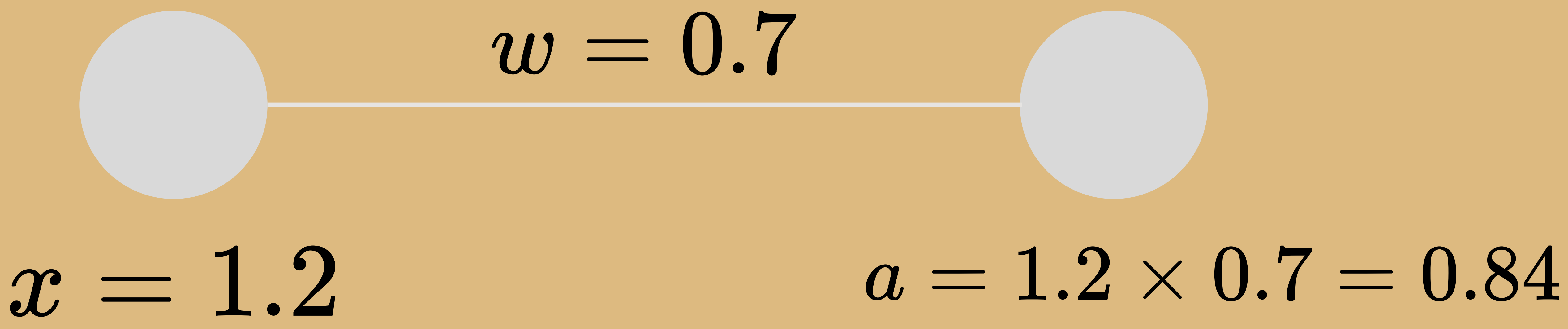
OUR NOTATIONS



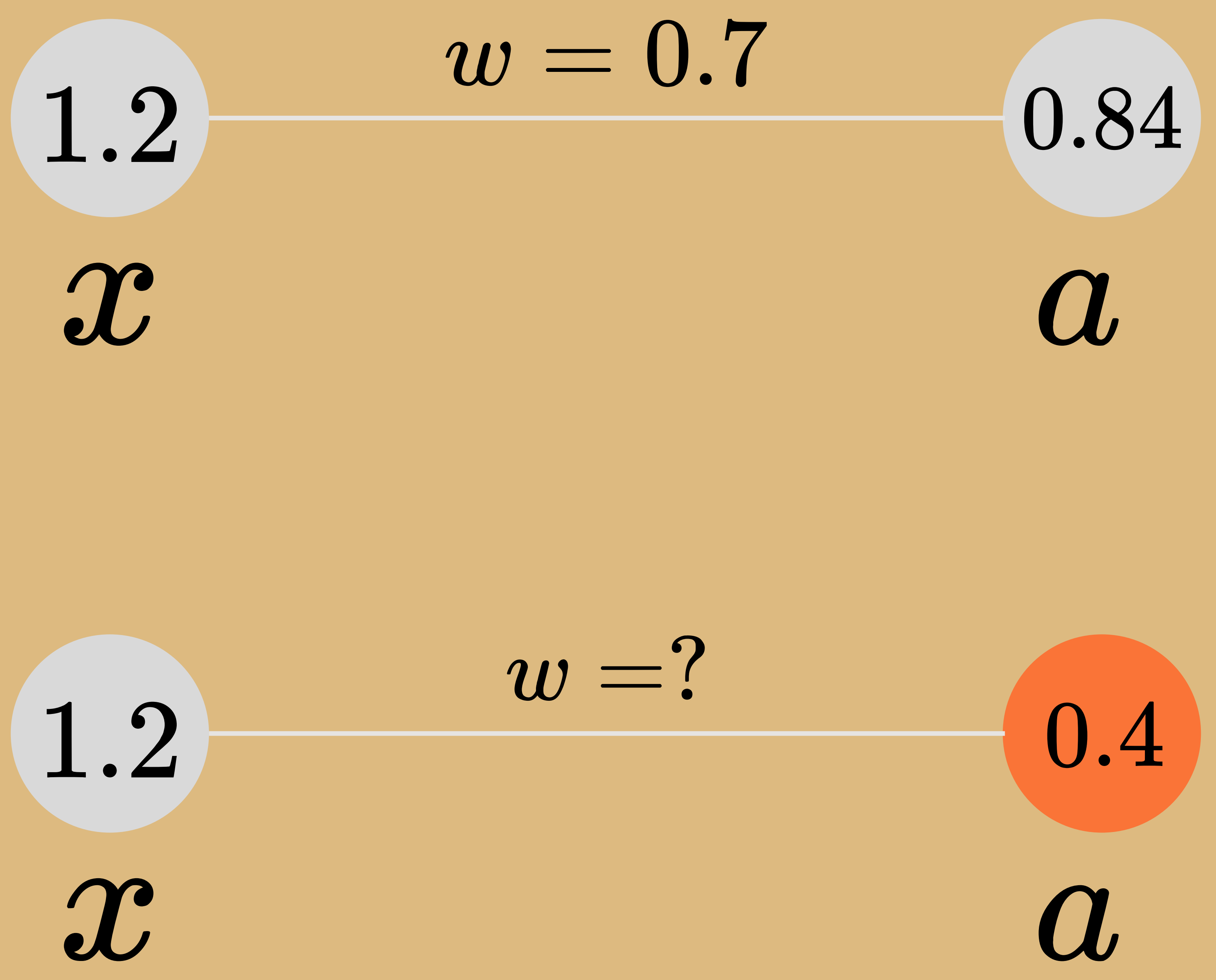
CALCULATION

aniruddha kudalkar

PREDICTION

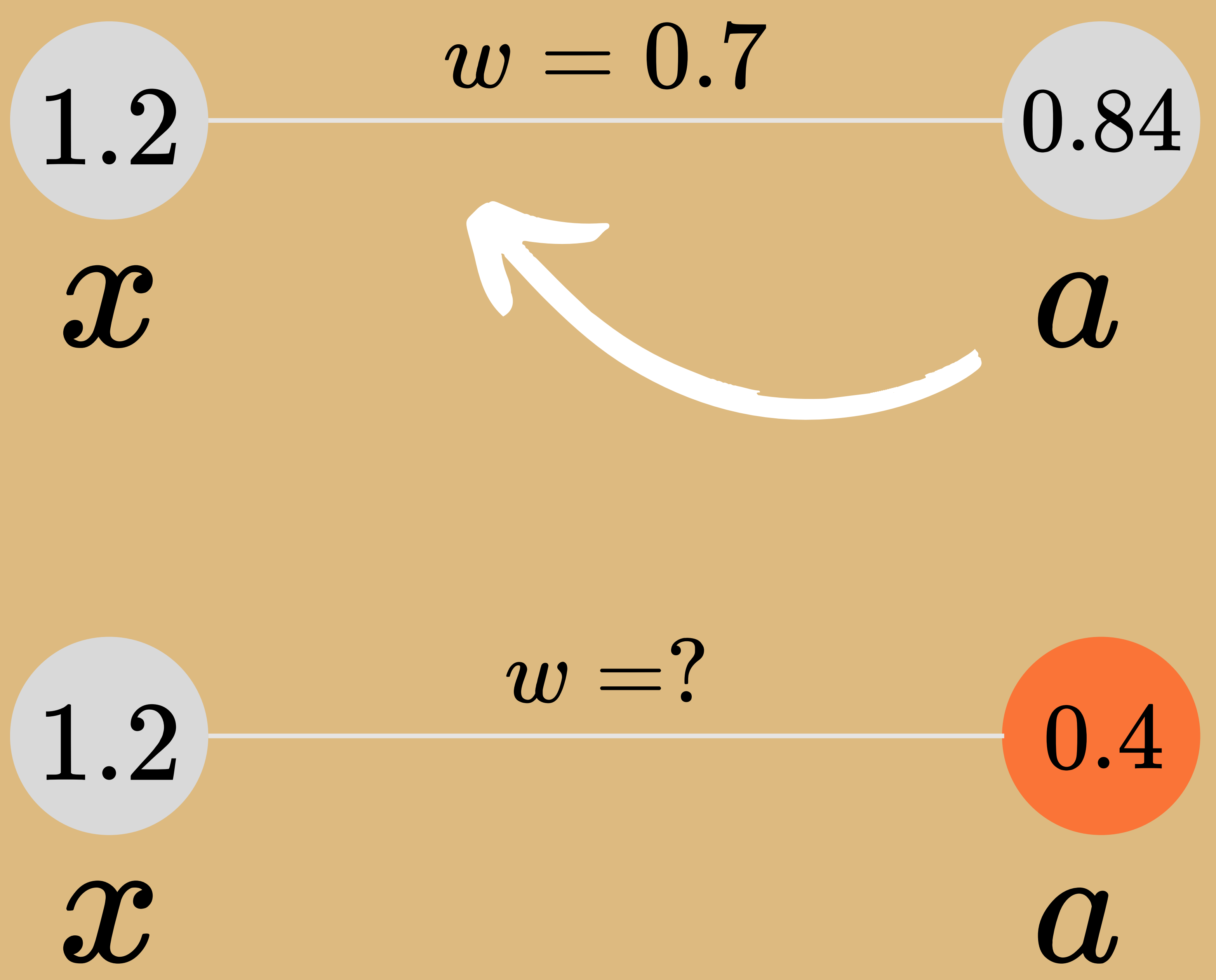


PREDICTION



WHAT CAN WE CHANGE ?

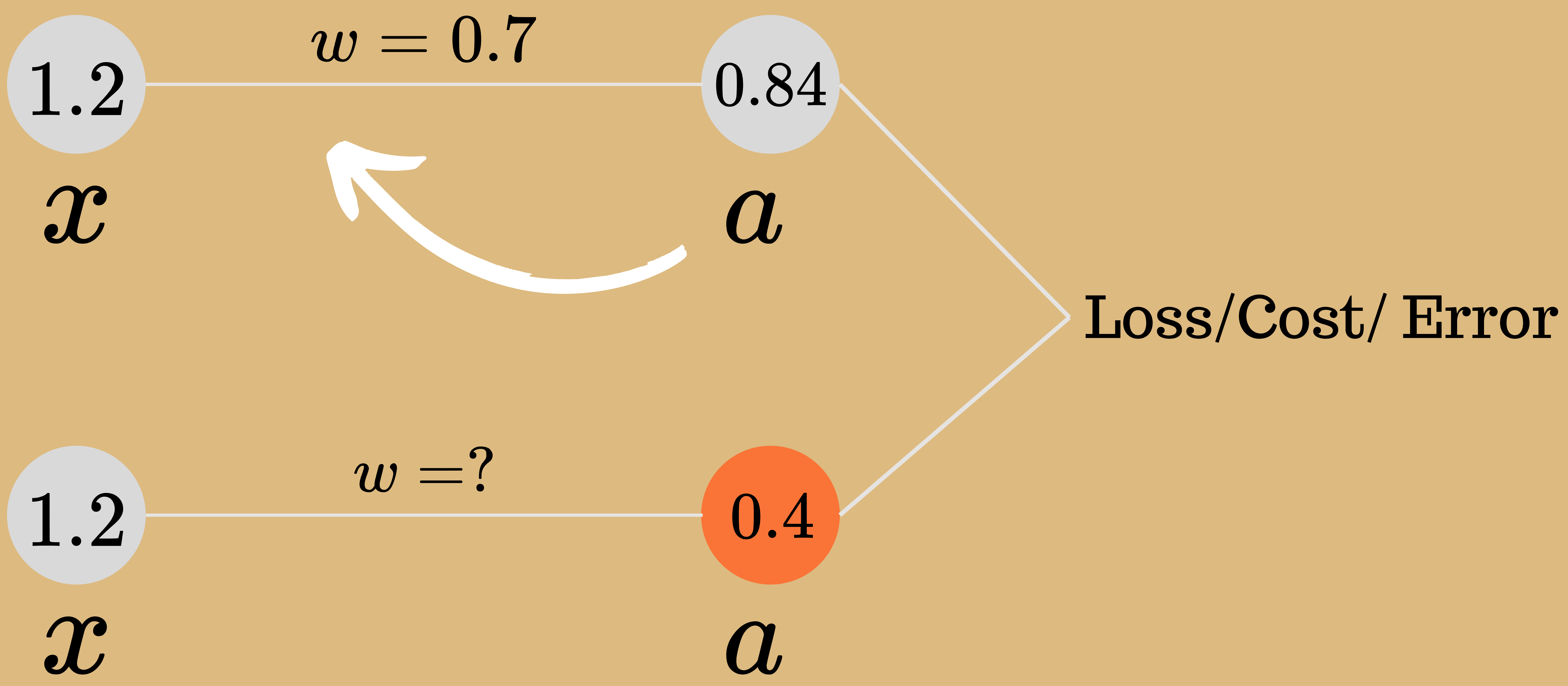
aniruddha kudalkar



- **a** is product of **x** and **w**, so we cant change **a** directly.
- We cant change **x** also as it input feature
- Only thing we can change is **w**, so it is called as <u>*trainable parameter.*</u>

WHAT CAN WE CHANGE ?

aniruddha kudalkar



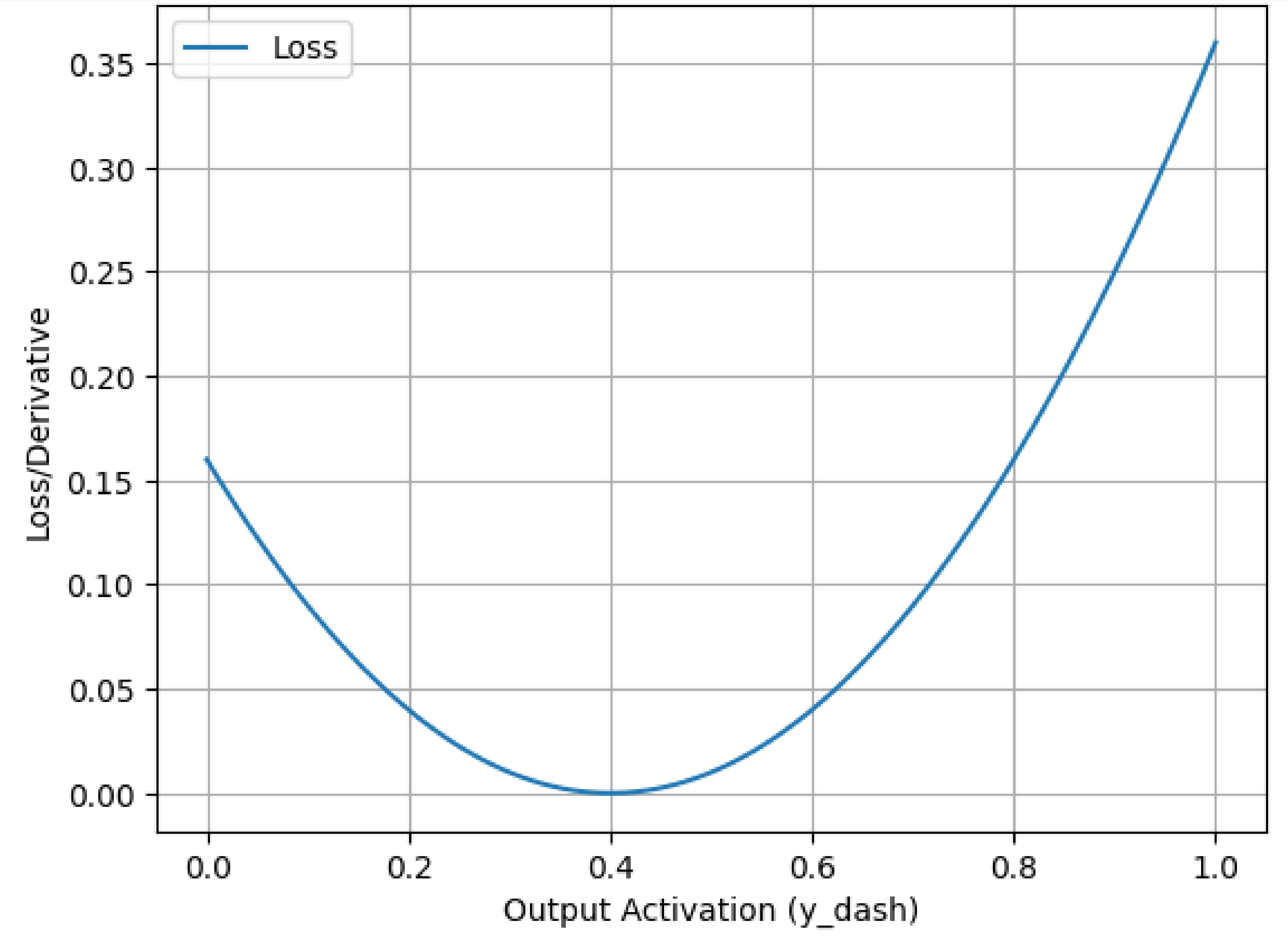
WHAT ?

COST FUNCTION

MSE
(Mean Squared Error)

$$loss = \left(\bar{y} - y\right)^2$$

difference squared



UPDATING WEIGHTS

aniruddha kudalkar

how much loss changes, if we make little change in weight.

it is the actual problem of Neural Networks i.e minimizing loss by updating weights

UPDATING WEIGHTS

aniruddha kudalkar

how much loss changes, if we make little change in weight i.e
we nee to find out rate of change loss w.r.t weight,
but how ?

MATHS says use DERIVATIVES

$$\frac{dloss}{dweights}$$

UPDATING WEIGHTS

aniruddha kudalkar

$$\frac{dloss}{dweights}$$

wait !! there is problem loss is not the directly associated with weight

UPDATING WEIGHTS

aniruddha kudalkar

cost is a function of a

$$C(a) = (a - y)^2$$

in turn, a is function of w

$$a(w) = x.w$$

Here we can see, one function is function of other

$$f(x) = f(g(x))$$

UPDATING WEIGHTS

aniruddha kudalkar

cost is a function of **a**

$$C(a) = (a - y)^2$$

in turn, **a** is function of **w**

$$a(w) = x.w$$

again maths says use CHAIN RULE of DERIVATIVES

$$\frac{d}{dx} f\left(g\left(x\right)\right) = g'\left(x\right). f'\left(g\right)$$

UPDATING WEIGHTS

aniruddha kudalkar

as output changes, loss also changes

$$\frac{dloss}{da} = ?$$

as weight changes, output also changes

$$\frac{da}{dweights} = ?$$

here is a chain rule for our example

$$\frac{dloss}{dweights} = \frac{dloss}{da} \times \frac{da}{dweights}$$

CALCULATE

Need to calculate derivative of loss wrt to output activation

$$\frac{dloss}{da} = ?$$

STEPS

1. $loss = \left(\bar{y} - y\right)^2$
2. Apply power rule in derivatives
3. $\frac{dloss}{da} = 2\left(\bar{y} - y\right)$

aniruddha kudalkar

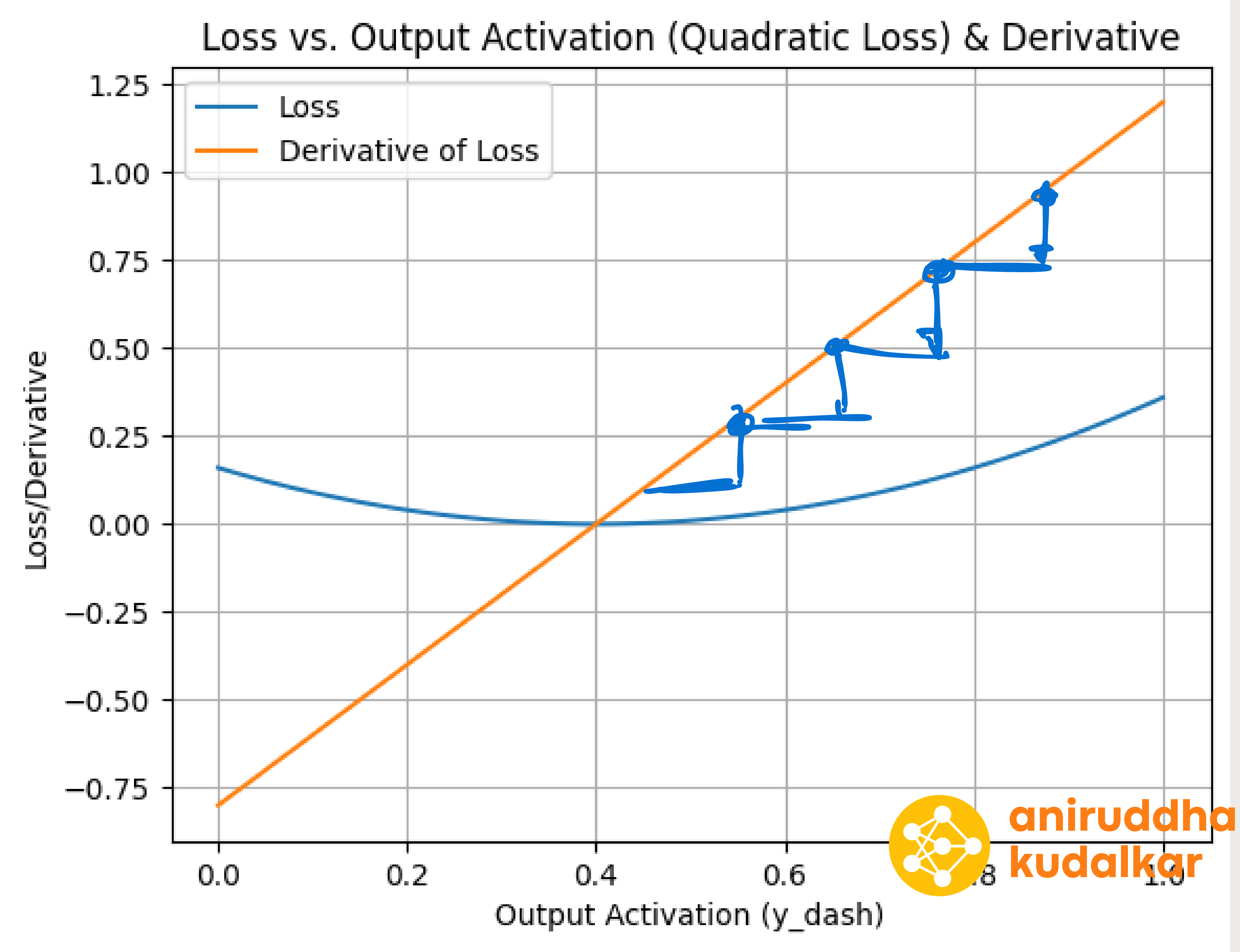
WHAT ?

SLOPE OF LOSS

Derivative of MSE

$$dloss = 2 \times \left(\bar{y} - y\right)$$

directions where we need to change the weight. -ve of gradient



aniruddha kudalkar

CALCULATE

Need to calculate derivative of output activation wrt weights

$$\frac{da}{dweights} = ?$$

STEPS

aniruddha kudalkar

1. $a = x.w$

2. derivative of equation of line is slope i.e a(w) = xw is x

3. $$\frac{da}{dweights} = x$$

LEARNING RATE

aniruddha kudalkar

$$lr = 0.1$$

it is the amount of adjustment we need make. Bigger value causes exploding gradients, and smaller value slows down training process

THE STEPS

Weight Adjustment

Learning Rate

Backpropagate adjustments

GRADIENT DESCENT

1. Slope of cost function gives adjustment direction. Adjust in -ve of gradient
2. Learning rate decide amount by which adjustment should be made.
3. Adjustments are backpropagated through all layers

aniruddha kudalkar

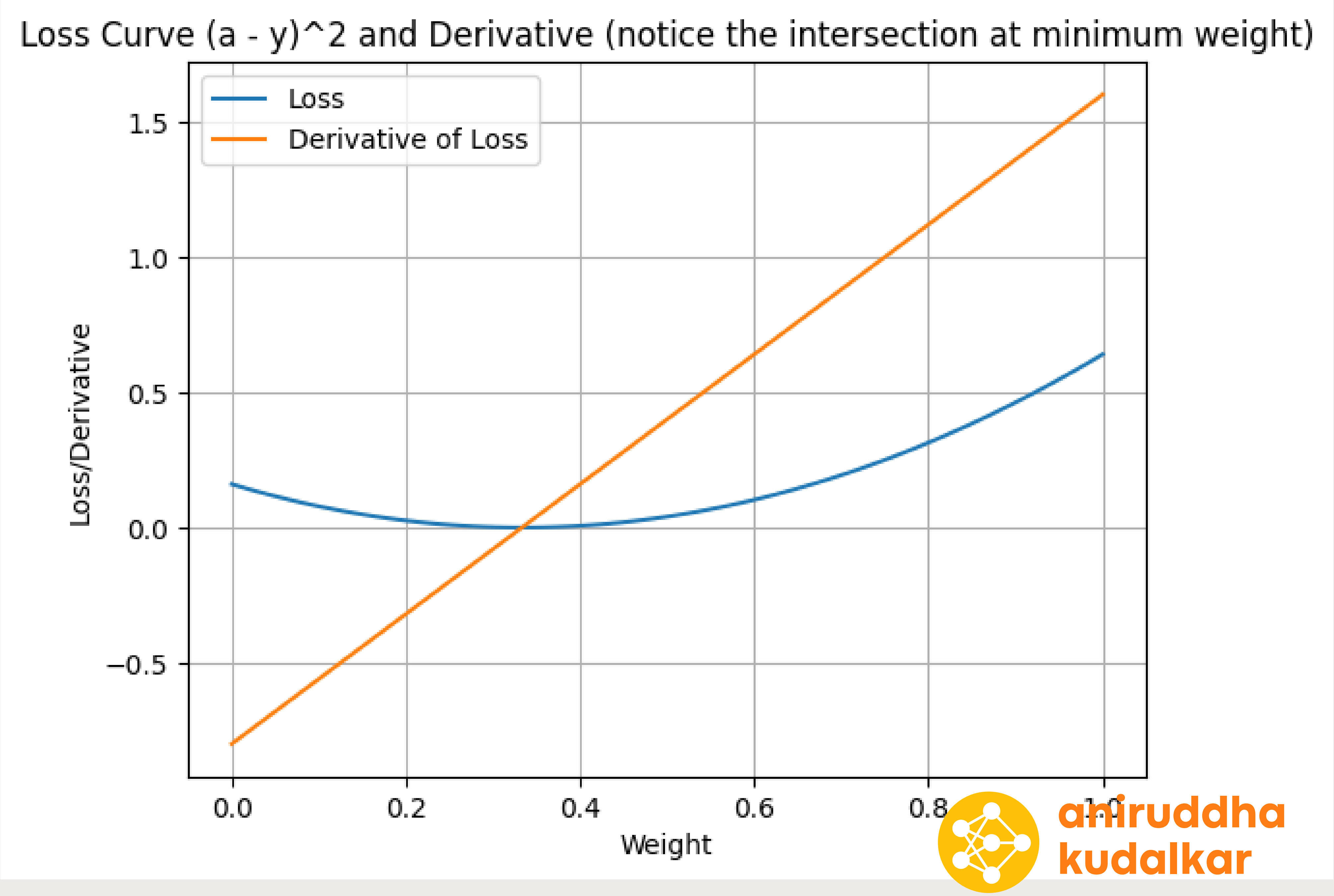
THE STEPS

Weight Adjustment

Learning Rate

Backpropagate adjustments

GRADIENT DESCENT



THE STEPS

Weight Adjustment

Learning Rate

Backpropagate adjustments

GRADIENT DESCENT

Loss 0.19359999999999997 Weight 0.5943999999999999 Output 0.84
Loss 0.09814435839999994 Weight 0.5192127999999999 Output 0.7132799999999999
Loss 0.04975369362472952 Weight 0.46567951359999993 Output 0.6230553599999998
Loss 0.025222336460894872 Weight 0.42756381368319996 Output 0.5588154163199999
Loss 0.012786312134831898 Weight 0.40042543534243835 Output 0.51307657641984
Loss 0.006481944218880216 Weight 0.3811029099638161 Output 0.480510522410926
Loss 0.003285982730096014 Weight 0.3673452718942371 Output 0.45732349195657934
Loss 0.0016658092291257965 Weight 0.3575498335886968 Output 0.44081432627308453
Loss 0.000844471993849945 Weight 0.3505754815151521 Output 0.42905980030643615
Loss 0.0004281000104502661 Weight 0.3456097428387883 Output 0.42069057781818253
Loss 0.00021702273169769994 Weight 0.3420741369012173 Output 0.414731691406546

anirudda
kudalkar