Our example will be to prove Moore's Law.

- Refer NB "2 Linear_Regression.ipynb"

Q.) What is Moore law?

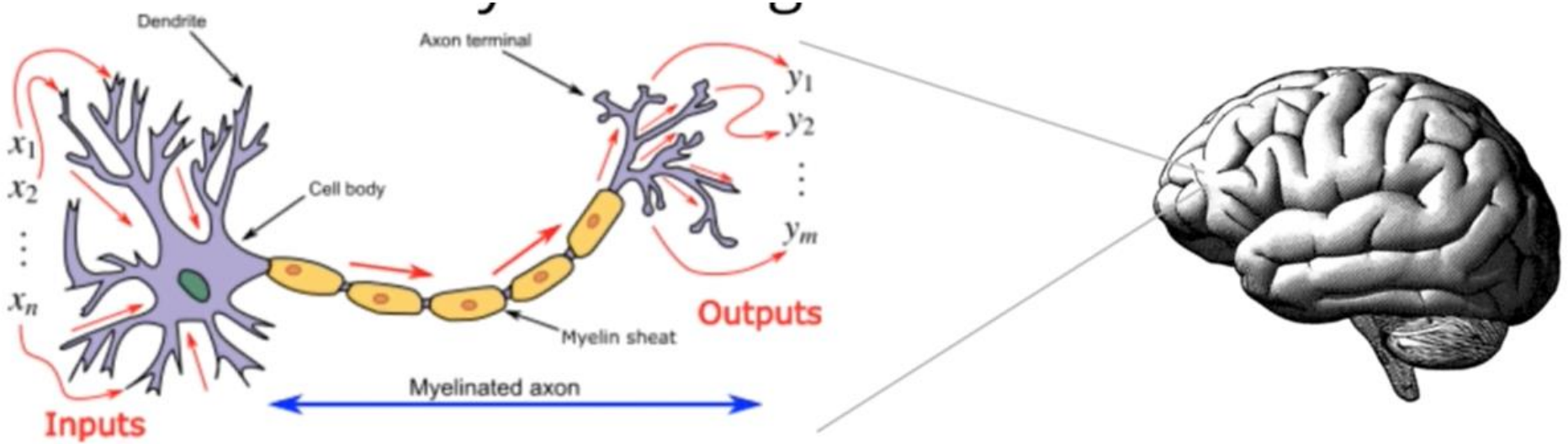
Soln: States that the # transistors / sq. inch on Integrated Circuits doubles approx. every 2 yrs. ⟹ Exponential Growth

∴

Standardization (Log)

Artificial Neural Network (ANN)

Where do ANN come from?



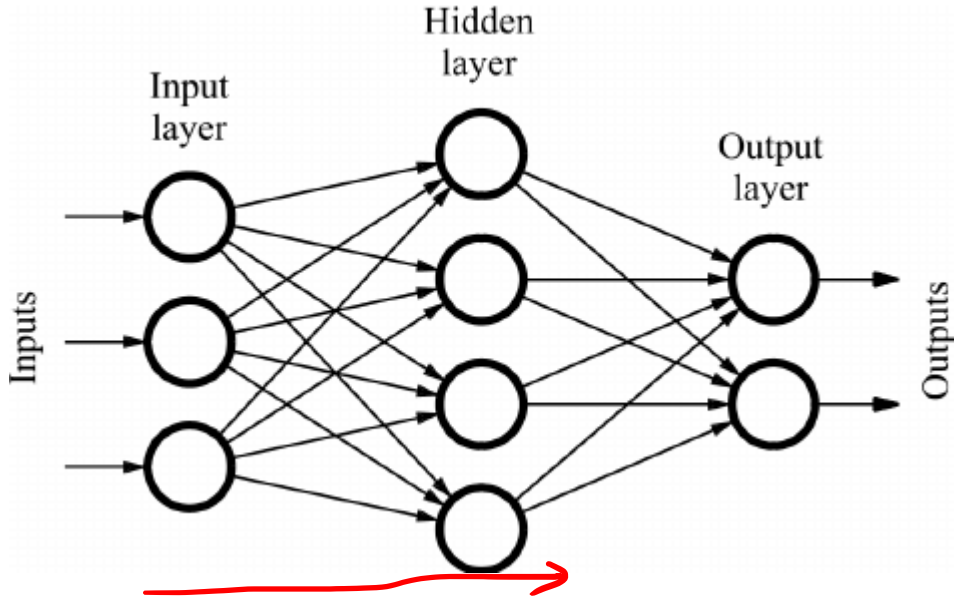
The obvious question

- We know that brain is made up of neurons, and we also know how neurons work (and thus can simulate them)
- It thus makes sense to ask: “Can we build brain?”
- If we connect a bunch of neurons together, will intelligence suddenly emerge?
- If so, it would be an Artificial Intelligence.

Feed Forward Neural Network

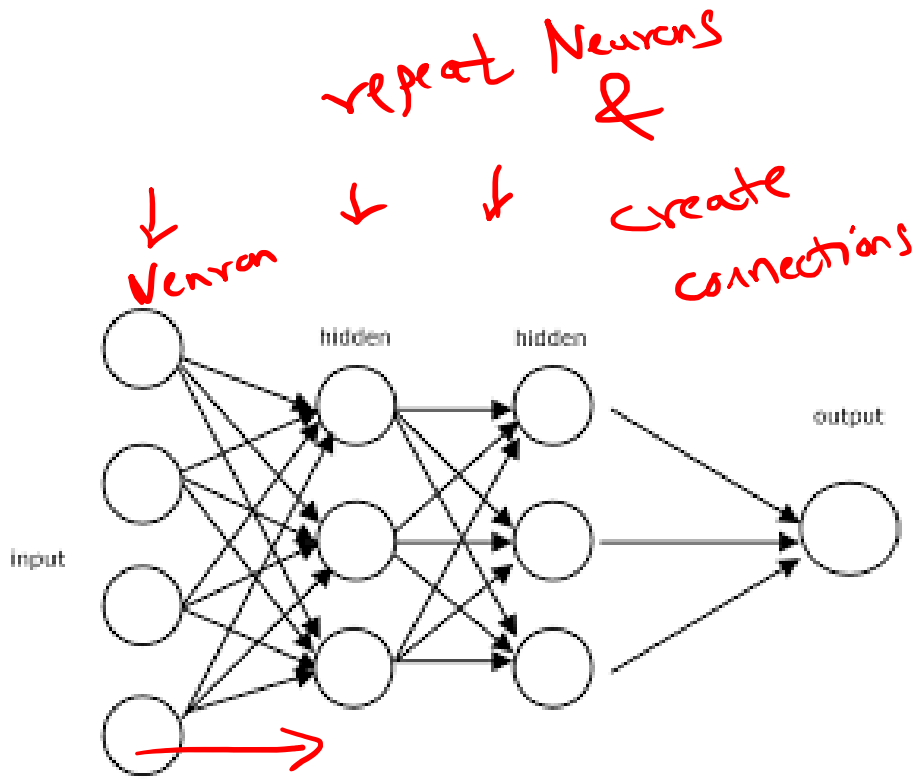
CNN (CV)
RNN (NLP)

Real brain: Wires can be criss-cross



Forward Propagation

Model — used for making predictions



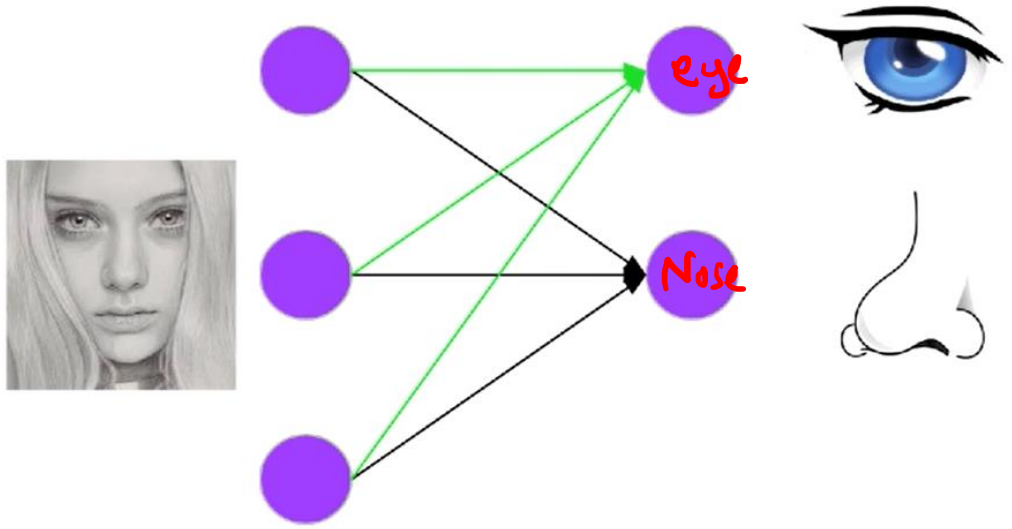
Repeating the single neuron

- Each of these neurons may be calculating something different, via different weights

eg: If the i/p is Face

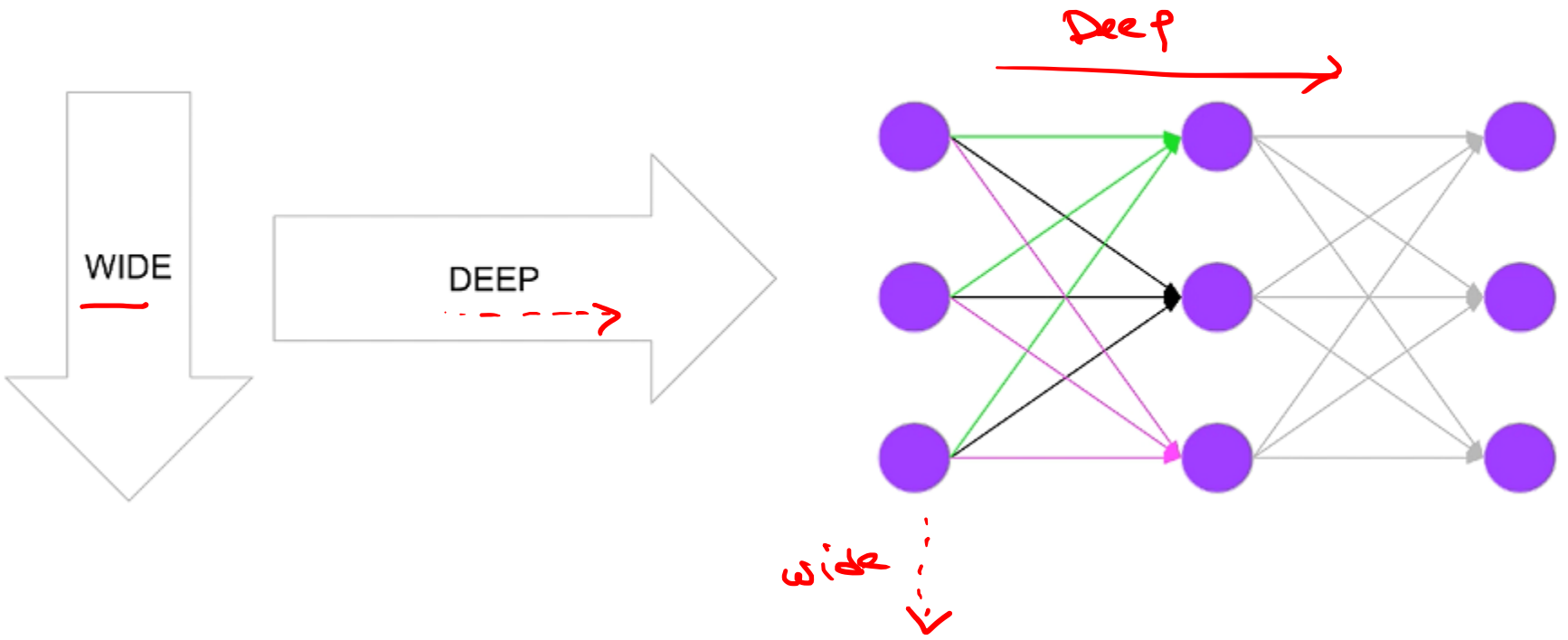
(a) One neuron → EYE
(b) One neuron → NOSE

} different features

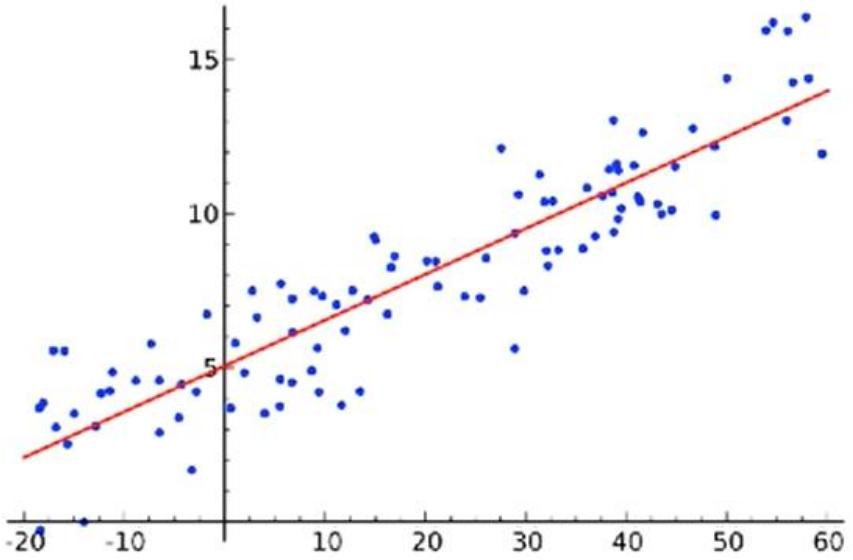


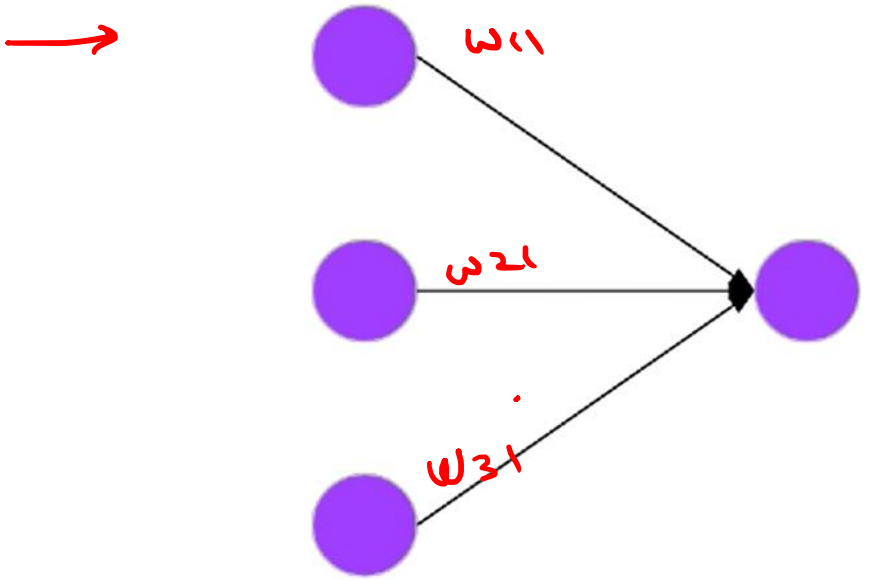
Two important ways to extend a single neuron

1. The same inputs can be fed to multiple different neurons, each calculating something different (more neurons per layer) WIDE
2. Neurons in one layer can act as inputs to another layer (DEEP)



Lines to Neurons



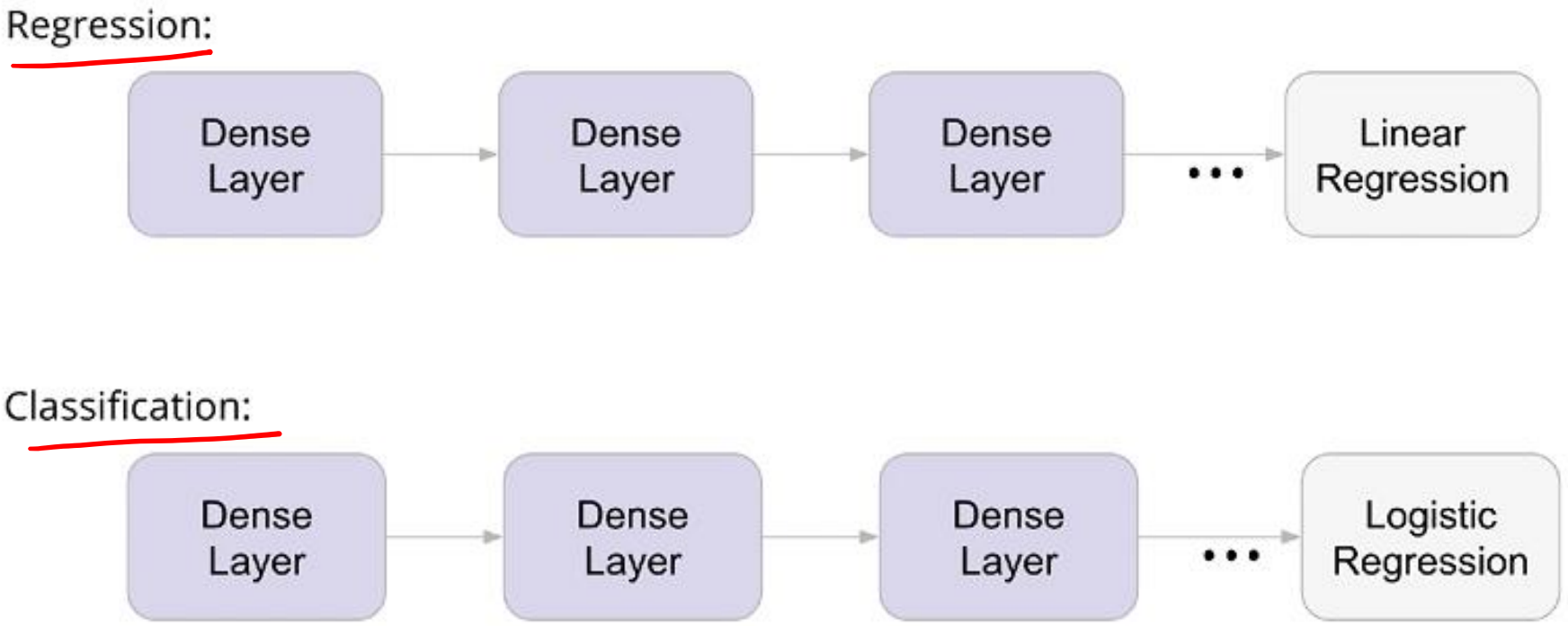


$$A\ line: \ ax + b$$

$$A\ neuron: \ \sigma(w^T x + b)$$

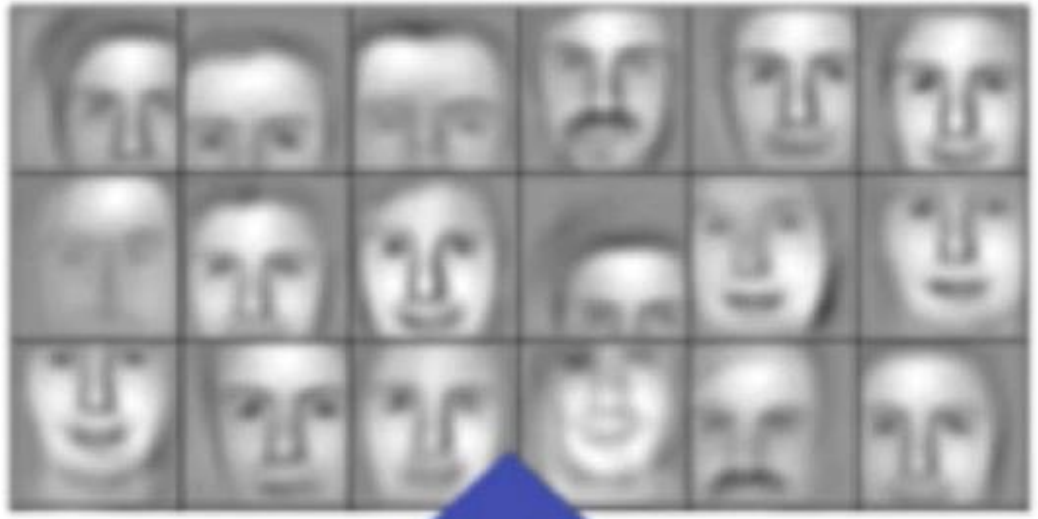
Another perspective

- Each neural network is a feature transformation

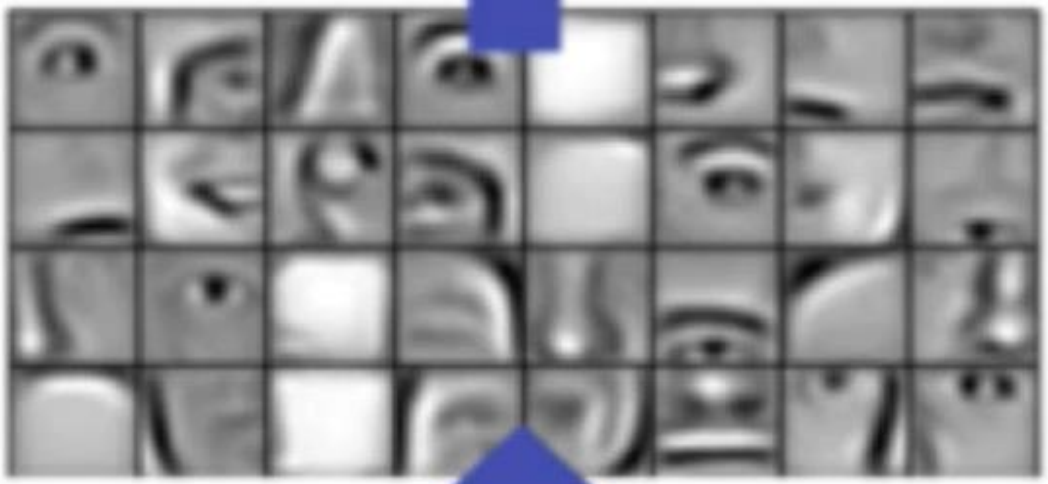


Feature hierarchies

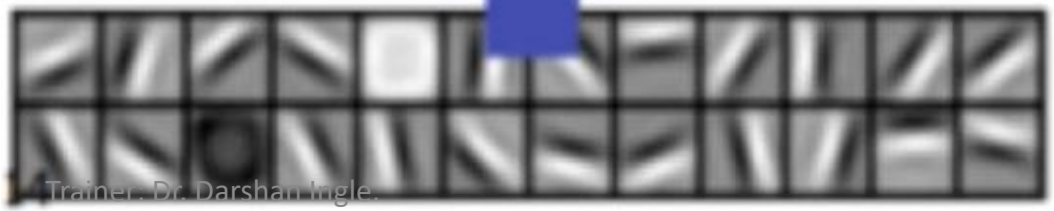
- Researchers noticed that each layer learns increasingly complex features



Layer 3



Layer 2



Layer 1

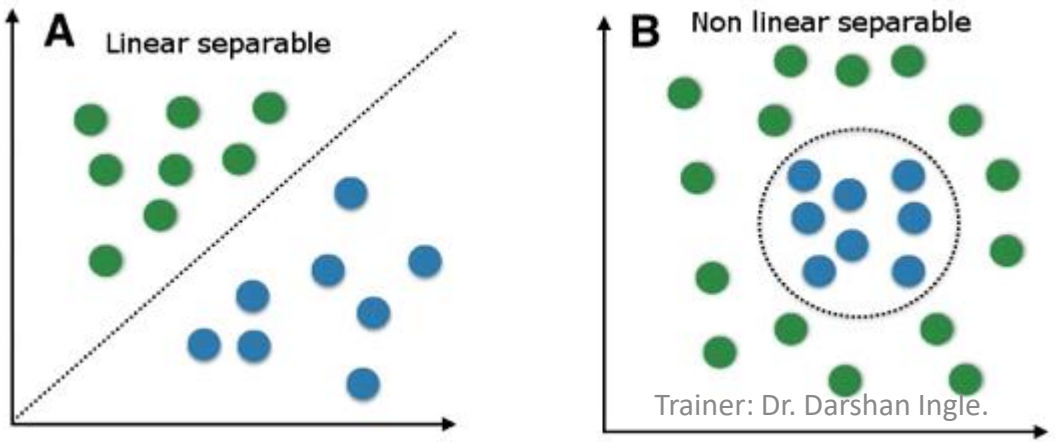
The Geometric Picture

- ML is nothing but a geometry problem
- Why are Neural Network so important?
- Why cant we just use a single neuron?
- The neuron is nice and interpretable
- Large weights = important feature
- Small weights = not important feature
- Unfortunately, the neuron (linear model) is not very expensive
- But true learning doesn’t happen with a single neuron

Making the line more complicated

- 2 ways to make our problem more complicated than “finding a line”

1. Adding more input dimensions
2. “Make the pattern non linear” (This is what we are concerned with now)



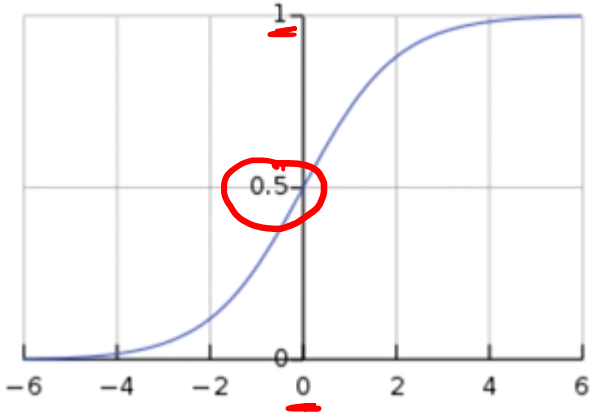
TensorFlow Playground

- https://playground.tensorflow.org/#activation=tanh&batchSize=10&dataset=circle&regDataset=reg-plane&learningRate=0.03&regularizationRate=0&noise=0&networkShape=4,2&seed=0.33964&showTestData=false&discretize=false&percTrainData=50&x=true&y=true&xTimesY=false&xSquared=false&ySquared=false&cosX=false&sinX=false&cosY=false&sinY=false&collectStats=false&problem=classification&initZero=false&hideText=false

Revisiting Activation Functions

Sigmoid: 0 and 1 (between)

$$\sigma(a) = \frac{1}{1+\exp(-a)}$$



mimics Biological Neuron

Problems? :. It is no longer used any more as exhaustively as it was used before.

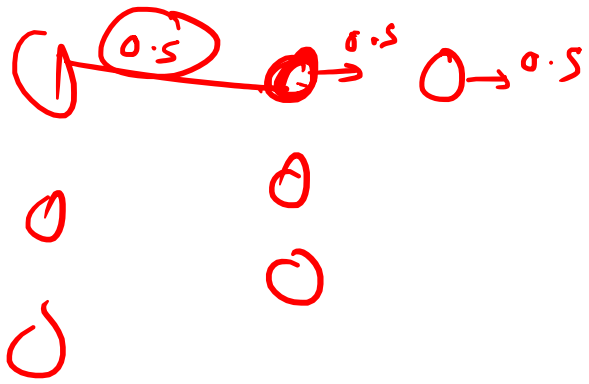
Standardization

I/p range (a) 1 to 5 million
(b) 0 to 0.0001
} i/p are not centered around 0

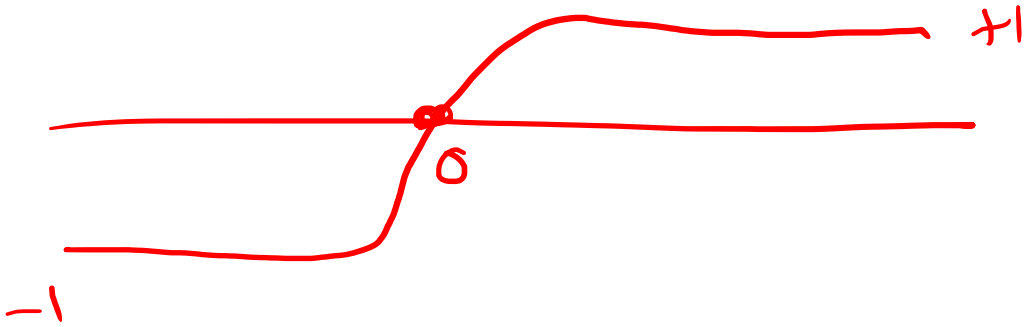
o/p of Sigmoid is b/w 0 & 1 & its middle is 0.5

∴ o/p of Sigmoid can never be centered around 0.

Idea of Uniformity.



Hyperbolic tangent (tanh)



$$\tanh(a) = \frac{\exp(2a)-1}{\exp(2a)+1}$$

Still more problems

tanh is a little better than sigmoid, but still both are problematic.

None of researchers wanted to break away from the classical way of doing things.

Because of that, Prob: Vanishing Gradient Problem.

Vanishing gradient problem

In 1980-1995, researchers were not able to create a Deep N.N.

bcz they were using Sigmoid in each & every Neuron.

ReLU was not invented that time.

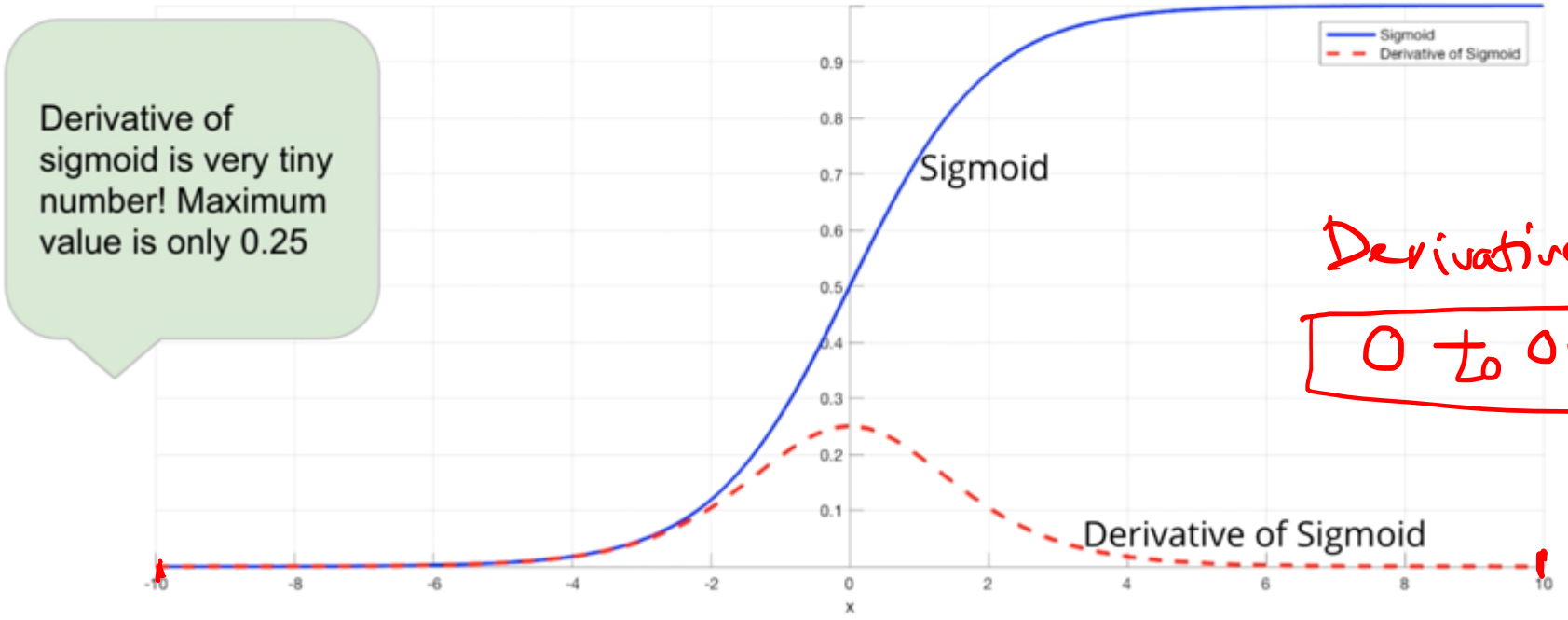
Bcz, everyone suffered from V.G. problem.

Vanishing gradient problem

Sigmoid

$w_{new} = w_{old} - \eta \cdot \frac{dL}{dw_{old}}$

← Backprop.



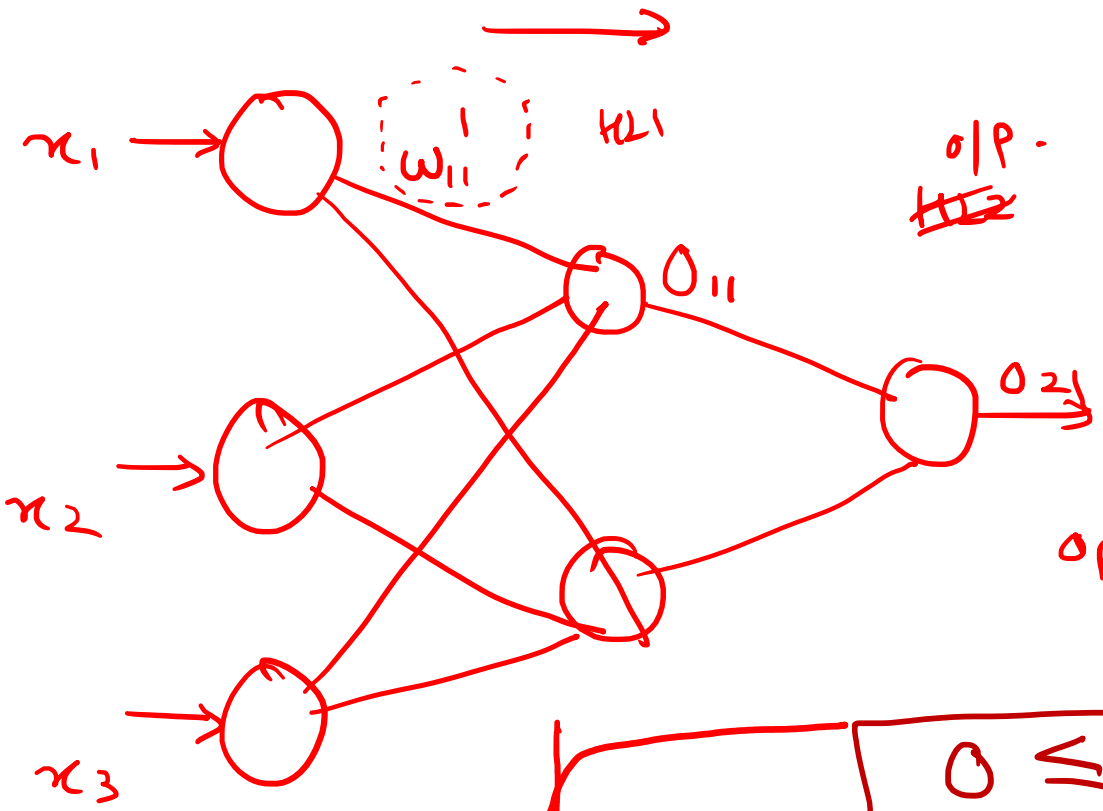
Derivative of sigmoid

0 to 0.25

Vanishing gradient problem

$\eta = 1$

$$w^1_{11\,new} = w^1_{11\,old} - \eta \cdot \frac{\partial L}{\partial w^1_{11\,old}}$$



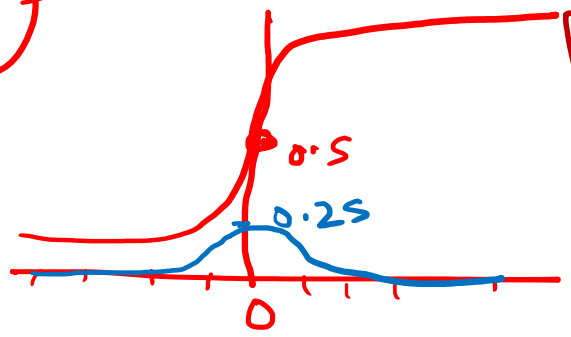
$$\frac{\partial L}{\partial w^1_{11}} = \frac{\partial O_{21}}{\partial O_{11}} \cdot \frac{\partial O_{11}}{\partial w^1_{11}}$$

$(0 \text{ to } 0.25) \quad (0 \text{ to } 0.25) = 0.0625$

$0.20 \quad 0.02 = 0.004$ (v. small)

$0.02 \quad 0.04 = 0.0008$

$$0 \le \sigma(z) \le 0.25$$



$$w^1_{11\,new} \approx w^1_{11\,old}$$

weight updation is not happening