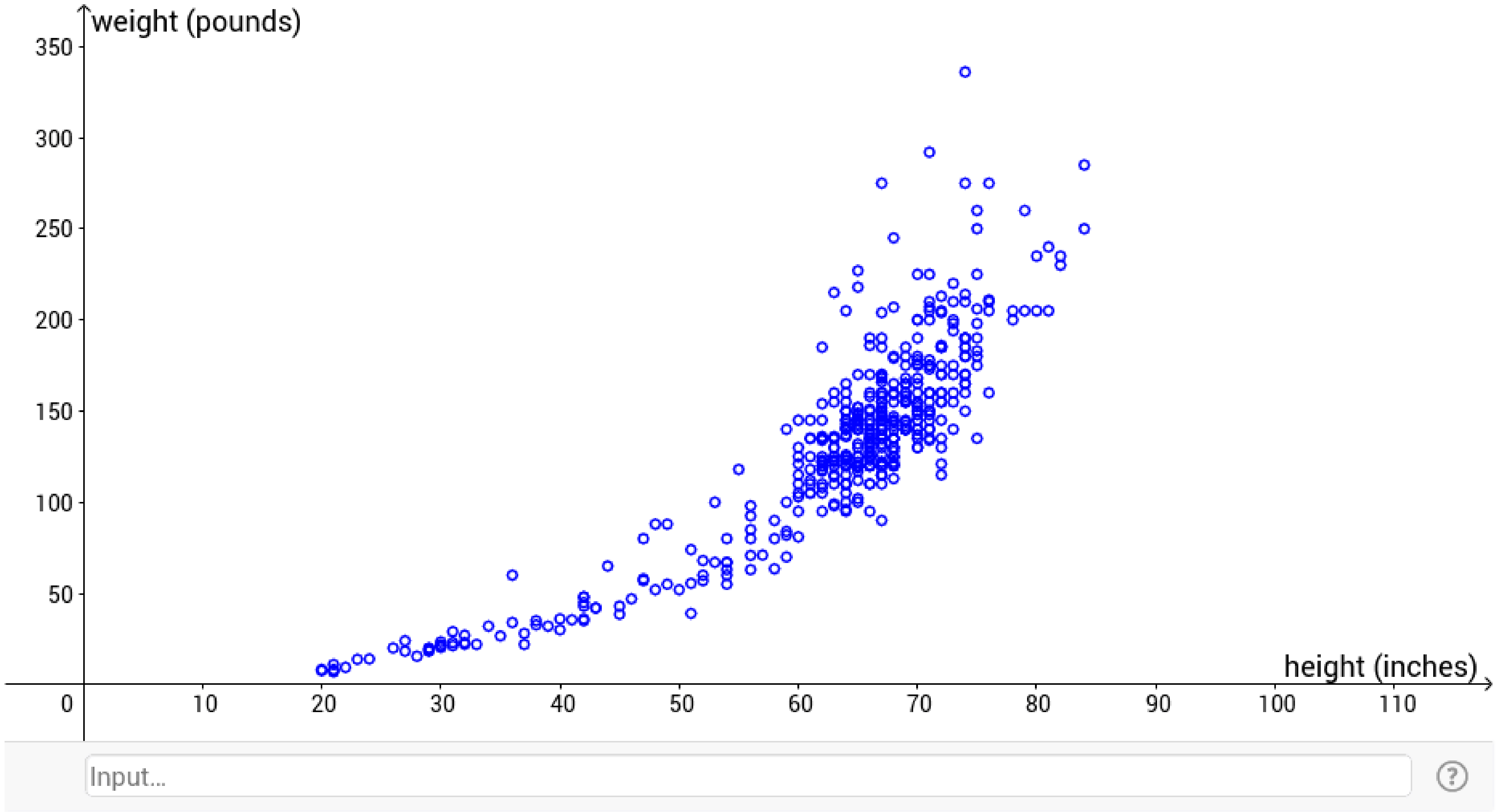
Deep Learning: Neural Networks

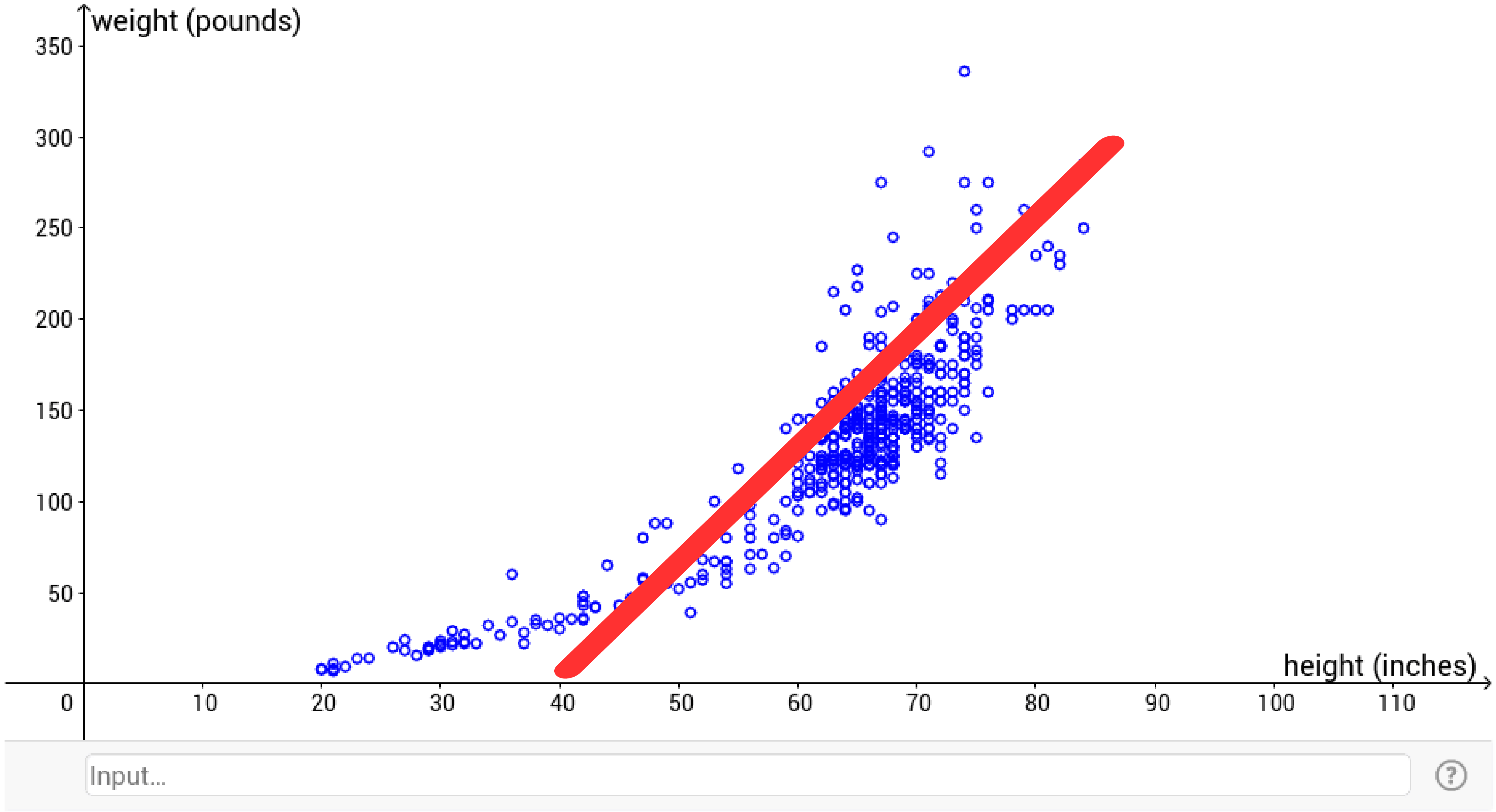
Delivered by Timilehin Owolabi

APWEN
—LAGOS CHAPTER—

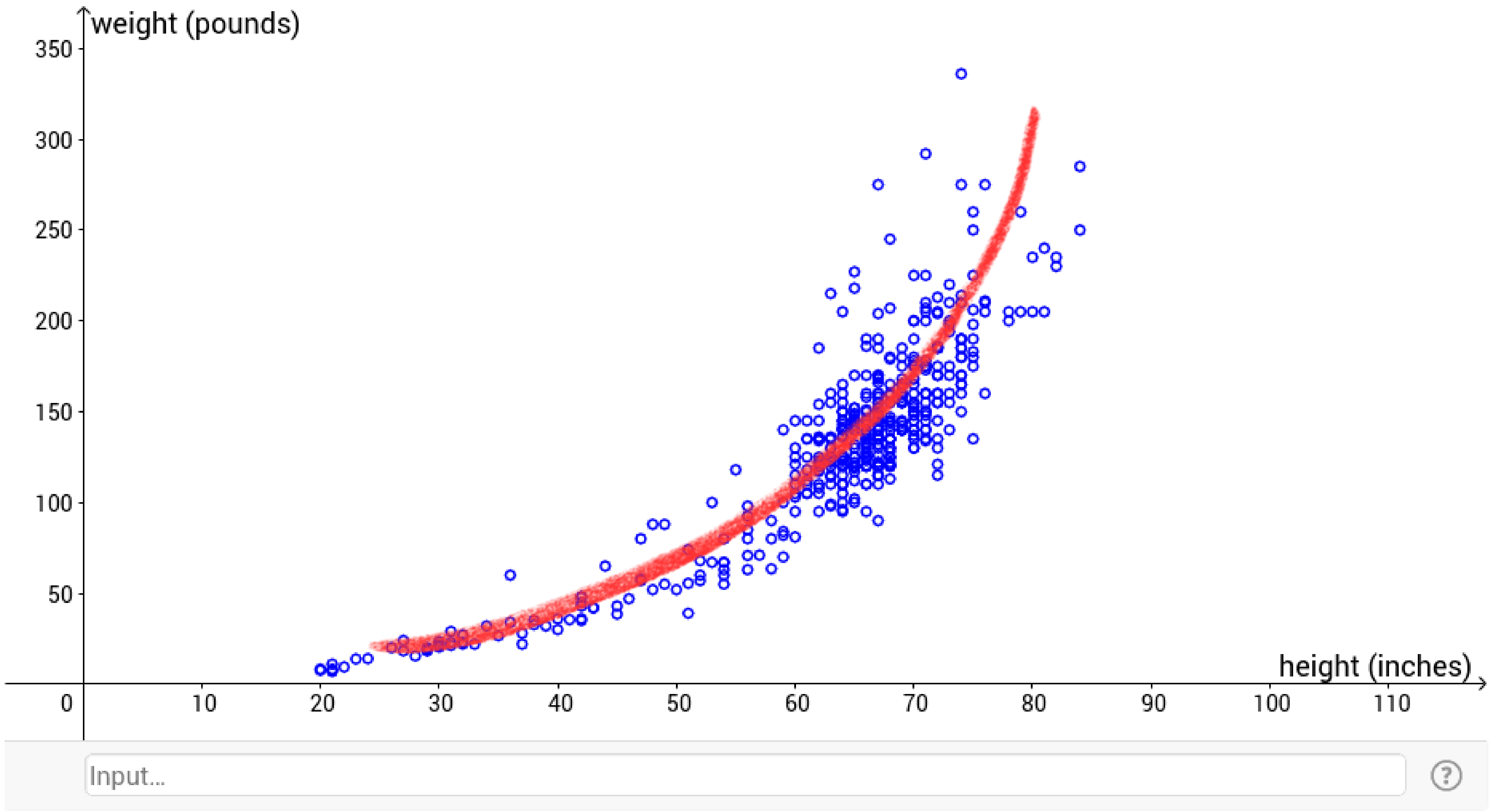
How do you fit a line to this data?



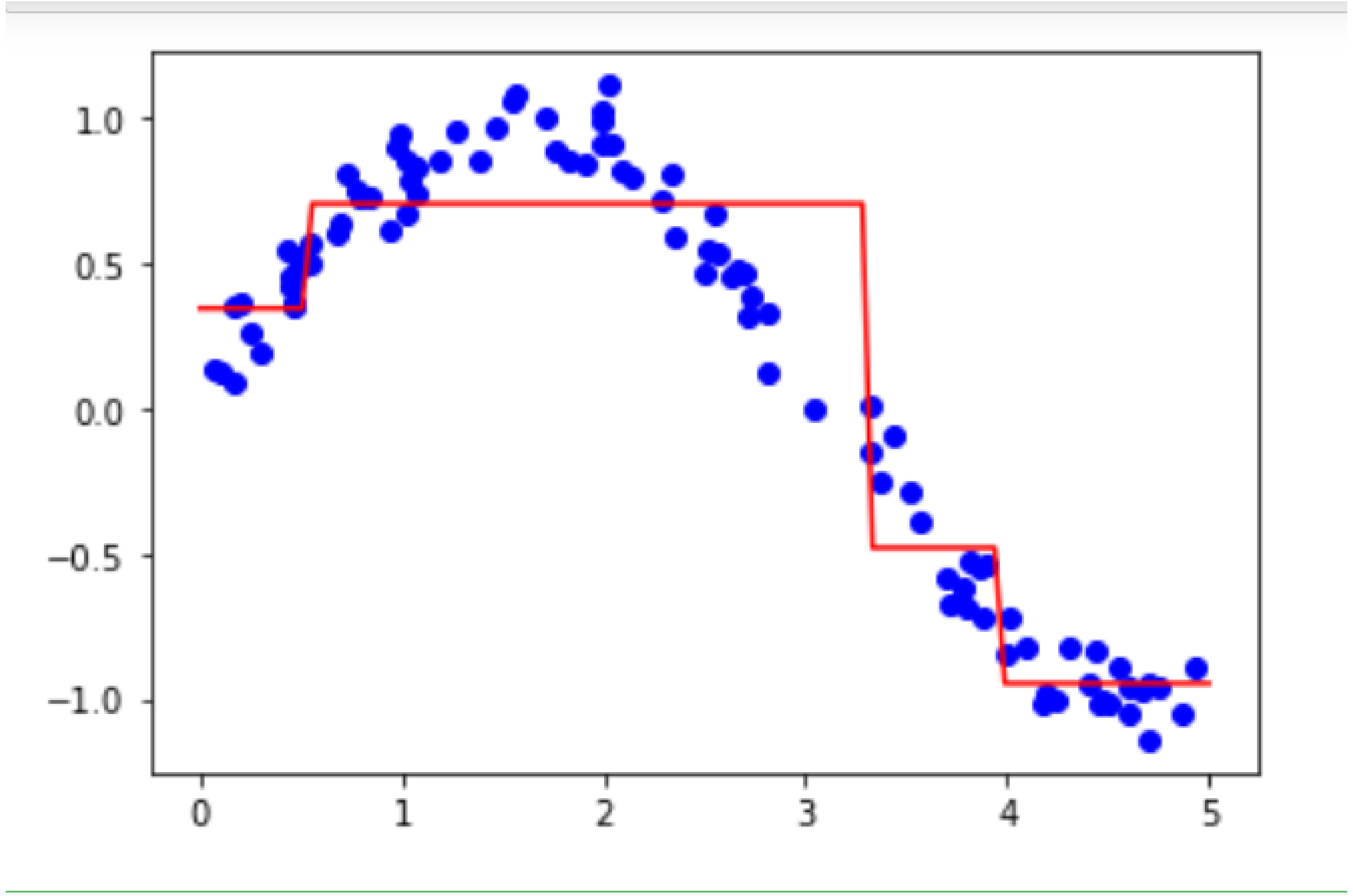
Linear Regression?



Exponential Regression?



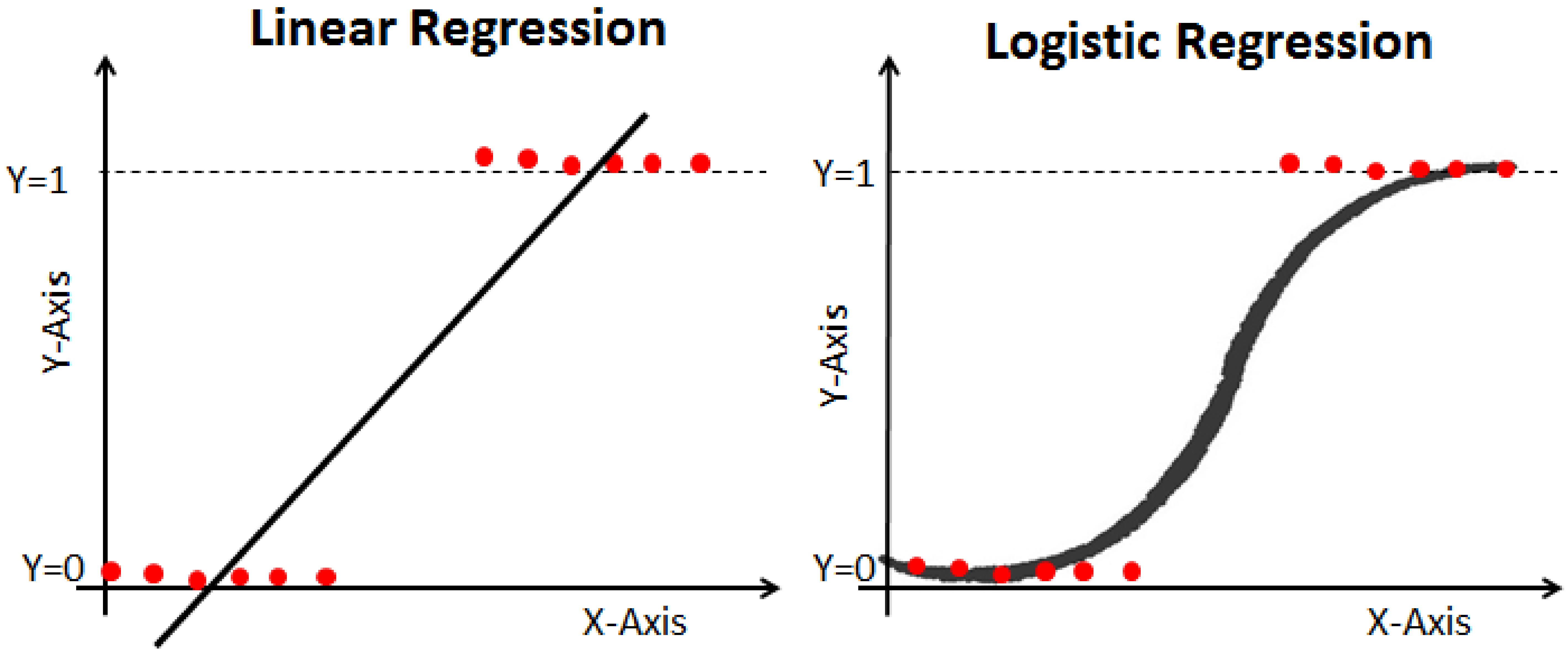
What of this?



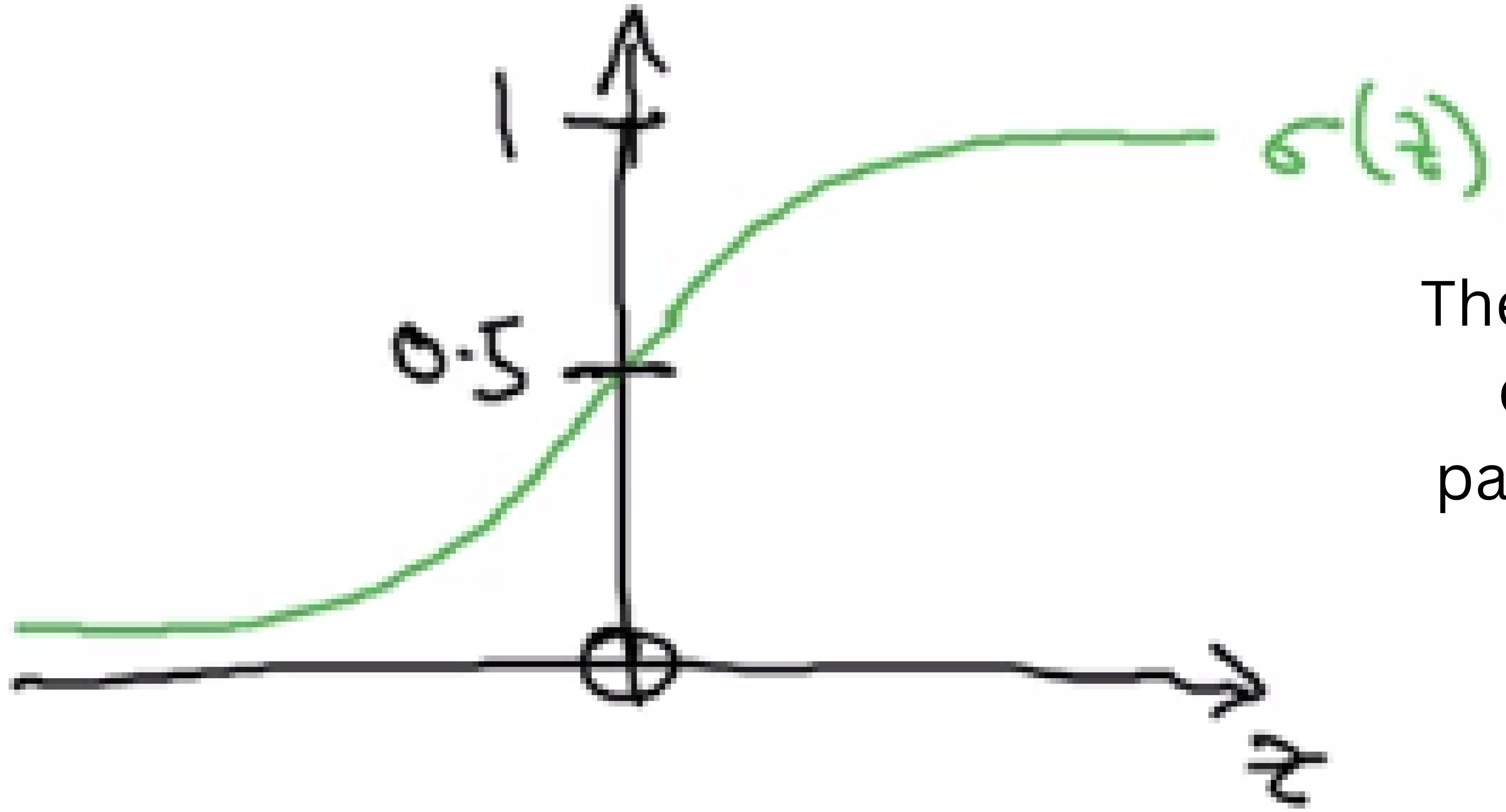
Logistic Regression

$$y = m\,x + b.$$

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

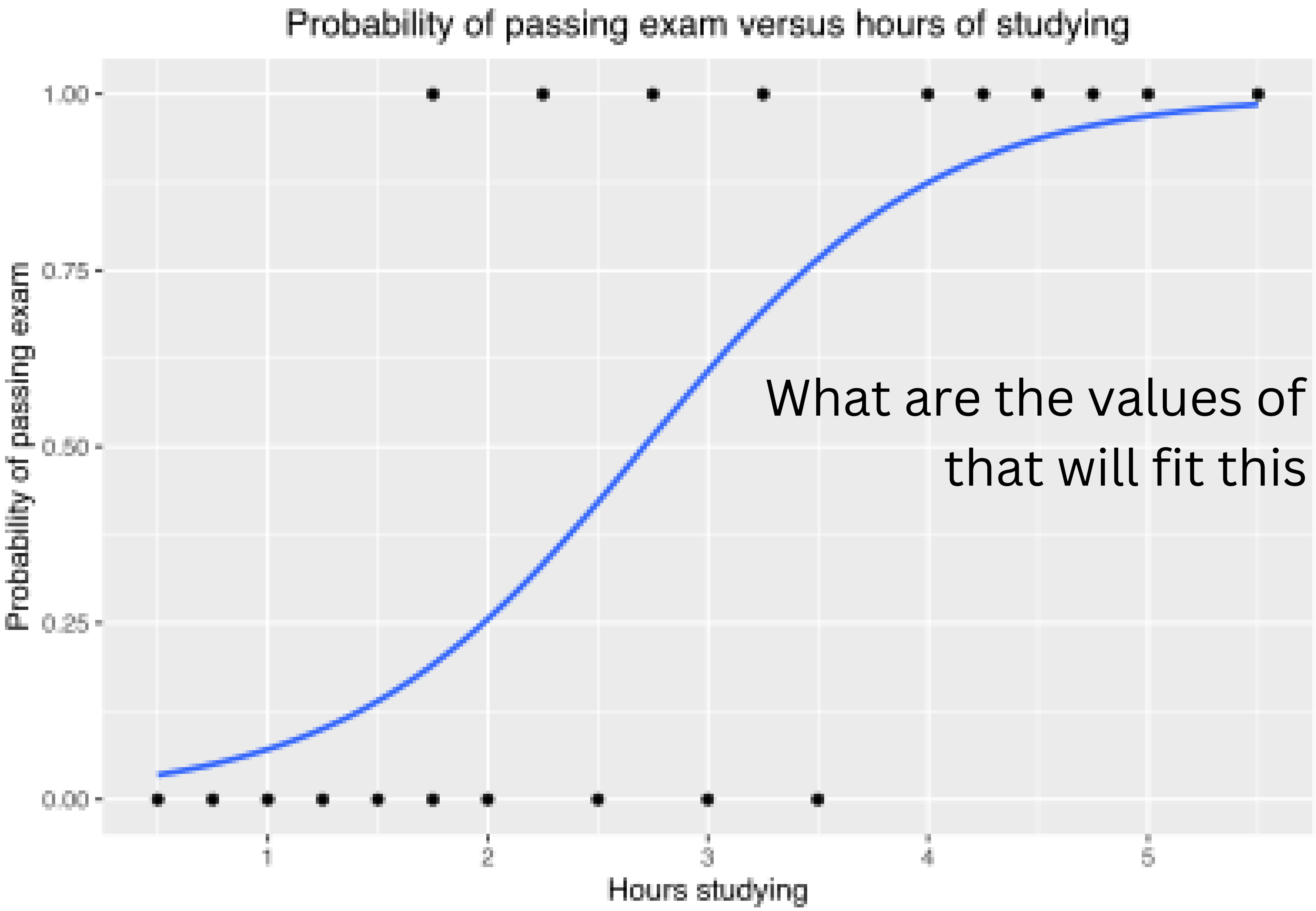


Output $$\hat{y} = \sigma(\underbrace{w^T x + b}_{z})$$



The shape of the line depends on the parameters w and b

How can we determine who passes?



It’s now a classification problem

What are the values of the parameters w and b that will fit this line to the data?

Cost Function: How do we determine the correctness of our line?

error = |y - y_hat|

error = (y - y_hat)^2

error = - (y log(y_hat) + (1 - y) log (1 - y_hat))

Total error = sum(error)/number of points

Cost = Total Error

Gradient Descent

Cost = - (y log(y_hat) + (1 - y) log (1 - y_hat))

y_hat = sigmoid(**w** x + **b**)

Let J = Cost

since y and x are constant, Cost can be seen as a function of **w** and **b**

J(w, b) = - (y log(y_hat) + (1 - y) log (1 - y_hat))

So how do we reduce the cost?

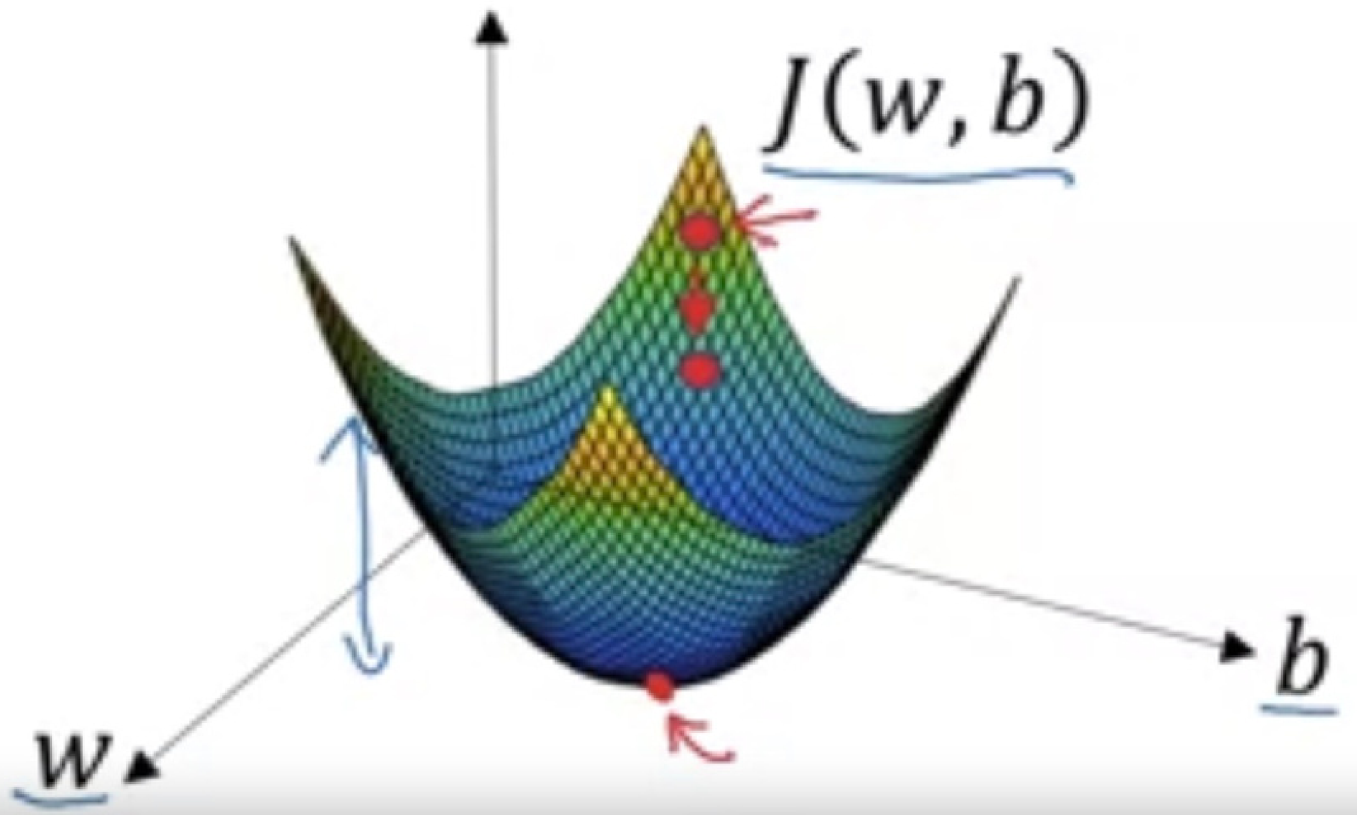
Cost is minimum when dJ(w,b) = 0

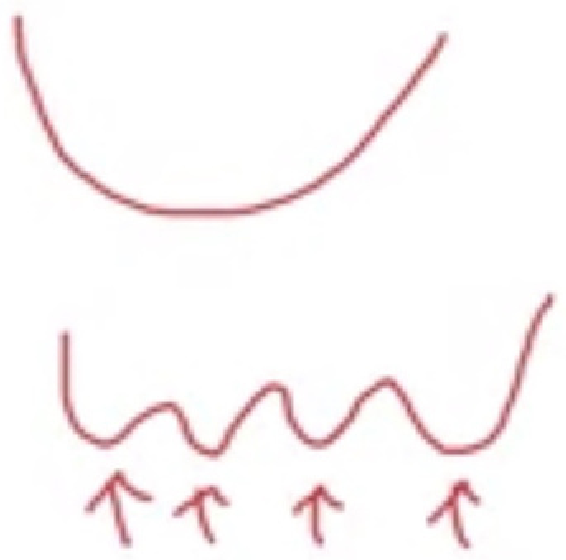
Gradient Descent

Recap: $\hat{y} = \sigma(w^T x + b)$, $\sigma(z) = \frac{1}{1+e^{-z}}$

$$J(w,b) = \frac{1}{m}\sum_{i=1}^{m} \mathcal{L}(\hat{y}^{(i)}, y^{(i)}) = -\frac{1}{m}\sum_{i=1}^{m} y^{(i)} \log \hat{y}^{(i)} + (1 - y^{(i)}) \log(1 - \hat{y}^{(i)})$$

Want to find w, b that minimize $J(w,b)$



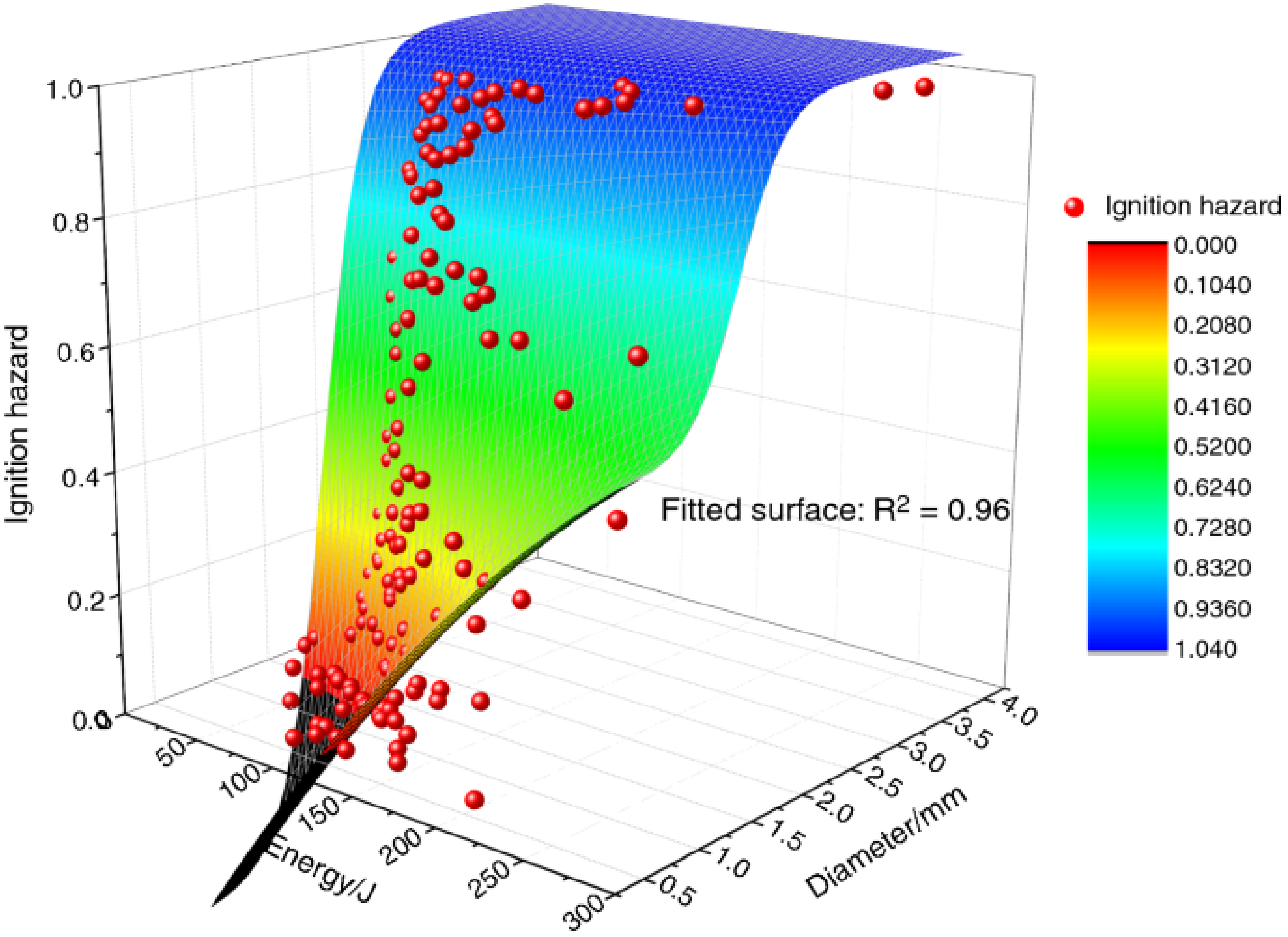


Andrew Ng

3:17 / 11:23

But what if we have 2 independent variables?

y = w1x1 + w2x2 + b



Our Cost function will become
J(w1, w2, b)

For each variable, we add a **w**
and a **b**

Solving for the minimum of J(w1, w2, ... wx, b) by partial differentiation will become very computationanly expensive

What if we have 3 independent variables?

J(w1, w2, w3, b)

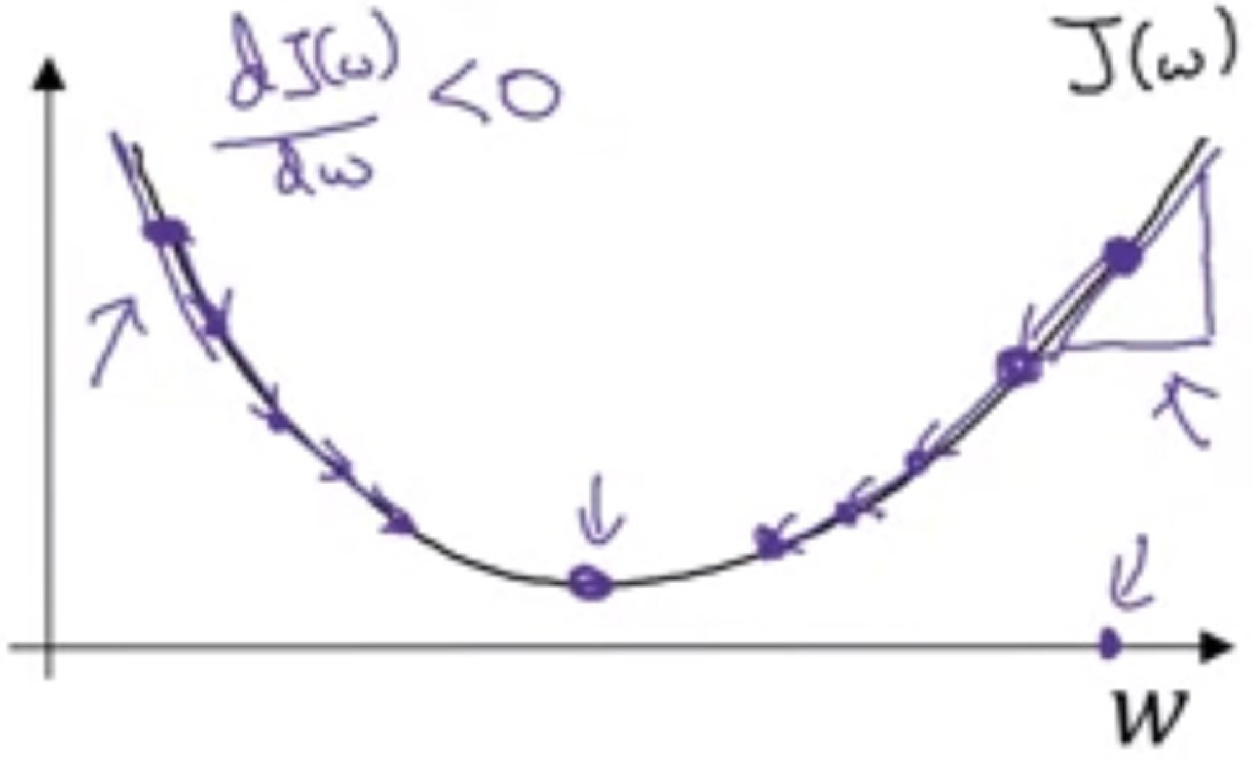
A 1920x1080 image has 2,073,600 x 3 variables

We can classify the image using the same principles

A pixel is just a number

J(w1, w2, ... w2073600, b)

Gradient Descent



Repeat {

$$w := w - \alpha \frac{dJ(w)}{dw}$$

learning rate

}

"dw"

$$w := w - \alpha \, dw$$

$$\frac{dJ(w)}{dw} = ?$$

$J(w, b)$

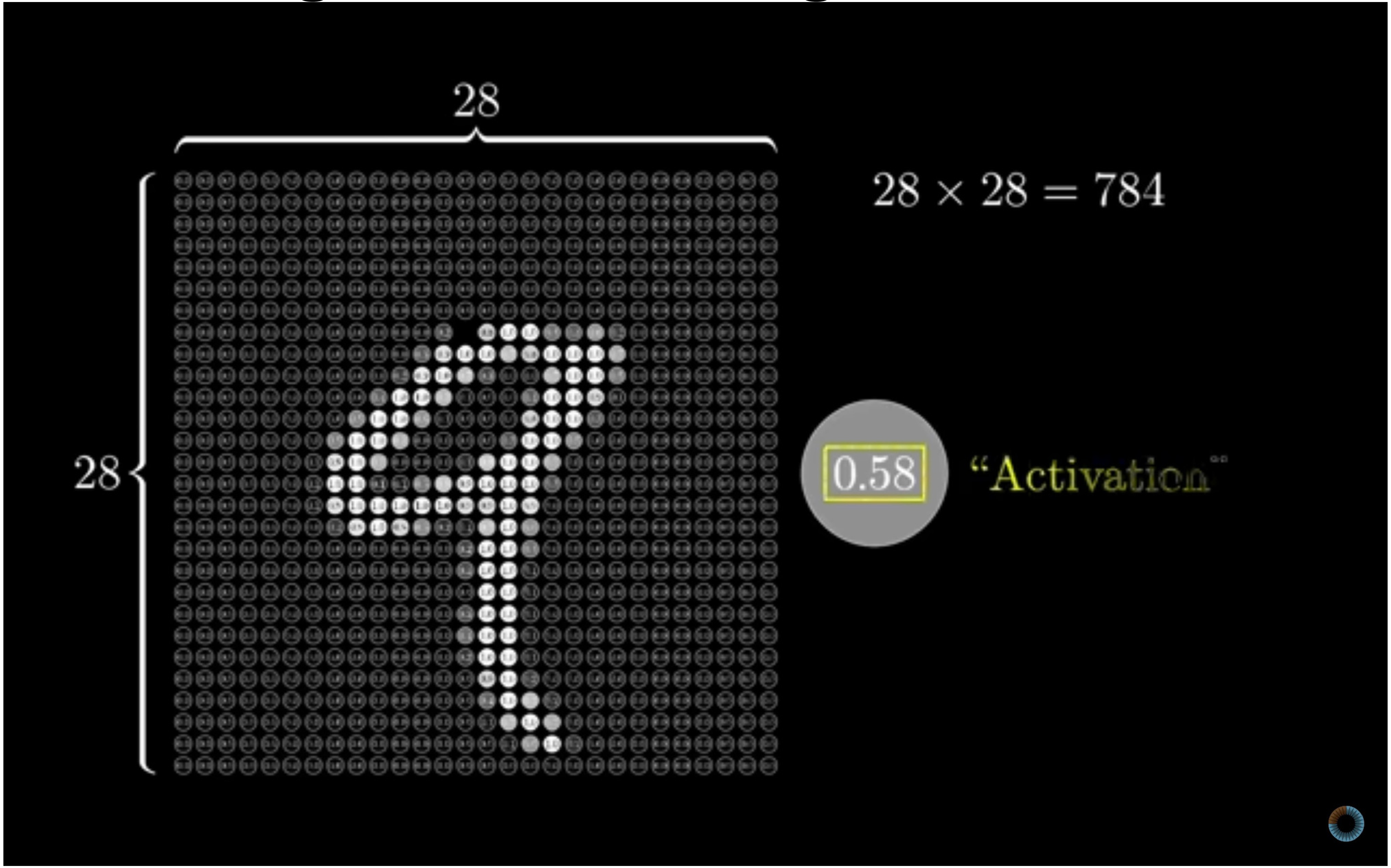
$$w := w - \alpha \frac{dJ(w,b)}{dw} \qquad \frac{\partial J(w,b)}{\partial w} \rightarrow dw$$

$$b := b - \alpha \frac{dJ(w,b)}{db} \qquad \frac{\partial J(w,b)}{\partial b} \rightarrow db$$

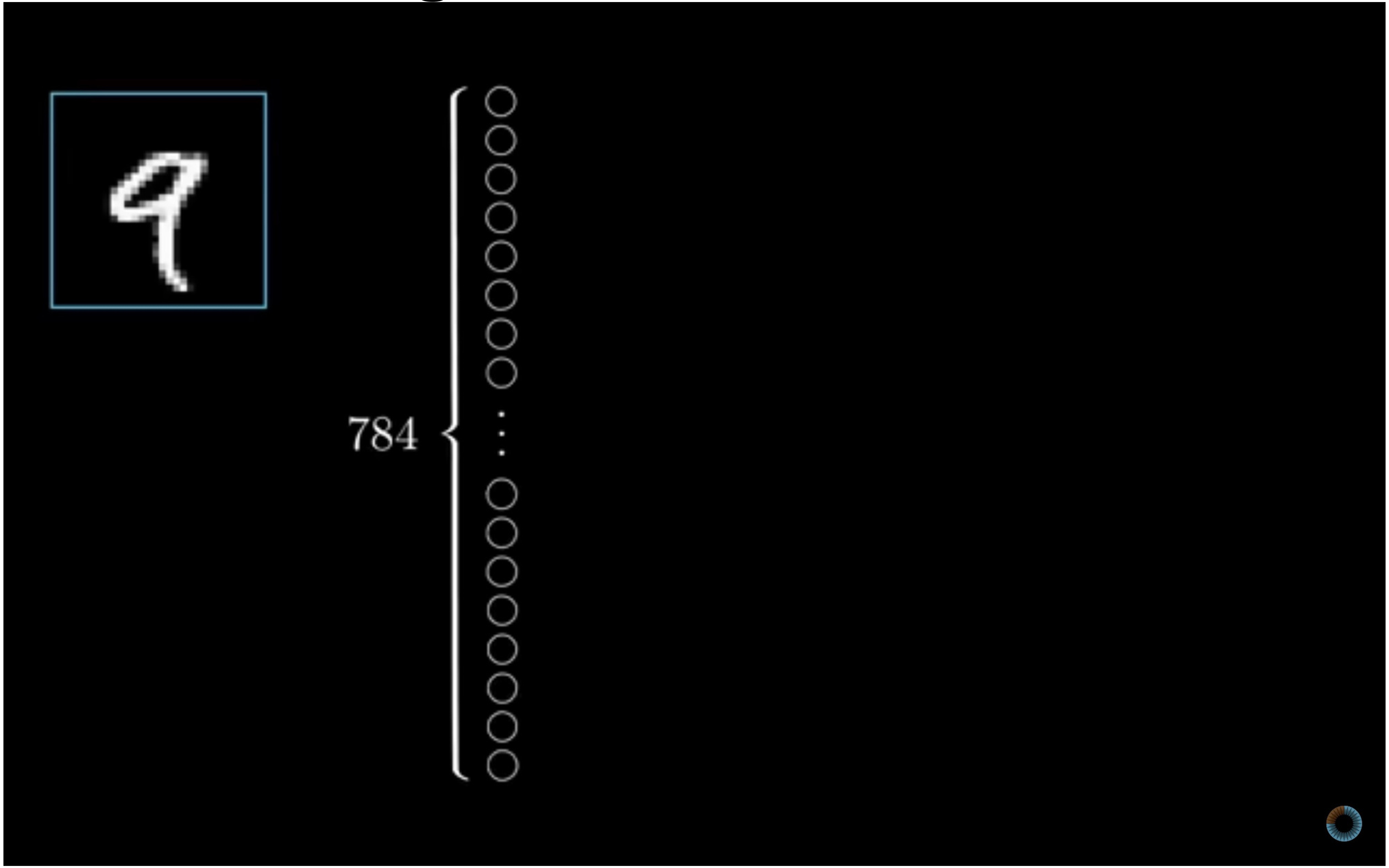
∂ "partial derivative"

d ← J

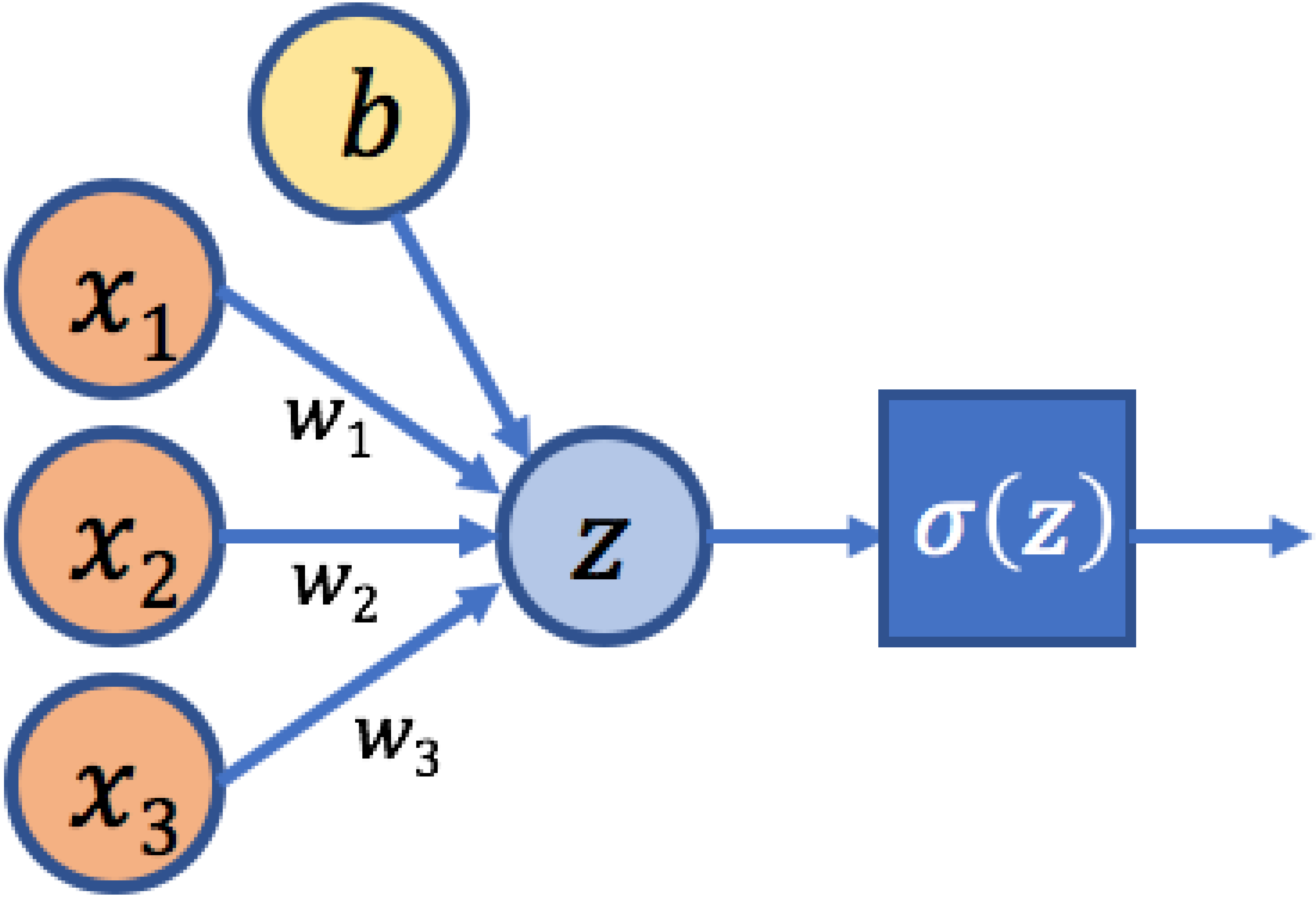
We will be dealing with 28x28 images



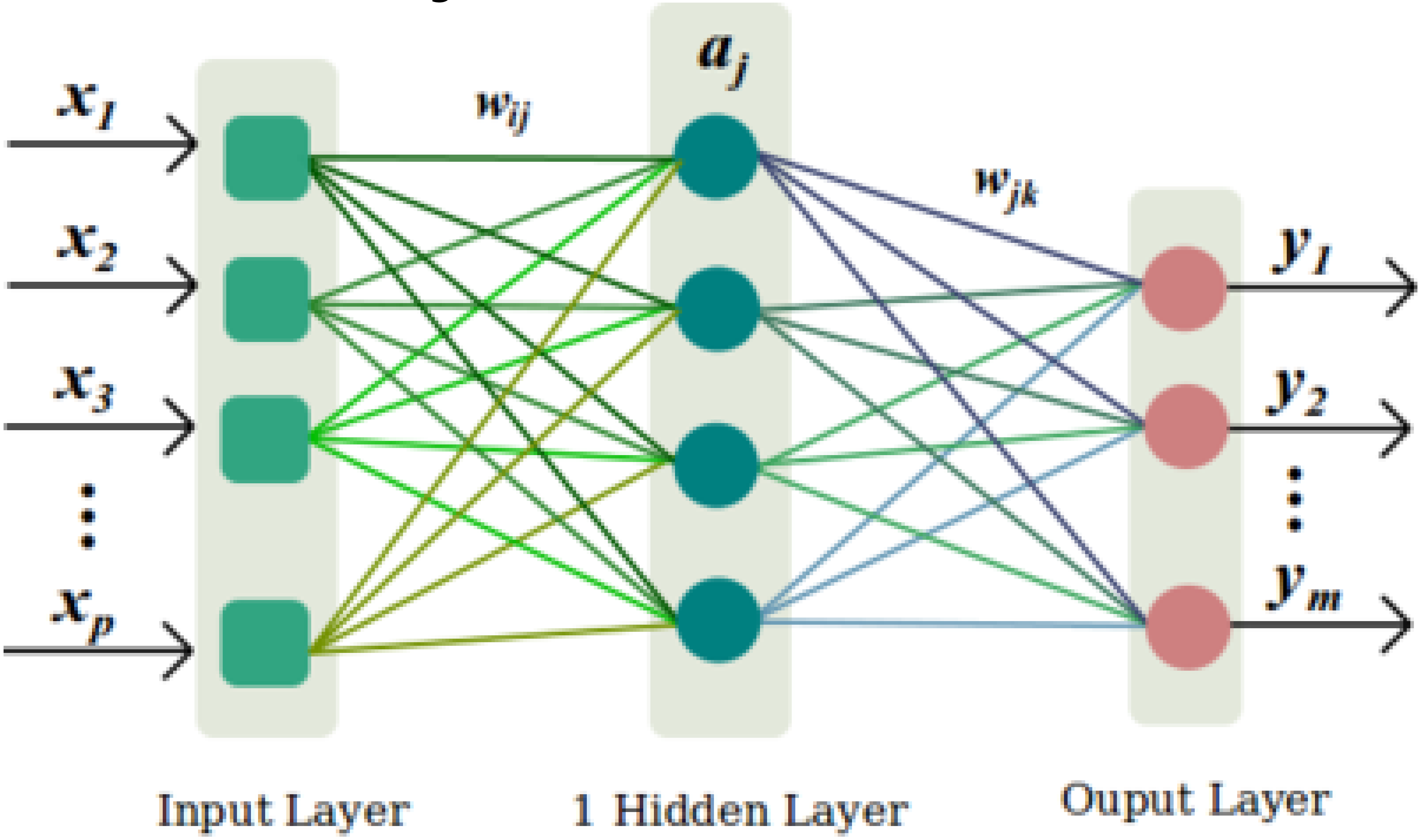
We flatten the image



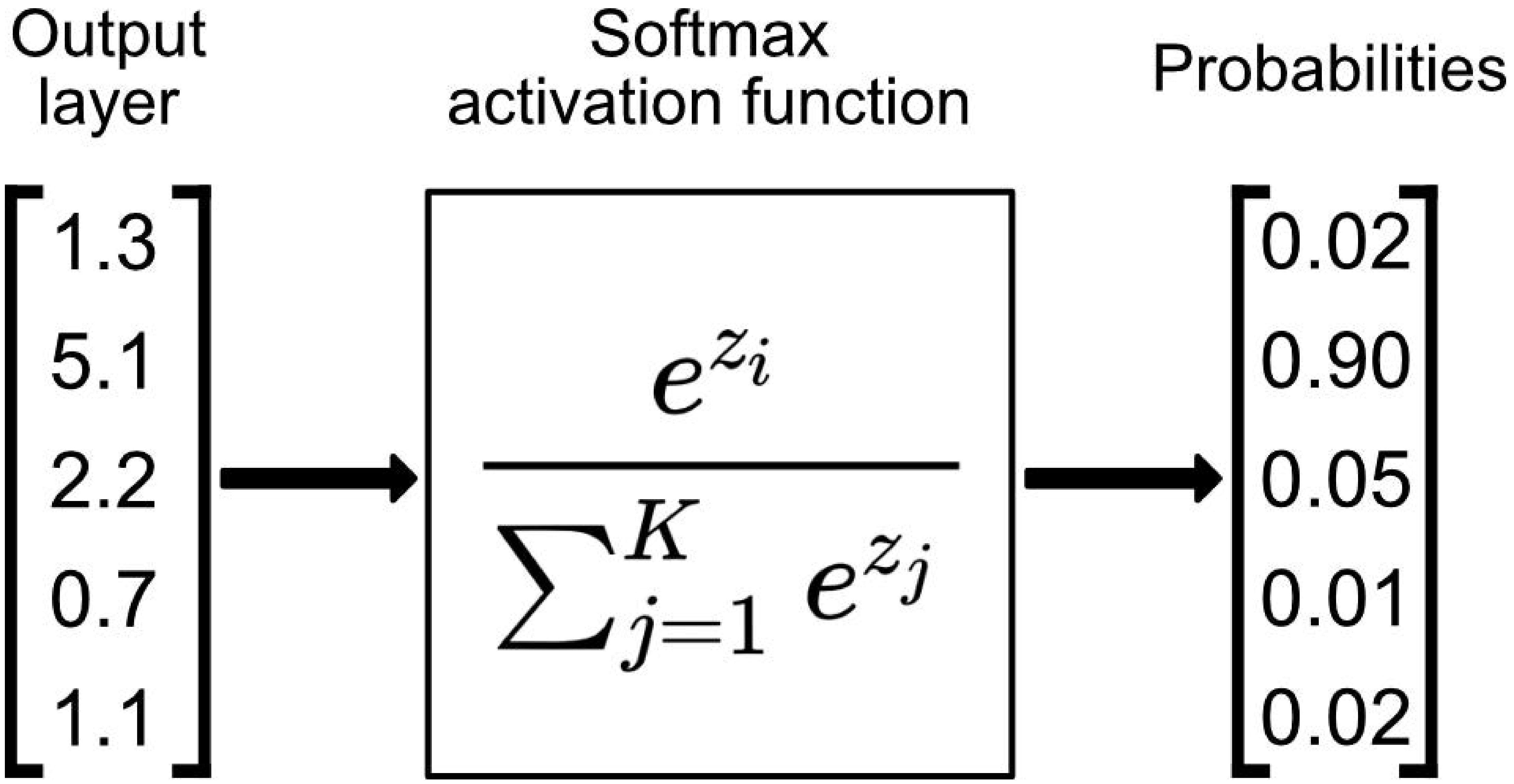
Each pixel of an image can be treated as an independent variable

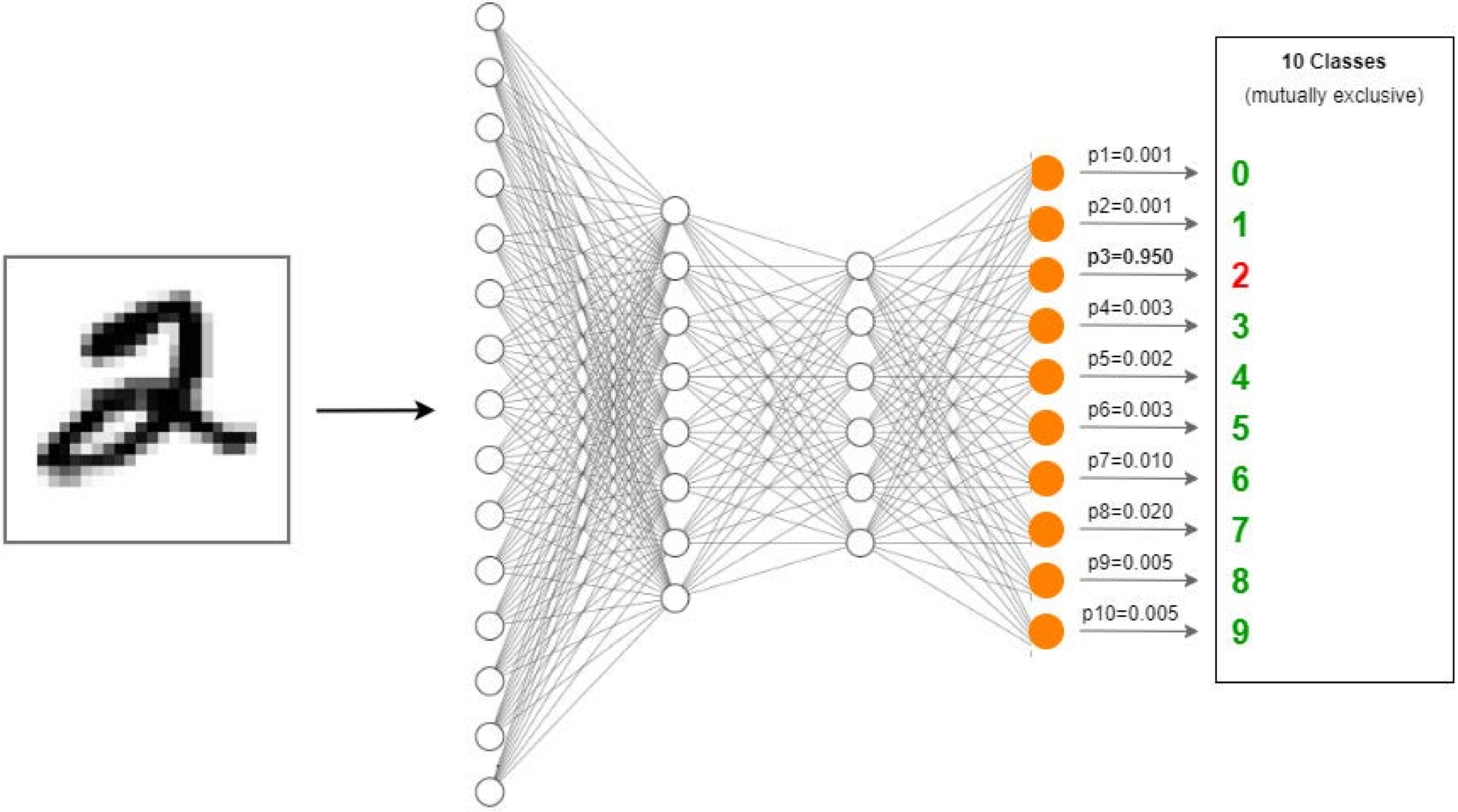


What are hidden layers?



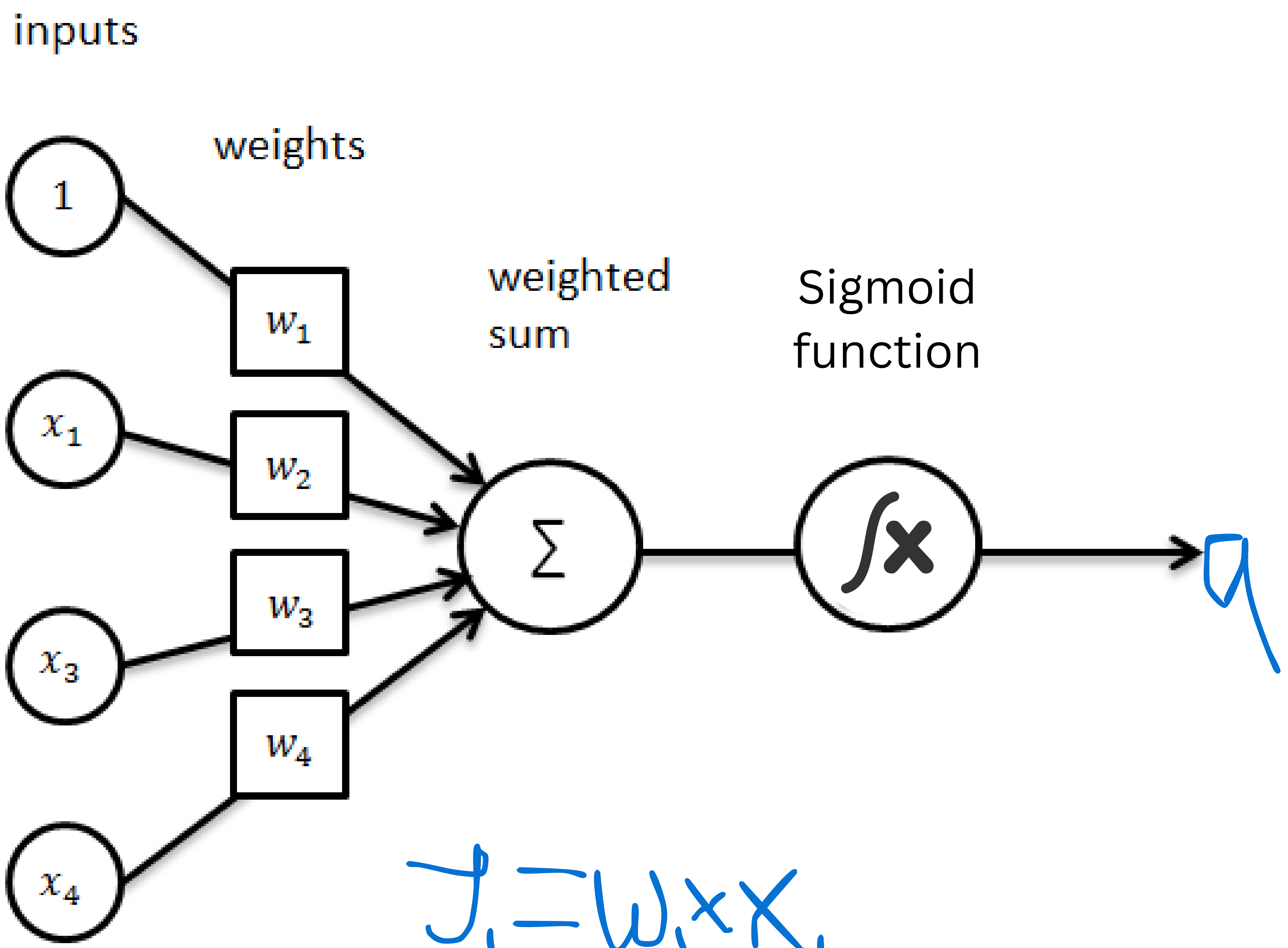
Softmax Function





10 Classes
(mutually exclusive)
p1=0.001
p2=0.001
p3=0.950
p4=0.003
p5=0.002
p6=0.003
p7=0.010
p8=0.020
p9=0.005
p10=0.005
0
1
2
3
4
5
6
7
8
9

Forward Propagation

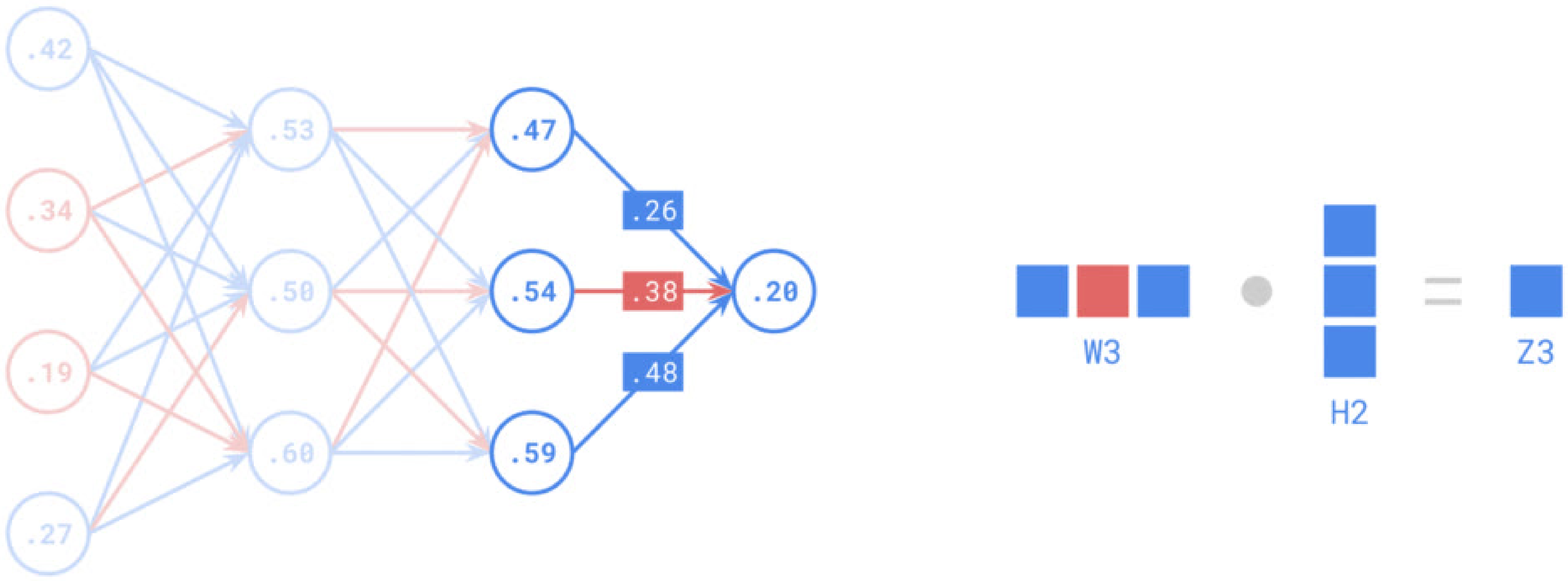


$$\begin{bmatrix} w_1 & w_2 & w_3 & w_n \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \\ \end{bmatrix} = Z$$

$$y_1 = w_1 \times x_1$$

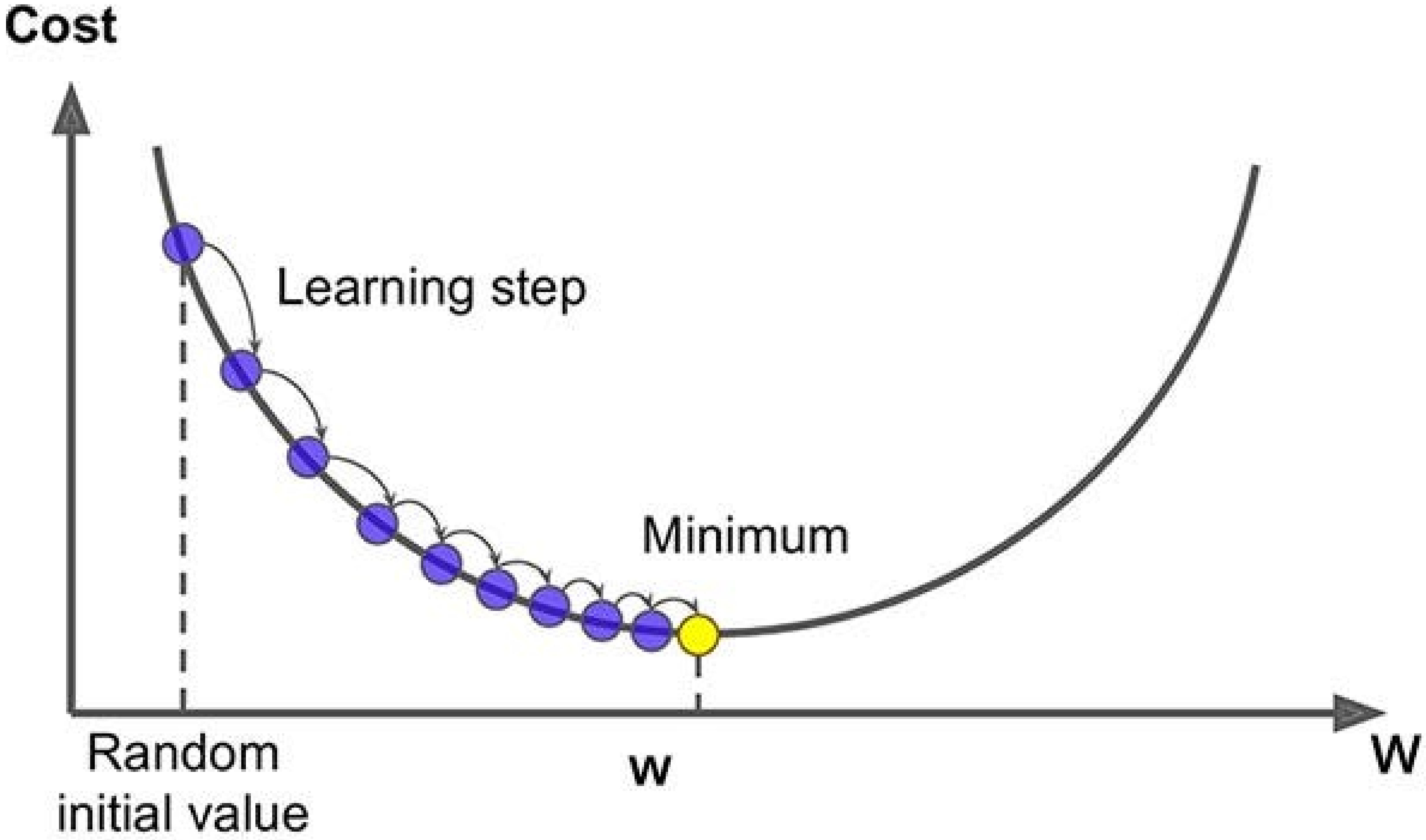
$$Z = y_1 + y_2 + y_3 + y_4 + b$$

$$a = \sigma(z)$$



.42
.34
.19
.27
.53
.50
.60
.47
.54
.59
.26
.38
.48
.20
W3
H2
Z3

Gradient Descent



Backpropagation

learning rate

observation *i*

$$\mathrm{w}^{(t+1)} = \mathrm{w} - \alpha \nabla f_i(\mathrm{w}^{(t)})$$

position of next iteration

position of previous step

step

Now do this for all **w** and **b**

Momentum

ReLU

Batch normalization

Regularisation

Attention

CNN

RNN

Transformers

Latent Diffusion

GANs

Advantages and Disadvantages

Neural Networks: Advantages and Disadvantages

Advantages	Disadvantages
1) The ability to learn by themselves	1) The black box nature and uncertain prediction rates
2) The ability to work with insufficient data	2) Long training processes and limited data efficiency
3) The ability of parallel processing	3) Economically and computationally expensive

clerk.io

PyTorch

TensorFlow

ANALYTIX LABS

Pytorch vs TensorFlow

Feature	PyTorch	TensorFlow
Ease of use	Relatively easy to learn, but can be complex for large projects	It can be difficult to understand, but it has a large community of users and resources
Speed	Faster than TensorFlow but not as fast as some other deep learning frameworks	Slower than PyTorch but faster than some other deep-learning frameworks
Flexibility	Very flexible, allowing for a wide variety of deep learning tasks	Less flexible than PyTorch, but still capable of a wide range of tasks

Credits and Materials

3blue1brown
Andrew Ng
Lightning AI

Awesome Course on Deep Learning by Lightning AI: **https://www.youtube.com/watch?v=6Py-tIEiXKw&list=PLaMu-SDt_RB4Ly0xb0qsQVpLwRQcjtOb-&pp=iAQB**

Andrew Ng’s Course: **https://www.coursera.org/learn/neural-networks-deep-learning/home/welcome**

Want to learn more theory😅😂?: **https://www.youtube.com/playlist?list=PLZHQObOWTQDNU6R1_67000Dx_ZCJB-3pi**

Thank you!