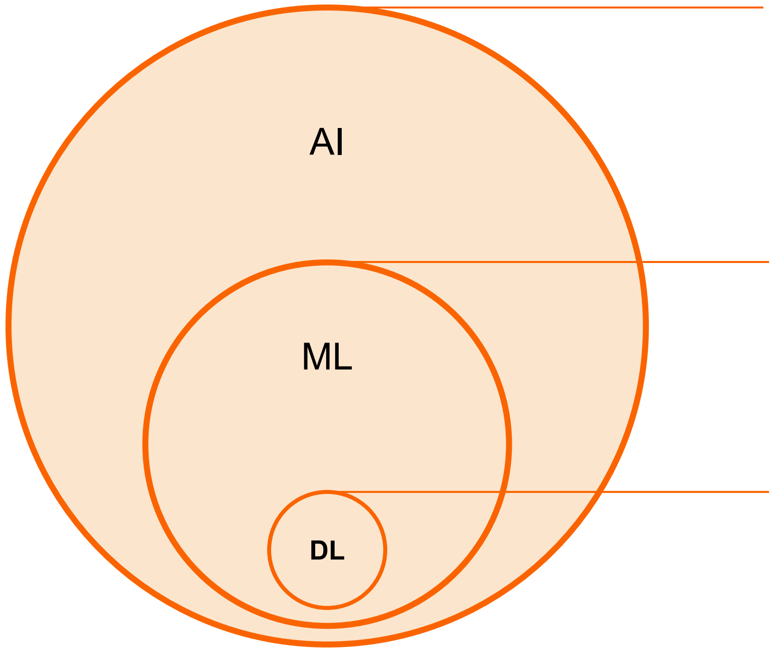
Webinar

Building a Neuron in JS

ACCOLITE DIGITAL
Transforming The Future, Now

Intelligence and Learning



Models for Thinking, Reasoning, Learning and Decision Making.

Encompasses subfields, like natural language processing, computer vision, robotics, and machine learning.

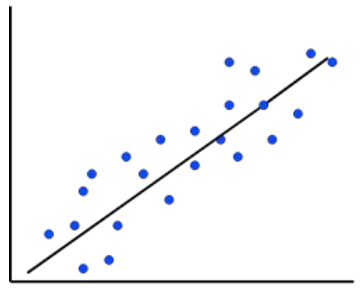
Learning from data using algorithms and statistical models.

These models can be supervised, unsupervised etc. Commonly used in image recognition, NLP etc.

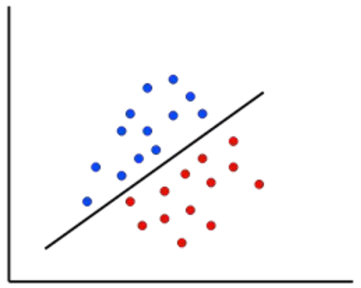
Neural networks that are complex and have multiple layers.

Learn from hierarchies of features in dataset. Commonly used in speech recognition, computer vision etc.

Regression and Classification



Regression



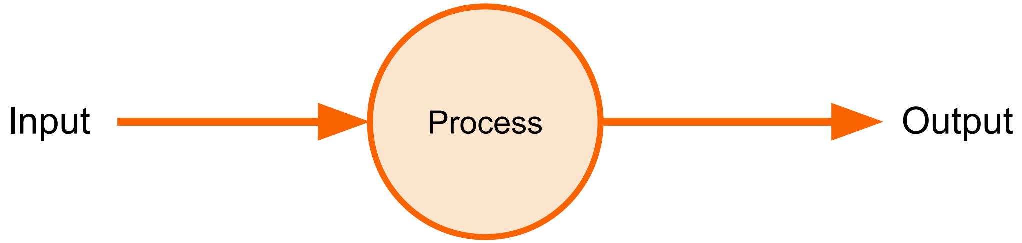
Classification

Goal is to predict a continuous numeric value as the output. e.g. price of a house

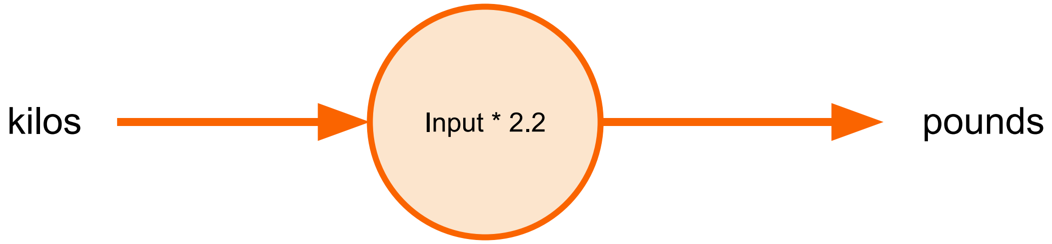
Goal is to predict a category label or class for a given input. e.g. spam detection

Regression statistical definition: a measure of the relation between the mean value of one variable (e.g. output) and corresponding values of other variables (e.g. time and cost).

A Weighing Scale - Just a Function

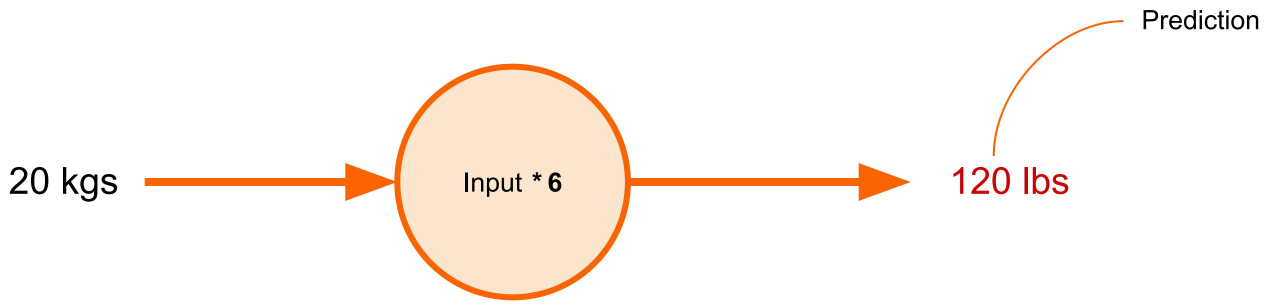


A Weighing Scale



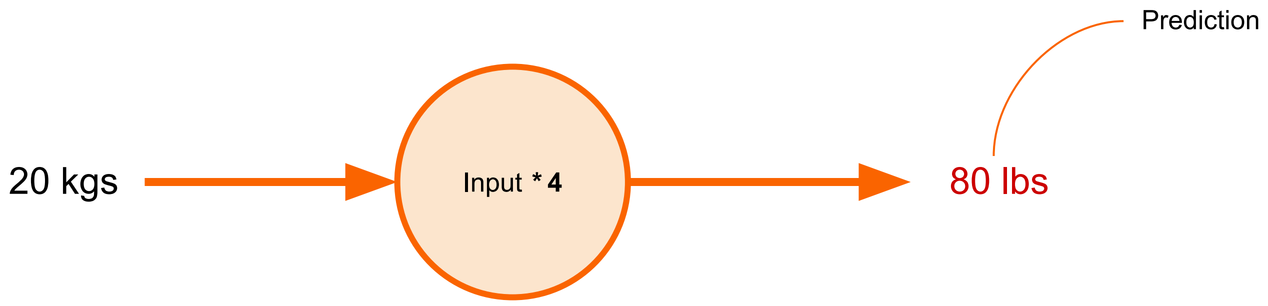
A linear relationship between the input and output.

A Weighing Scale - Predicting



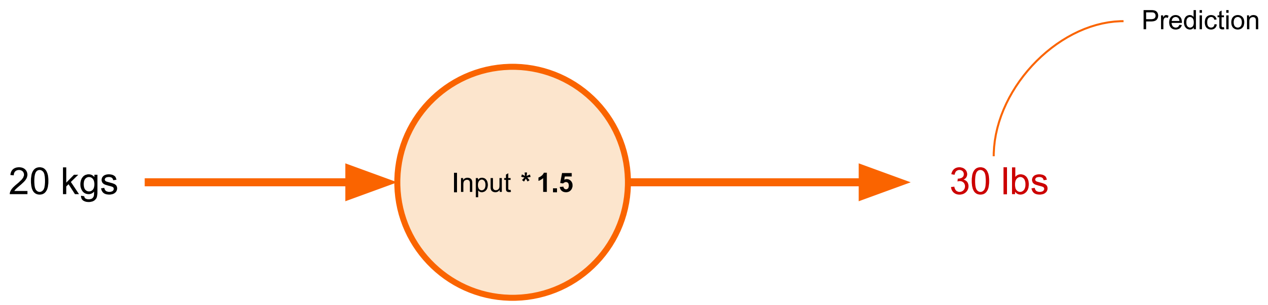
Target **44 lbs** and Error = **76 lbs**

A Weighing Scale - Predicting



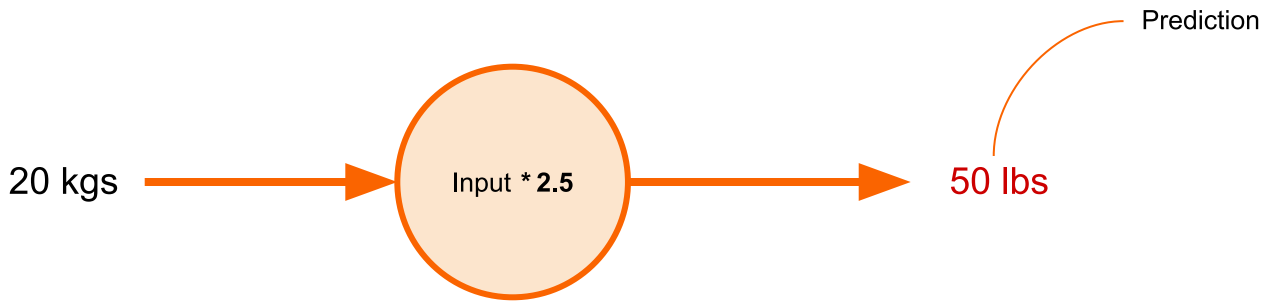
Target **44 lbs** and Error = **36 lbs**

A Weighing Scale - Predicting



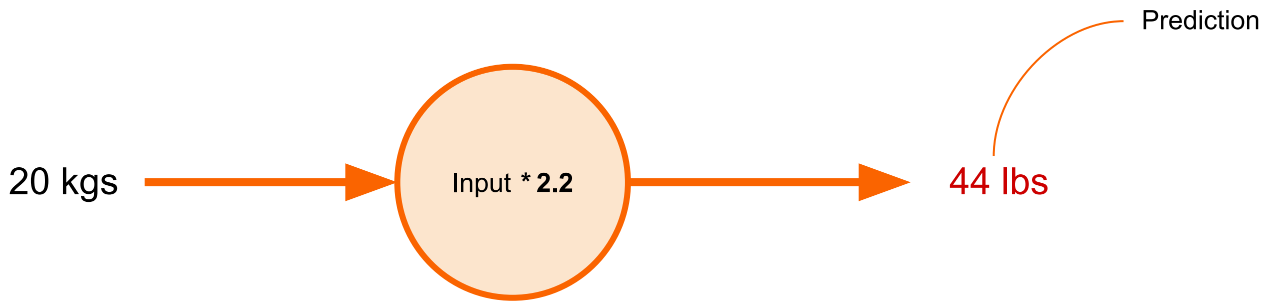
Target **44 lbs** and Error = **-14 lbs**

A Weighing Scale - Predicting



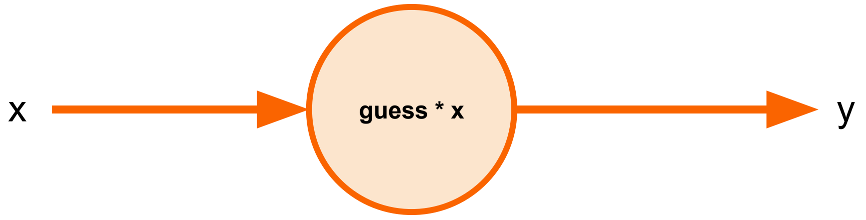
Target **44 lbs** and Error = **6 lbs**

A Weighing Scale - Predicting



Target **44 lbs** and Error = **0 lbs**

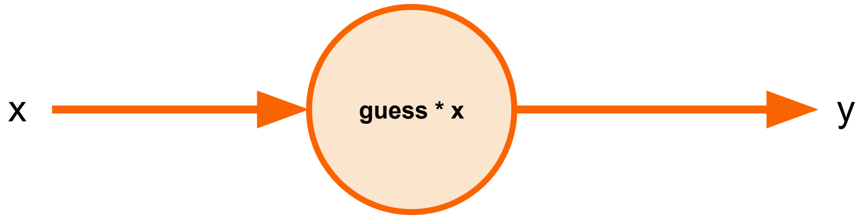
A Weighing Scale - Better Prediction



Prediction

$y = m * x$

A Weighing Scale - Better Prediction



Prediction

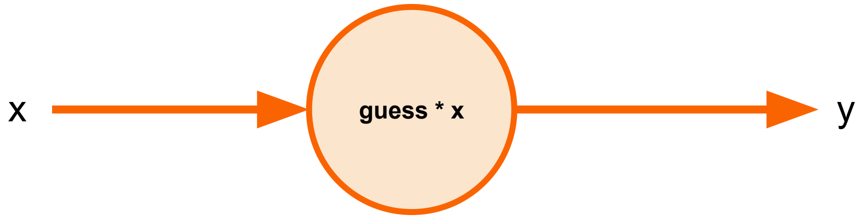
$$y = m * x$$

Target

$$t = (m + \Delta m) * x$$

Accolite Digital

A Weighing Scale - Better Prediction



Prediction

$$y = m * x$$

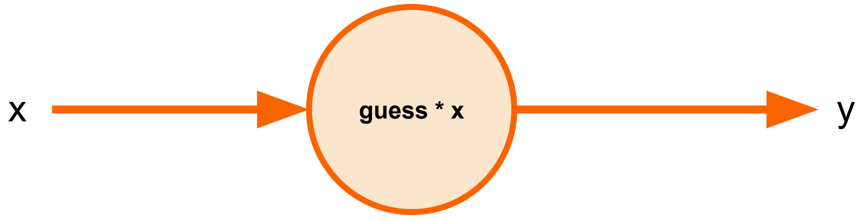
Target

$$t = (m + \Delta m) * x$$

Error = Prediction - Target

$$E = y - t$$

A Weighing Scale - Better Prediction



Prediction

$$y = m * x$$

Target

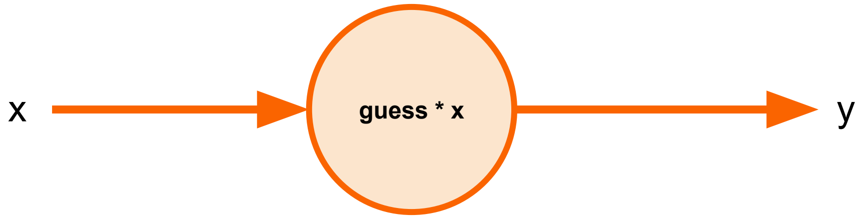
$$t = (m + \Delta m) * x$$

Error = Prediction - Target

$$E = y - t$$

$$E = m * x - (m + \Delta m) * x$$

A Weighing Scale - Better Prediction



Prediction

$$y = m * x$$

Target

$$t = (m + \Delta m) * x$$

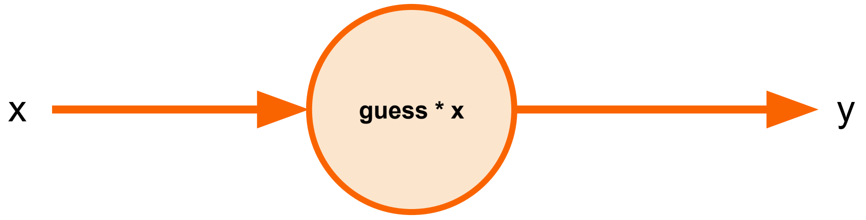
Error = Prediction - Target

$$E = y - t$$

$$E = m * x - (m + \Delta m) * x$$

$$E = -\Delta m * x$$

A Weighing Scale - Better Prediction



Prediction

$$y = m * x$$

Target

$$t = (m + \Delta m) * x$$

Error = Prediction - Target

$$E = y - t$$

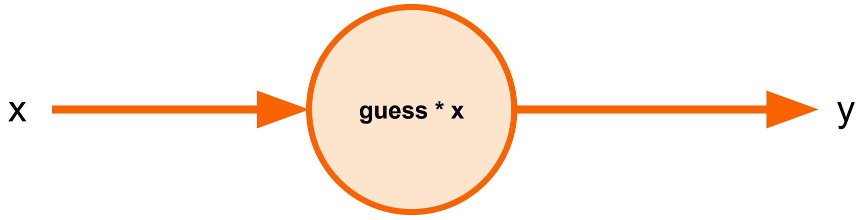
$$E = m * x - (m + \Delta m) * x$$

$$E = -\Delta m * x$$

$$\Delta m = -E/x$$

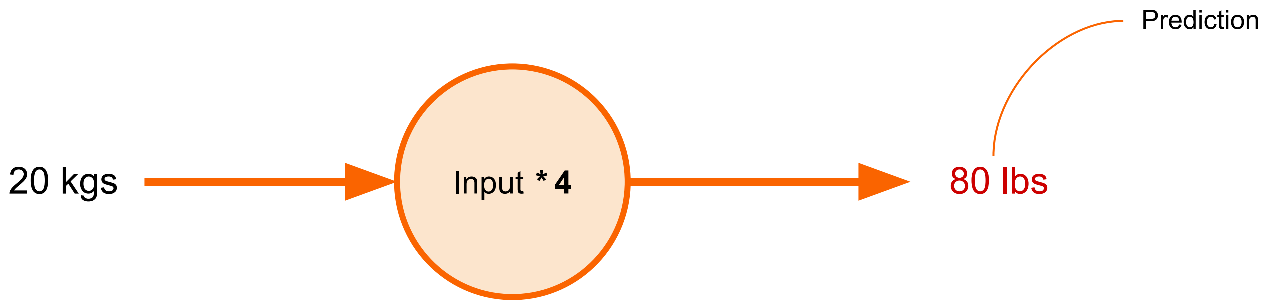
Accolite Digital

A Weighing Scale - Better Prediction



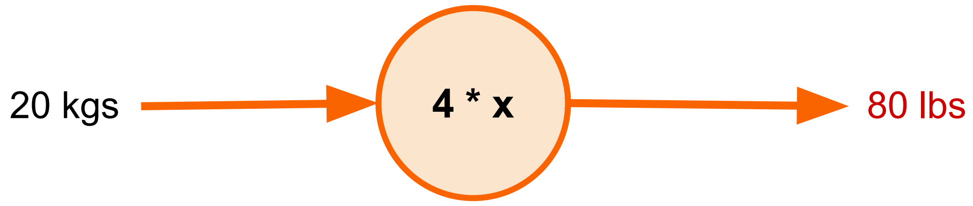
$$\Delta m = -E/x$$

A Weighing Scale - Revisiting



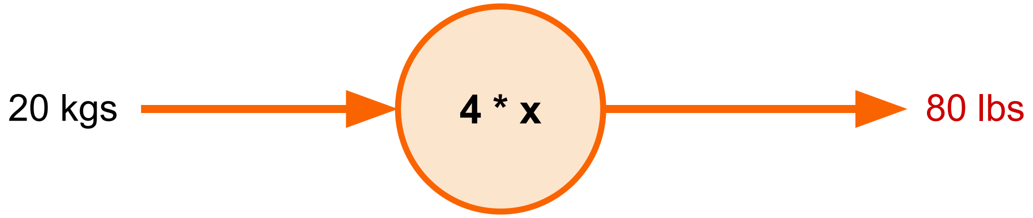
Target **44 lbs** and Error = **36 lbs**

A Weighing Scale - Prediction from Error



$$\Delta m = -36/20$$
$$\Delta m = -1.8$$

A Weighing Scale - Prediction from Error



$$\Delta m = -1.8$$

$$(4 - 1.8) * x$$

$$2.2 * x$$

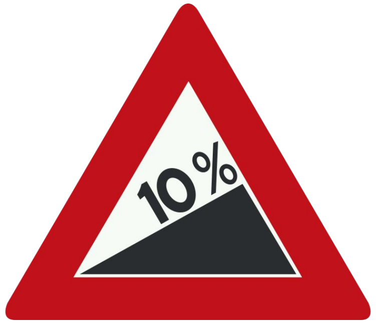
Accolite Digital

A relatively simple method of learning by refining the slope of a line.

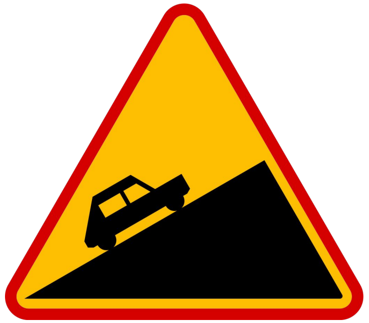
Could we classify using this process? maybe.. maybe not..

Slope & Gradient Descent

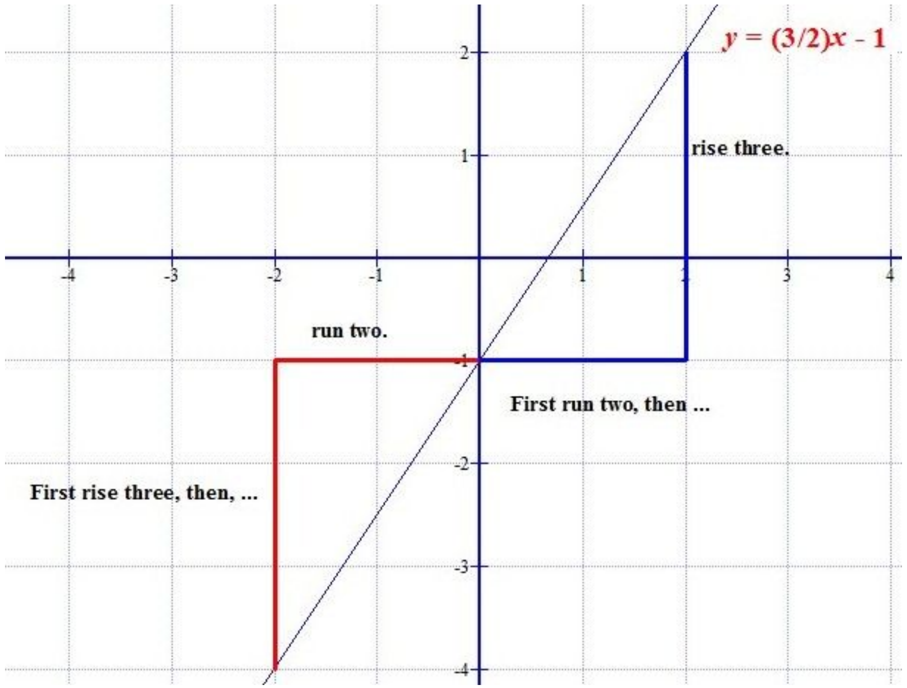
🤔



Slope warning sign
in Netherland



Slope warning sign
in Poland

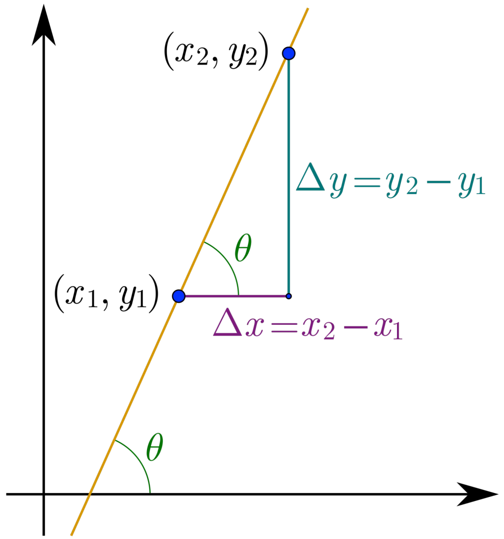


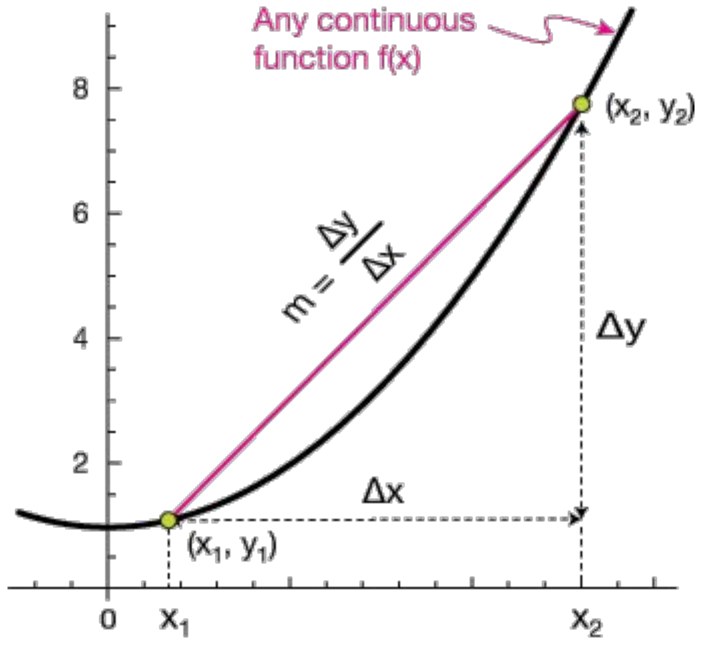
$y = (3/2)x - 1$
rise three.
run two.
First run two, then ...
First rise three, then, ...
-4
-3
-2
-1
1
2
3
4
2
1
-1
-2
-3
-4

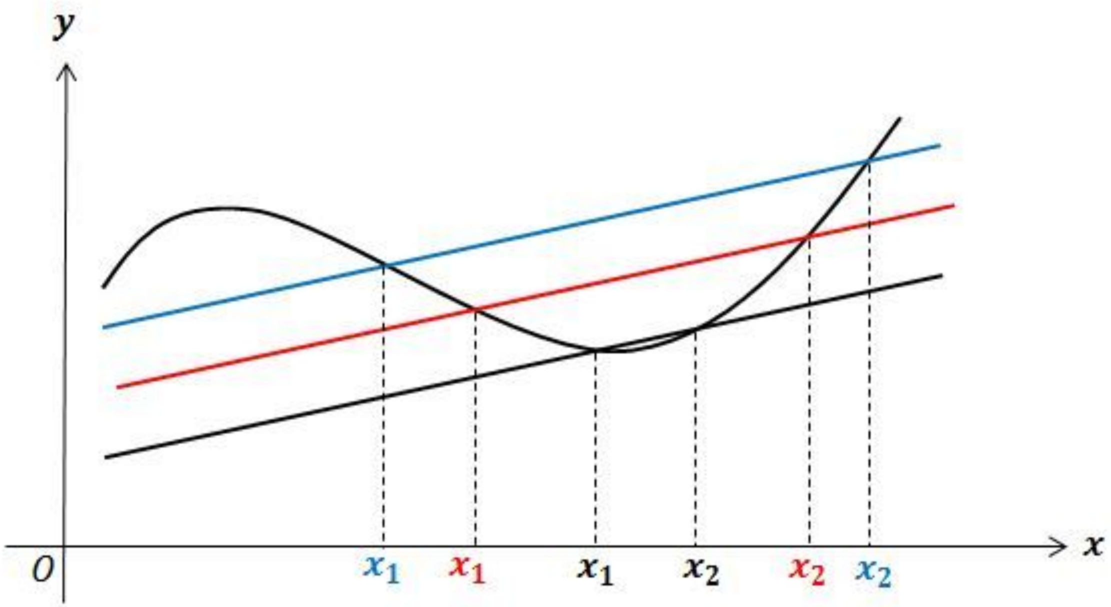
$$m = \frac{\Delta y}{\Delta x} = \frac{\text{vertical change}}{\text{horizontal change}} = \frac{\text{rise}}{\text{run}}.$$

$$m = \frac{y_2 - y_1}{x_2 - x_1}.$$

$$m = \frac{\Delta y}{\Delta x} = \tan(\theta)$$




Any continuous function f(x)
8
6
4
2
0
x_1
x_2
(x_1, y_1)
(x_2, y_2)
$m = \Delta y / \Delta x$
Δx
Δy



y
x
O
x_1
x_1
x_1
x_2
x_2
x_2

Calculus

🏃

Accolite Digital

Calculus

History & why is it needed in Neural Networks

- The word calculus is Latin for "small pebble". Back in the past, finding the average speed of falling objects was a straightforward process. You start your timer before the apple drops and stop it when it comes to rest. It was the time around when Galileo showed us that objects fall at the same rate or at the same time. Average speed is total distance/total time. Newton was curious about what would be the speed of the apple, which is constantly accelerating, at every point in time along its way to the ground. What would be the apple's velocity halfway? quarter way? etc.
- This was only possible if you could look at not the total time but a fraction of time A and time B and then compared the distance travelled. And the more we make the fraction smaller the more accurate velocity could be determined. Newton was interested in finding the velocity of a falling apple for a given instant. He knew that the velocity kept increasing until it comes to 0 when it hits the ground.
- Similarly, you can scale that example to a practical problem like, a planet on its path, approaching the sun and then going around it and then moving away and coming back again. In this example the velocity of the planet increases and then decreases. So does the planet have a constant velocity? What is the velocity of the planet on any given day or time?
- The slope of a straight line was pretty early on know but the slope of a curved line was something difficult. The slope of a line can be calculated as the ratio of the change in the dependent variable (Δy) to the change in the independent variable (Δx) between any two points on the line. In other words, the slope of the line gives the rate at which the value of y changes for each unit increase in x.
- But finding the rate of change becomes difficult if the object's movement doesn't follow a straight line, eg. an apple falling, a planet's movement, even if you look at the graph of the car going the fastest quarter mile, is not a straight line. So finding the slope or velocity of an object from the previous example. Also, this slope is a really good indicator in terms of whether something is moving faster or slowing down.

- Derivatives come into play when we are dealing with functions that are non-linear or in other words, curves. So the slope is just a number but the derivative is a function that gives you the slope(the number). That's why for a line, y = 2x, the slope is 2 but the derivative of y = x^3 is 3x^2 and now when you plug the value of x at 3x^2, you get the slope. So now finding the derivative is similar to finding the slope but because with the derivatives, the distance between (x1,y1) and (x2,y2) is infinitely small. In mathematical terms, it tends to be zero. You should check out the Calculus series on Youtube channel 3Blue1Brown.

- So it boils down to how steeper a curve is at a given value of x. The steeper the climb, the higher the slope and vice versa. And we need this value so that we can know 'how' should we decrease/increase the parameters used to make our guess. And that is how it is related to neural networks. If you recall the example of the weighing scale, we were actually updating the slope of our problem statement because the kgs to lbs relationship is linear and can be represented as y = guess * x or y =mx or y = slope*x

Calculus

How is it applied in neural network?

And all this calculus basically helps us simply do the following math for our cost function which could look like this:

$$y = x^3$$

then how do I find in which way to increase or reduce the value of **x** in order to reduce the value of **y** to the minimum.
By using the knowledge of slope we know that we can find the slope by taking a small step **h** and then this way the equation turns out to be. It's of the form

$$\frac{(y2 - y1)}{(x2 - x1)}$$

so, slope for our equation becomes

$$\frac{h^3 + 3h^2x + 3hx^2}{h} = 3x^2 + h^2 + 3hx \quad = \quad 3x^2$$

and this equation simplifies because the value of **h** is so miniscule and as it tends to zero we can ignore it altogether. And the final result that we get here is also same as applying power rule of derivatives to the equation.

$$y = x^n = nx^{(n-1)}$$

So, derivatives comes in real handy is simplifying such lengthy expansions. So again, the whole idea here is find a way for us to know how can we find the value of **x** that minimizes the **y.** And more importantly, we want the equation itself to tell this, we don't want to be guessing this. And slope or slope function(derivatives) tells us exactly that.

Calculus

rate of change

area under a curve

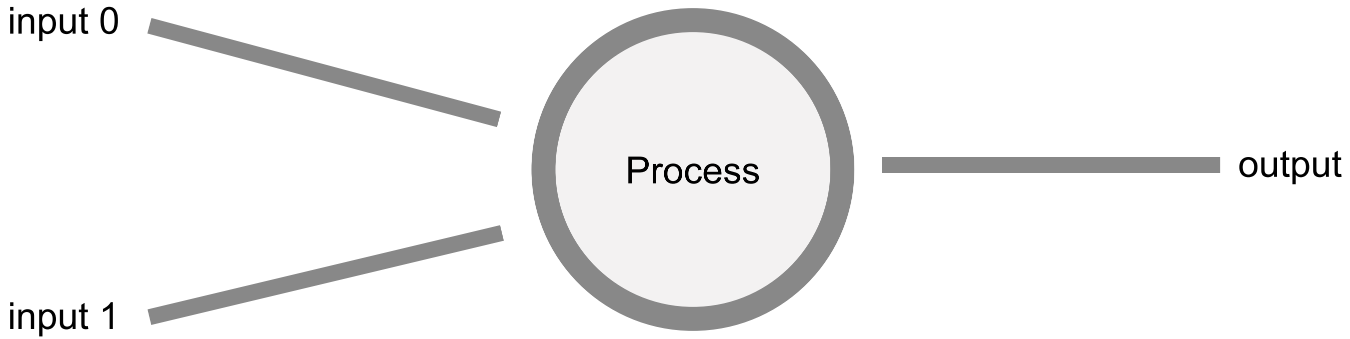
differential calculus

integral calculus

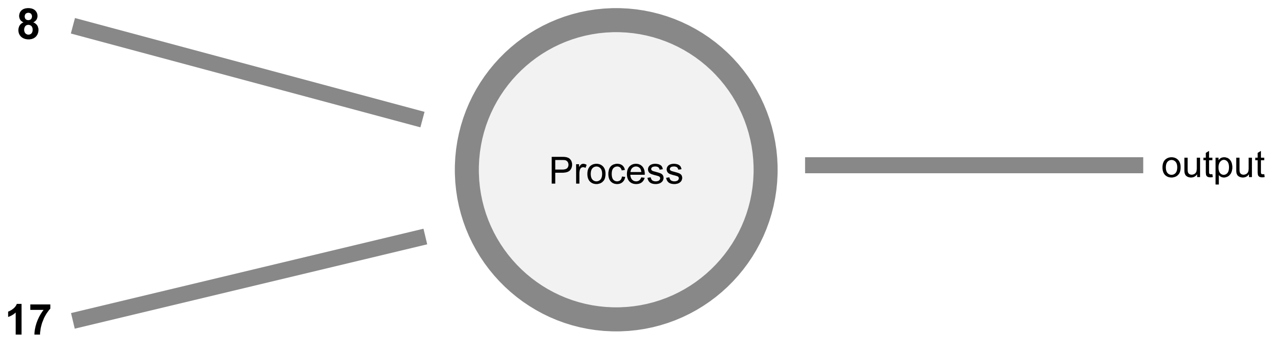
these tools share the same idea of

we can do something infinitely many times with approximations & yet get a finite answer

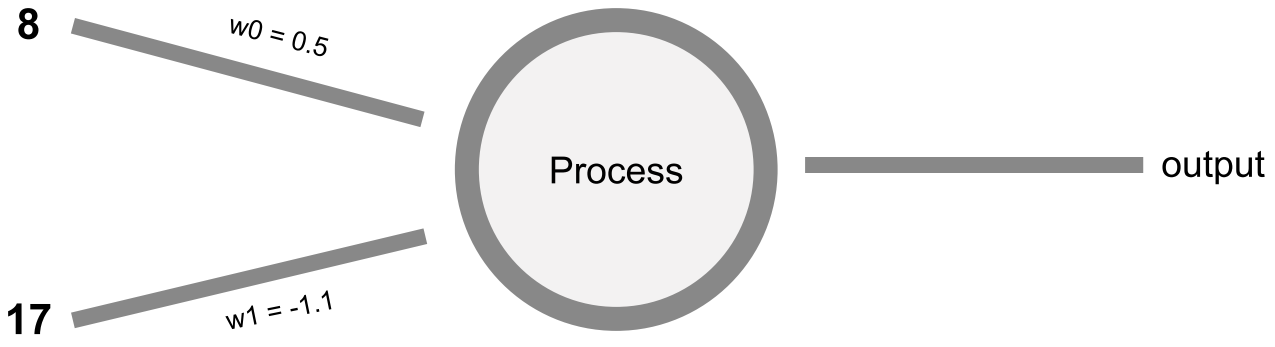
Perceptron


input 0
input 1
Process
output

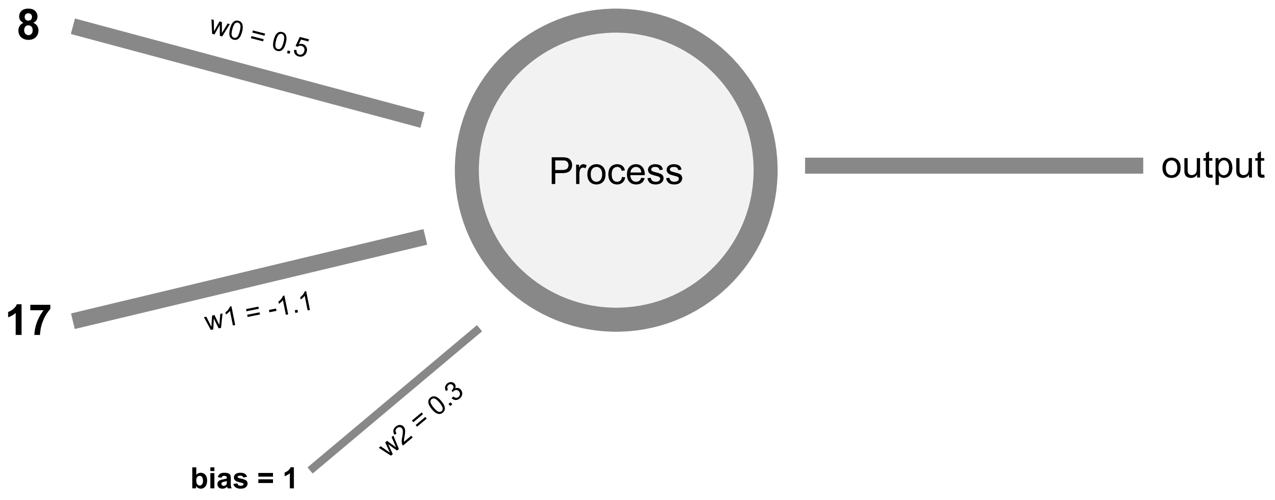
Step 1 — Receive Inputs



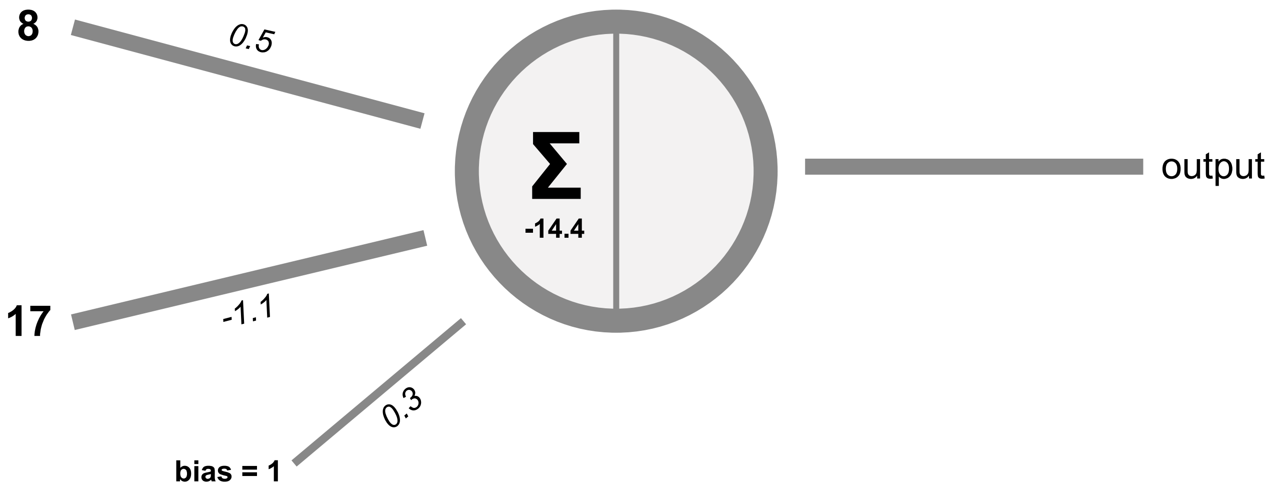
Step 2 — Weight Inputs



Step 3 — Add Bias



Step 4 — Sum Inputs

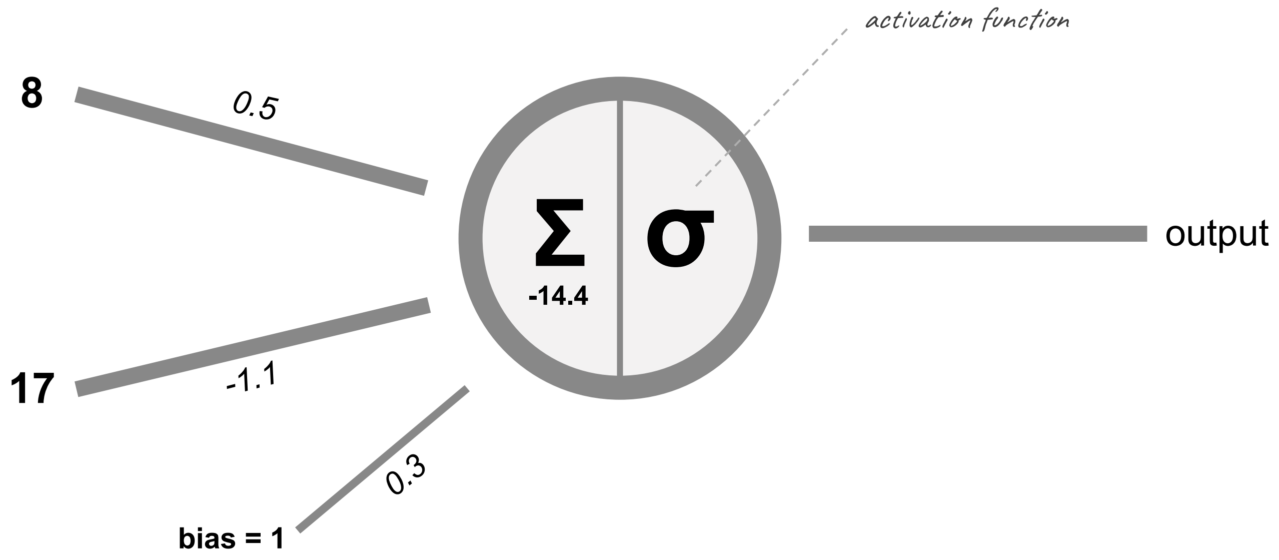


Input 0 * Weight 0 + Input 1 * Weight 1 + Bias * Weight 2

8 * 0.5 + 17 * -1.1 + 1 * 0.3

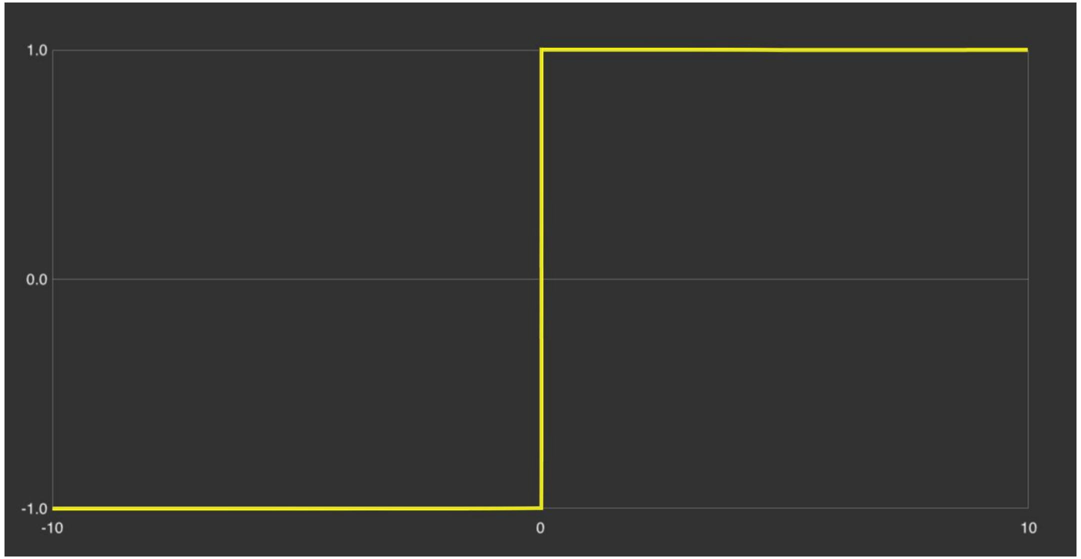
-14.4

Step 5 — Activation Function

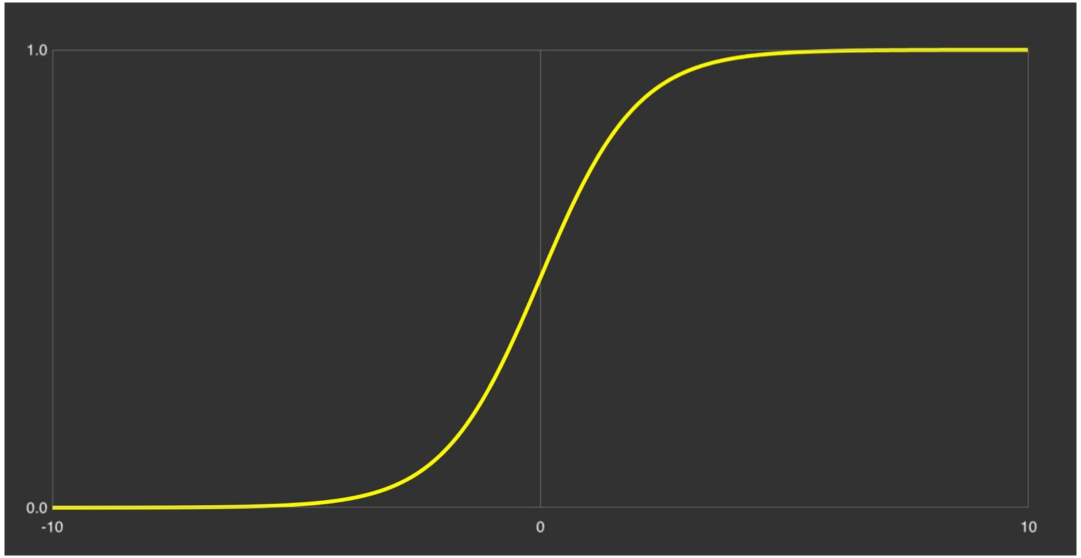


$$\sigma(x) = \frac{1}{1+e^{-1}}$$

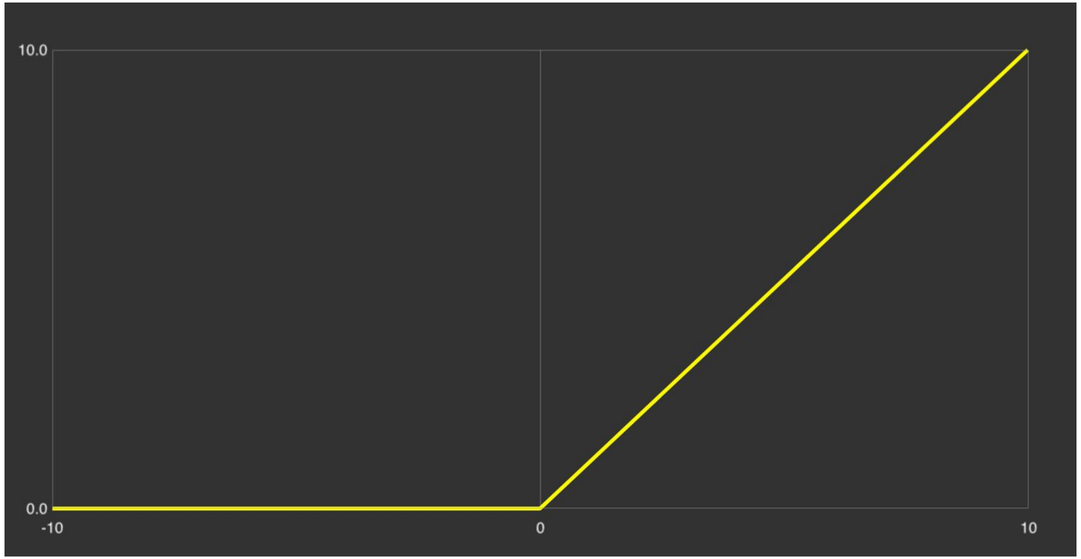
Sign Function



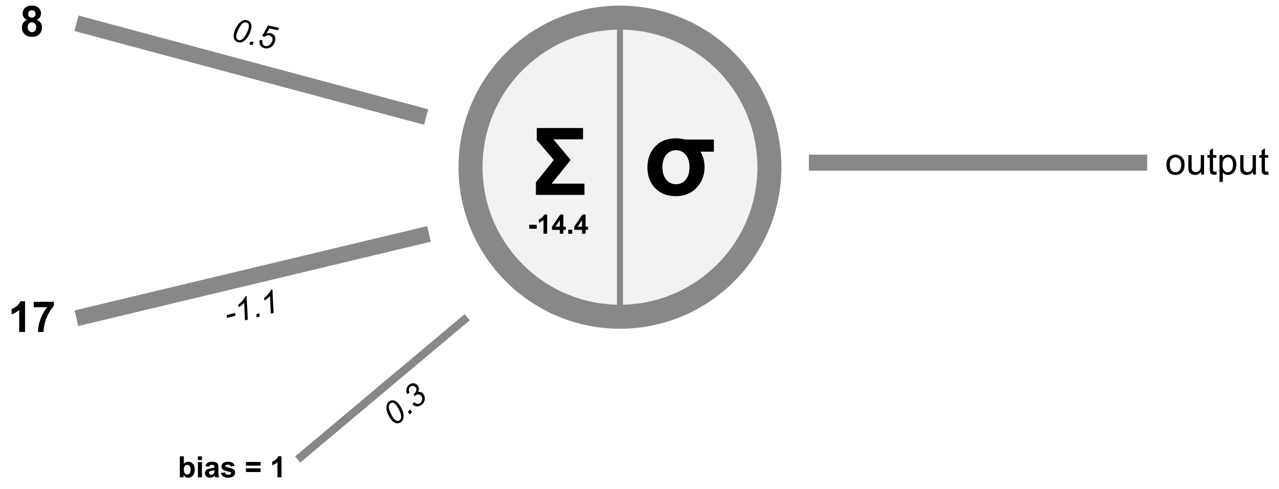
Sigmoid Function



ReLU Function

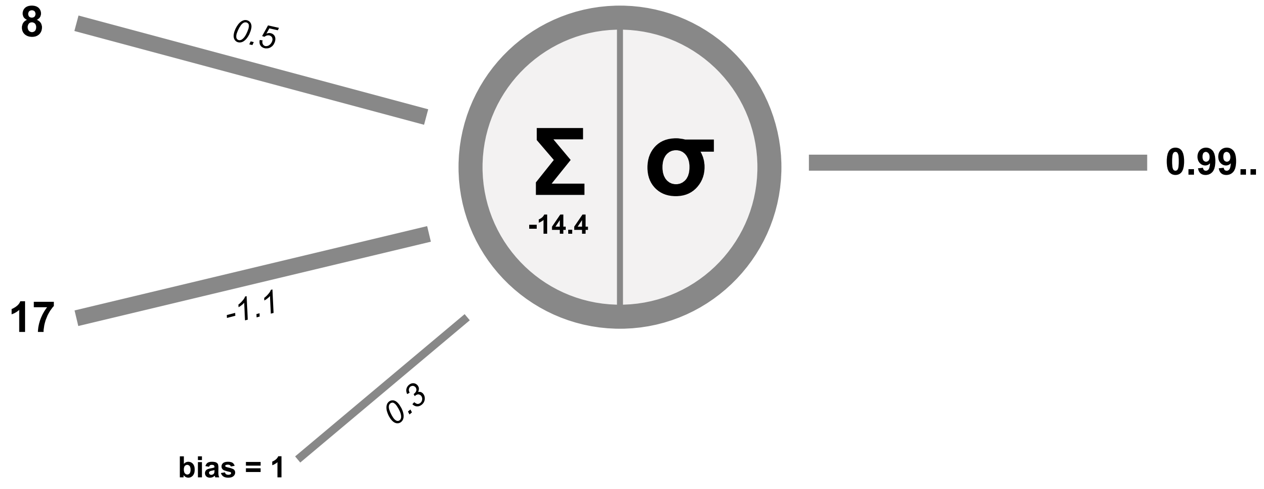


Step 6 — Generate Output

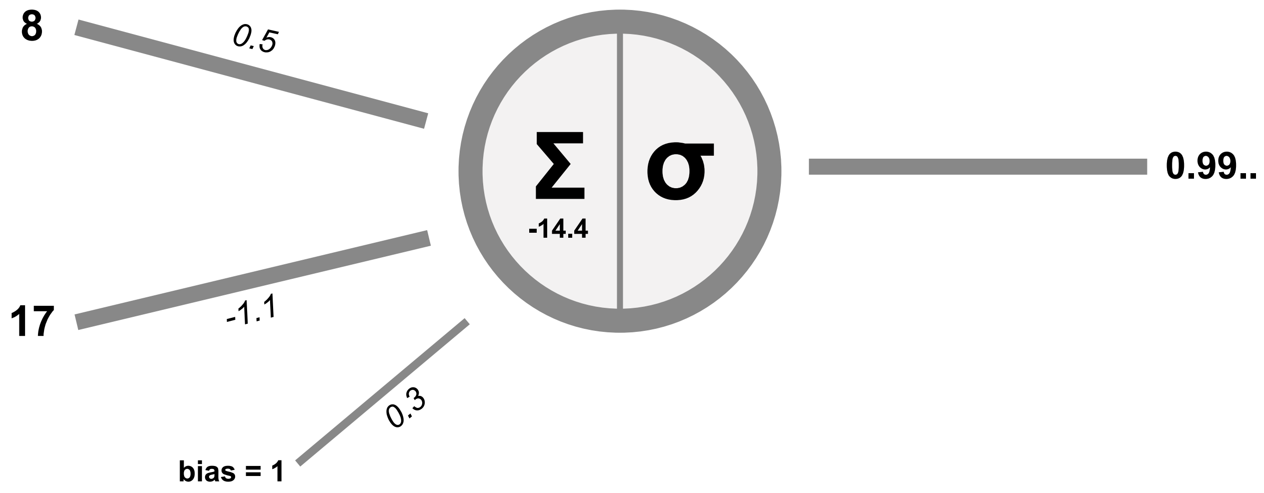


$$\sigma(-14.4) = 0.9999994426099414$$

Step 7 — Feedforward



Step 8 — Compute Error



Error = Prediction - Target

Squaring the error

Network Output	Target Output	Error (target-actual)	Error abs(target-actual)	Error (target-actual)^2
0.4	0.5	0.1	0.1	0.01
0.8	0.7	-0.1	0.1	0.01
1.0	1.0	0	0	0
Sum		**0**	0.2	0.02

Rate of Change for **m**

$$error = prediction - target$$

$$mx + b - y$$

$$f'(x) = x$$

$$J = (prediction - target)^2$$

$$y = x^2$$

$$f'(x) = 2x$$

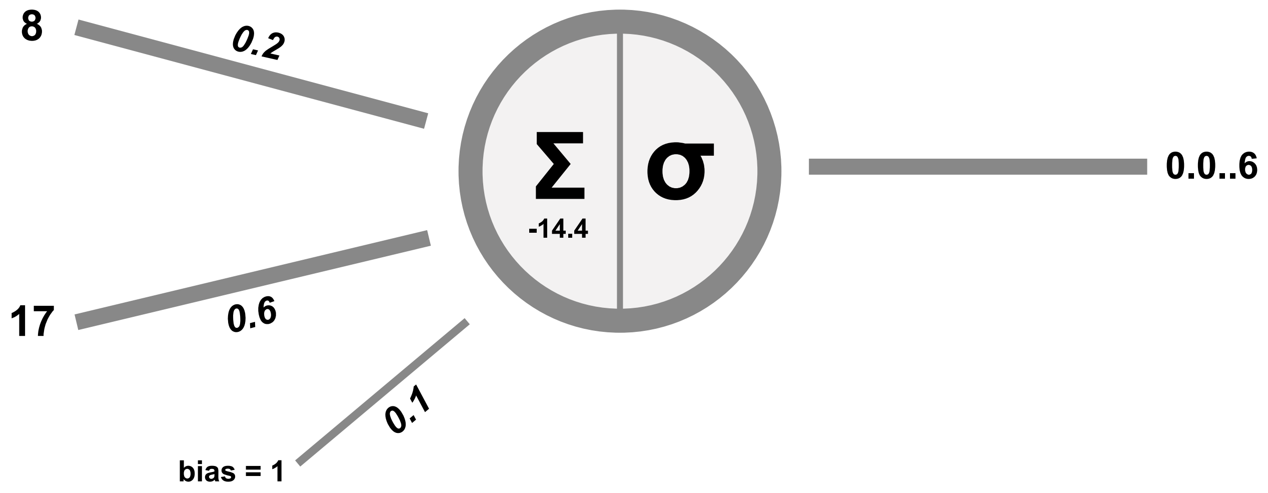
$$\frac{\delta J}{\delta m} = \frac{\delta J}{\delta error} * \frac{\delta error}{\delta m}$$

$$\frac{\delta J}{\delta m} = 2 * error * x$$

Rate of Change for **b**

exercise for you to try it yourself

Step 9 — Adjust Weights



Step 10 — Repeat 7 8 9

Code Repository

https://github.com/meherranjan/neuron-in-js

THANK YOU FROM ACCOLITE