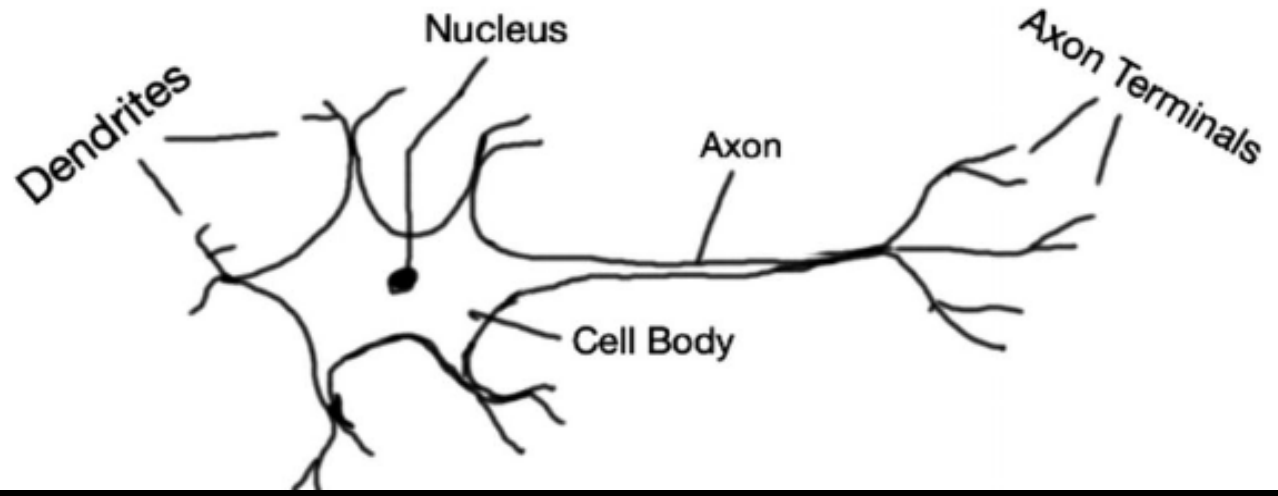
Neural networks– Perceptron

Neuron

neuron is made of the following:

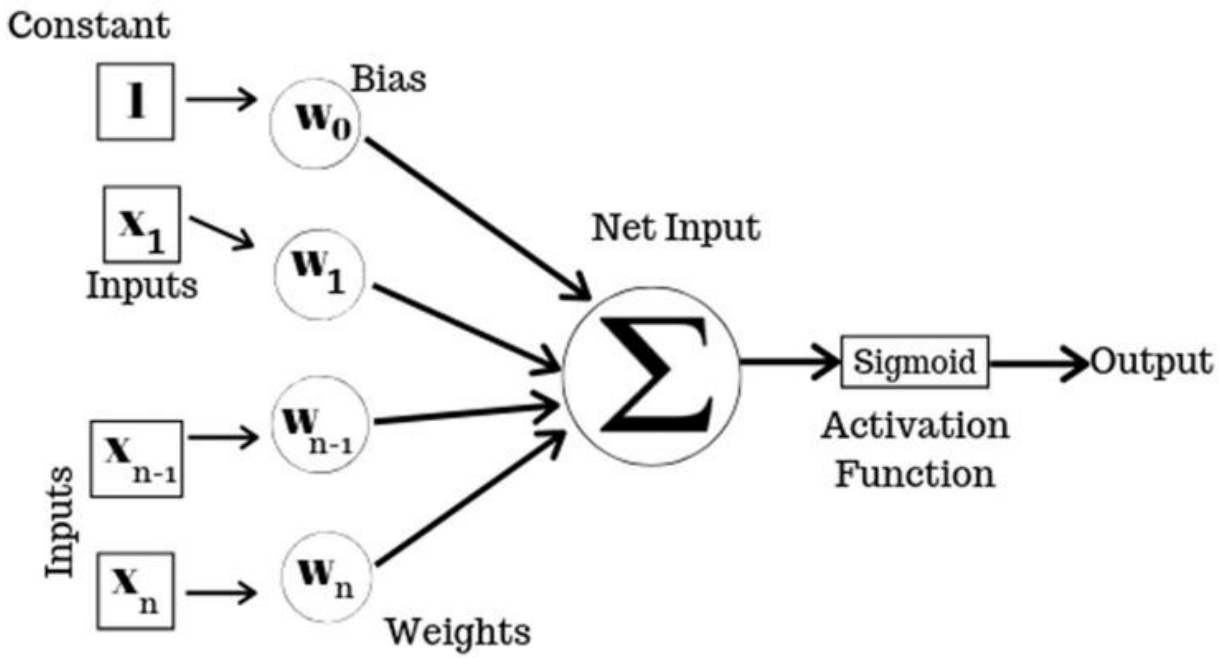
- Cell body (also called the *soma*)
- Dendrites
- Axon



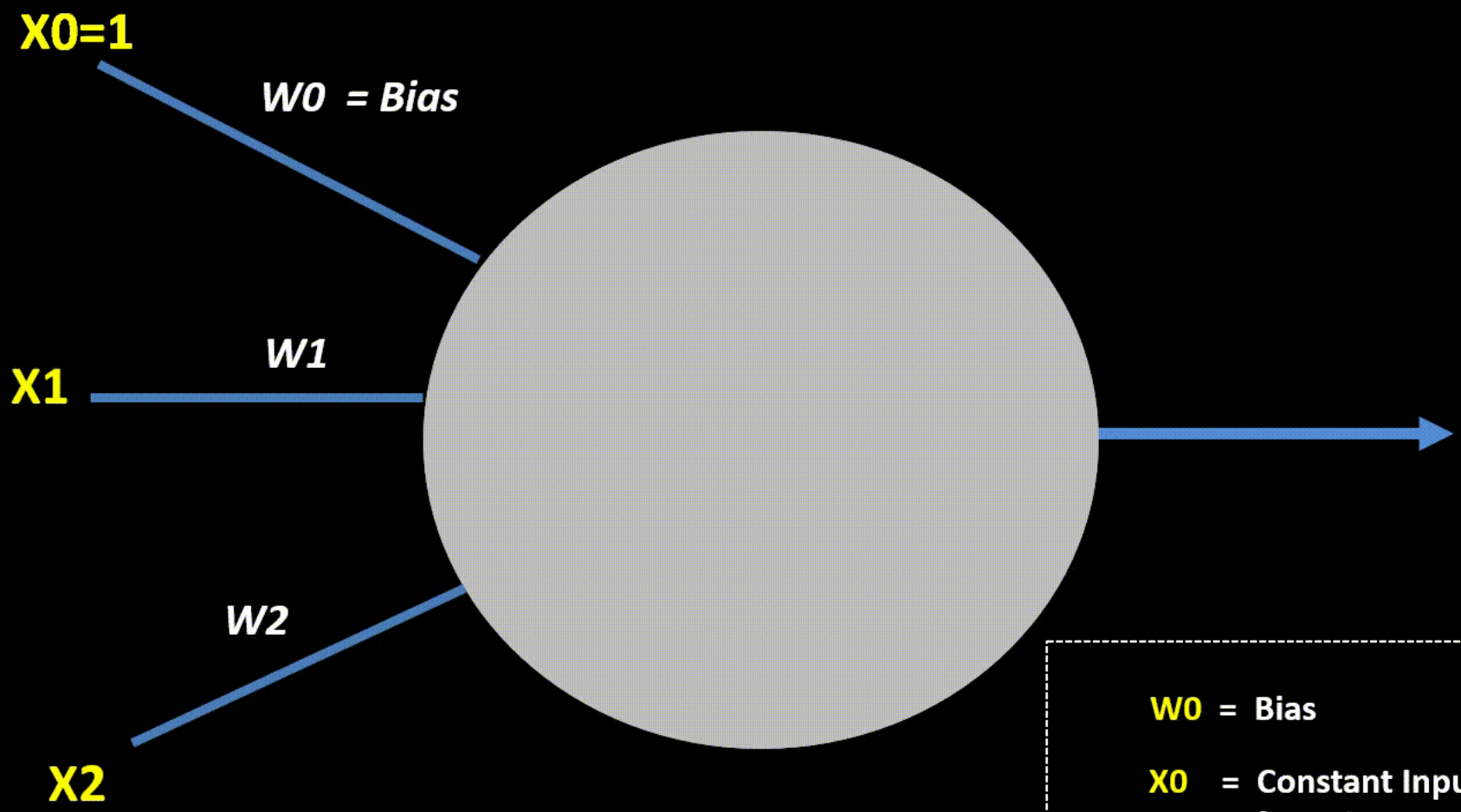
Perceptron = 1 Neuron

Perceptrons in Action

A perceptron process consists of four stages (see Figure 2-5). The following sections discuss each one.



Artificial Neuron



W0 = Bias

X0 = Constant Input 1 for Bias

X1,X2 = Attribute Inputs

W1,W2 = Weights of Inputs X1,X2

Artificial Neurons

An *artificial neuron* is a mathematical function, modeled on the working of a biological neuron. Each neuron takes inputs, weighs them separately, sums them up, and passes this sum through a nonlinear function to produce an output. Every neuron holds an internal state called the *activation signal.* Each neuron is connected to another neuron via a connection link.

Components of an artificial neuron include the following:

- Input signal
- Weights
- Bias
- Net input
- Activation function
- Output signal

Activation function

The activation function provides nonlinearity to the perceptron. In the current example, we use the Sigmoid function as the activation function. A Sigmoid function is defined for real input values and has a non-negative derivative at each point. The output lies between 0 and 1.

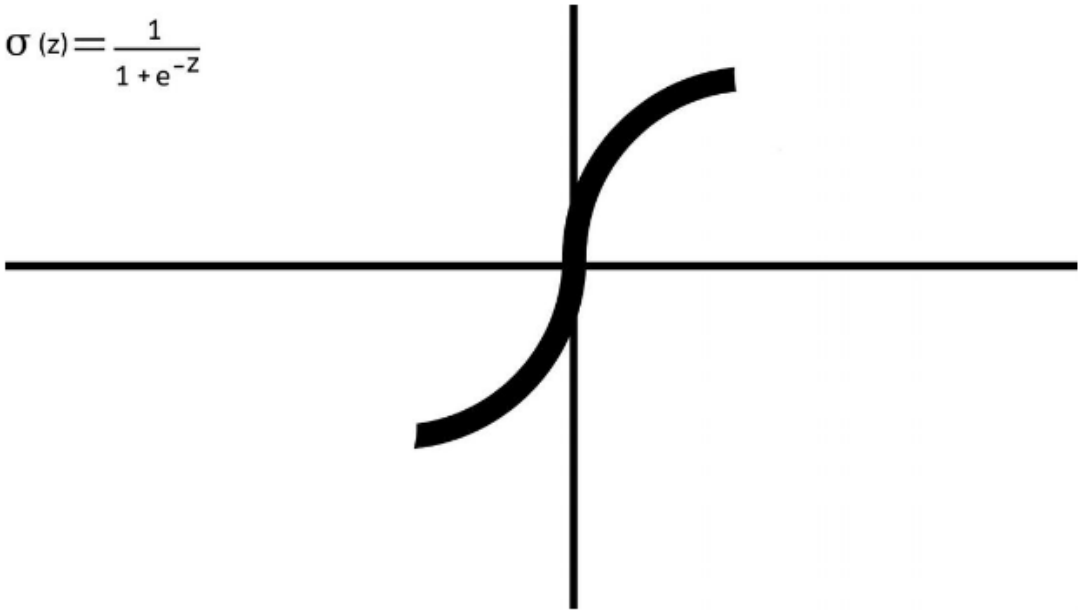
The Sigmoid function is expressed as `f(x) = 1/1+e^-x`.

At this point, forward propagation is complete, but the beauty of perceptrons and artificial neurons in general lies with the process of backward propagation, where a recalculation occurs.

THE SIGMOID ACTIVATION FUNCTION

A Sigmoid function is a mathematical function with a Sigmoid curve, also called an "S" curve (see Figure 2-2). It is a special case of the logistic function and leads to a probability of the value between 0 and 1.

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

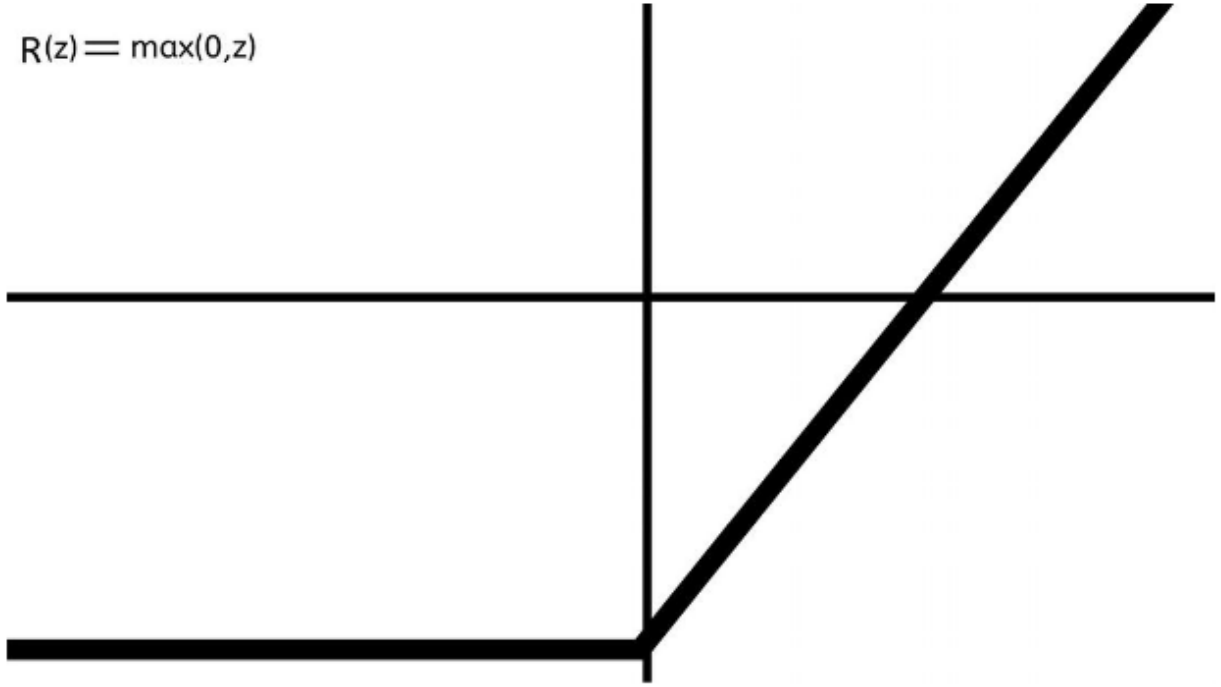


- It is especially used for models in which we have to predict the probability as an output. Since the probability of anything exists only between the range of 0 and 1, Sigmoid is the right choice.

- The Sigmoid function cannot be used in networks with many layers due to the vanishing gradient problem.

THE RELU FUNCTION

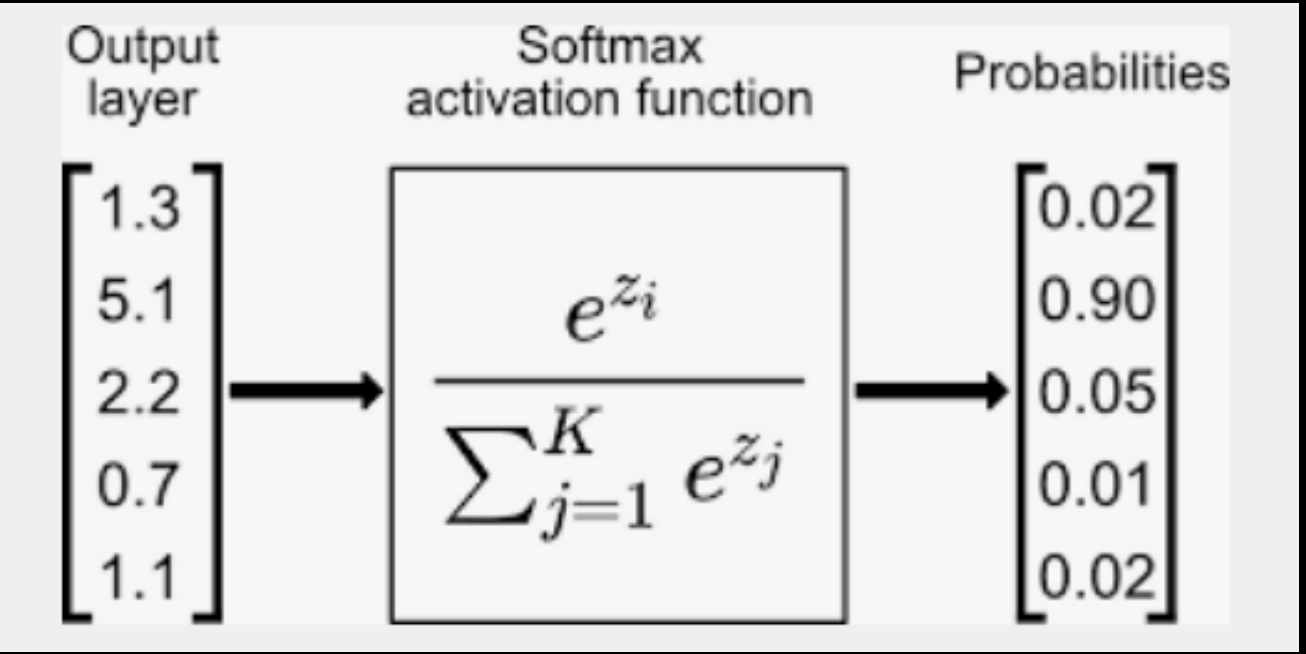
A rectifier or ReLU (Rectified Linear Unit) allows us to eliminate negative values in an artificial neural network, as it is a piecewise linear function. It will output the input directly if positive; otherwise, it will give 0 as the output (see Figure 2-3).



- Sparse activation of only about 50% of units in a neural network (as negative values are eliminated).
- Efficient gradient propagation, which means no vanishing or exploding gradient problems.

THE SOFTMAX FUNCTION

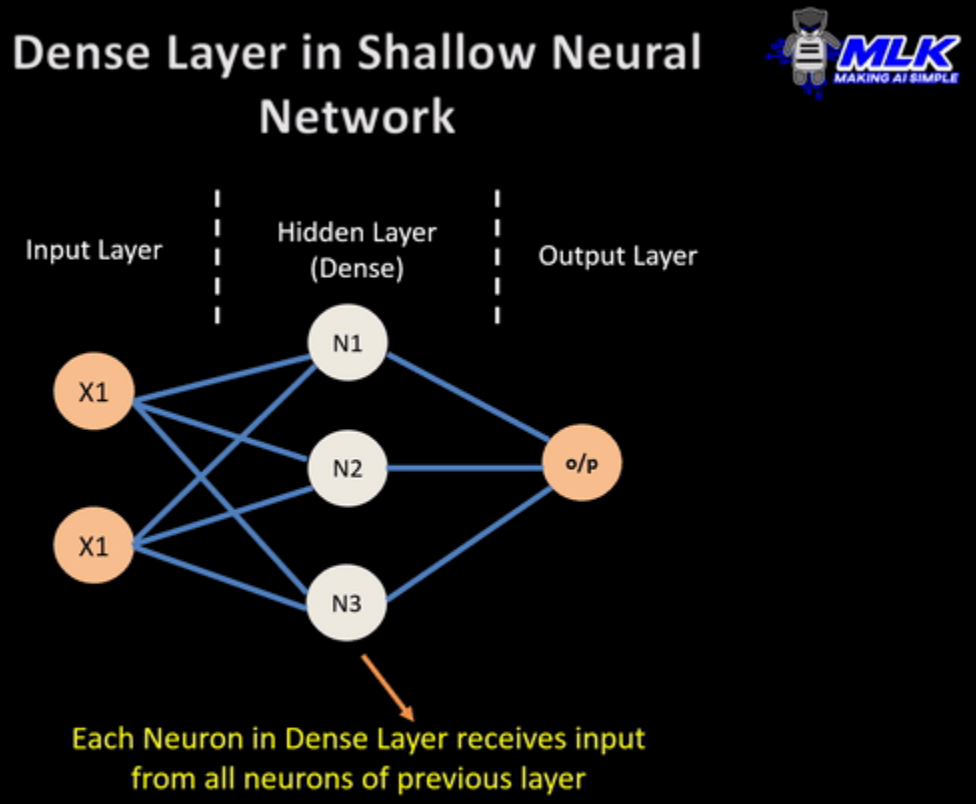
The Softmax, or normalized exponential, function is a generalization of the logistic function that outputs the probability of the result belonging to a certain set of classes. It converts a K-dimensional vector of arbitrary real values to a K-dimensional vector of real values in the range (0, 1) that add up to 1 (see Figure 2-4). It is akin to categorization logic at the end of a neural network.



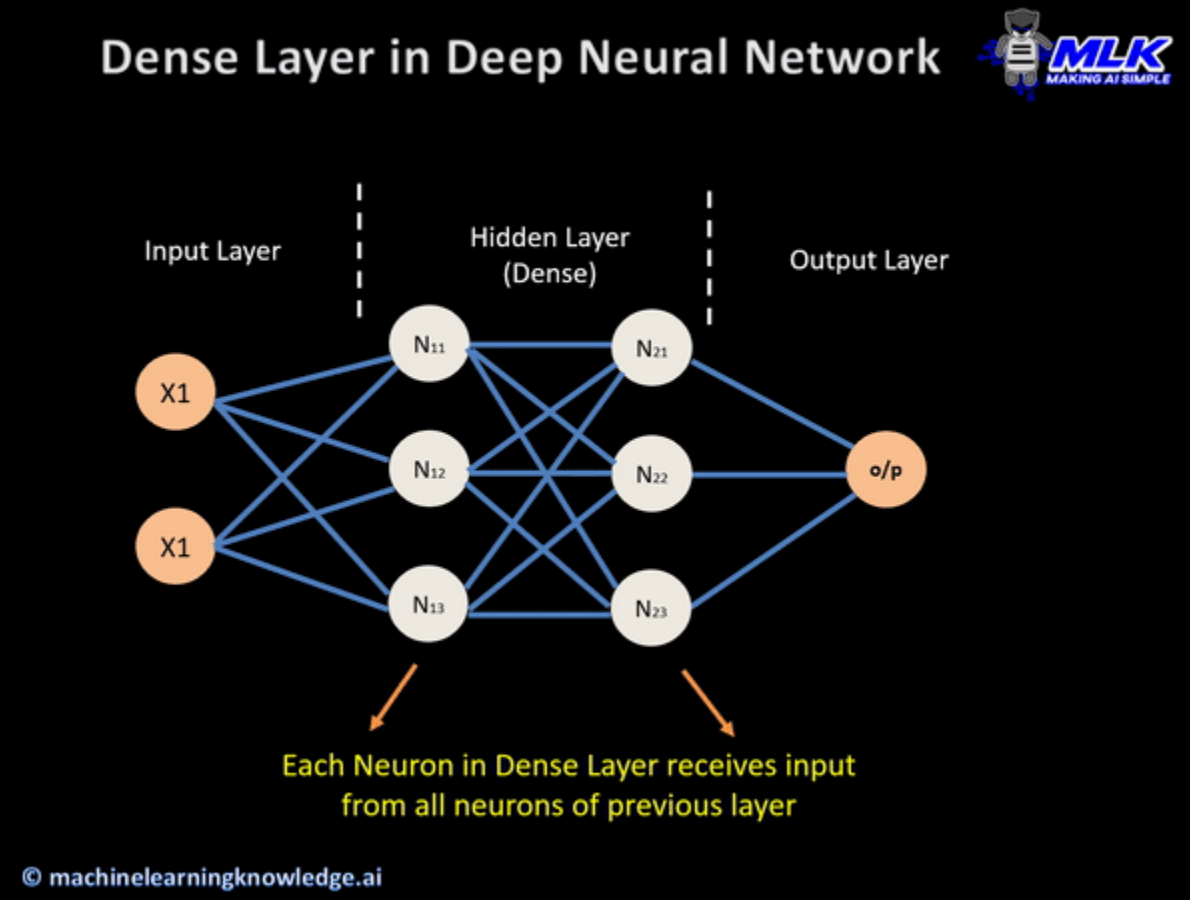
IMPORTANT TERMINOLOGY

- **Target/label:** The expected output of the function.
- **Loss function:** It computes the error for each iteration. The error is calculated as the difference between the expected output value and the prediction value.
- **Optimizer:** It minimizes the loss function.
- **Iteration:** The number of times training is to be done.
- **Confusion matrix:** It compares the predicted value with the target value and represents the number of correct and incorrect observations in a matrix.

Layers with neurons

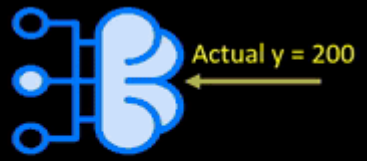


Layers with neurons



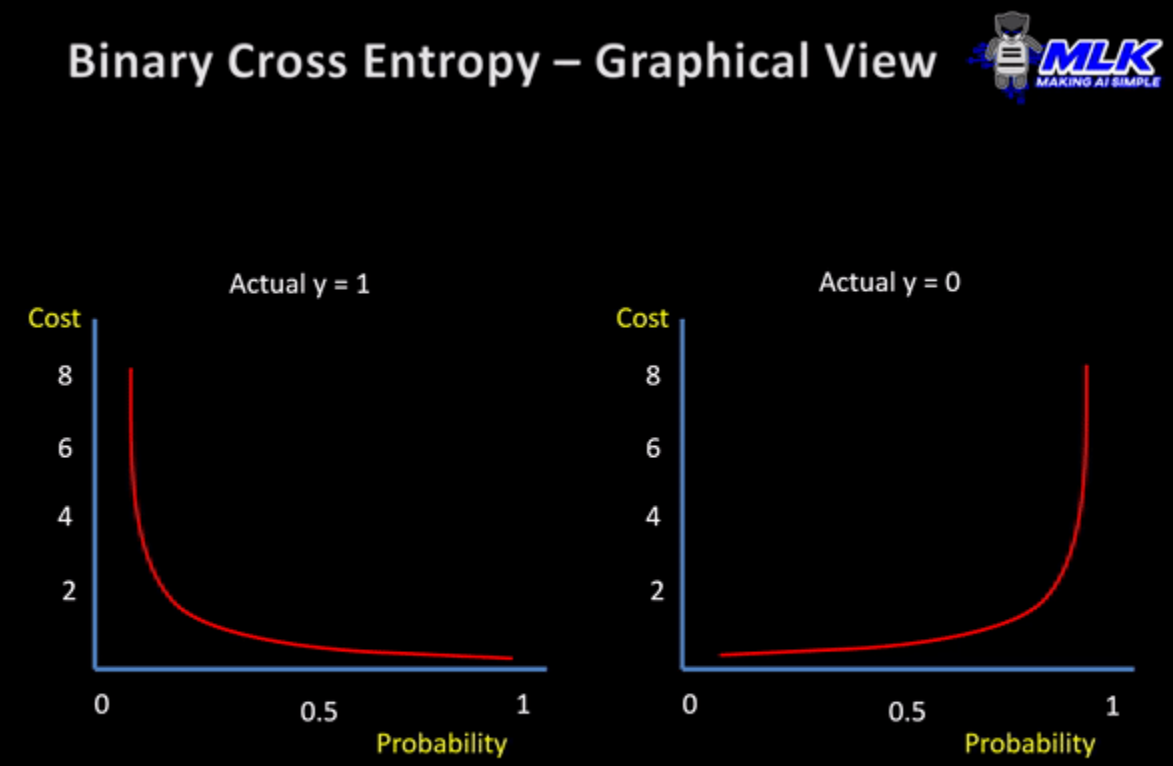
Cost function

Model

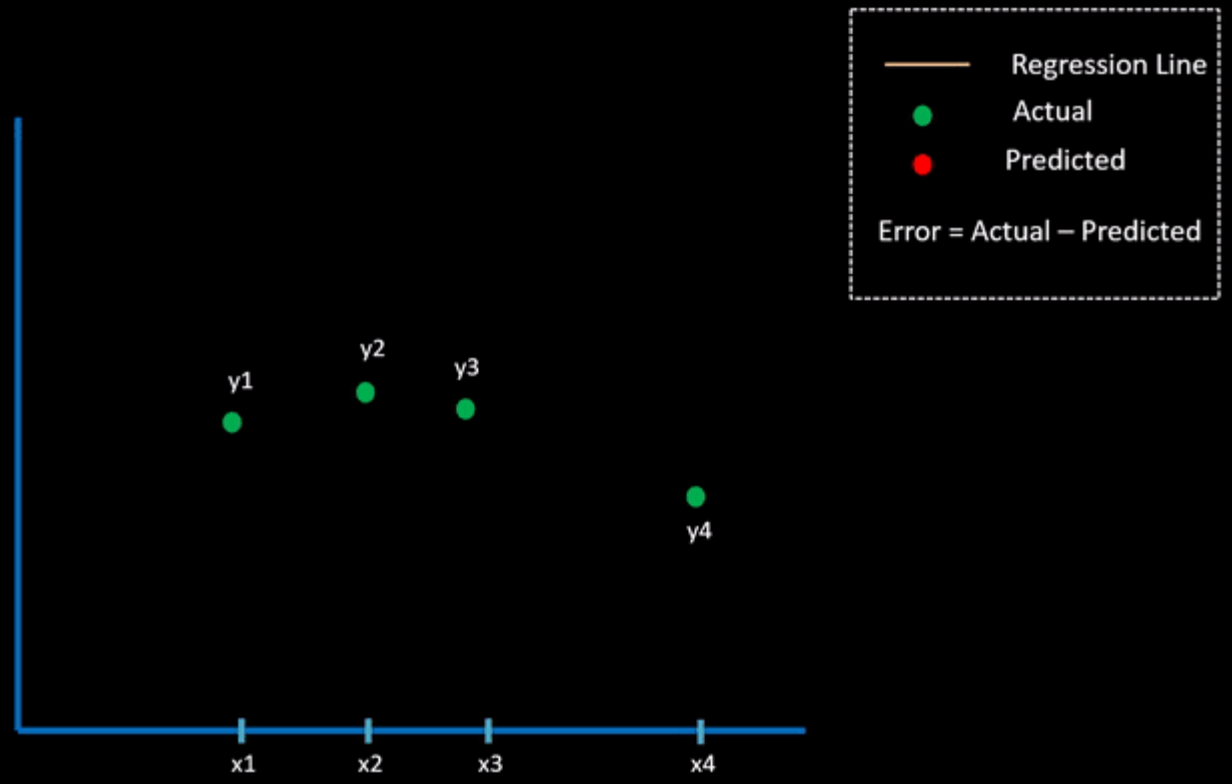


Cost Function = y – y'

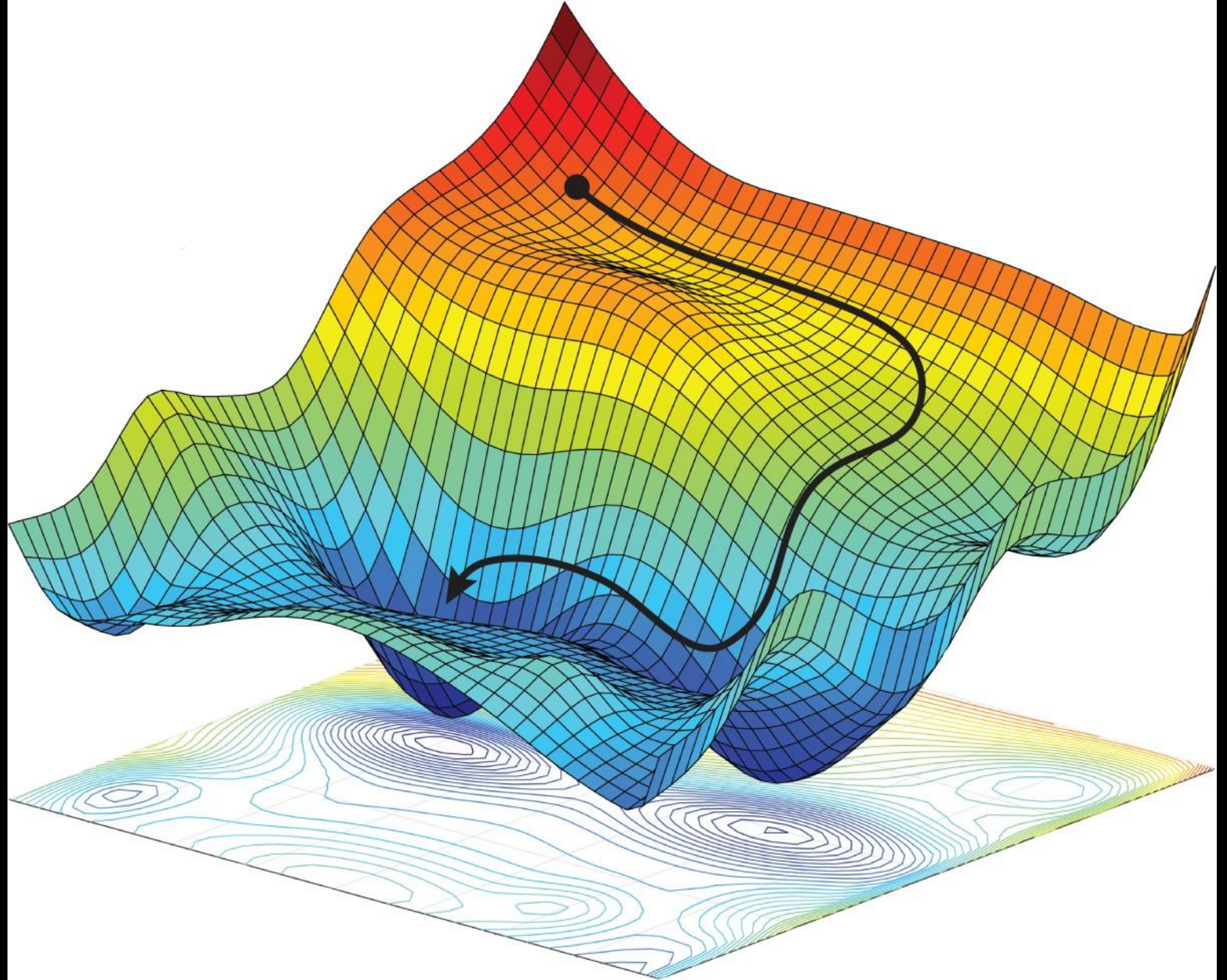
Cost function – example

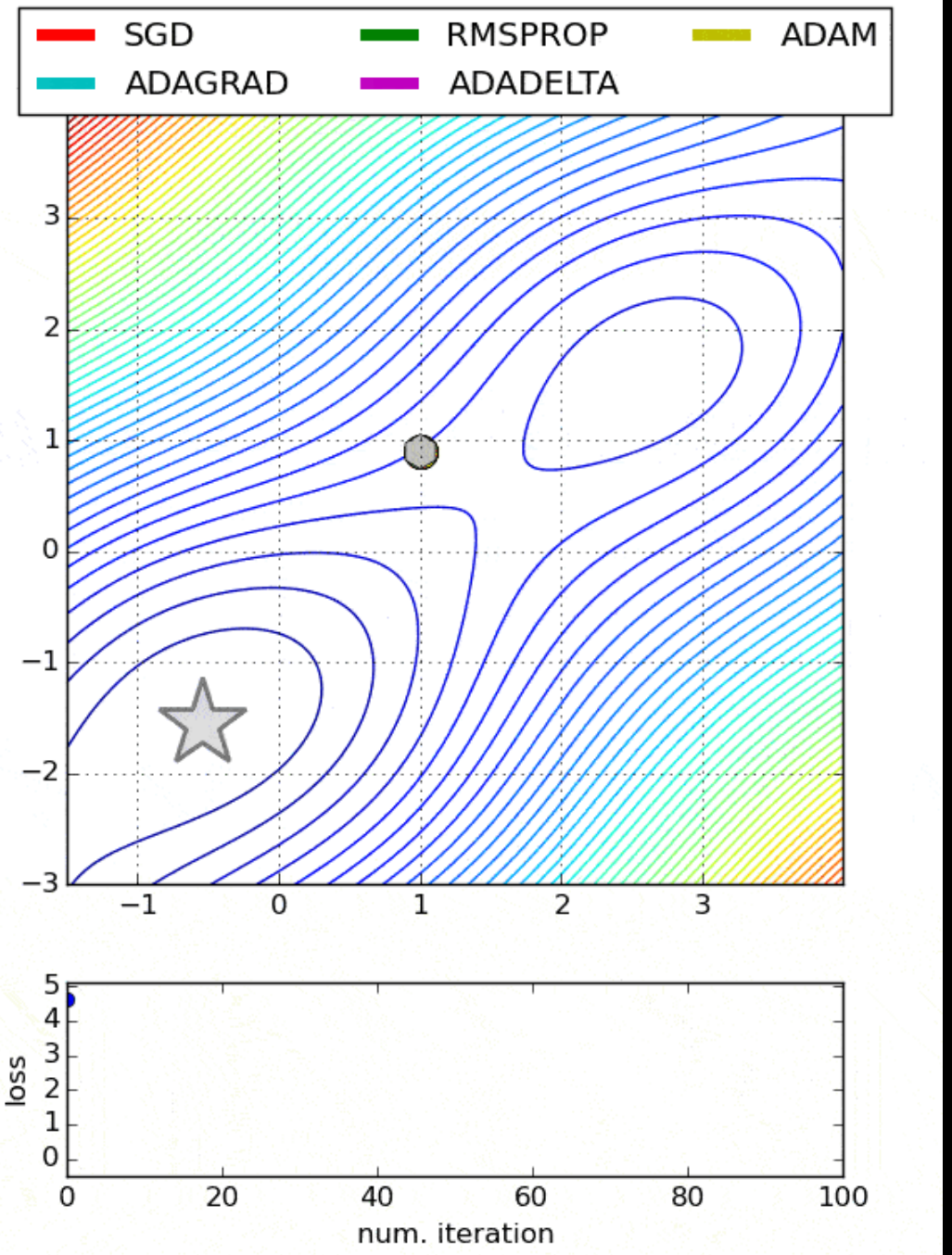


Cost function – example



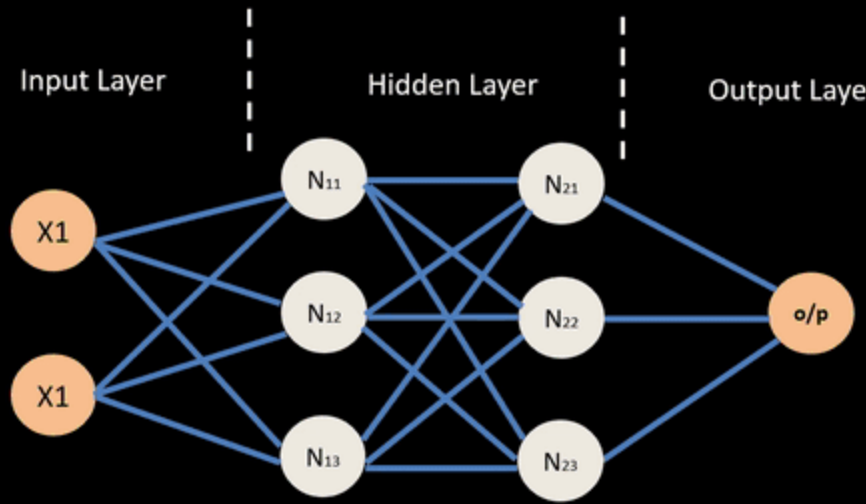
Cost function and Optimizer




SGD
RMSPROP
ADAM
ADAGRAD
ADADELTA
3
2
1
0
−1
−2
−3
−1
0
1
2
3
loss
5
4
3
2
1
0
0
20
40
60
80
100
num. iteration

Backpropagation

Neural Network – Backpropagation



Vanishing gradient
and
Exploiting gradient