**NATURAL LANGUAGE PROCESSING**

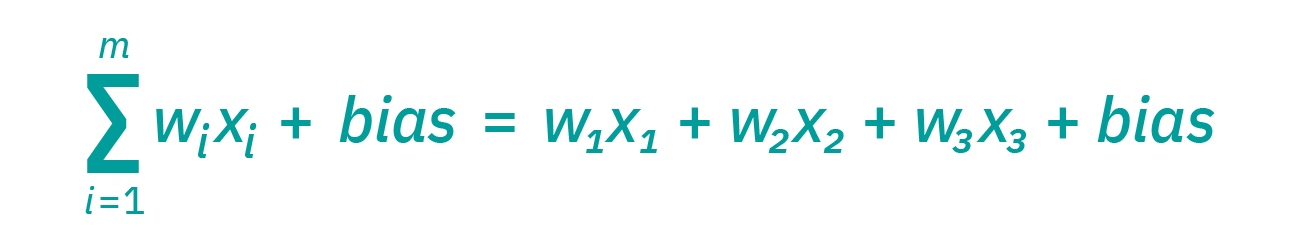
RINSHAN KOLAYIL

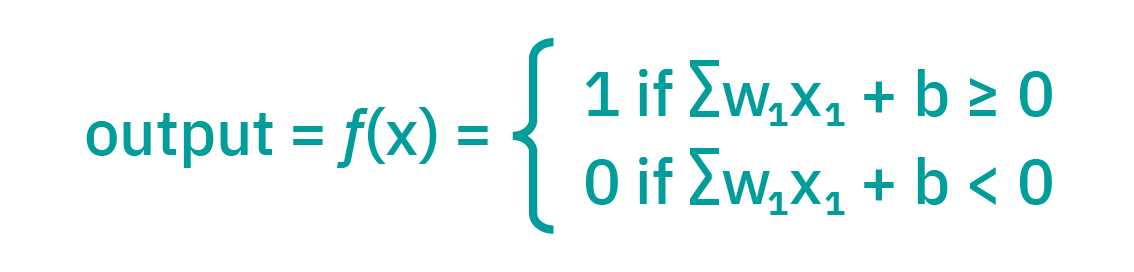
**INDEX**

|  |  |  |
| --- | --- | --- |
| Index | Topic | Page |
| **1** | **Neural networks** |  |
| 1.1 | Introduction to neural networks | 3 |
| 1.2 | Primitive neuron | 4 |
| 1.3 | Introduction to neuron and perceptron |  |
| 1.4 | Sigmoid neuron |  |
| 1.5 | Activation function & Types of activation functions |  |
| 1.6 | Cost functions |  |
| 1.7 | Gradient descent |  |
| 1.8 | Stochastic gradient descent |  |
| 1.9 | The feedforward model of neural networks |  |
| 1.10 | Disadvantages of feedforward model |  |
| 1.11 | Applying weights to the feedforward model |  |
| 1.12 | Back propagation algorithm |  |
|  |  |  |
|  |  |  |

**INTRODUCTION TO NEURAL NETWORKS**

Neural networks, also known as artificial neural networks inspired from human brain. It consist of input layer, one or more hidden layers and an output layer. Each node or artificial neuron connects one another with weight and bias. If the output of any node is above the threshold, the node is activated and then sending the data to the next layer. Otherwise no data is passed along the next layer of network.





The weights help determine the importance of any given variable with large one contributing more significantly to the output compared to other outputs.

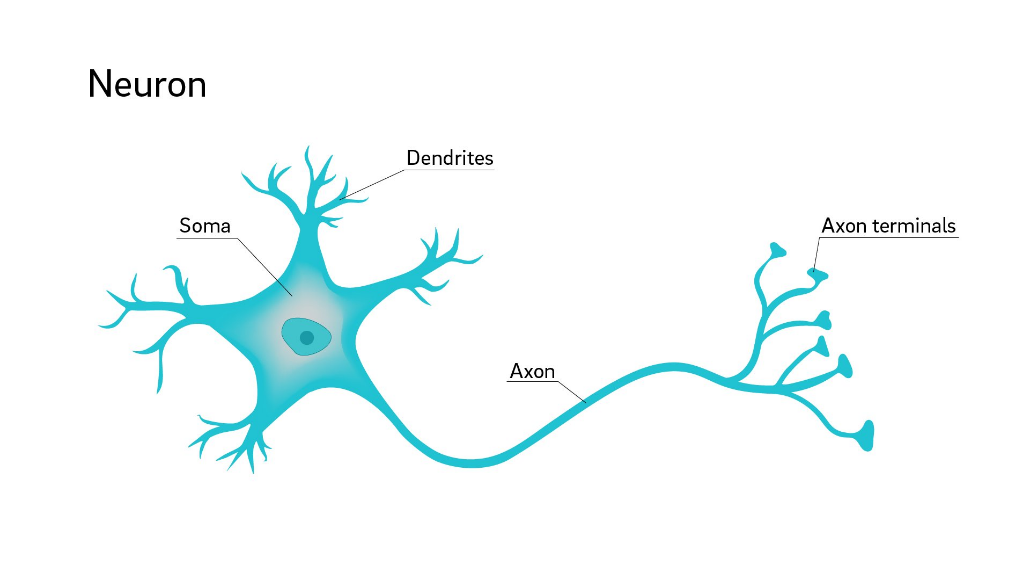
The output layer or decisions depend on the previous layers. Let’s say an example, since yesterday rains heavy you might bring an umbrella today.

History

* Warren McCulloch and Walter Pitts open the subject for creating computational model of NN
* D.O Hebb created the hypothesis based on the mechanism of neural plasticity
* Frank Rosenblatt invented the perceptron, the first neural network
* Ivakhnenko - First function NN with many layers

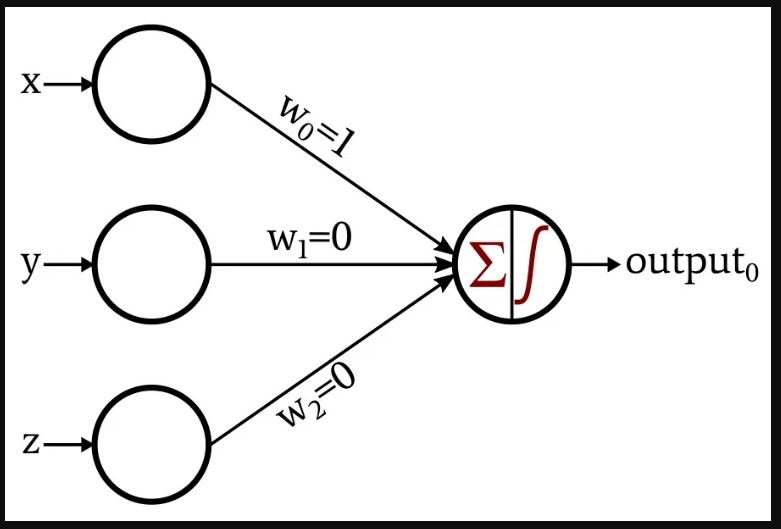
**HOW ARE NEURAL NETWORK SIMILAR TO BIOLOGICAL NEURON**

|  |  |
| --- | --- |
| **BIOLOGICAL NEURON** | **ARTIFICIAL NEURON** |
| Dendrites Brings inputs from other neuron to current neuron | X1, X2, X3 |
| Synaptic gap modifies the signal (Signals in the form of electrical and chemical form) | Weights |
| Axon | Output connection |
| Soma act as the processing unit and process the input from dendrites | Activation function |



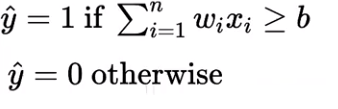
**PRIMITIVE NEURON**

The smallest neuron of processing available to a programmer of a particular machine

**INTRODUCTION TO NEURONS AND PERCEPTRON**

Perceptron act as a linear classifier and it can do binary classifier. Perceptron output is one if weighted sum of inputs greater than the threshold.

Founder - Frank Rosenblatt



**HOW DO WE TRAIN A PERCEPTRON?**

We have following for training,

* One model (Perceptron)
* One loss function

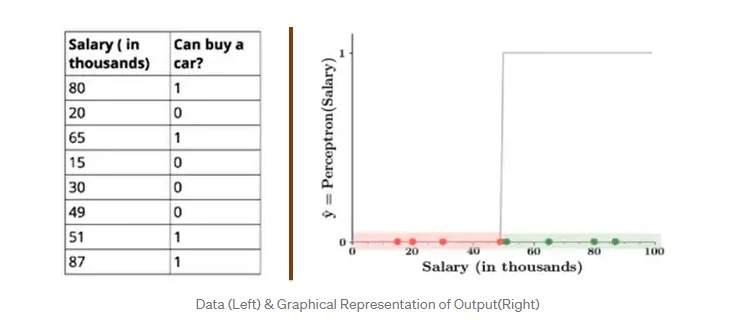
**SIGMOID NUERON**

In a perceptron, the output is equal to one if the weighted sum of inputs is greater than the threshold, otherwise zero but in a sigmoid neuron, small changes in input cause only small changes in output.

Let’s say an example, threshold is 50K

* One person will buy a card if he earns 50K or more
* In the case of perceptron, he will not buy a car if he has 49.9K. In the case of sigmoid neurons, he will.

Diagram

Description automatically generated with medium confidence

**ACTIVATION FUNCTION**

Activation function maps the resulting values between 0 to 1 or -1 to 1 (Depending on the function). Activation function decides whether the input is important or not.

Types of activation functions

* Linear activation functions
  + Binary step function
  + Linear or identity activation function
* Nonlinear activation function
  + Sigmoid or Logistic activation function
  + Tanh or hyperbolic tangent activation function
  + ReLU (Rectified Linear unit) activation function
  + Leaky ReLU

**BINARY STEP FUNCTION**

The input fed to the activation function is compared to a certain threshold, if input greater than it, then the neuron is activated, else it is deactivated meaning that the output not passed to the next hidden layer.

Chart, line chart, box and whisker chart

Description automatically generated

Shape

Description automatically generated with medium confidence

**LINEAR ACTIVATION FUNCTION**

The activation proportional to the input. In other words, no activation or identity function

F(X) = X

Chart, line chart

Description automatically generated

**WHY NONLINEAR ACTIVATION FUNCTION.**

Linearity, an example if body weight effects the diabetes risk depend on height. In other words, height and weight have an interaction effect. If a variable on x axis and the predictions on y axis and it is not a straight line, in such cases we will use nonlinear functions (or curved everwhere).

Non-linear means that the output cannot be reproduced from the linear combination of the inputs.

Without a non-linear activation function in the network, no matter how many layers it had, it would behave just like a single-layer perceptron. Because summing these layers give you just another linear function.

Chart

Description automatically generated

Video reference

* <https://www.youtube.com/watch?v=wgATTmC0JSQ>
* <https://www.youtube.com/watch?v=uulfOWzIe9o&t=178s>

**SIGMOID ACTIVATION FUNCTION**

Sigmoid function will compress the output ranges. Small change in the input will cause only small change in output. Founder – Pierre Francios Verhulst between 1838 and 1847

Link - <https://medium.com/analytics-vidhya/activation-function-sigmoid-7673dc0efcbe>

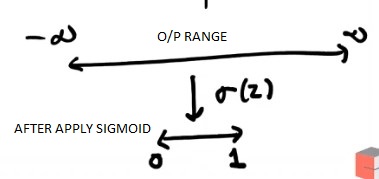


If z is 100, then sigma (z) close to 1. If z is -200, then sigma (z) close to 0. In other words, the output range is lies between 0 and 1 after applying the sigmoid function.

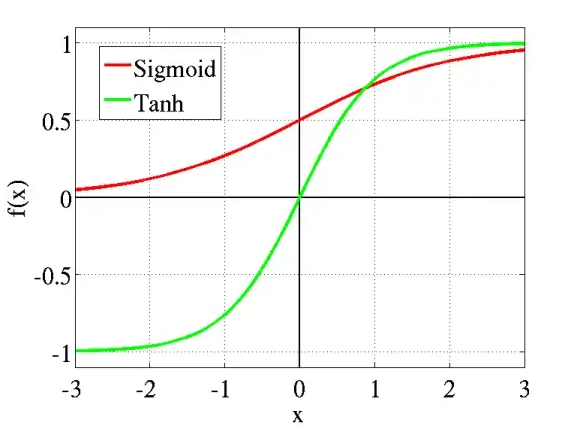
Let’s say an example (For binary classification), if output of a class 1 is 0.7, then the probability of class zero is (1 - 0.7).

* P(Y=1) = sigma (z)
* Then P(Y=0) = 1 – (z)

Sigmoid function if differentiable. That means, we can find the slope of the curve at any two points.

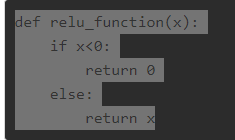
**TANH ACTIVATION FUNCTION**

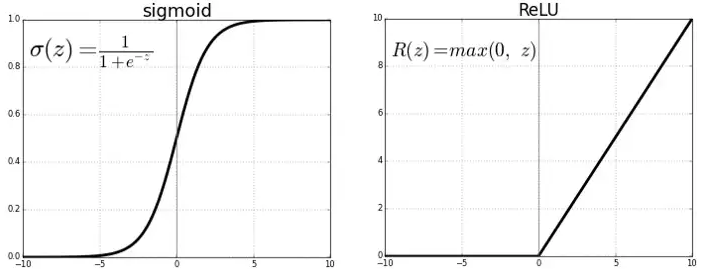
* Similar to sigmoid.
* Range of output lies between -1 and 1
* Differentiable
* Advantage is negative inputs mapped to strongly negative and zero input will be mapped to zero in the graph
* Used mainly in classification of two classes



**RELU ACTIVATION FUNCTION**

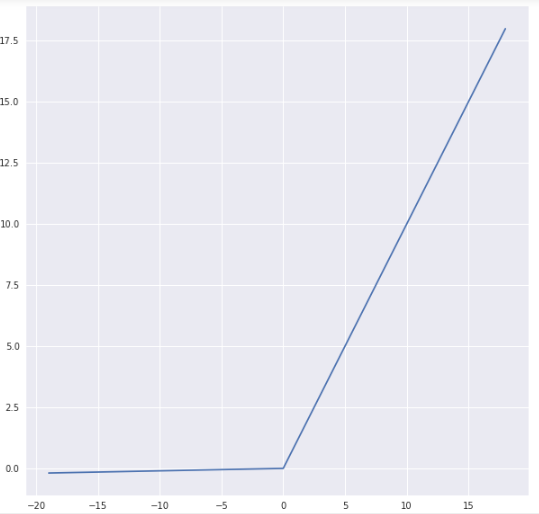
* Rectified Linear unit
* Founder – Fukushima in 1975.
* Research paper - <https://link.springer.com/article/10.1007/BF00342633>
* F(X) = max (0, z)
* It does not activate all the neurons at the same time.
* Most used activation function
* For negative input values, the result is zero, which means the neuron is not activated. Since certain number of neuron does not get activated.





**LEAKY RELU ACTIVATION FUNCTION**

* Ranges between (–infinity, +infinity)
* Both Leaky ReLU and ReLU are monotonic and also their derivative
* In ReLU activation function, the gradient is zero for values less than zero which would deactivate the neuron in that region.
* In Leaky ReLU activation function, F(X) = max(αx, x)
  + α is slope (example it can be 0.01)
  + x is small linear component
  + so the gradient can be negative as well



If you carefully see at the graph, the values less than zero, slight negative values.

**COST FUNCTIONS**

The difference between actual and predicted value is loss. The function to determine the loss is called as loss functions over a single training example.

Cost function used to find the average of loss function over the entire dataset.

**Error = | Y\_Actual – Y\_Predicted |**

**Error over all example =**

**Average Mean Error over all example =**

**Average Mean Squared Error over all example = \*\*2**

* Because of modulus order does not matter
* Mean squared error is differentiable, so we are squaring error term

**TYPES OF COST FUNCTION**

Many Cost functions are available based of problem we are solving, data quality and distribution and algorithm we are using.

1. Regression
   1. Mean squared error (MSE)
   2. Mean absolute error
   3. Root Mean squared error
   4. Mean bias error
   5. Huber loss
2. Binary classification
   1. Likelihood Loss
   2. Binary cross entropy
   3. Hing loss and Squared Hing loss
3. Multinomial classification
   1. Categorical cross entropy
   2. Kullback Leibler Divergence

**MEAN SQUARED ERROR**

Loss =

**MEAN ABSOLUTE ERROR**

* This is the better measure than mean squared error

Loss =

**ROOT MEAN SQUARED ERROR**

* This is an ideal solution if we don’t penalize the larger errors

**MEAN BIAS ERROS**

It is similar to mean absolute error but it can have negative values, which leads to disadvantage of negative and positive values can cancel each other. So it is better applied when the error has only one direction.

Loss =

**HUBER LOSS**

* Combines Mean absolute error and mean squared error