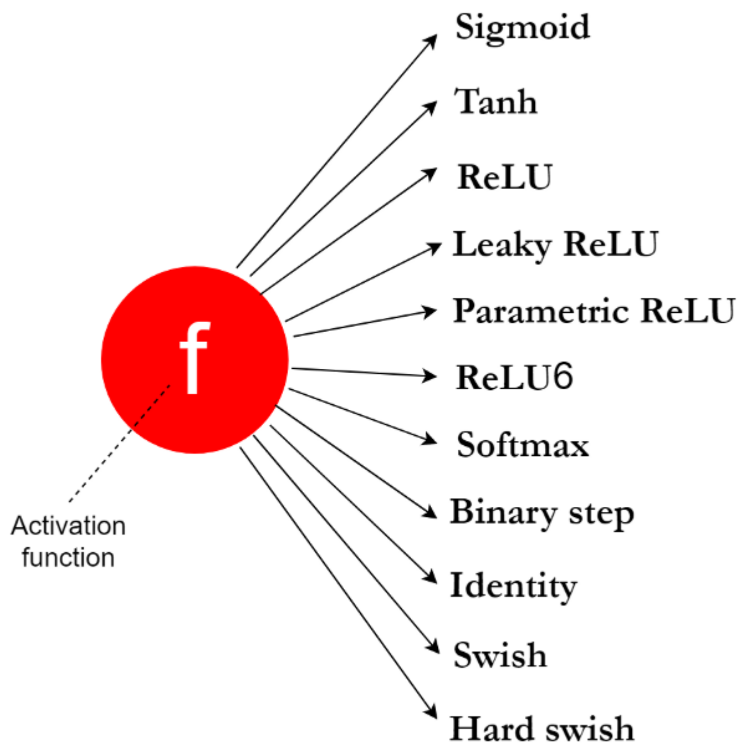
Module-5

Activation functions



Sigmoid

Tanh

ReLU

Leaky ReLU

Parametric ReLU

ReLU6

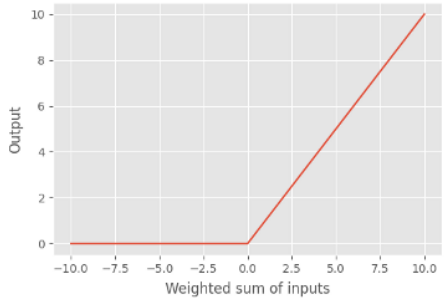
Softmax

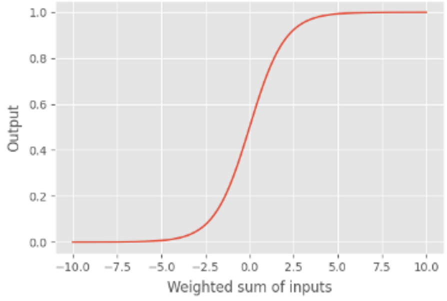
Binary step

Identity

Swish

Hard swish





Activation functions

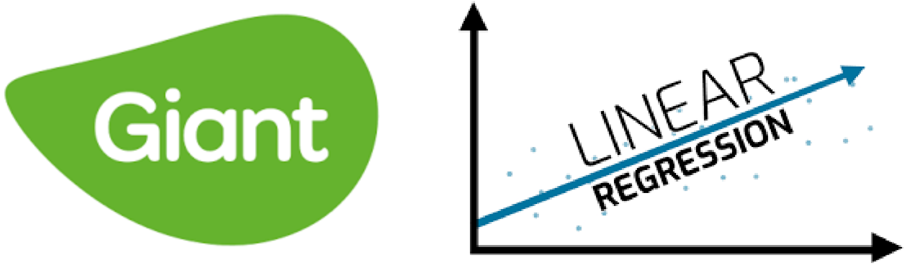
- **An Activation Function** decides whether a neuron should be activated or not.
- This means that it will decide whether the neuron’s input to the network is important or not in the process of prediction using simpler mathematical operations.
- Well, the purpose of an activation function is to add non-linearity to the neural network.

Activation Functions

- *a gatekeeper between layers and it decides which weights and biases (i.e. learning of the model) should propagate further to other layers, significant values are allowed to pass and rest are dropped.*

Activation functions in hidden layers

- This is because we need to introduce non-linearity to the network to learn complex patterns.
- Without non-linear activation functions, a neural network with many hidden layers will act as a



that is useless for learning complex patterns from real-world data.

- The performance of a neural network model will vary significantly depending on the type of activation function we use inside the hidden layers.

Sigmoid

$$y = \frac{1}{1+e^{-x}}$$

Tanh

$$y = \tanh(x)$$

Step Function

$$y = \begin{cases} 0, & x<n \\ 1, & x\geq n \end{cases}$$

Softplus

$$y = \ln(1+e^{x})$$

ReLU

$$y = \begin{cases} 0, & x<0 \\ x, & x\geq 0 \end{cases}$$

Softsign

$$y = \frac{x}{(1+|x|)}$$

ELU

$$y = \begin{cases} \alpha(e^{x}-1) & ,x<0 \\ x & ,x\geq 0 \end{cases}$$

Log of Sigmoid

$$y = \ln\left(\frac{1}{1+e^{-x}}\right)$$

Swish

$$y = \frac{x}{1+e^{-x}}$$

Sinc

$$y = \frac{\sin(x)}{x}$$

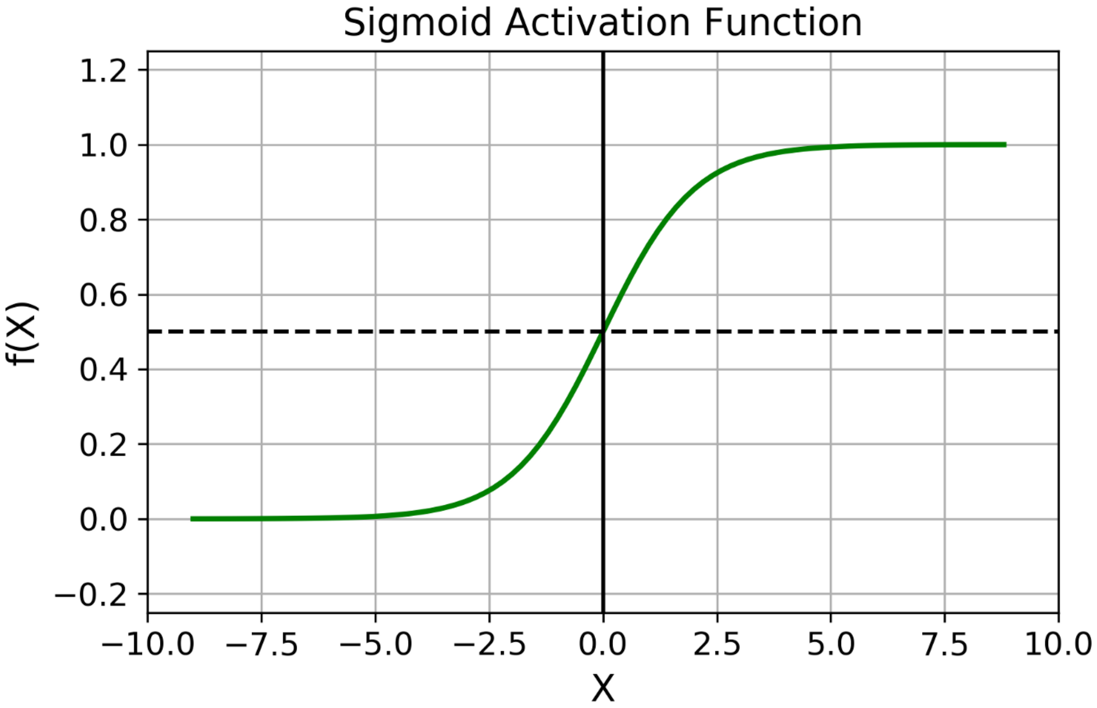
Leaky ReLU

$$y = \max(0.1x, x)$$

Mish

$$y = x(\tanh(\text{softplus}(x)))$$

- **f(z) = 1/(1+ e^-z)**
- The output of a sigmoid function **ranges between 0 and 1**. Since, output values bound between 0 and 1, it **normalizes** the output of each neuron.
- Specially used for models where 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 **perfect** choice.

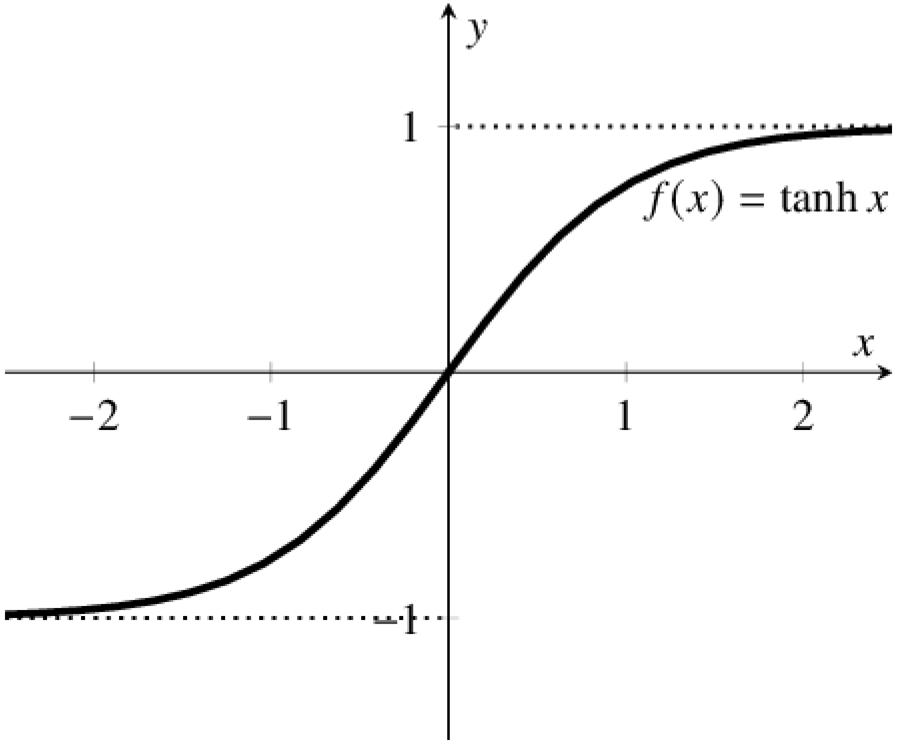


- Prone to gradient vanishing (when the **sigmoid** function value is either too high or too low, the derivative becomes very small i.e. << 1. This causes **vanishing gradients** and poor learning for deep networks.)
- The function output is **not centered on 0,** which will reduce the efficiency of weight update.
- The sigmoid function performs exponential operations, which is slower for computers.

Tanh or Hyperbolic Tangent Activation Function

$$f(x) = tanh(x) = \frac{2}{1+e^{-2x}} - 1$$

- First of all, when the input is large or small, the output is **almost smooth and the gradient is small**, which is not conducive to weight update. The difference is the output interval. The output interval of tanh is 1, and the whole function is **0-centric, which is better than sigmoid.**
- The **major advantage** is that the **negative inputs** will be mapped strongly **negative** and the **zero inputs** will be mapped **near zero** in the tanh graph.

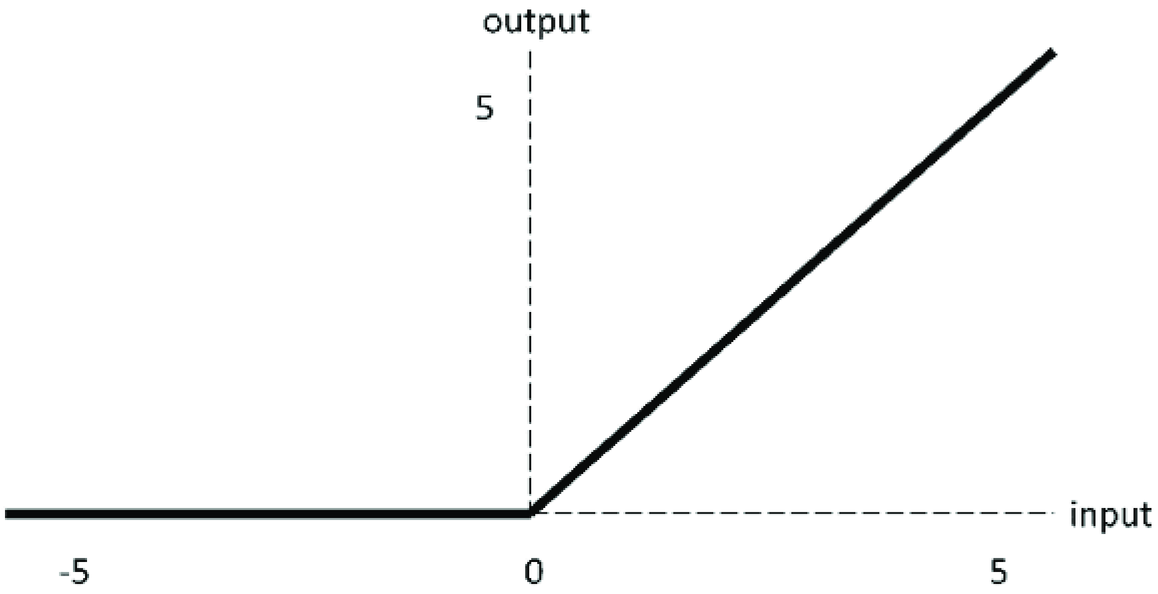


- *In general* ***binary classification problems****, the tanh function is used for the* ***hidden layer*** *and the sigmoid function is used for the* ***output layer****.*
- *However, these are* ***not static****, and the specific activation function to be used must be analyzed according to the specific problem, or it depends.*

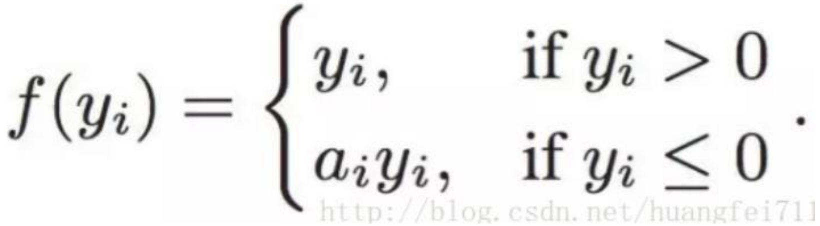
ReLu – Rectified Linear unit

$$\sigma(x) = \begin{cases} max(0, x) & , x >= 0 \\ 0 & , x < 0 \end{cases}$$

- Compared with the sigmoid function and the tanh function, it has the following **advantages**:
- When the input is positive, there is **no gradient saturation problem.**
- The calculation speed is much **faster**. The ReLU function has only a linear relationship. Whether it is forward or backward, it is much faster than sigmoid and tanh. (Sigmoid and tanh need to calculate the exponent, which will be slower.)
- **Range:** [0 to infinity)



Leaky ReLU Activation Function

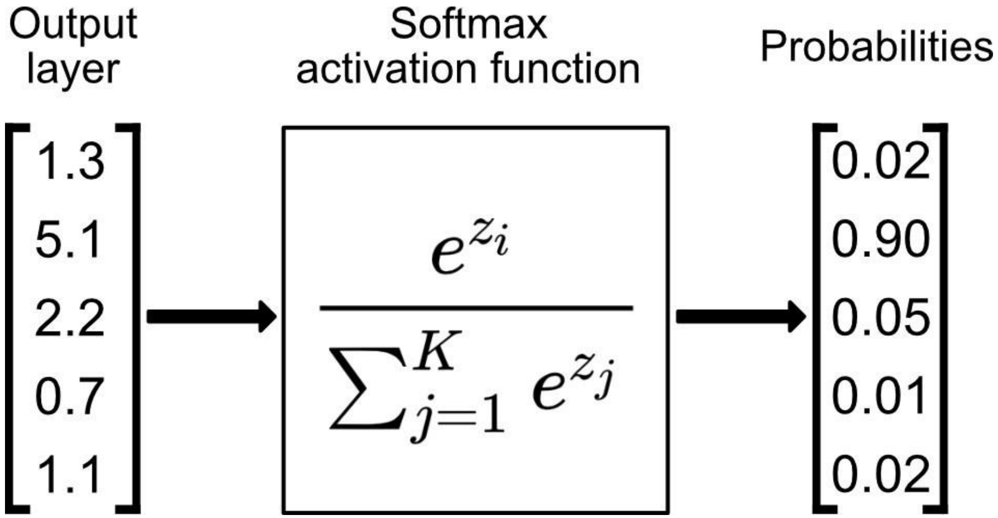


$$f(y_i) = \begin{cases} y_i, & \text{if } y_i > 0 \\ a_i y_i, & \text{if } y_i \le 0 \end{cases}.$$

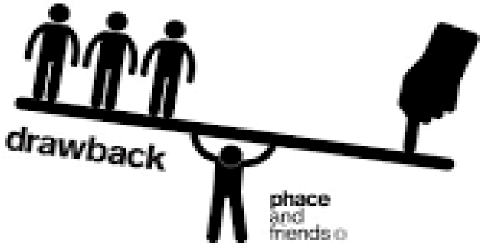
- The leaky ReLU **adjusts the problem of zero gradients** for negative value, by giving a very **small linear component of x** to negative inputs**(0.01x)**.
- The leak helps to increase the range of the ReLU function. Usually, the value of **a** is **0.01** or so.
- Range of the Leaky ReLU is **(-infinity to infinity).**

Softmax

- **Softmax** is used as the **activation function** for multi-class classification problems where class membership is required on more than two class labels.



Output layer
Softmax activation function
Probabilities
1.3
5.1
2.2
0.7
1.1
$\frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$
0.02
0.90
0.05
0.01
0.02



1. Non-differentiable at zero and ReLU is unbounded.

2. The gradients for negative input are zero, which means for activations in that region, the weights are not updated during backpropagation. This can create dead neurons that never get activated.

ACTIVATION FUNCTION	EQUATION	RANGE
Linear Function	$f(x) = x$	$(-\infty, \infty)$
Step Function	$f(x) = \begin{cases} 0 \; for \; x < 0 \\ 1 \; for \; x \geq 0 \end{cases}$	$\{0, 1\}$
Sigmoid Function	$f(x) = \sigma(x) = \dfrac{1}{1 + e^{-x}}$	$(0, 1)$
Hyperbolic Tanjant Function	$f(x) = \tanh(x) = \dfrac{(e^{x} - e^{-x})}{(e^{x} + e^{-x})}$	$(-1, 1)$
ReLU	$f(x) = \begin{cases} 0 \; for \; x < 0 \\ x \; for \; x \geq 0 \end{cases}$	$[0, \infty)$
Leaky ReLU	$f(x) = \begin{cases} 0.01 \; for \; x < 0 \\ \quad x \; for \; x \geq 0 \end{cases}$	$(-\infty, \infty)$
Swish Function	$f(x) = 2x\sigma(\beta x) \; = \begin{cases} \beta = 0 \; for \; f(x) = x \\ \beta \to \infty \; for \; f(x) = 2\max(0, x) \end{cases}$	$(-\infty, \infty)$