



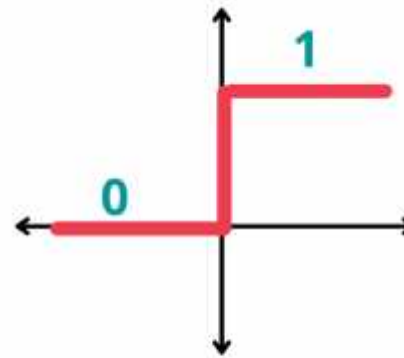
Activation Functions

A Quick Revision

Prasad Deshmukh

Step Function

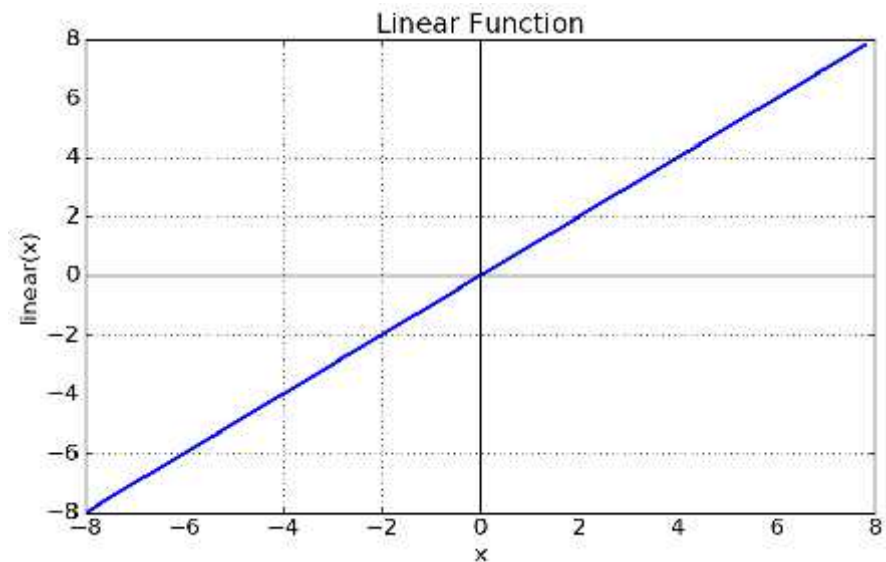
- ▶ **Principle:** The output is binary, either 0 or 1, based on a threshold.
- ▶ **Applications:** Binary classification problems, perceptron models.
- ▶ **Advantages:** Simplicity, easy interpretation.
- ▶ **Limitations:** Not suitable for complex problems, not differentiable.



Binary Step Activation Function

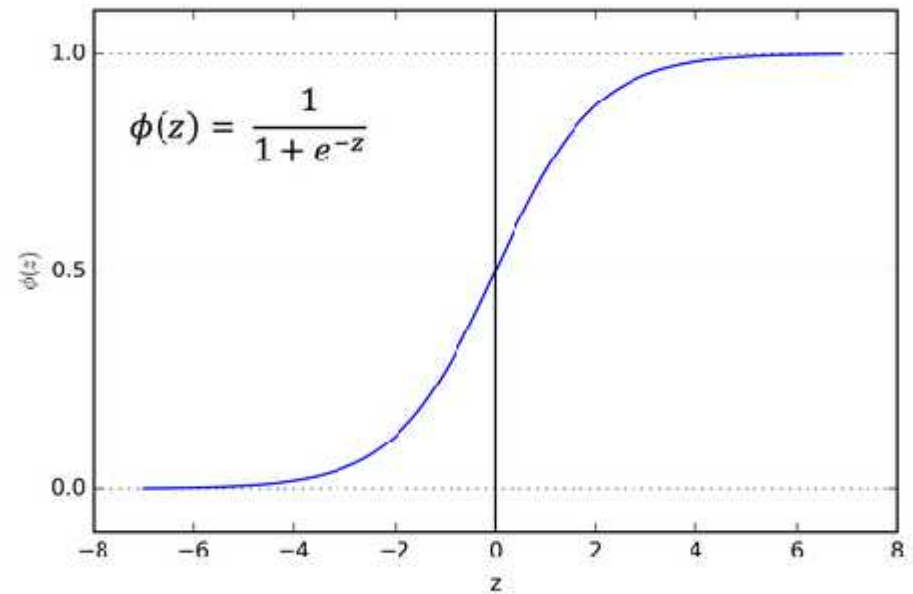
Linear Function

- ▶ **Principle:** The output is proportional to the input, without any non-linearity.
- ▶ **Applications:** Linear regression, simple tasks without non-linear relationships.
- ▶ **Advantages:** Simplicity, computational efficiency.
- ▶ **Limitations:** Unable to capture complex patterns, limited representation power.



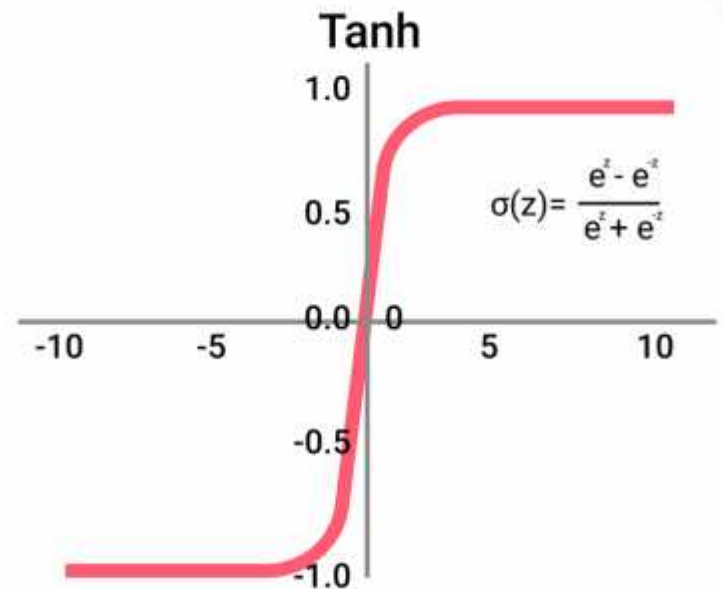
Sigmoid Function (Logistic Function)

- ▶ **Principle:** Smooth transition between 0 and 1 using a logistic function.
- ▶ **Applications:** Binary classification, feedforward neural networks.
- ▶ **Advantages:** Non-linear, differentiable, squashes outputs to a range.
- ▶ **Limitations:** Prone to vanishing gradients, output saturates at the extremes.



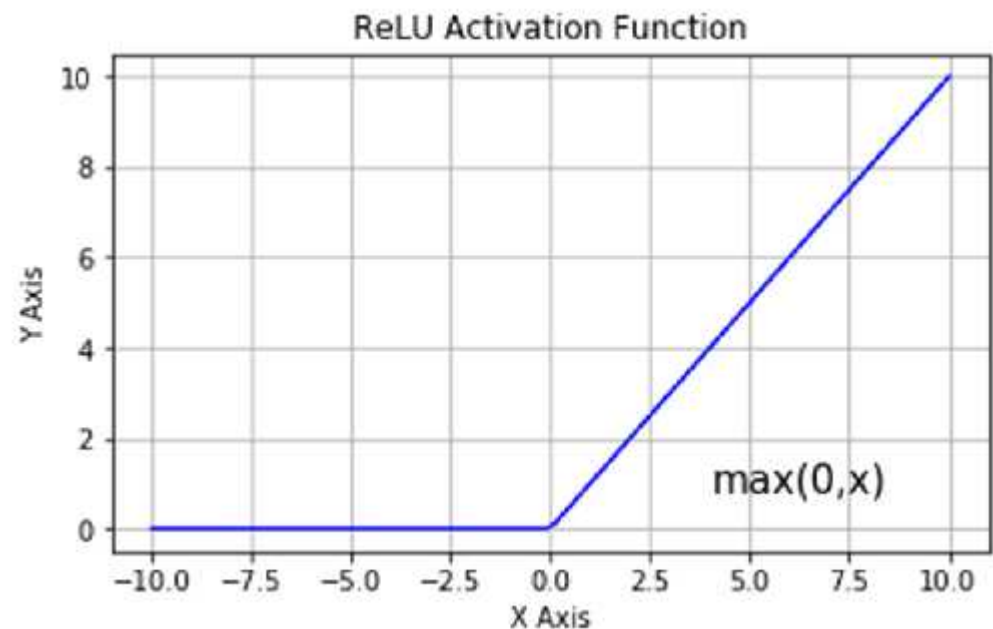
Tanh Function (Hyperbolic Tangent)

- ▶ **Principle:** Similar to the sigmoid function, but output ranges from -1 to 1.
- ▶ **Applications:** Feedforward neural networks, recurrent neural networks.
- ▶ **Advantages:** Non-linear, differentiable, zero-centered outputs.
- ▶ **Limitations:** Prone to vanishing gradients, output saturates at the extremes.



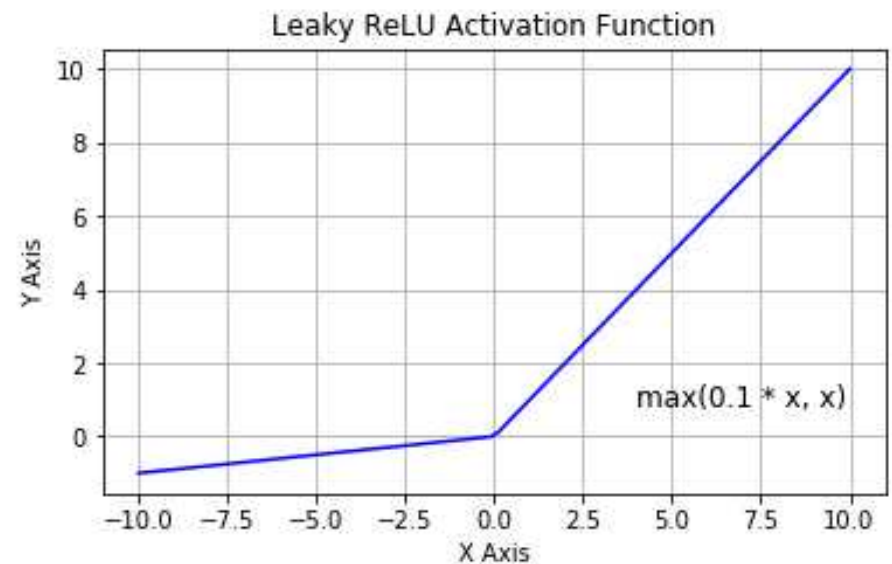
ReLU Function (Rectified Linear Unit)

- ▶ **Principle:** Activation is 0 for negative inputs and linear for positive inputs.
- ▶ **Applications:** Deep learning models, convolutional neural networks.
- ▶ **Advantages:** Avoids vanishing gradients, computationally efficient.
- ▶ **Limitations:** Can lead to dead neurons (dying ReLU problem) during training.



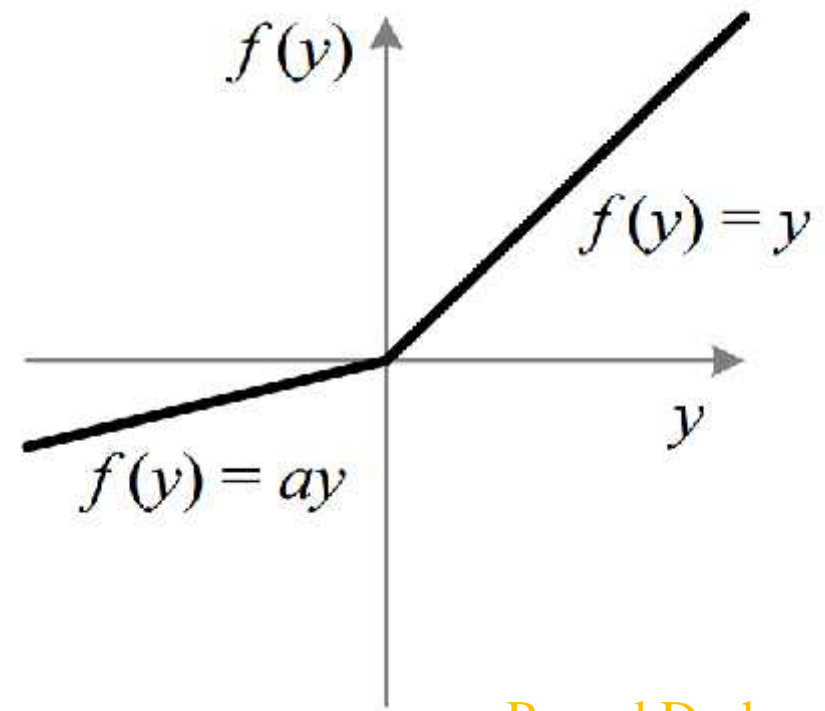
Leaky ReLU Function

- ▶ **Principle:** Similar to ReLU, but with a small slope for negative inputs.
- ▶ **Applications:** Deep learning models, especially when ReLU causes dying neurons.
- ▶ **Advantages:** Avoids dying neurons, computationally efficient.
- ▶ **Limitations:** Not entirely zero-centered, still prone to dead neurons.



Parametric ReLU (PReLU) Function

- ▶ **Principle:** Extension of Leaky ReLU with a learnable parameter for the slope.
- ▶ **Applications:** Deep learning models, improved flexibility over Leaky ReLU.
- ▶ **Advantages:** Avoids dying neurons, allows adaptation to data.
- ▶ **Limitations:** Requires more training parameters, potential overfitting.

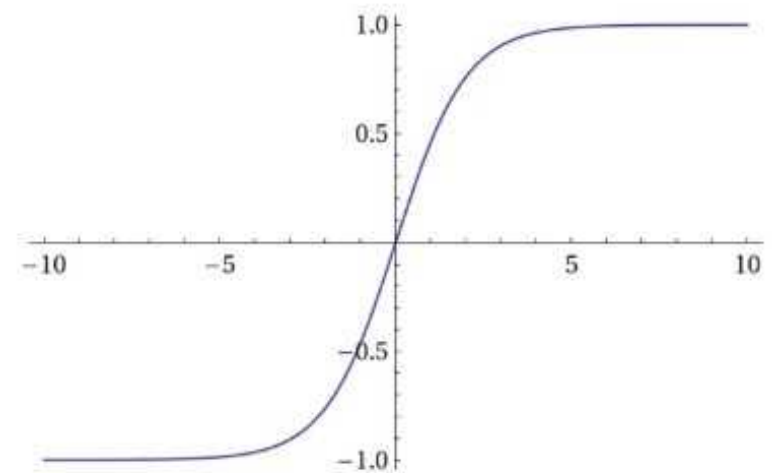


Prasad Deshmukh

Softmax Function

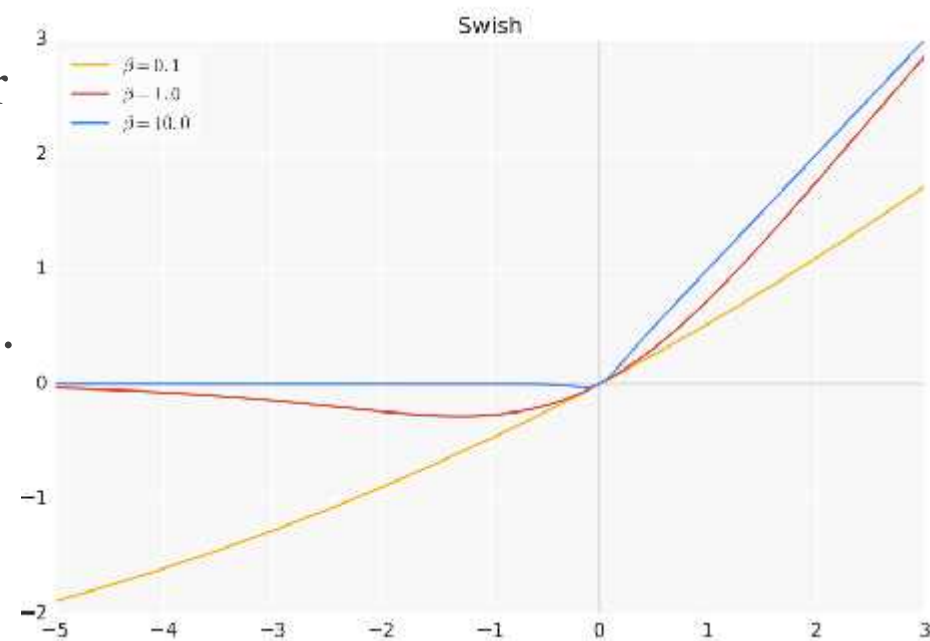
- ▶ **Principle:** Converts a vector of real numbers into a probability distribution.
- ▶ **Applications:** Multiclass classification, output layer in neural networks.
- ▶ **Advantages:** Provides probabilities for mutually exclusive classes.
- ▶ **Limitations:** Sensitive to outliers, not suitable for regression problems.

Softmax Activation Function




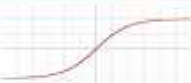






Swish Function

- **Principle:** Non-linear activation function using a combination of sigmoid and linear functions.
- **Applications:** Deep learning models, potentially better performance than ReLU.
- **Advantages:** Non-monotonic, avoids vanishing gradients, smoothness.
- **Limitations:** Computationally more expensive than ReLU.





ACTIVATION FUNCTION	PLOT	EQUATION	DERIVATIVE	RANGE
Linear		$f(x) = x$	$f'(x) = 1$	$(-\infty, \infty)$
Binary Step		$f(x) = \begin{cases} 0 & \text{if } x < 0 \\ 1 & \text{if } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{if } x \neq 0 \\ \text{undefined} & \text{if } x = 0 \end{cases}$	$\{0, 1\}$
Sigmoid		$f(x) = \sigma(x) = \frac{1}{1 + e^{-x}}$	$f'(x) = f(x)(1 - f(x))$	$(0, 1)$
Hyperbolic Tangent(tanh)		$f(x) = \tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$	$f'(x) = 1 - f(x)^2$	$(-1, 1)$
Rectified Linear Unit(ReLU)		$f(x) = \begin{cases} 0 & \text{if } x < 0 \\ x & \text{if } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{if } x < 0 \\ 1 & \text{if } x > 0 \\ \text{undefined} & \text{if } x = 0 \end{cases}$	$[0, \infty)$
Softplus		$f(x) = \ln(1 + e^x)$	$f'(x) = \frac{1}{1 + e^{-x}}$	$(0, 1)$
Leaky ReLU		$f(x) = \begin{cases} 0.01x & \text{if } x < 0 \\ x & \text{if } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0.01 & \text{if } x < 0 \\ 1 & \text{if } x \geq 0 \end{cases}$	$(-1, 1)$
Exponential Linear Unit(ELU)		$f(x) = \begin{cases} \alpha(e^x - 1) & \text{if } x < 0 \\ x & \text{if } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} \alpha e^x & \text{if } x < 0 \\ 1 & \text{if } x > 0 \\ 1 & \text{if } x = 0 \text{ and } \alpha = 1 \end{cases}$	$[0, \infty)$

Prasad Deshmukh



These are some of the commonly used activation functions, each with its own characteristics and suitability for different tasks. It's worth noting that the choice of activation function depends on the specific problem, network architecture, and experimentation to find the most effective option.



THANK YOU

Prasad Deshmukh