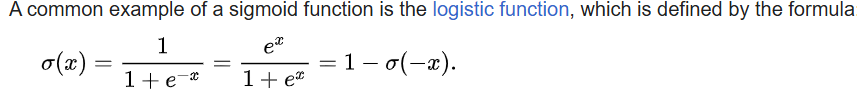
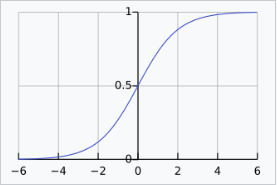
**DIFFERENT TYPES OF ACTIVATION FUNCTION IN DEEP LEARNING**

Activation functions are crucial in deep learning as they introduce non-linearity into the model, enabling it to learn complex patterns. Here are some common types of activation functions:

**1. Sigmoid**

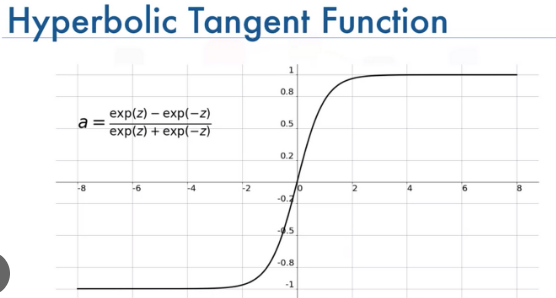
The sigmoid function maps input values to a range between 0 and 1. It's often used in binary classification problems.





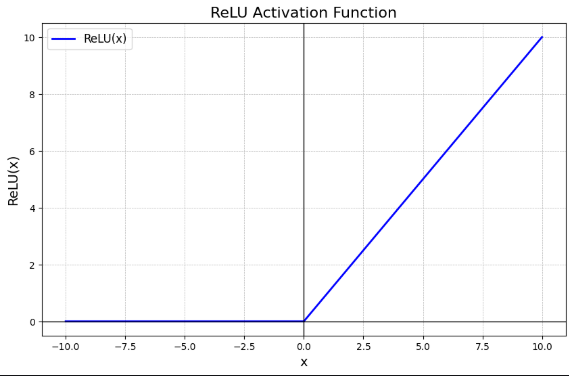
**2. Hyperbolic Tangent (Tanh)**

The tanh function maps input values to a range between -1 and 1. It's zero-centered, which often helps with faster convergence.



**3. Rectified Linear Unit (ReLU)**

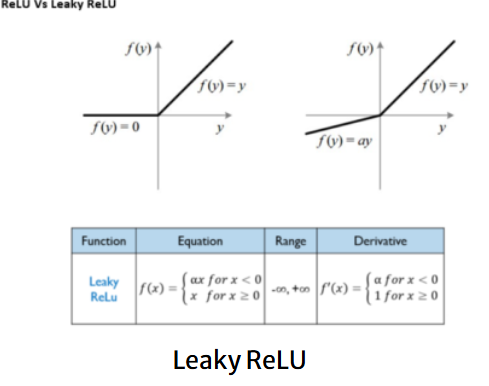
The ReLU function is widely used due to its simplicity and effectiveness. It outputs the input directly if it is positive; otherwise, it outputs zero.



**4. Leaky ReLU**

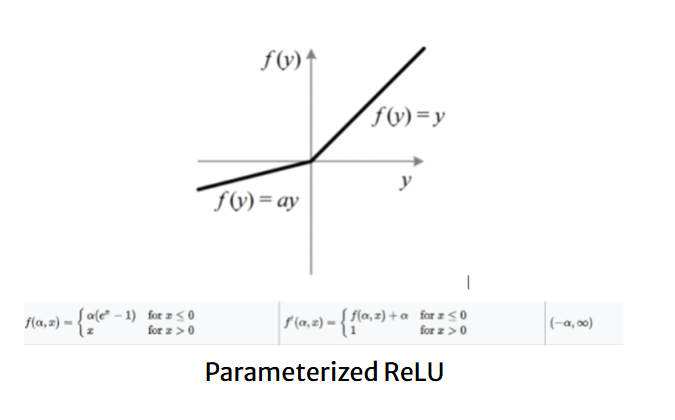
Leaky ReLU function is nothing but an improved version of the ReLU function, As we saw that for the ReLU function, the gradient is 0 for x<0, which would deactivate the neurons in that region.

Leaky ReLU is defined to address this problem, instead of defining the ReLU function as 0 for negative values of x, we define it as an extremely small linear component of x.



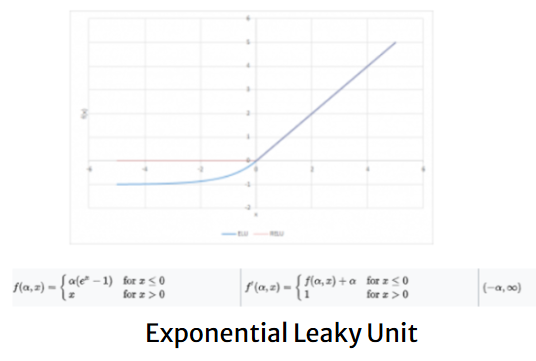
**5. Parametric ReLU (PReLU)**

This is another variant of ReLU that aims to solve the problem of gradient’s becoming zero for the left half of the axis. The parameterized ReLU, as the name suggests, introduces a new parameter as a slope of the negative part of the function.

****

**6. Exponential Linear Unit (ELU)**

The ELU function tends to converge faster and produce more accurate results as it reduces the bias shift. $$ \text{ELU}(x) = \begin{cases} x & \text{if } x > 0 \\ \alpha (e^x - 1) & \text{if } x \leq 0 \end{cases}



**7. Swish**

Swish is a newer activation function that often performs better than ReLU on deep networks. $$ \text{Swish}(x) = x \cdot \sigma(x) = x \cdot \frac{1}{1 + e^{-x}} $$

**8. Softmax**

Softmax is used in the output layer of a neural network for multi-class classification problems. It converts the logits into probabilities. $$ \text{Softmax}(x\_i) = \frac{e^{x\_i}}{\sum\_{j} e^{x\_j}} $$

**9. Gelu**

Gaussian Error Linear Unit (GELU) is used in BERT and many other transformer models. It is defined as: $$ \text{GELU}(x) = x \cdot \Phi(x) $$ where Φ(x)\Phi(x) is the cumulative distribution function of the standard normal distribution.

These activation functions each have their own strengths and are chosen based on the specific requirements of the neural network being designed. Let me know if you want to dive deeper into any of these functions!

| **Activation Function** | **Formula** | **Output Range** | **Advantages** | **Disadvantages** | **Use Case** |
| --- | --- | --- | --- | --- | --- |
| **ReLU** | f(x)=max⁡(0,x)*f*(*x*)=max(0,*x*) | [0,∞)[0,∞) | - Simple and computationally efficient | - Dying ReLU problem (neurons stop learning) | Hidden layers of deep networks |
|  |  |  | - Helps mitigate vanishing gradient problem | - Unbounded positive output |  |
|  |  |  | - Sparse activation (efficient computation) |  |  |
|  |  |  |  |  |  |
| **Leaky ReLU** | f(x)={xx>0αxx≤0*f*(*x*)={*xαx*​*x*>0*x*≤0​ | (−∞,∞)(−∞,∞) | - Solves the dying ReLU problem | The slope α*α* needs to be predefined | Hidden layers, as an alternative to ReLU |
| **Parametric ReLU (PReLU)** | Same as Leaky ReLU, but  α*α*  is learned | (−∞,∞)(−∞,∞) | - Learns the slope for negative values | - Risk of overfitting with too much flexibility | Deep networks where ReLU fails |
| **Sigmoid** | f(x)=11+e−x*f*(*x*)=1+*e*−*x*1​ | (0, 1) | - Useful for binary classification | - Vanishing gradient problem | Output layers for binary classification |
|  |  |  | - Smooth gradient | - Outputs not zero-centered |  |
| **Tanh** | f(x)=ex−e−xex+e−x*f*(*x*)=*ex*+*e*−*xex*−*e*−*x*​ | (-1, 1) | - Zero-centered output, better than sigmoid | - Still suffers from vanishing gradients | Hidden layers, when data needs to be zero-centered |