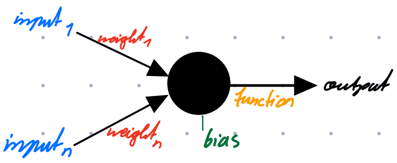
# Single Neuron → is just a function




→ Output = function($w_1 \cdot i_1 + \ldots + w_n \cdot i_n + b$)

**weights** → how much influence a specific input has

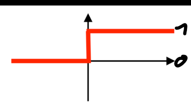
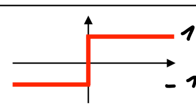
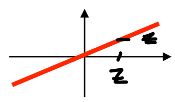
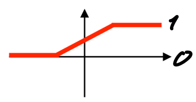
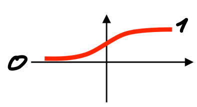
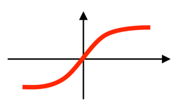
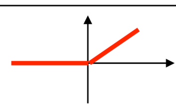
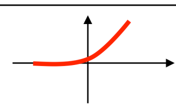
**bias** → how much influence the neuron has
↳ positive bias: small input → big output
↳ negative bias, vice versa → neuron less important / influence

**activation functions** → allow to capture and model complex non-linear relationships within inputs
↳ enabling to learn & generalize from complex data

→ any growing function is possible

→ but some non-linear are necessary or neural net is just an linear regression

examples:

| Activation function | Equation | Example | 1D Graph | |
|---|---|---|---|---|
| Unit step (Heaviside) | $\phi(z) = \begin{cases} 0, & z < 0, \\ 0.5, & z = 0, \\ 1, & z > 0, \end{cases}$ | Perceptron variant |   | converts negative to 0<br>converts positive to 1 |
| Sign (Signum) | $\phi(z) = \begin{cases} -1, & z < 0, \\ 0, & z = 0, \\ 1, & z > 0, \end{cases}$ | Perceptron variant |   | converts negative → set negative<br>" positive → set positive |
| Linear | $\phi(z) = z$ | Adaline, linear regression |   | Linear |
| Piece-wise linear | $\phi(z) = \begin{cases} 1, & z \ge \frac{1}{2}, \\ z + \frac{1}{2}, & -\frac{1}{2} < z < \frac{1}{2}, \\ 0, & z \le -\frac{1}{2}, \end{cases}$ | Support vector machine |   | Signum with linear transition |
| Logistic (sigmoid) | $\phi(z) = \frac{1}{1 + e^{-z}}$ | Logistic regression, Multi-layer NN |   | Signum smooth transition between 0 and 1 |
| Hyperbolic tangent | $\phi(z) = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}}$ | Multi-layer Neural Networks |  | Sigmoid w/o boundries |
| Rectifier, ReLU (Rectified Linear Unit) | $\phi(z) = max(0, z)$ | Multi-layer Neural Networks |  | sets negative input to zero |
| Rectifier, softplus | $\phi(z) = \ln(1 + e^{z})$ | Multi-layer Neural Networks |  | |



converts numbers into probabilities
↳ numbers between 0-1