Explanatory Notes for 6.390

Shaunticlair Ruiz (Current TA)

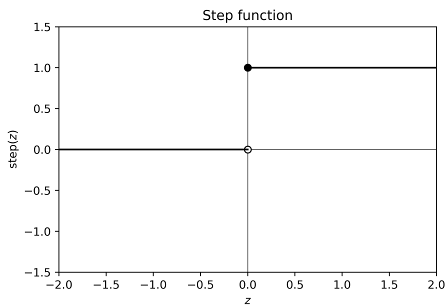
Fall 2022

Example of Activation Functions

So, let's look at some possible **activation** functions:

- **Step** function step(z):

$$\text{step}(z) = \begin{cases} 1 & \text{if } z \geqslant 0 \\ 0 & \text{if } z < 0 \end{cases} \tag{1}$$



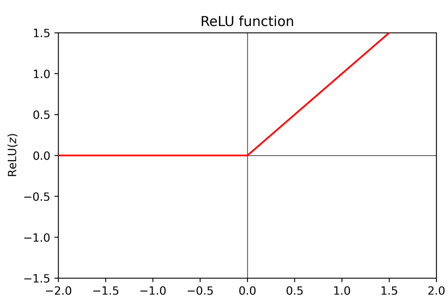
  – This function is basically a **sign** function, but uses $\{0, 1\}$ instead of $\{-1, +1\}$.

  – Step functions were a common early choice, but because they have a **zero** gradient, we can't use **gradient descent**, and so we basically **never** use them.

Same reason we replaced the sign function with sigmoid.

- **Rectified Linear Unit** ReLU(z):

$$\text{ReLU}(z) = \max(0, z) = \begin{cases} z & \text{if } z \geqslant 0 \\ 0 & \text{if } z < 0 \end{cases} \tag{2}$$



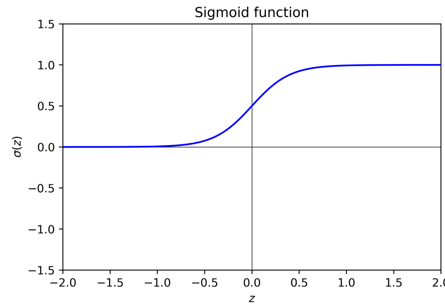
  – This is a very **common** choice for activation function, even though the derivative is undefined at 0.

  – We specifically use it for internal ("**hidden**") layers: layers that are neither the **first** nor **last** layer.

They're "hidden" because they aren't visible to the input or output.

- **Sigmoid** function $\sigma(z)$:

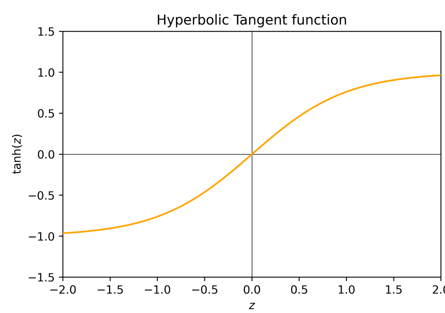
$$\sigma(z) = \frac{1}{1+e^{-z}} \tag{3}$$



- This is the **activation** function for our **LLC** neuron from before.
- Just like it was then, it's useful for the **output neuron** in **binary classification**.
- Can be interpreted as the **probability** of a positive (+1) binary classification.

• **Hyperbolic Tangent** $\tanh(z)$:

$$\tanh(z) = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}} \tag{4}$$



- This is function looks similar to sigmoid over a different **range**.
- Unfortunately, it will not get much use in this class.

• **Softmax** function $\text{softmax}(z)$:

$$\text{softmax}(z) = \begin{bmatrix} \exp(z_1)/\sum_i \exp(z_i) \\ \vdots \\ \exp(z_n)/\sum_i \exp(z_i) \end{bmatrix} \tag{5}$$

- Behaves a like a **multi-class** version of **sigmoid**.
- Appropriately, we use it as the **output neuron** for **multi-class** classification.

- Can be interpreted as the **probability** of our k possible classifications.

Concept 1

For the different **activation functions**:

- $\text{sign}(z)$ is **rarely** used.
- $\text{ReLU}(z)$ is often used for "**hidden**" layers.
- $\sigma(z)$ is often used as the **output** for **binary classification**.
- $\text{softmax}(z)$ is often used as the **output** for **multi-class classification**.

$\tanh(z)$ is useful, but not a focus of this class.

Loss functions and activation functions

As we can see above, your **activation** function depends on what kind of **problem** you're dealing with.

The same is true for our **loss** function: we used **different** loss functions for classification and regression.

Classification can be further broken up into **binary** versus **multiclass** classification.

To summarize our findings, we'll **sort** this information:

Concept 2

Each of our **tasks** requires a different **loss** and output **activation** function.

We emphasize that we specifically mean the output activation function: the activation function used in **hidden layers** doesn't have to match the loss function.

task	f^L		Loss	
Regression	Linear	z	Squared	$(g-y)^2$
Binary Class	Sigmoid	$\sigma(z)$	NLL	$y \log g + (1-y)\log(1-g)$
Multi-Class	Softmax	$\text{softmax}(z)$	NLLM	$\sum_j y_j \log(g_j)$