# Metrics

Discover the most commonly used metrics for evaluating a neural network.

Chapter Goals:

- Convert logits to probabilities
- Obtain model predictions from probabilities
- Calculate prediction accuracy

A. Sigmoid

As discussed in the previous chapter, our model outputs logits based on the input data. These logits represent real number mappings from probabilities. Therefore, if we had the inverse mapping we could obtain the original probabilities. Luckily, we have exactly that, in the form of the sigmoid function.
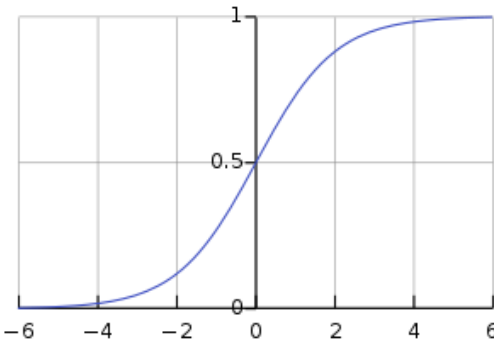
The above plot shows a sigmoid function graph. The x-axis represents logits, while the y-axis represents probabilities.

> **Note** the horizontal asymptotes at $y = 0$ and $y = 1$.

Using the sigmoid function (`tf.nn.sigmoid` in TensorFlow), we can extract probabilities from the logits. In binary classification, i.e. `output_size = 1`, this represents the probability the model gives to the input data

being labeled 1. A probability closer to 0 or 1 means the model is pretty sure in its prediction, while a probability closer to 0.5 means the model is unsure (0.5 is equivalent to a random guess of the label's value).

B. Predictions and accuracy

A probability closer to 1 means the model is more sure that the label is 1, while a probability closer to 0 means the model is more sure that the label is 0. Therefore, we can obtain model predictions just by rounding each probability to the nearest integer, which would be 0 or 1. Then our prediction accuracy would be the number of correct predictions divided by the number of labels.

In TensorFlow, there is a function called `tf.reduce_mean` which produces the overall mean of a tensor's numeric values. We can use this to calculate prediction accuracy as the mean number of correct predictions across all input data points.

We use these metrics to evaluate how good our model is both during and after training. This way we can experiment with different computation graphs, such as different numbers of neurons or layers, and find which structure works best for our model.