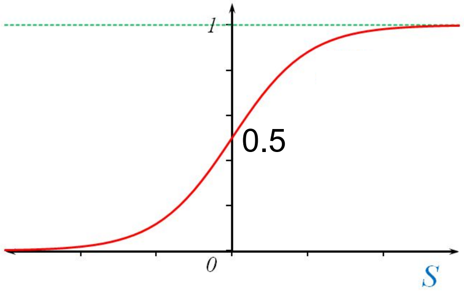
Logistic regression with TensorFlow:

- The linear combinations of variables X_i is the so-called "Logit" denoted here as S :

$$S = b + w_1 X_1 + w_2 X_2 + \cdots + w_k X_k$$

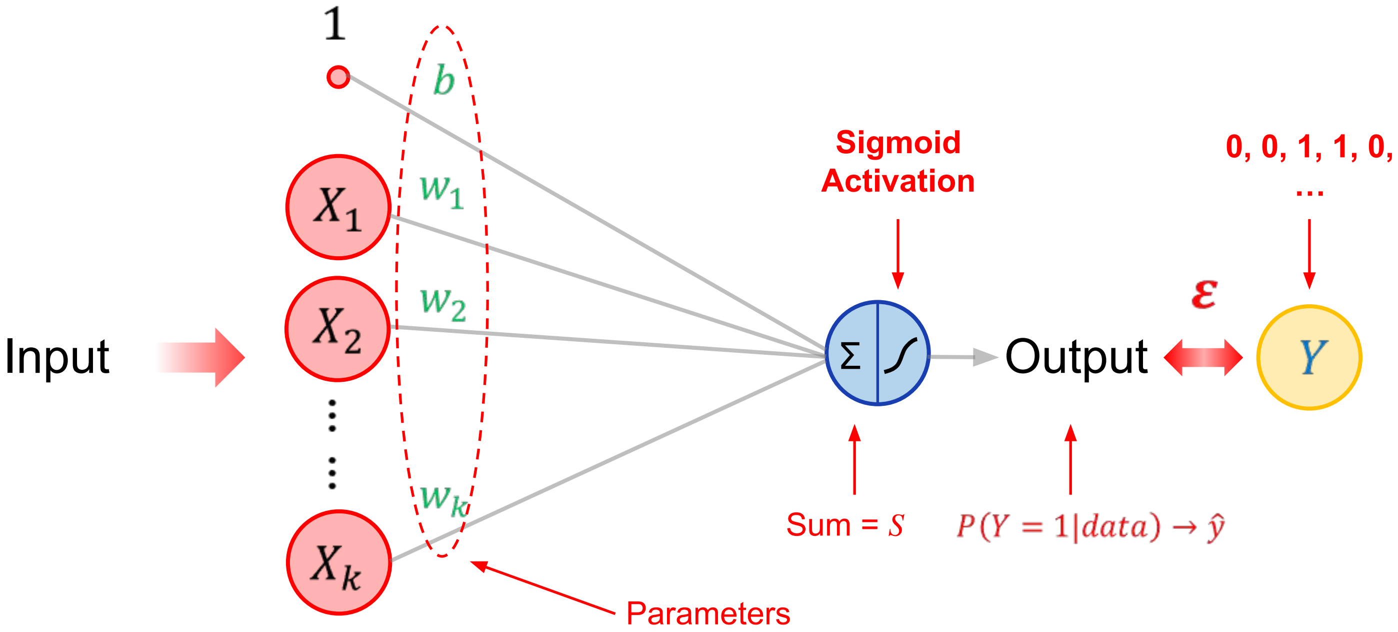
- The conditional probability of Y being equal to 1 is denoted as $P(Y{=}1|\{x_i\})$.
- A "Sigmoid" function connects the probability with the logit:

$$\sigma(S) = \frac{e^{S}}{1 + e^{S}}$$



- This Sigmoid is the "activation function" ⇐ the biggest difference with the linear regression.

Logistic regression with TensorFlow:



- The mismatch between the predicted $\hat{y}$ and the true y has to be minimized by training.

Logistic regression with TensorFlow:

- We can recall the negative of logarithmic likelihood with the redefined $y_i = -1 \text{ or } +1$. (*)

$$-\sum_{i=1}^{n} Log(1 + e^{-y_i\, S})$$

- We now rename this as the "entropy" and express it in terms of the predicted probability $\widehat{p}_i$.

$$L = -\frac{1}{n}\sum_{i=1}^{n}(y_i log(\widehat{p}_i) + (1 - y_i)\, log(1 - \widehat{p}_i))$$

  - We went back to the definition $y_i = 0 \text{ or } 1$.
  - This is the loss function we'd like to minimize by the gradient descent algorithm.

- Binary logistic regression can be generalized to the **multi-class version** using the Softmax activation and multi-class cross entropy as the loss function.