There could be better constants than 0.01 (perhaps for larger NNs?)

$W^{[2]} = np.random.randn((1,2)) * 0.01$

$b^{[2]} = 0$

---

## Week 3 quiz notes

- $a^{[l](m)}_{i}$ → layer; → train. example; → node

- $X = \begin{bmatrix} \vdots & \vdots & & \vdots \\ x^{(1)} & x^{(2)} & \cdots & x^{(m)} \\ \vdots & \vdots & & \vdots \end{bmatrix}$

- tanh works better than σ for hidden units, because the mean of its output ≈0, and so it centers the data better for the next layer

- Vectorised for. prog. for layer $l$, where $1 \leq l \leq L$:

$$Z^{[l]} = W^{[l]} A^{[l-1]} + b^{[l]}$$

$$A^{[l]} = \sigma(Z^{[l]}) \qquad g(z) = \sigma(z)$$

- Cucumbers ($y=1$) v. Watermelons ($y=0$), which act. func. for output layer?
  ↳ SIGMOID.
  The output value from a sigmoid can be easily understood as a probability. Sigmoid outputs 0≤values≤1, so it can be classified as 0 if <0.5 and 1 if >0.5. tanh (−1≤values≤1) is less convenient

- $A = \begin{bmatrix} \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \end{bmatrix}$, $B = [\cdot\ \cdot\ \cdot]$ | $B = np.sum(A, axis=1, keepdims=True)$

  =0 : vert. (y)
  =1 : hor. (x)