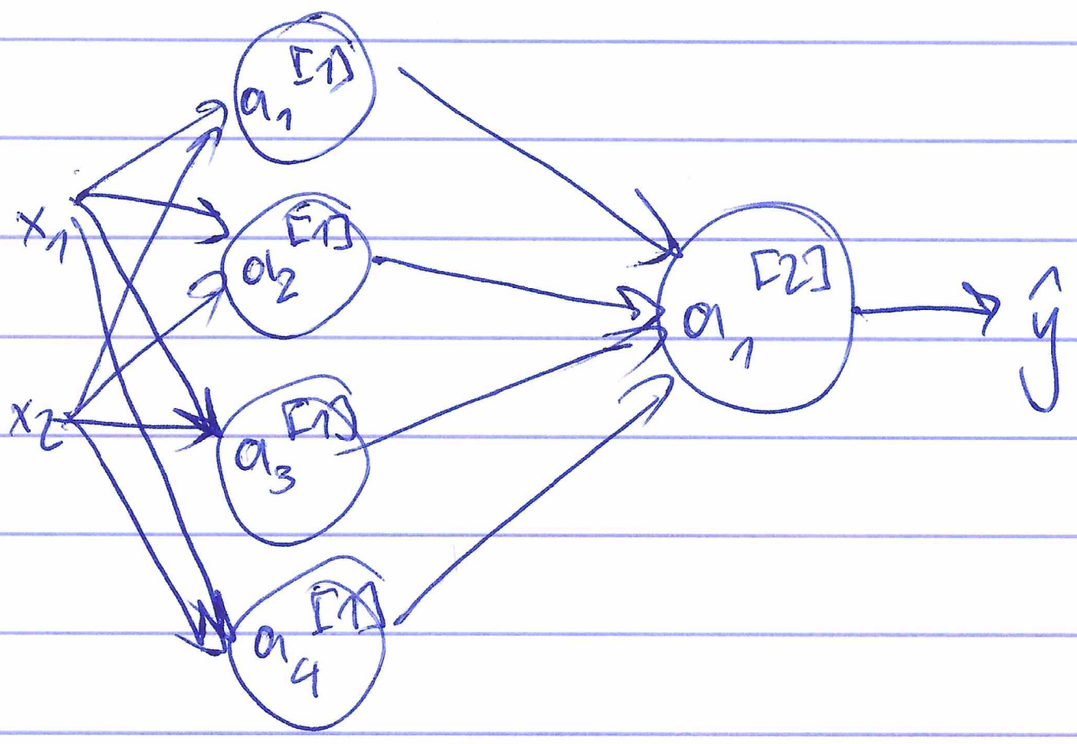
- NN; init weights & biases = 0:
  - Each neuron in the 1st h.l. will perform the same computation. Thus, even after multiple GD iterations, each neuron will be computing the same thing as other neurons

- Log. reg doesn't have a hidden layer. If you init. the weights to 0, the first ex. x fed will output 0 but the derivatives of the log. reg depend on input x (because no hidden layer), and $x \neq 0$. Thus, at the 2nd iter., the weights follow x's distribution and are different from each other if x isn't a const. vector. So the log. reg wasn't fail to "break symmetry" if we init to all 0s.

- NN w/ tanh; init. w's using eg. random.randn(...,
  - Inputs of tanh become very large, so gradients ≈ 0,
  - ∴ slow optimization



$W^{[l]}$ is $(n^{[l]}, n^{[l-1]})$
$b^{[l]}$ is $(n^{[l]}, 1)$

$W^{[1]}$ has shape	$(n^{[1]}, n^{[0]})$	or $(4,2)$	$Z^{[1]}$	$(4, m)$
$b^{[1]}$	$(n^{[1]}, 1)$	$(4,1)$	$A^{[1]}$	$(4, m)$
$W^{[2]}$	$(n^{[2]}, n^{[1]})$	$(1,4)$		
$b^{[2]}$	$(n^{[2]}, 1)$	$(1,1)$		

↑ h.n. ↑ t.e.