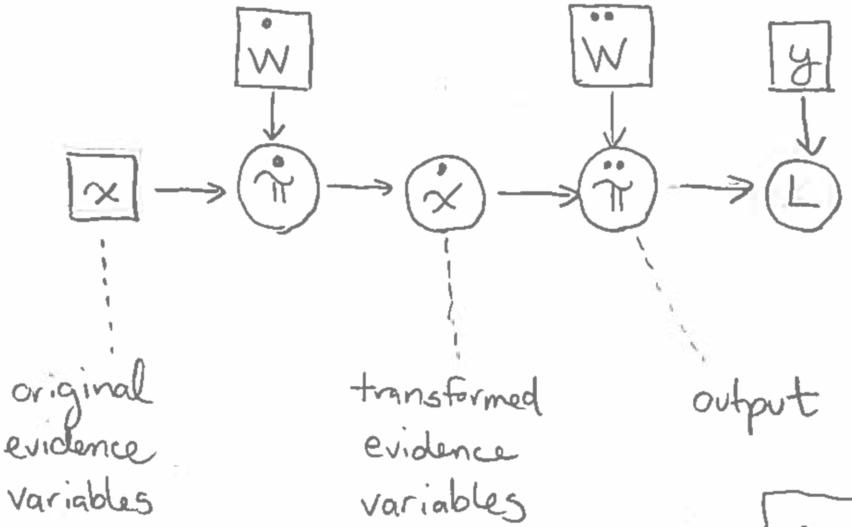
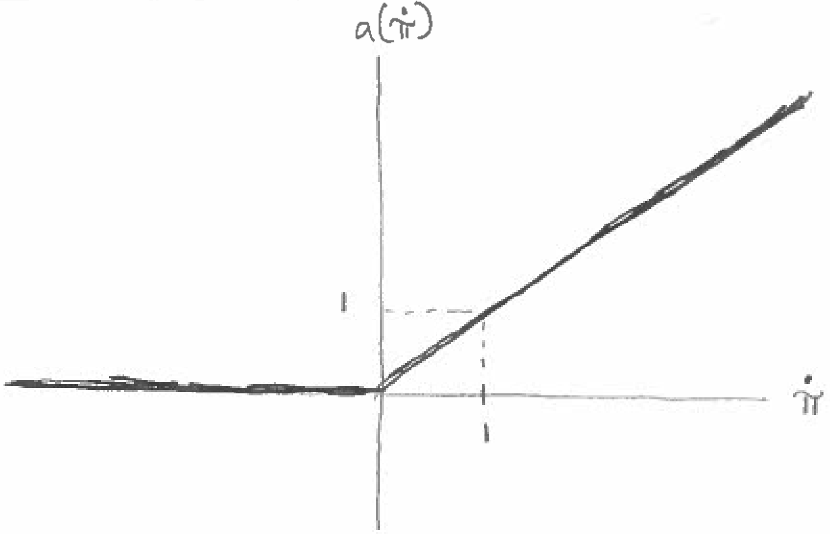
ACTIVATION FUNCTIONS

① Let's briefly consider the role of the ReLU function in a two-layer neural network:



$$\boxed{\begin{aligned}\dot{\pi} &= \dot{W}^T x \\ \dot{x} &= a(\dot{\pi}) \\ \ddot{\pi} &= \ddot{W}^T \dot{x}\end{aligned}}$$

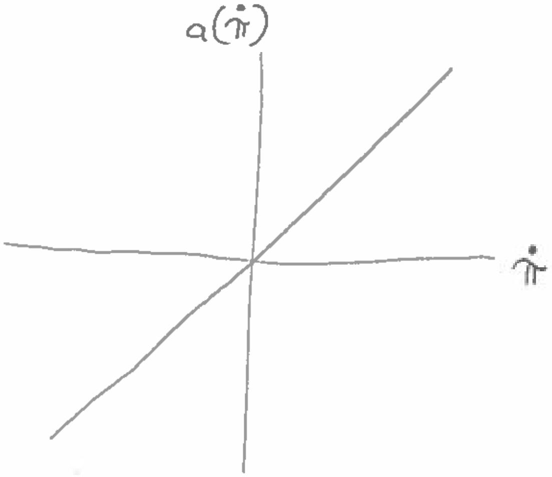
② Recall that we previously used the ReLU as our <u>activation function</u> a:



ACTIVATION FUNCTIONS

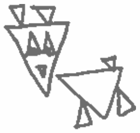
3 By applying the activation function to linear combinations of our original evidence variables (height and mass), we obtained transformed evidence variables (overweight and underweight) that were amenable to logistic regression.

4 But what if we hadn't applied ReLU? Or, equivalently, what if we applied a linear activation function $a(\dot{\pi}) = \dot{\pi}$?



5 Then our output would be:

$$\begin{aligned}
\ddot{\pi} &= \ddot{W}^T \dot{x} \\
&= \ddot{W}^T a(\dot{\pi}) \\
&= \ddot{W}^T a(\dot{W}^T x) \\
&= \ddot{W}^T (\dot{W}^T x) \\
&= (\ddot{W}^T \dot{W}) x \\
&= W' x
\end{aligned}$$

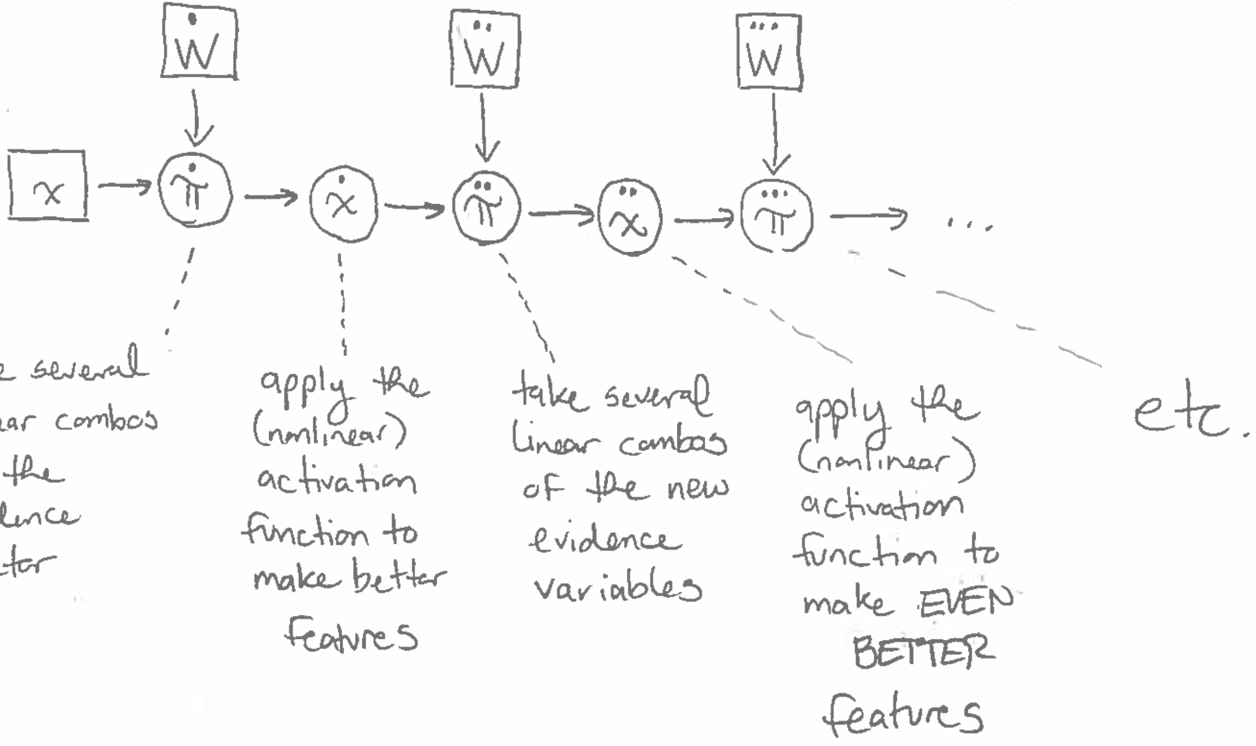


it's just a linear combination of the original evidence variables!

ACTIVATION FUNCTIONS

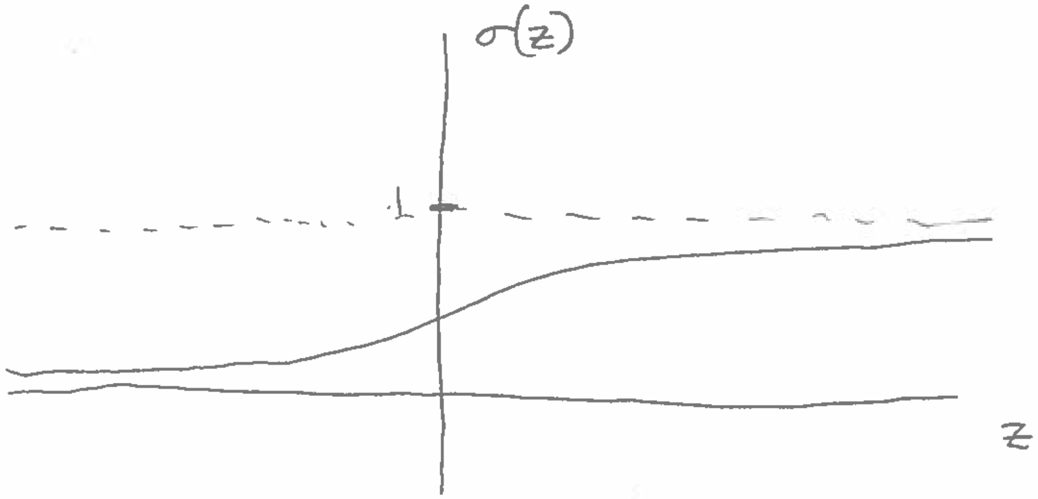
⑥ So we haven't gained any advantage over plain regression by adding layers. It's only because of applying the nonlinear ReLU activation function that we had the ability to create "logistic regression friendly" evidence variables like "underweight" and "overweight".

⑦ Thus, the key to "feature discovery" in neural networks lies in the activation function:

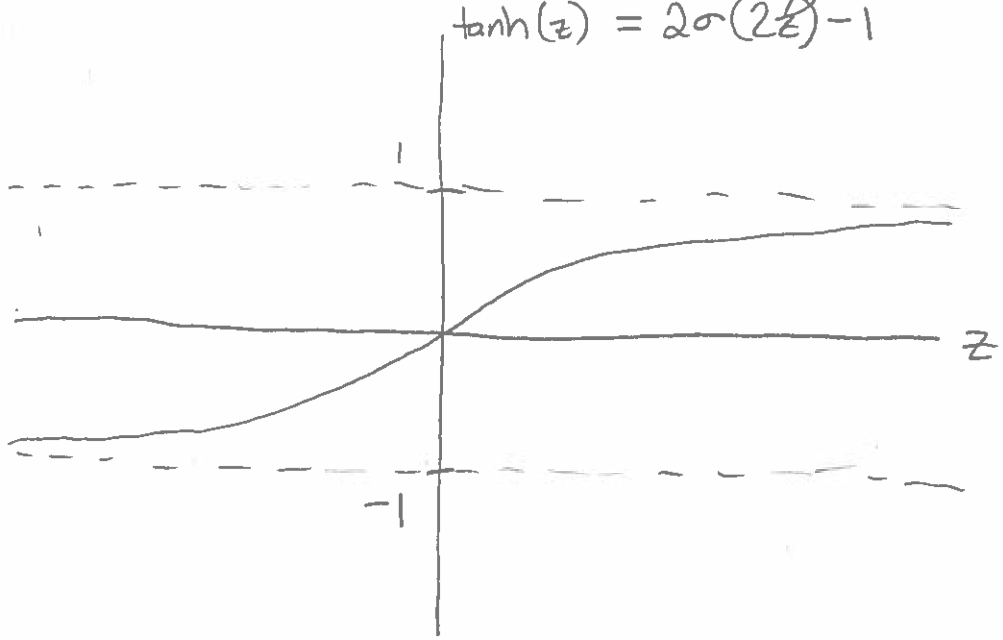

$\dot{W}$
$\ddot{W}$
$\dddot{W}$
x
$\dot{\pi}$
$\dot{x}$
$\ddot{\pi}$
$\ddot{x}$
$\dddot{\pi}$
...
take several linear combos of the evidence vector
apply the (nonlinear) activation function to make better features
take several linear combos of the new evidence variables
apply the (nonlinear) activation function to make EVEN BETTER features
etc.

ACTIVATION FUNCTIONS

8 Traditionally, the most popular activation functions were the logistic function:



and the (quite similar) hyperbolic tangent (tanh) function



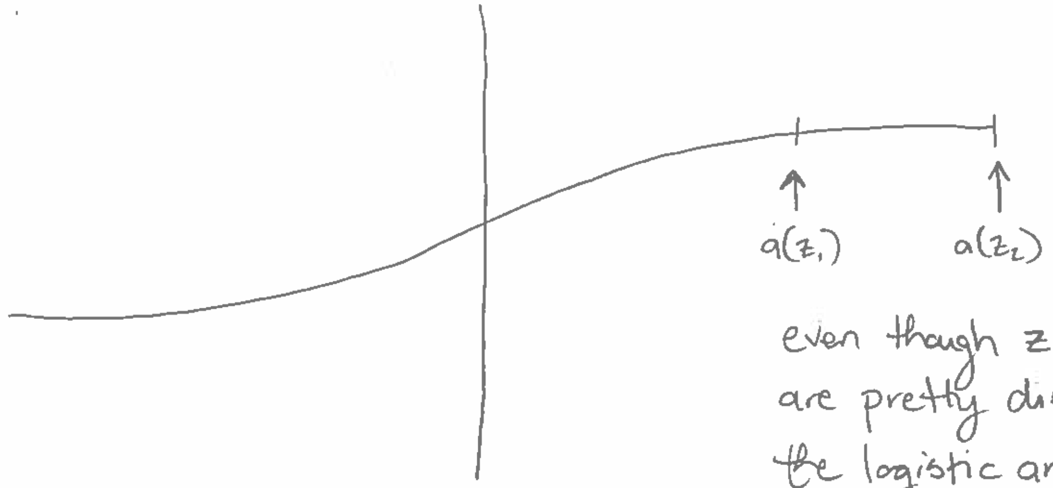
Note that the tanh function is just a rescaled and shifted logistic function:

$$\tanh(z) = 2\sigma(2z) - 1$$

ACTIVATION FUNCTIONS

9) Despite the fact that $\sigma(z)$ and $\tanh(z)$ are both differentiable (while ReLU is only piecewise differentiable), it was discovered in the early 2000s that ReLU tends to produce much better trained neural networks.

This behavior has been attributed to the fact that $\sigma(z)$ and $\tanh(z)$ both saturate, i.e.



even though z_1 and z_2 are pretty different, the logistic and tanh functions map them to nearly the same value

By contrast, ReLU only saturates in one direction (as $z \to -\infty$), leaving plenty of room to distinguish between values in the positive direction.