دوره‌ی آموزشی «علم داده»
Data Science Course

جلسه بیست و هشتم

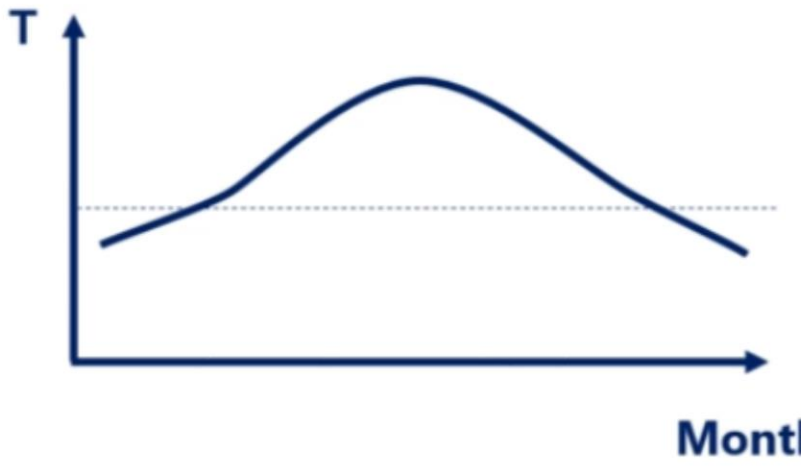
مقدمه‌ای بر شبکه‌های عصبی عمیق (همراه با معرفی کتاب)

Deep Neural Networks

مدرس: محمد فزونی
عضو هیئت علمی دانشگاه گنبدکاووس

Real life is not linear

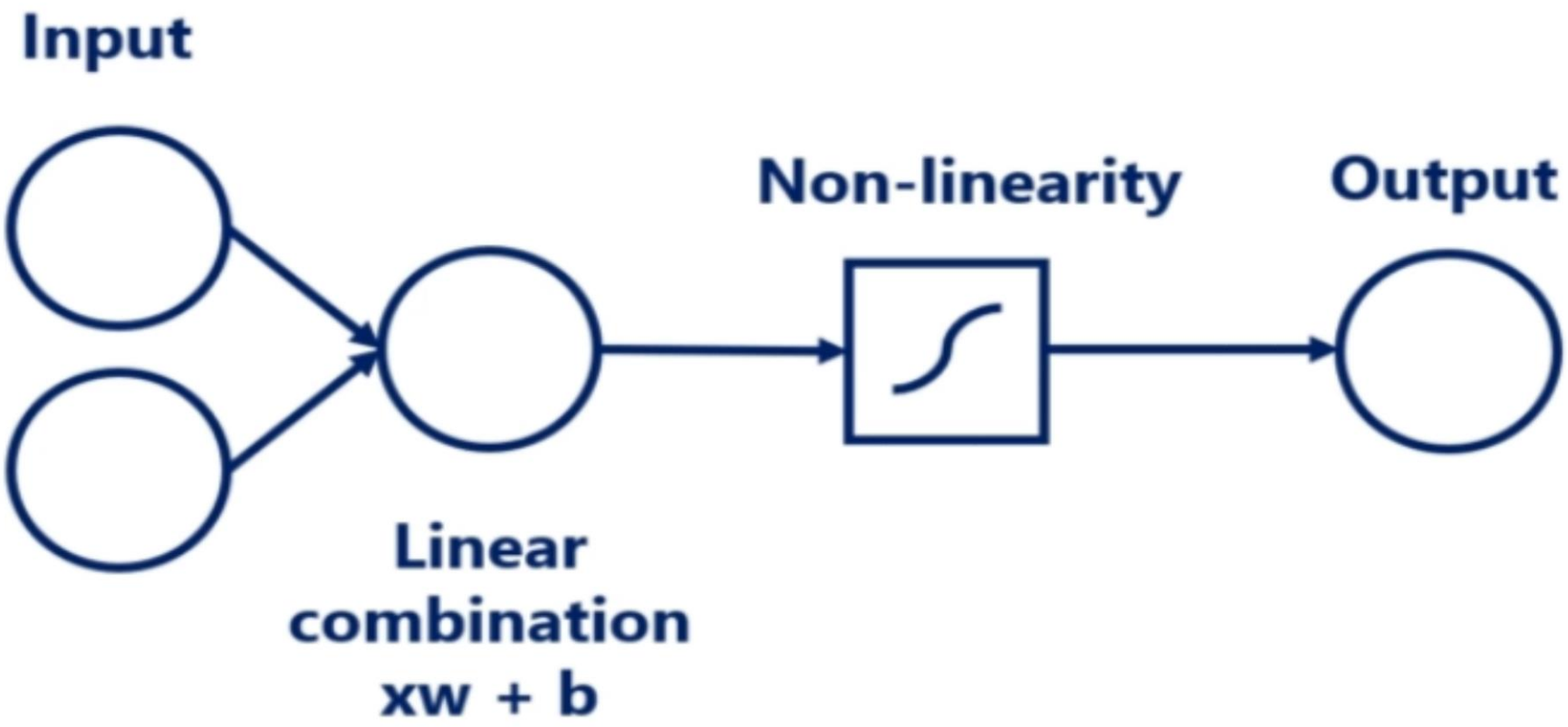
Temperature Northern hemisphere

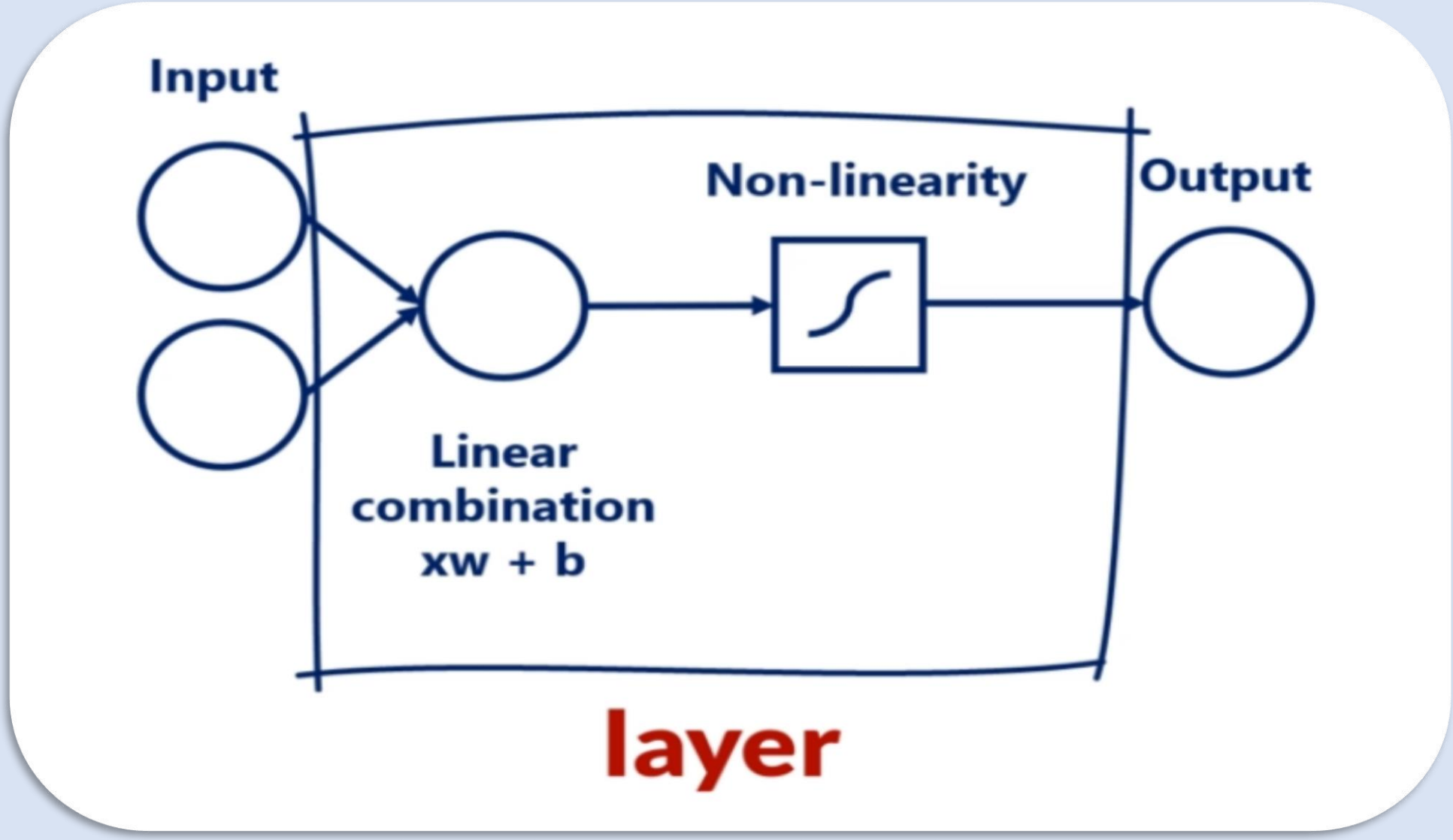


Income distribution in the US

Because we want to be better forecasters, we need better models

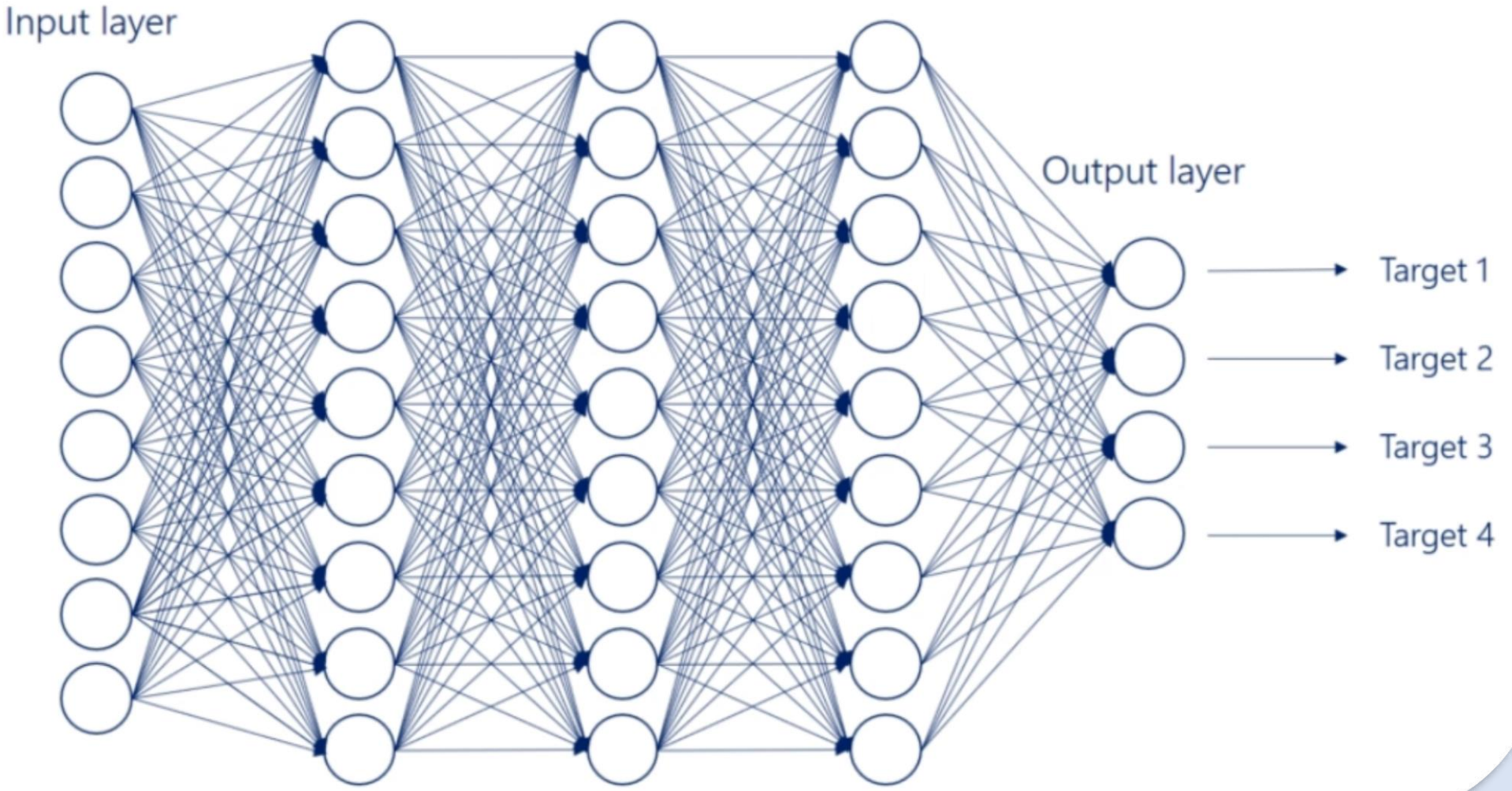
Linear + Non-linear



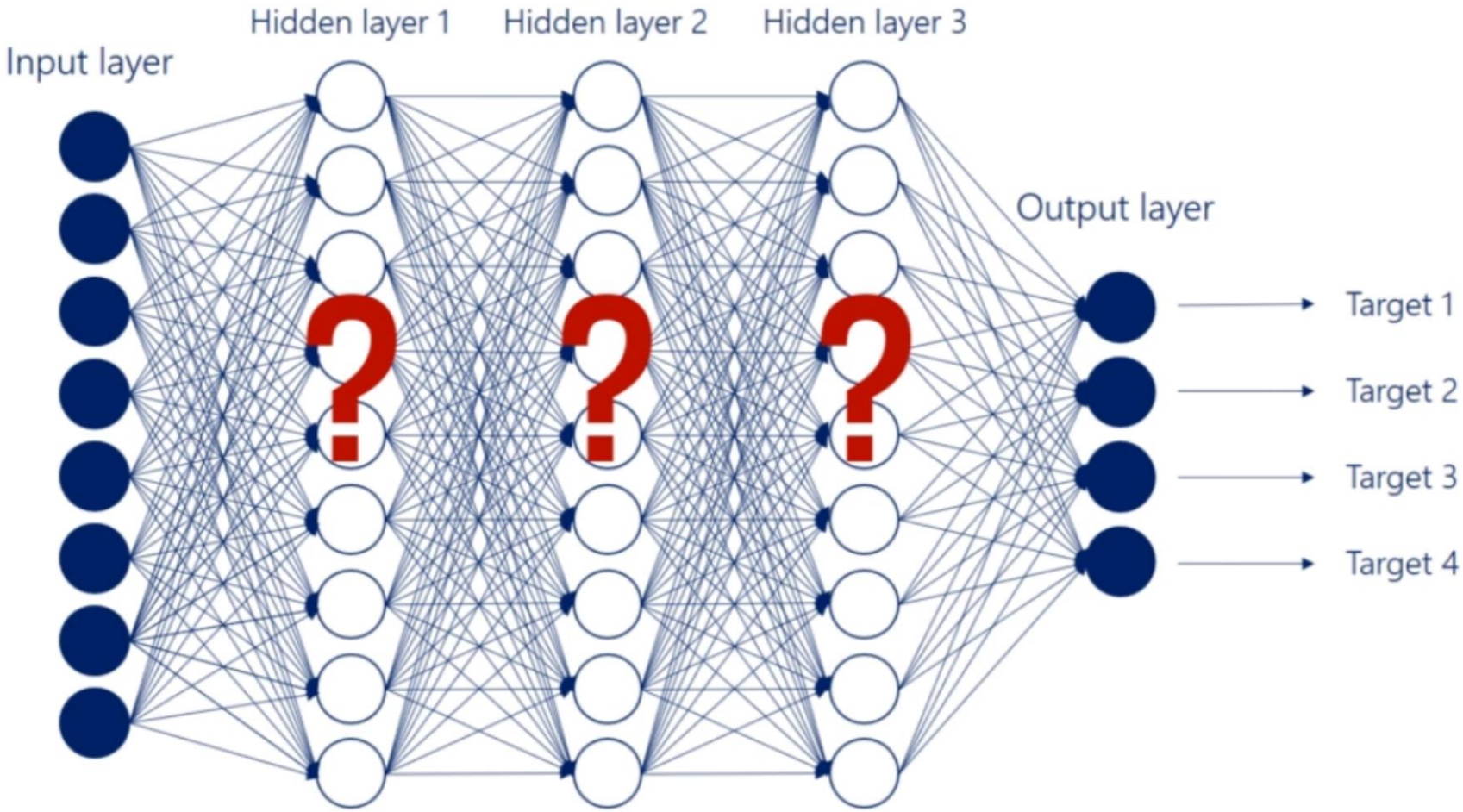


When we have more than one layer, we are talking about a **deep neural network**

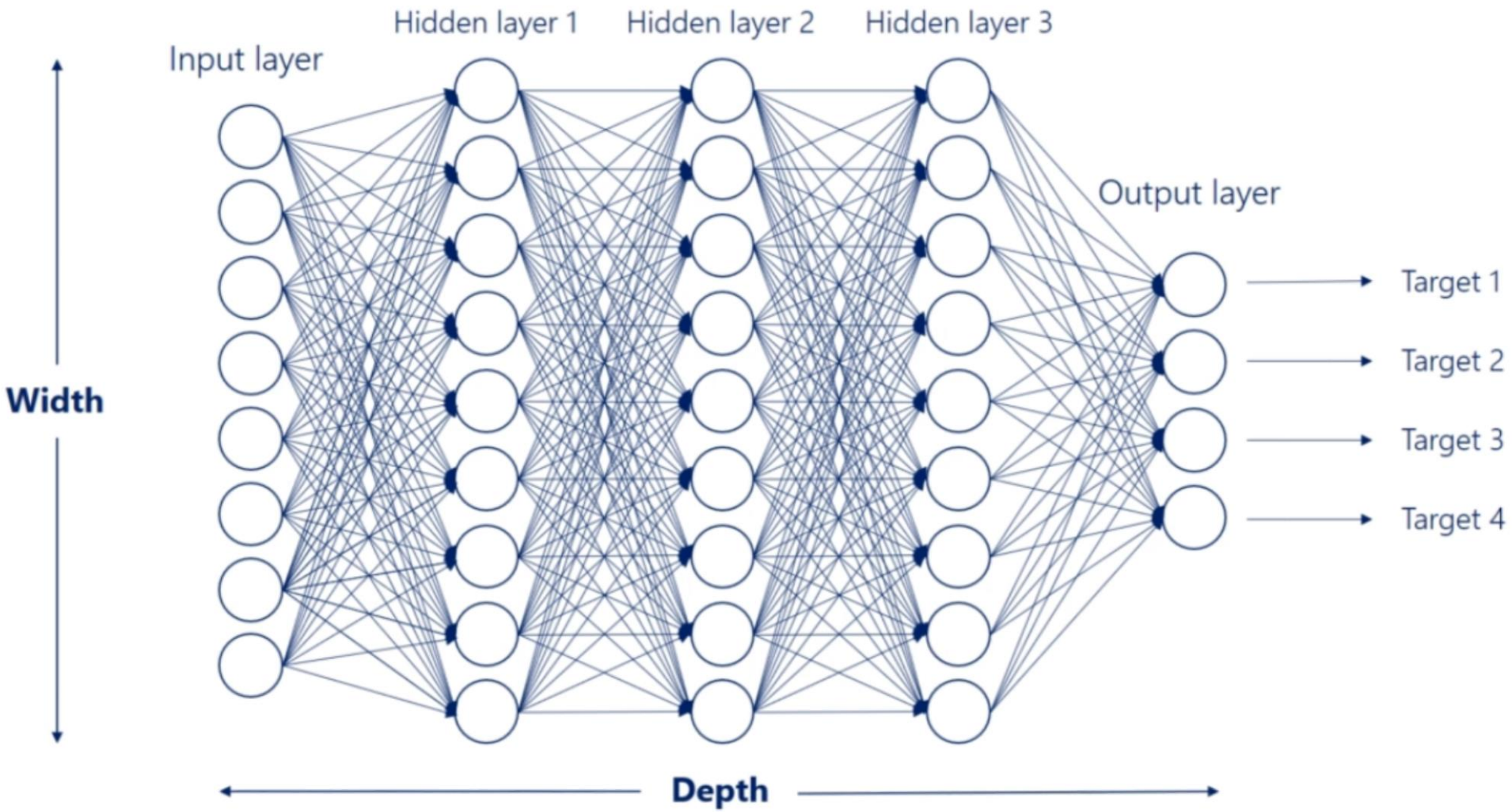
What is a deep net?



What is a deep net?



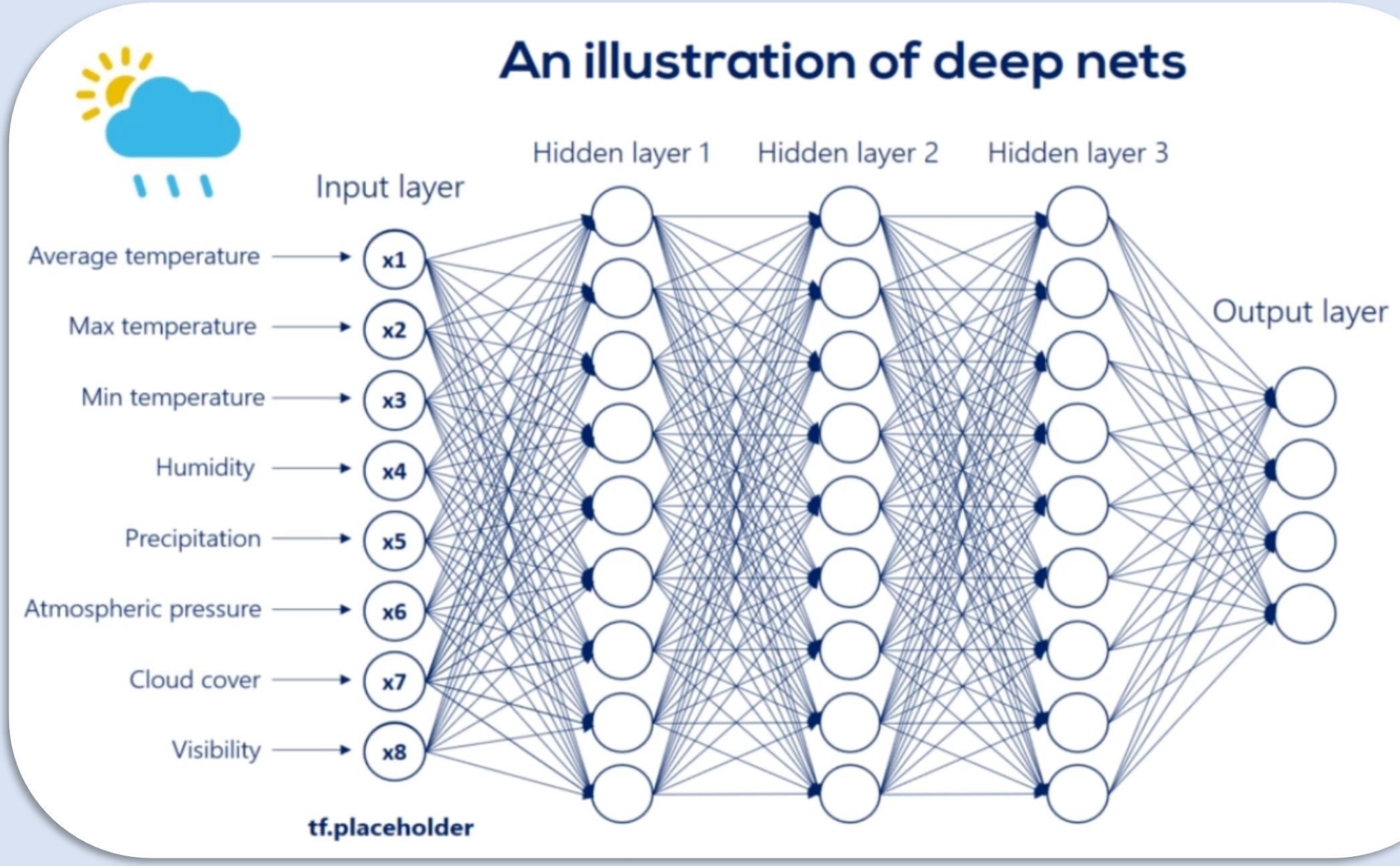
A deep neural network



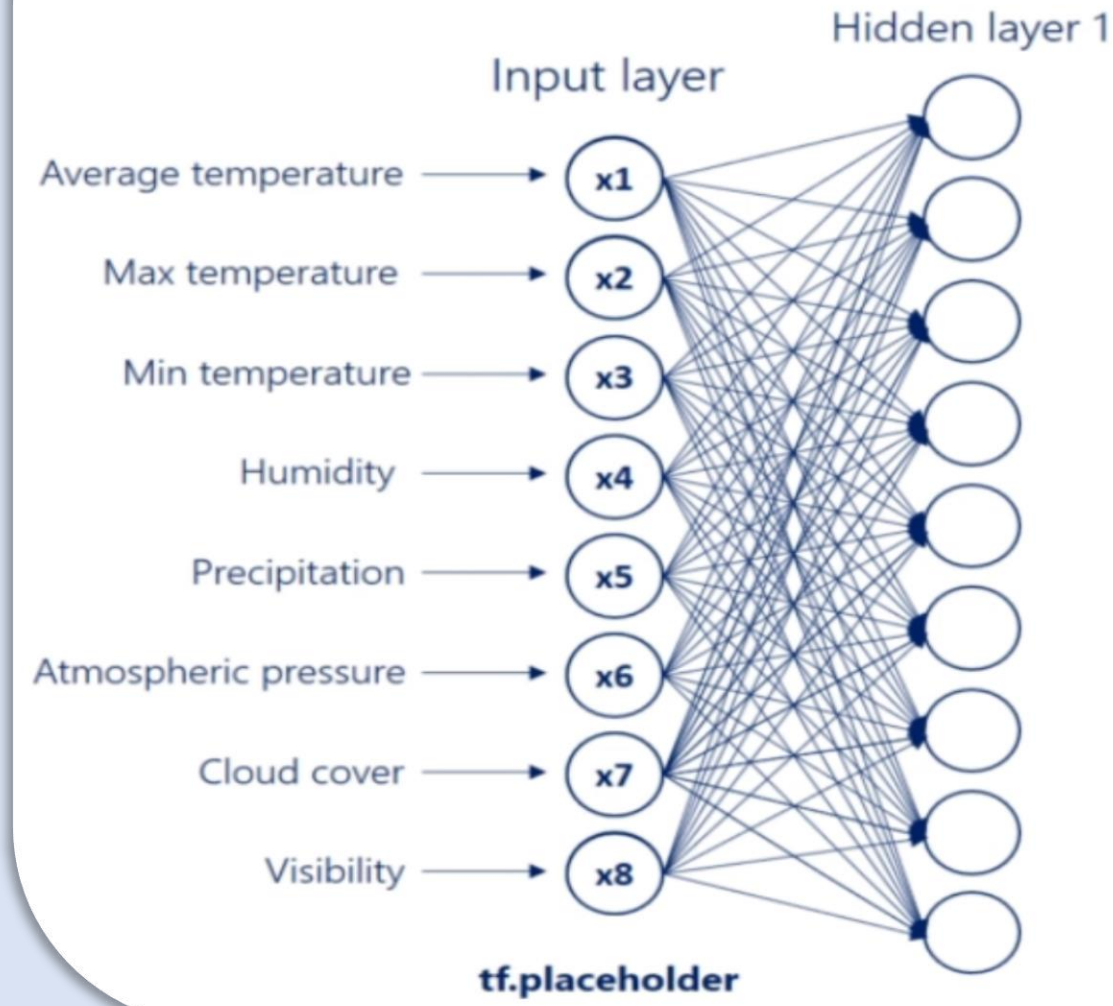
Hyperparameters vs Parameters

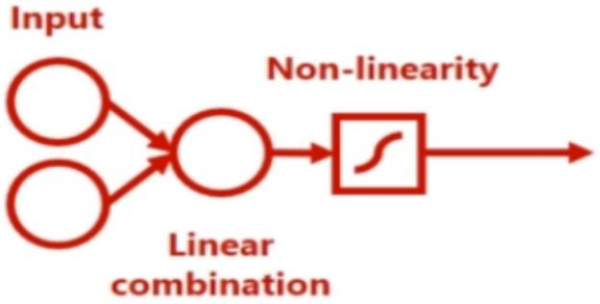
Hyperparameters	Parameters
Width	Weights (w)
Depth	Biases (b)
Learning rate (η)	

Hyperparameters pre-set by us	vs	Parameters found by optimizing
Width		Weights (w)
Depth		Biases (b)
Learning rate (η)		


An illustration of deep nets
Input layer
Hidden layer 1
Hidden layer 2
Hidden layer 3
Output layer
Average temperature
Max temperature
Min temperature
Humidity
Precipitation
Atmospheric pressure
Cloud cover
Visibility
x1
x2
x3
x4
x5
x6
x7
x8
tf.placeholder

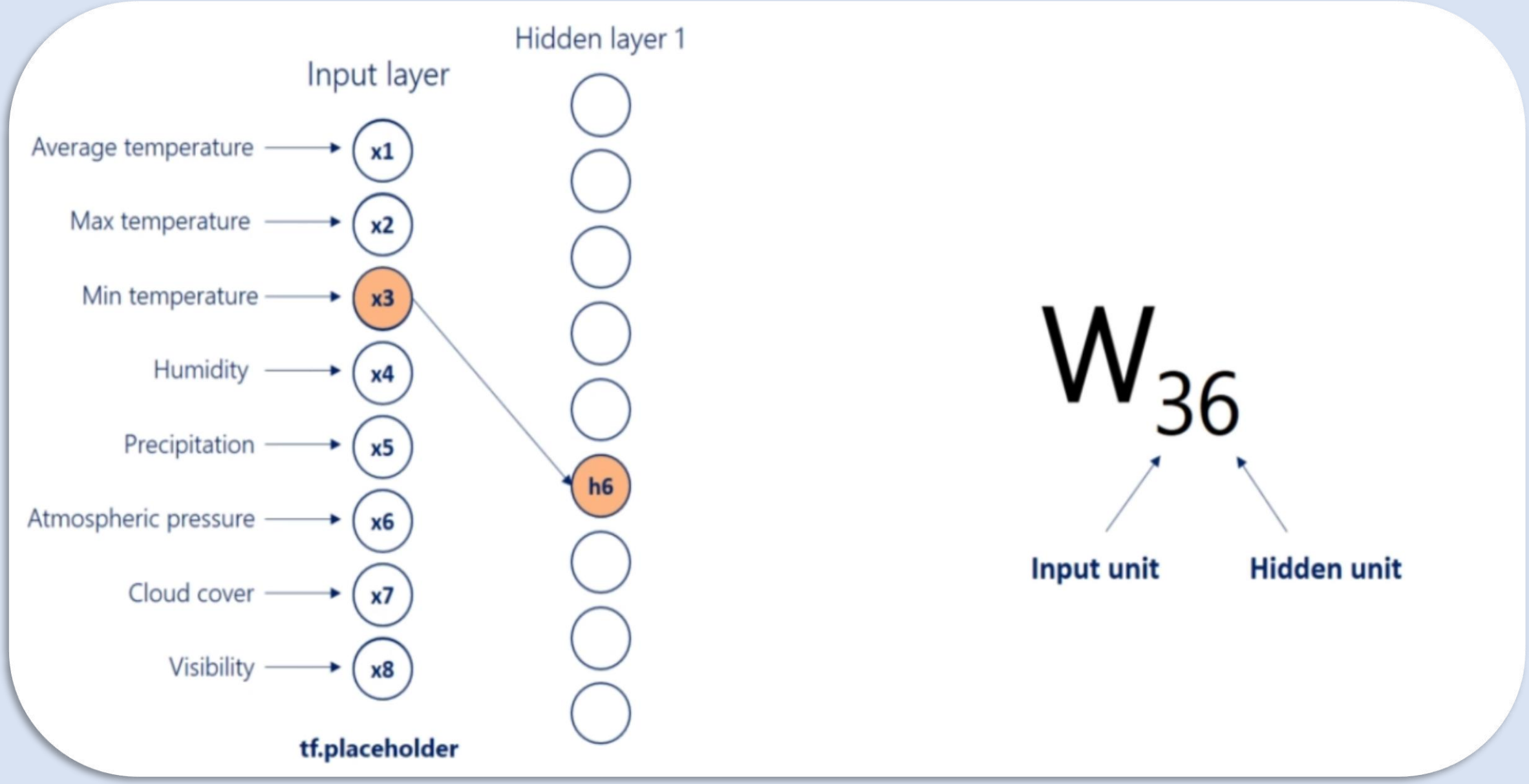
An illustration of deep nets





$$X * W$$

$$1 \times 8 \quad * \quad 8 \times 9 = 1 \times 9$$

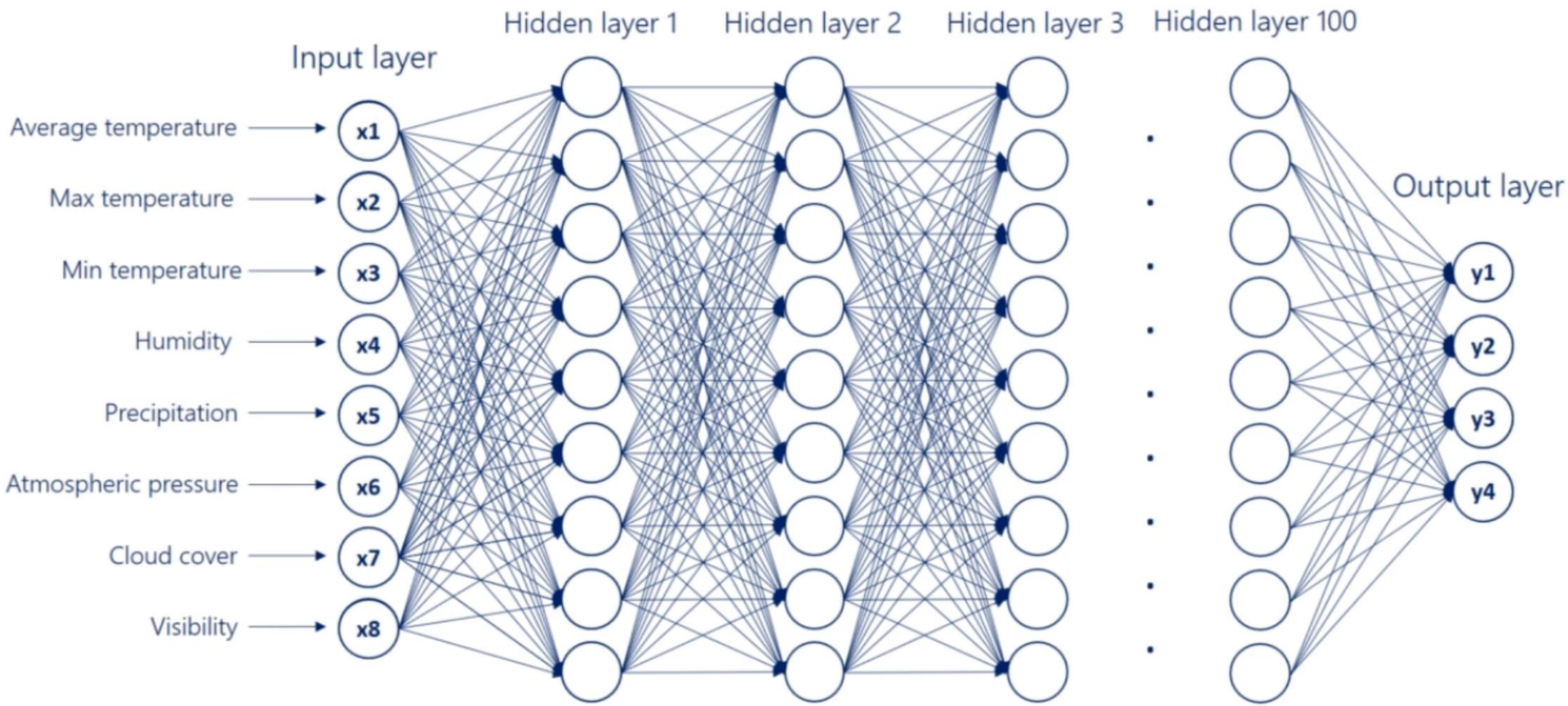


W_{36}

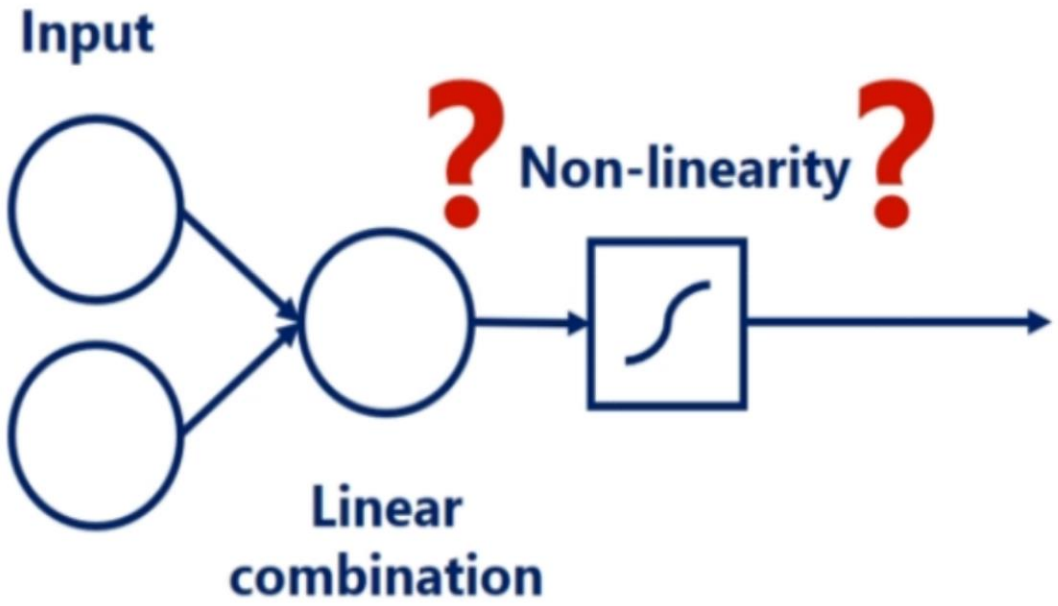
Input unit

Hidden unit

An illustration of deep nets



Why do we need non-linearities?

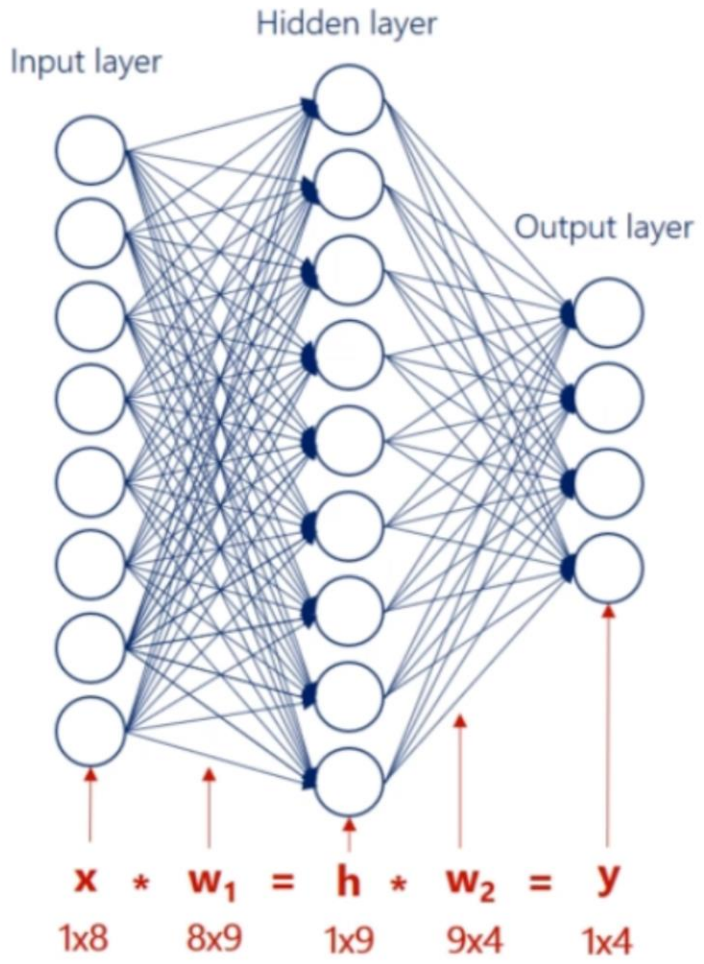


"Non-linearities are needed so we can break the linearity and represent more complicated relationships"

We cannot stack layers when we only have linear relationships

Why do we need non-linearities?

Imagine a network with no **non-linearities,** just linear combinations



$$h = x * w_1$$

$$y = h * w_2$$

$$h = x * w_1$$

$$y = h * w_2$$

$$h = x * w_1$$

$$y = x * w_1 * w_2$$

$$h = x * w_1$$

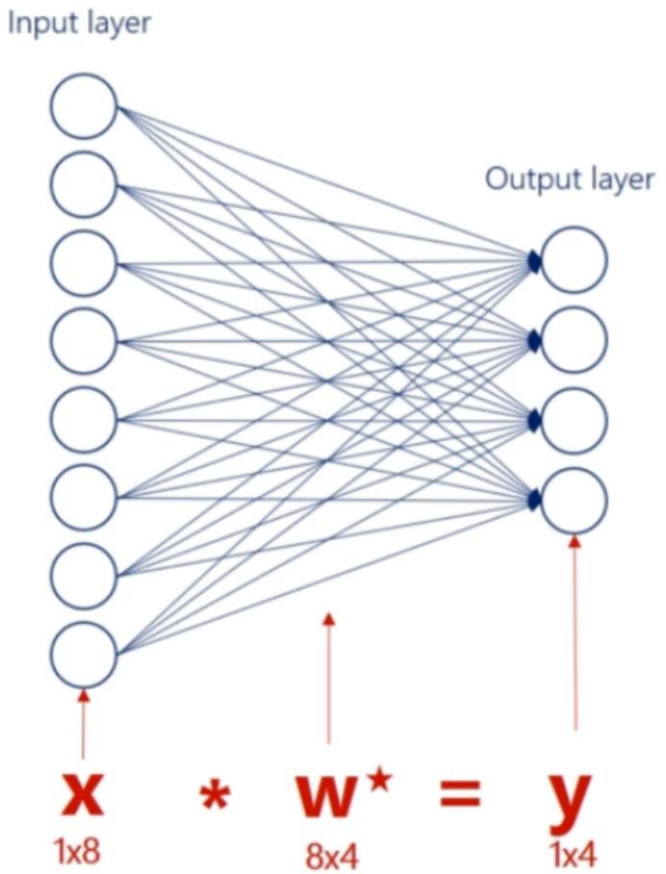
$$y = x * \underbrace{w_1 * w_2}_{8x9 \quad 9x4} =$$

$$= x * w^{\star}$$

$$8x4$$

Why do we need non-linearities?

Two consecutive linear transformations are equivalent to a single one.



Non-linearities

↓

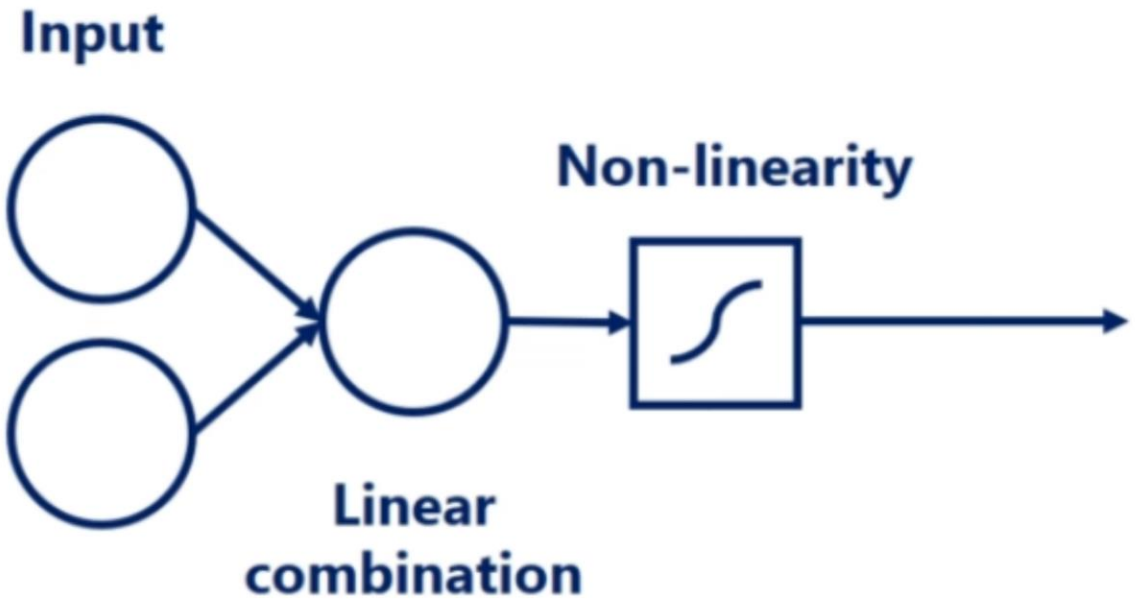
Stacking layers

↓

Depth

↓

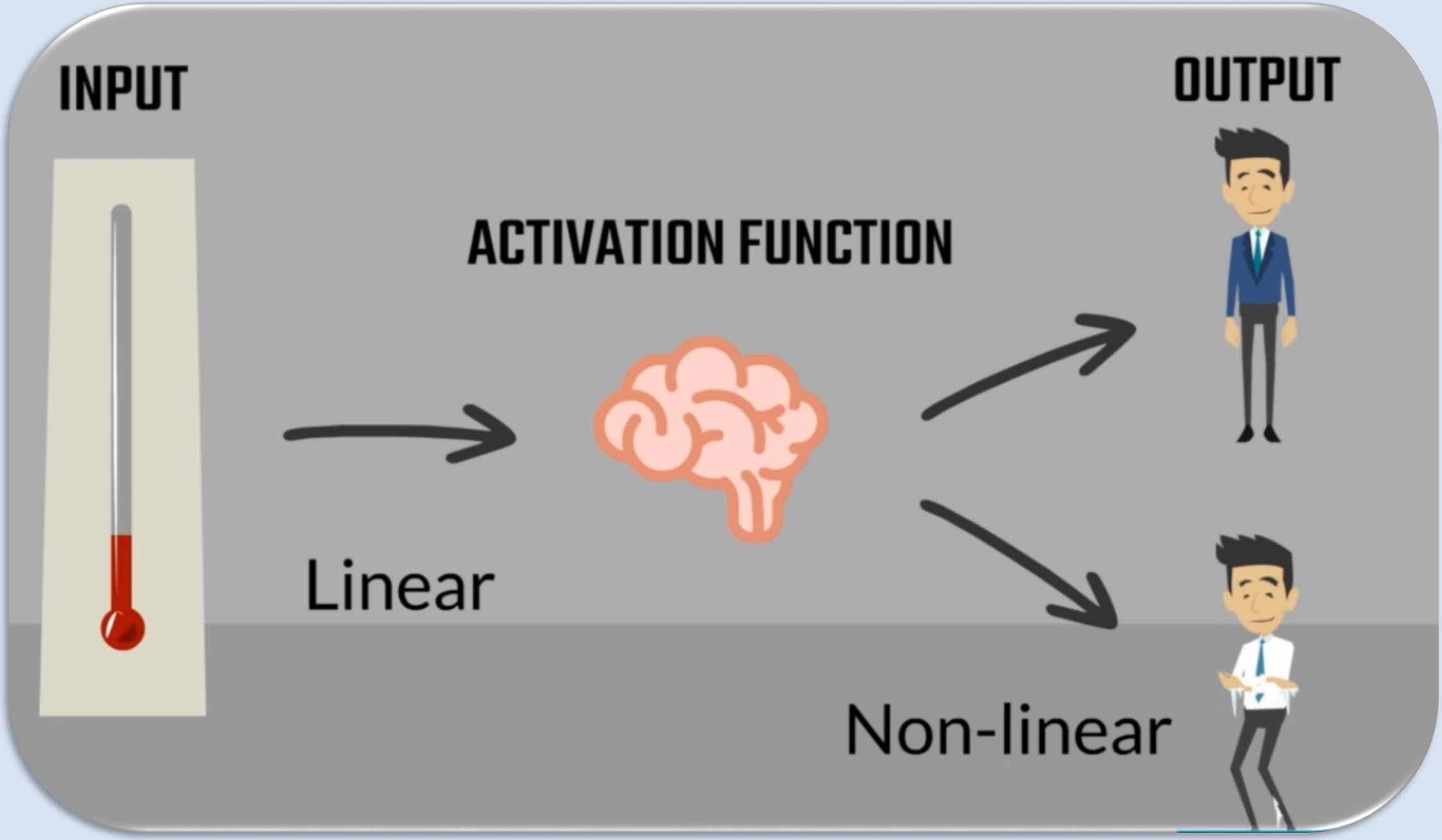
Deep learning



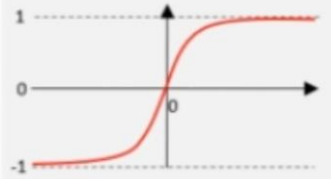
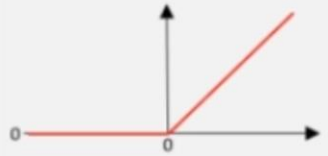
In order to have deep nets and find complex relationships through arbitrary functions, we need non-linearities.

یک مثال ساده از عملکرد جز غیر–خطی

Activation Function
Non-linearity
Transfer Function


INPUT
OUTPUT
ACTIVATION FUNCTION
Linear
Non-linear

Common activation functions

Name	Formula	Derivative	Graph	Range
sigmoid (logistic function)	$\sigma(a) = \frac{1}{1+e^{-a}}$	$\frac{\partial \sigma(a)}{\partial a} = \sigma(a)(1-\sigma(a))$		(0,1)
TanH (hyperbolic tangent)	$\tanh(a) = \frac{e^{a}-e^{-a}}{e^{a}+e^{-a}}$	$\frac{\partial \tanh(a)}{\partial a} = \frac{4}{(e^{a}+e^{-a})^2}$		(-1,1)
ReLu (rectified linear unit)	relu(a) = max(0,a)	$\frac{\partial \text{ relu}(a)}{\partial a} = \begin{cases} 0, if\ a \le 0 \\ 1, if\ a > 0 \end{cases}$		(0,∞)
softmax	$\sigma_i(\boldsymbol{a}) = \frac{e^{a_i}}{\sum_j e^{a_j}}$	$\frac{\partial \sigma_i(\boldsymbol{a})}{\partial a_j} = \sigma_i(\boldsymbol{a})\left(\delta_{ij} - \sigma_j(\boldsymbol{a})\right)$ Where δ_{ij} is 1 if i=j, 0 otherwise	different every time	(0,1)

یک مثال ساده از سافت‌مکس

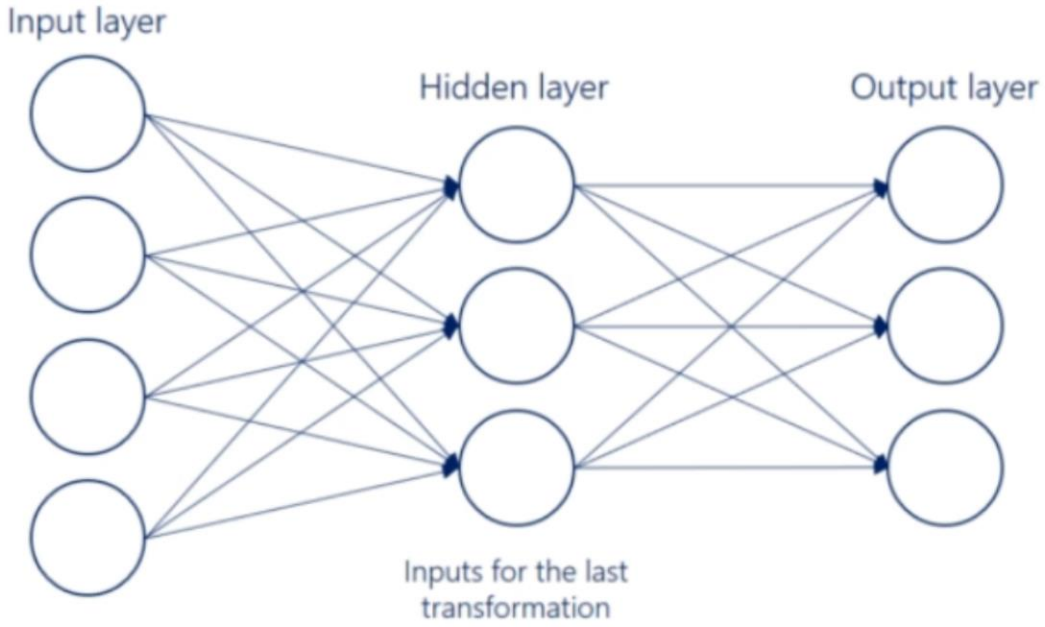
Softmax

$$a = xw+b$$

Linear combination

$$y = \text{softmax}(\mathbf{a})$$

Activation



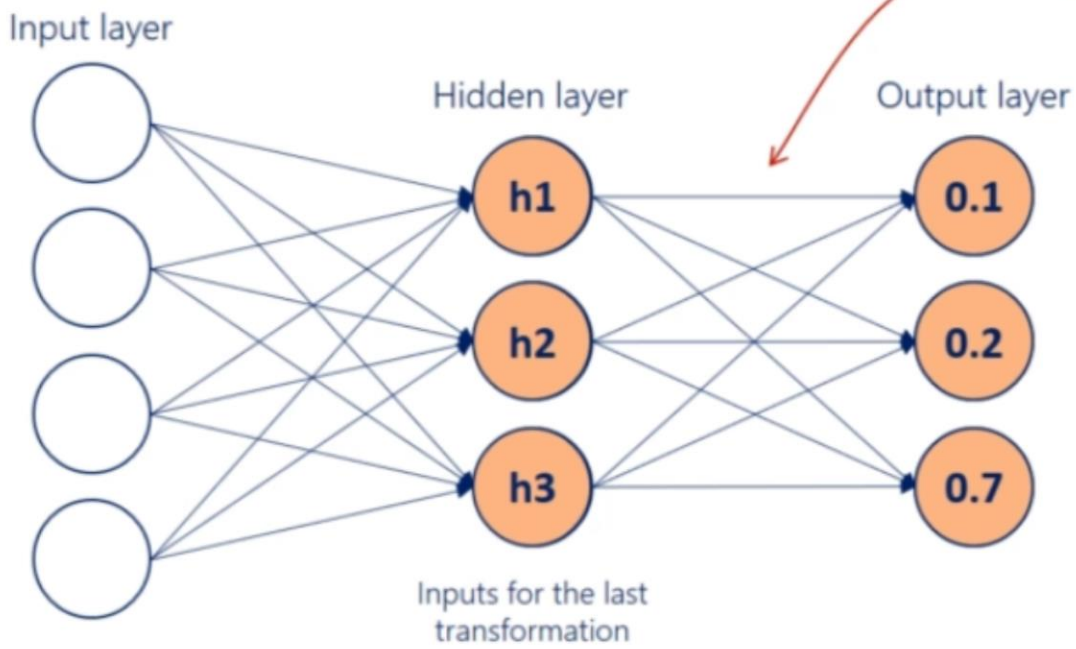
Softmax

$$a = xw+b$$

Linear combination

$$y = \text{softmax}(\mathbf{a})$$

Activation



$$\mathbf{a}_h = hw+b$$

$$\mathbf{a} = [-0.21, 0.47, 1.72]$$

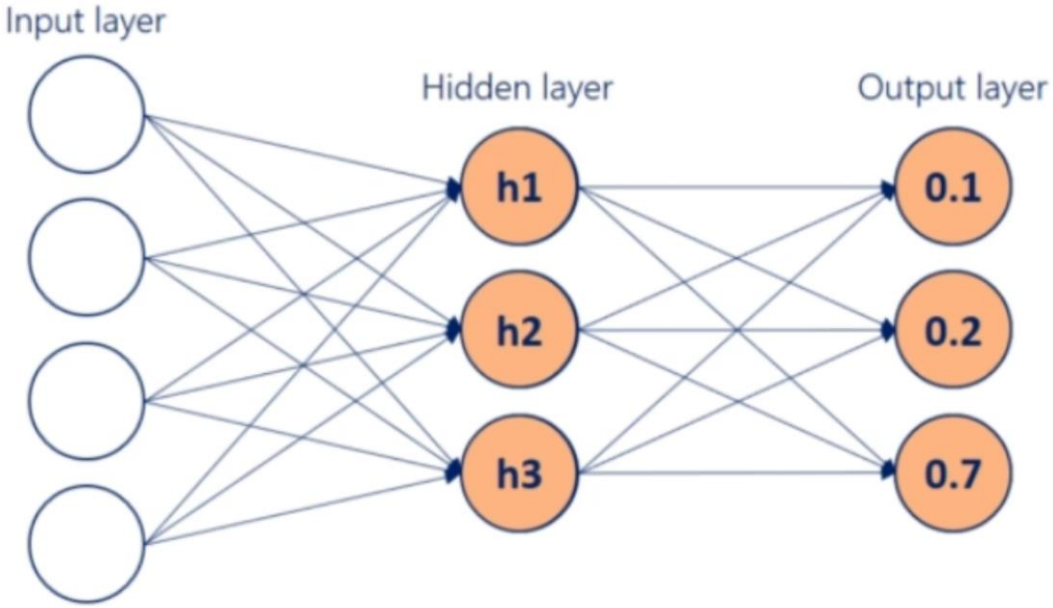
$$\text{softmax}(\mathbf{a}) = \frac{e^{a_i}}{\sum_j e^{a_j}}$$

$$\sum_j e^{a_j} = e^{-0.21} + e^{0.47} + e^{1.72} = 8$$

$$\text{softmax}(\mathbf{a}) = \left[\frac{e^{-0.21}}{8}, \frac{e^{0.47}}{8}, \frac{e^{1.72}}{8}\right]$$

$$y = [0.1, 0.2, 0.7]$$

Softmax



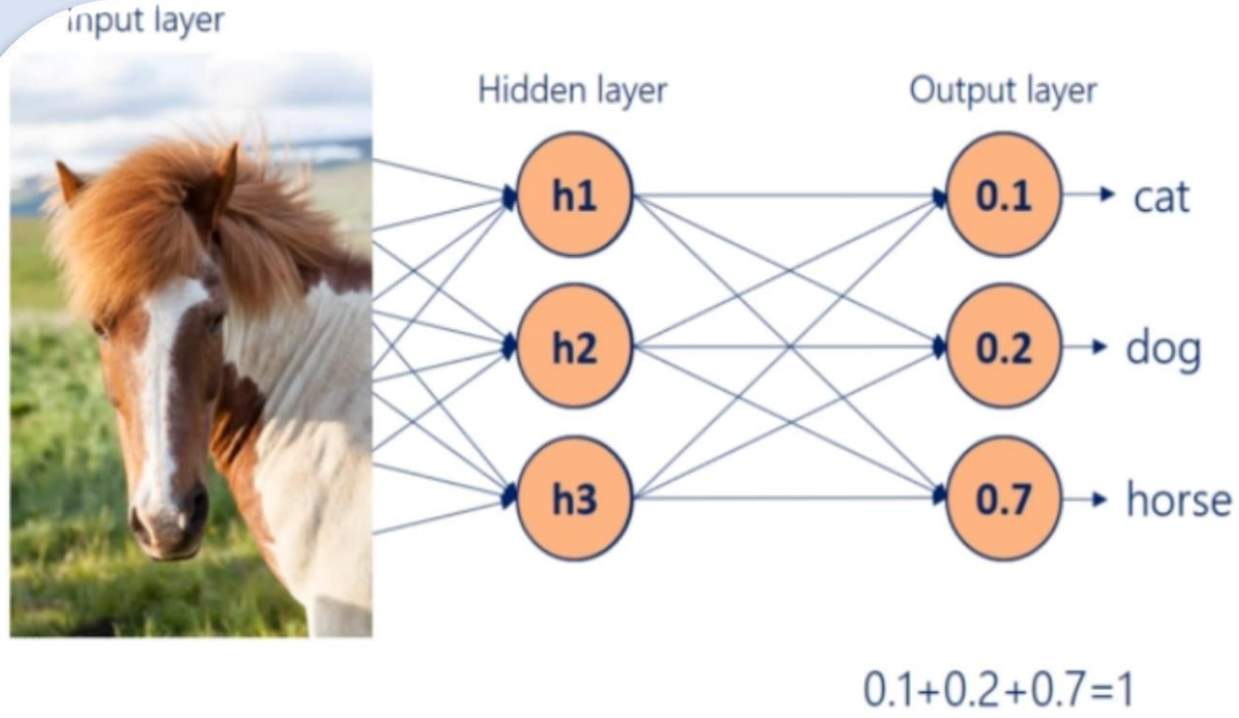
0.1+0.2+0.7=1

1. Range: (0,1)

2. They always sum up to 1

PROBABILITIES

The softmax transformation transforms a bunch of arbitrarily large or small numbers into a valid probability distribution



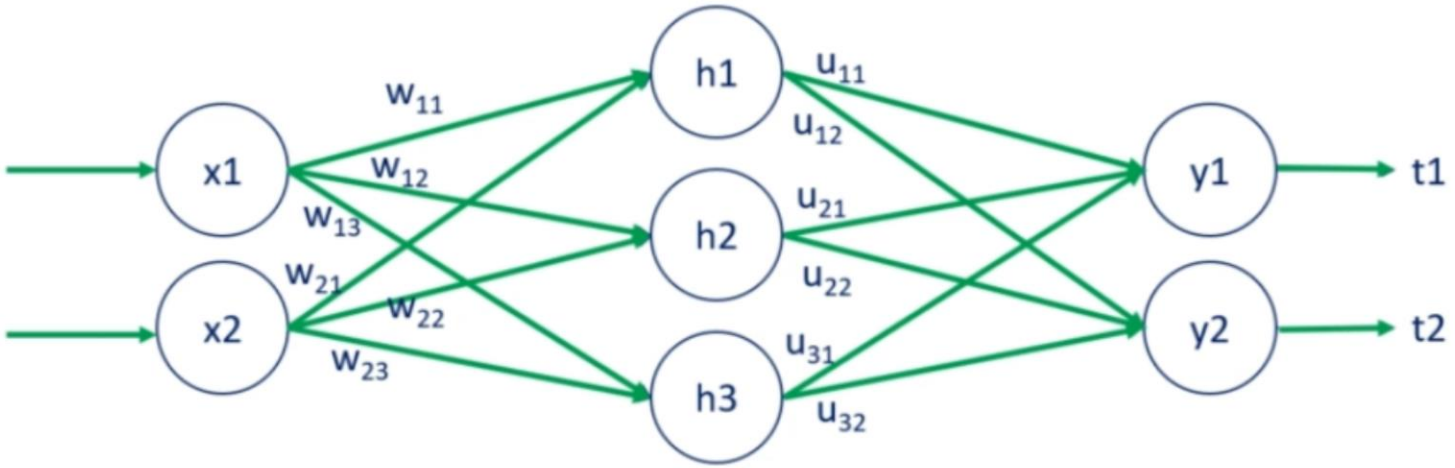
PROBABILITIES

The softmax transformation transforms a bunch of arbitrarily large or small numbers into a valid probability distribution

0.1+0.2+0.7=1

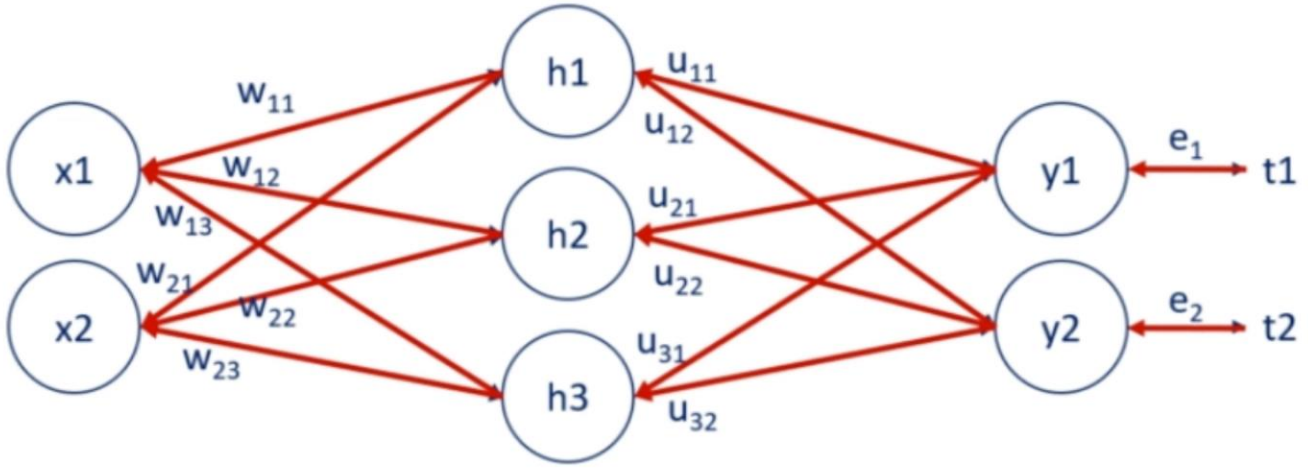
Softmax is often used as the activation of the output layer in classification problems

Forward propagation



Forward propagation is the process of pushing inputs through the net

Backpropagation

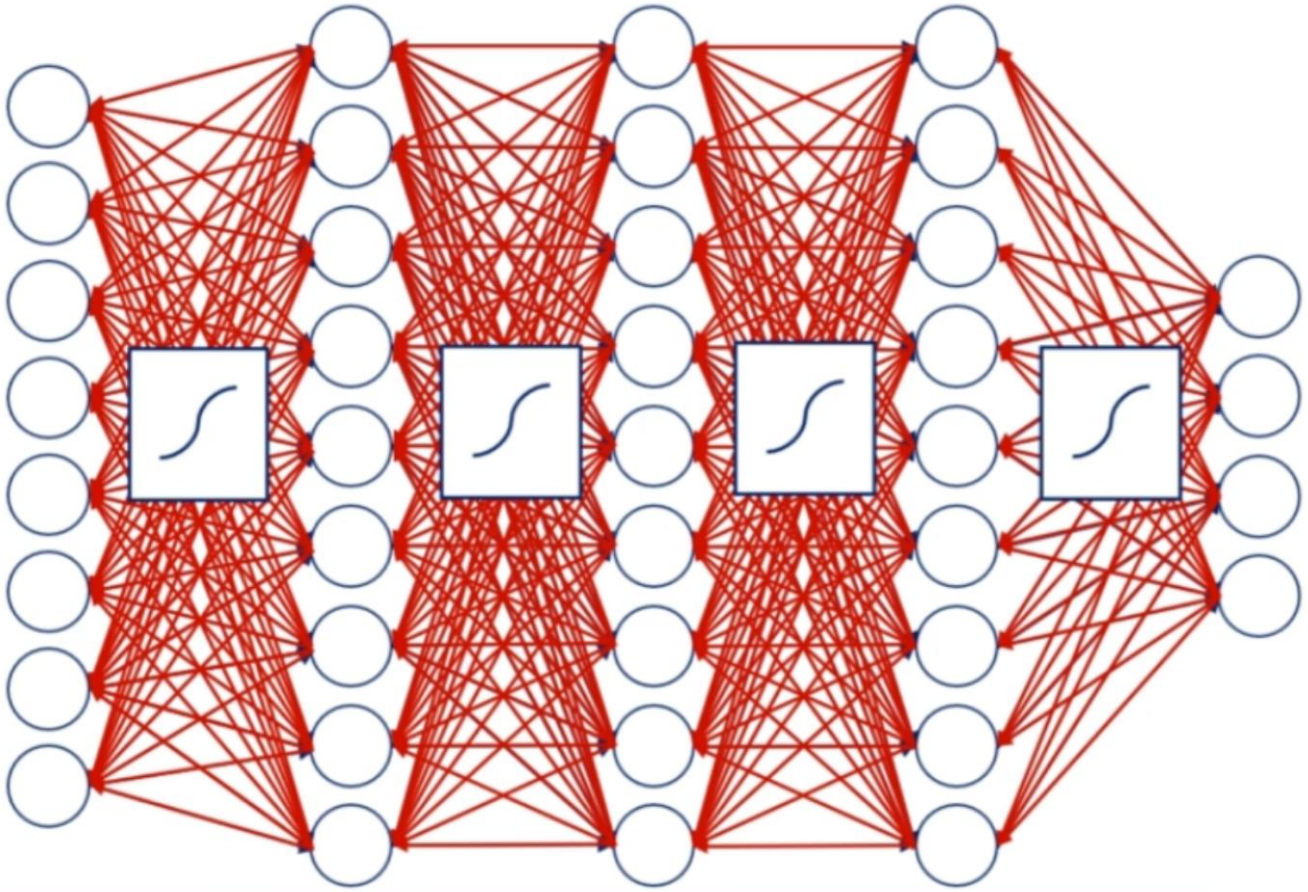


End of epoch

At the end of each epoch, we **backpropagate** and change each parameter accordingly

Backpropagation

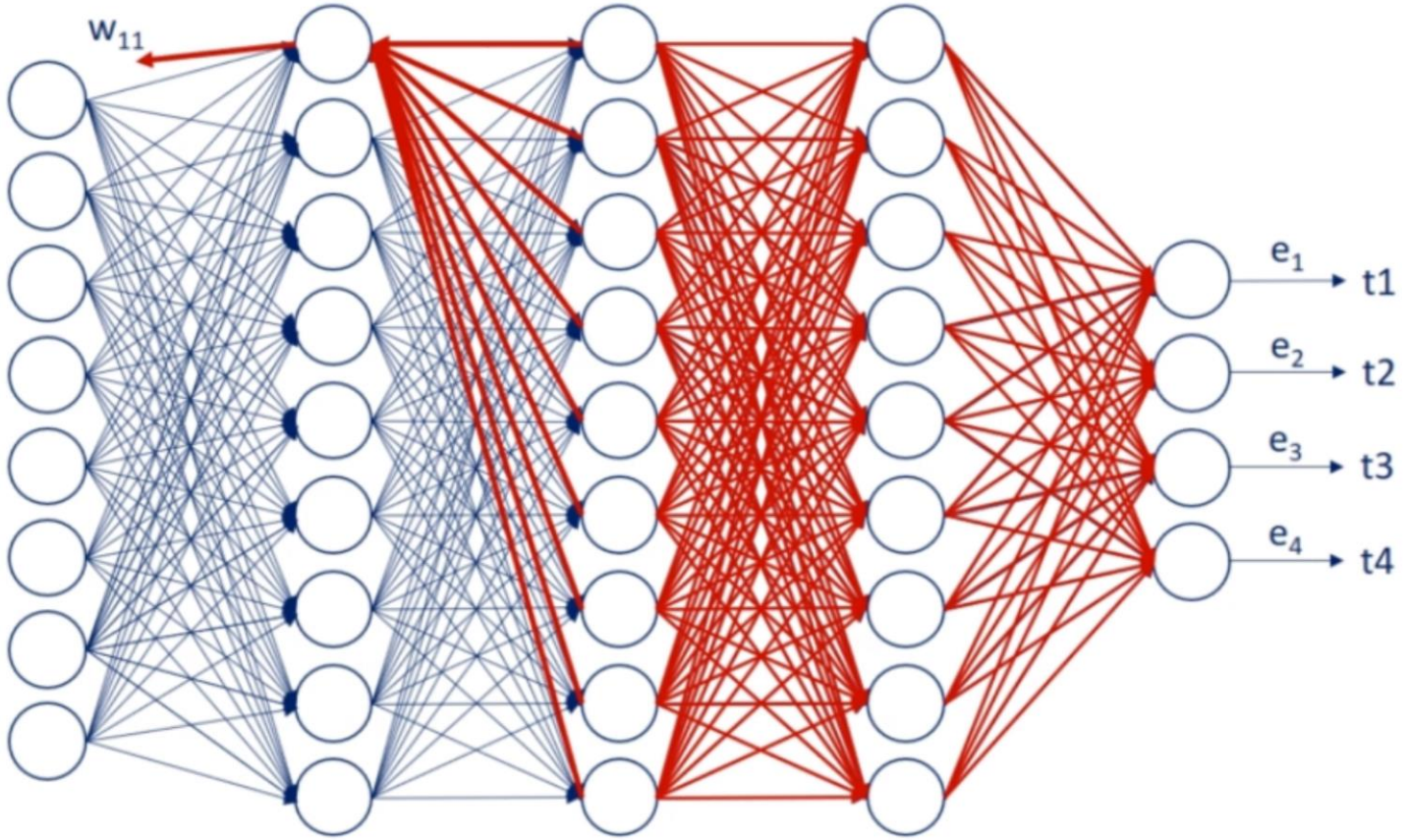
$\delta_{i,hidden} = ?$ $\delta_{i,hidden} = ?$ $\delta_{i,hidden} = ?$



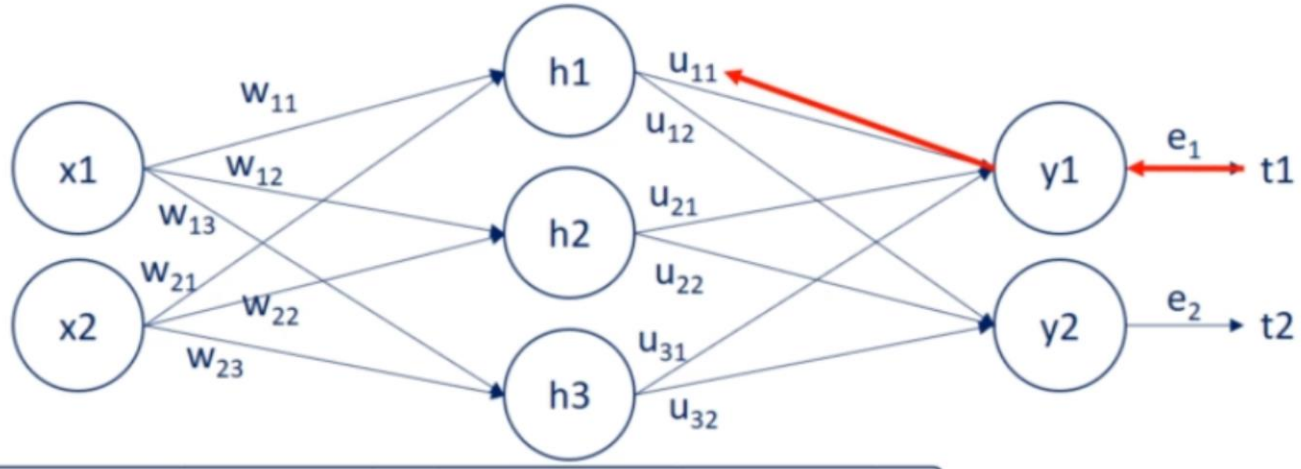
$\delta_{i,output} = y_i - t_i$

We have to update the parameters **as if we had "hidden targets"**

Backpropagation



Backpropagation

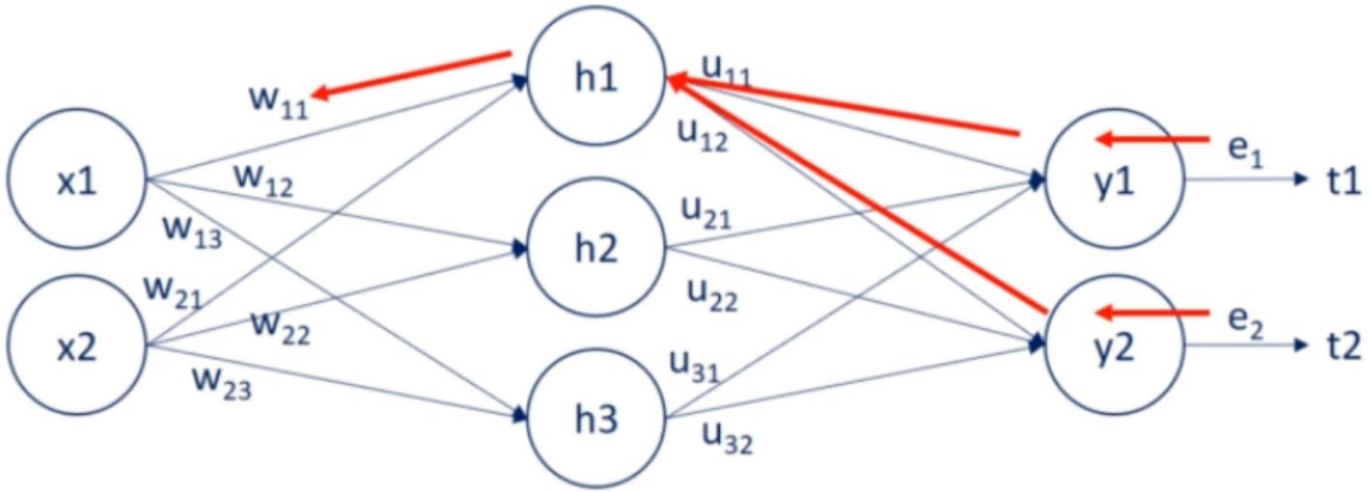


U_{11} contributes to a single error e_1

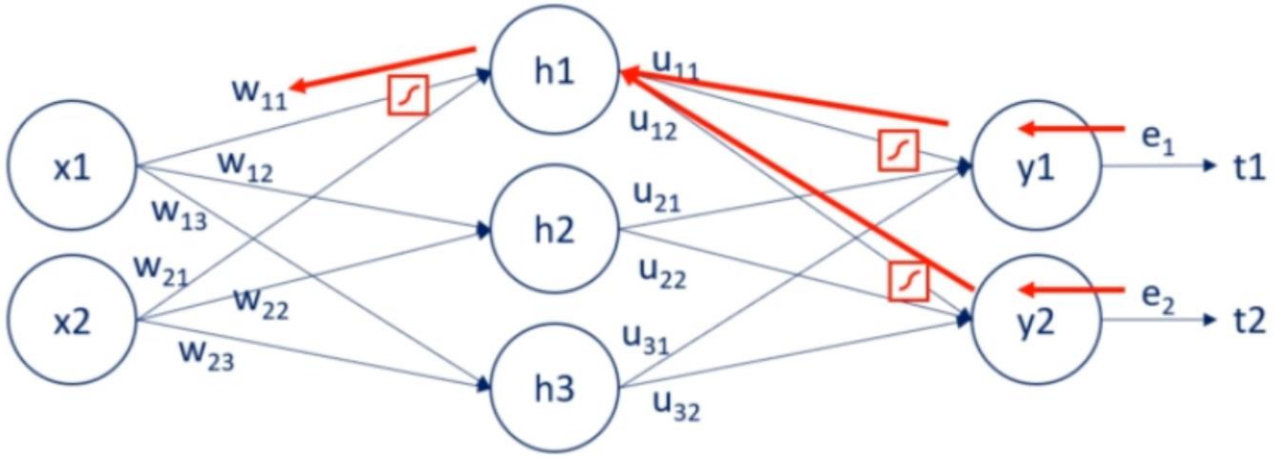
we find its derivative (gradient) $\boldsymbol{u}_{i+1} = \boldsymbol{u}_i - \eta \nabla_{\mathbf{u}} L(\boldsymbol{u}_i)$

and update the coefficient.

Backpropagation

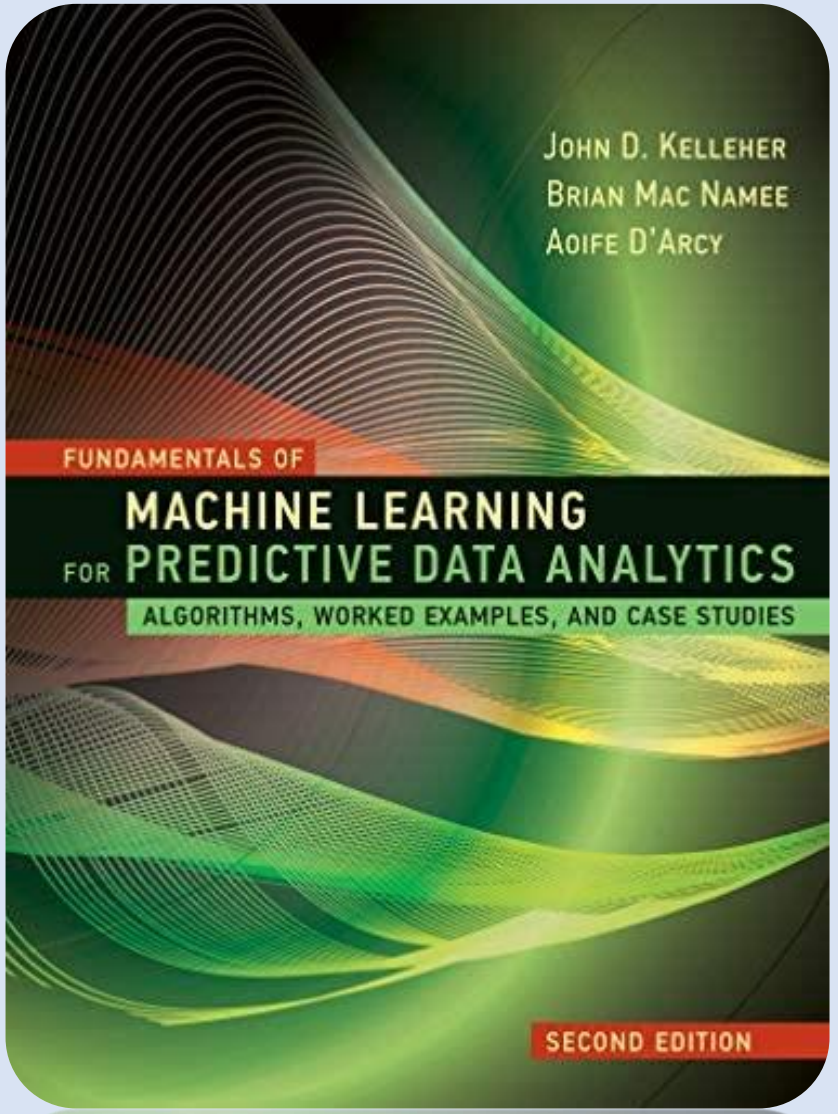


Backpropagation mathematics



L^2 −Norm Loss and Sigmoid Activation

$$\frac{\partial L}{\partial w_{ij}} = \delta_j x_i \text{ , where } \delta_j = \sum_k \delta_k w_{jk}\, y_j(1 - y_j)$$


JOHN D. KELLEHER
BRIAN MAC NAMEE
AOIFE D'ARCY
FUNDAMENTALS OF
MACHINE LEARNING
FOR PREDICTIVE DATA ANALYTICS
ALGORITHMS, WORKED EXAMPLES, AND CASE STUDIES
SECOND EDITION

CONTENTS