

Title: Machine Learning Lecture

Speakers: Mohamed Hibat Allah

Collection: Machine Learning 2023/24

Date: April 09, 2024 - 9:00 AM

URL: <https://pirsa.org/24040049>

Lecture 3

Recall the goal of supervised learning (SL). Given a dataset

$D = \{(\vec{x}, \vec{y})\}$, fit a function $f(\vec{x})$ to \vec{y}

$$\vec{x} = (x_1, x_2, \dots, x_{d_x})$$

$$\vec{y} = (y_1, y_2, \dots, y_{d_y})$$

$$f: \mathbb{R}^{d_x} \rightarrow \mathbb{R}^{d_y}$$

Sofar, we have studied two SL algorithms
"Linear regression" and "logistic regression"

* Outline for today Intro to SL with feed-forward Neural Networks (FFNNs)

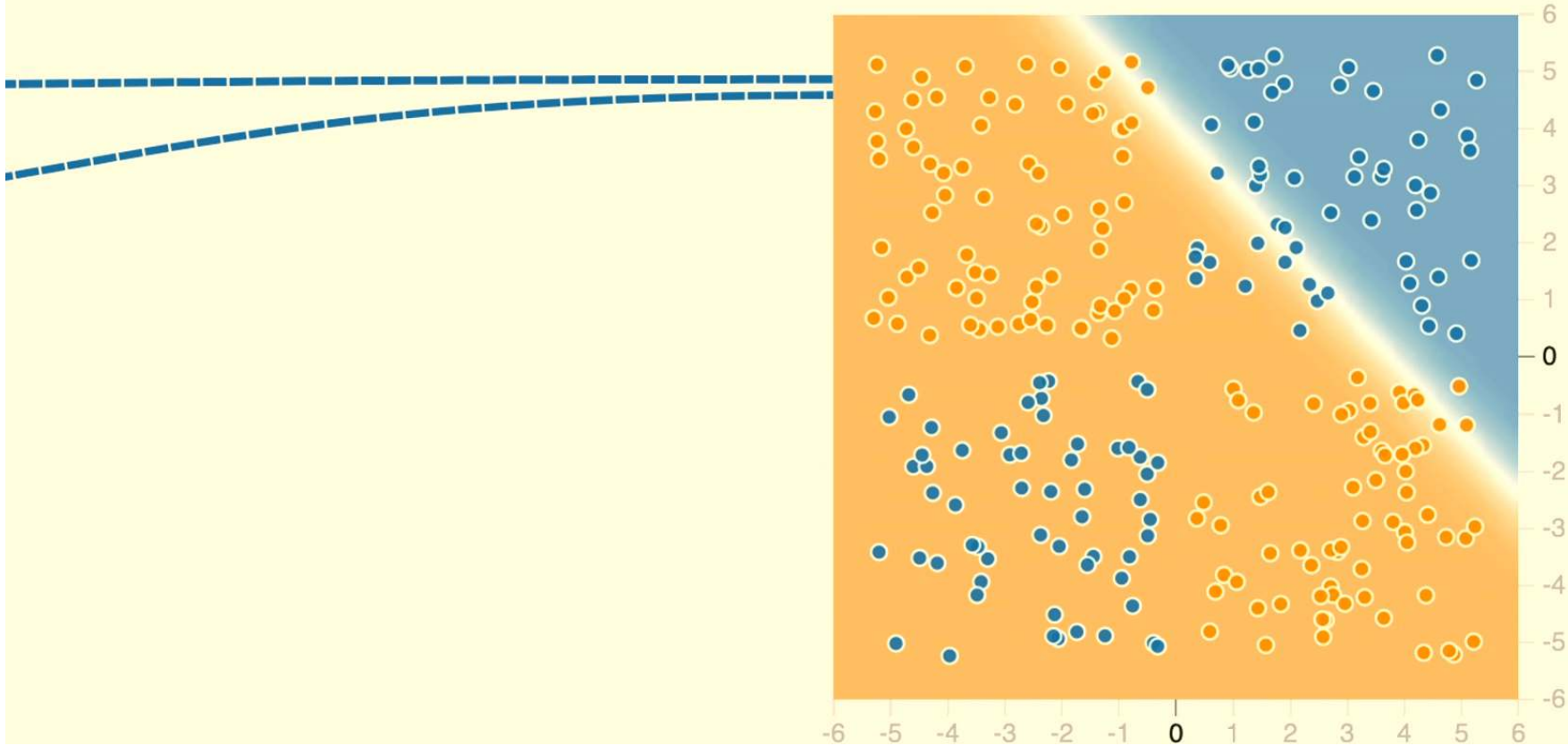
- ↳ Network architecture.
 - Weights & Biases.
 - Activation functions
- ↳ Expressivity of NNs
- ↳ Basics of training.
 - Cost functions.

0 HIDDEN LAYERS

OUTPUT

Test loss 0.599

Training loss 0.549



Historic motivation

Neural networks have been studied since the 70's, until the breakthrough moment in 2012 in the field of **computer vision**.

Nowadays, neural networks are doing wonders in the field of **natural language processing**.

ImageNet Classification with Deep Convolutional Neural Networks

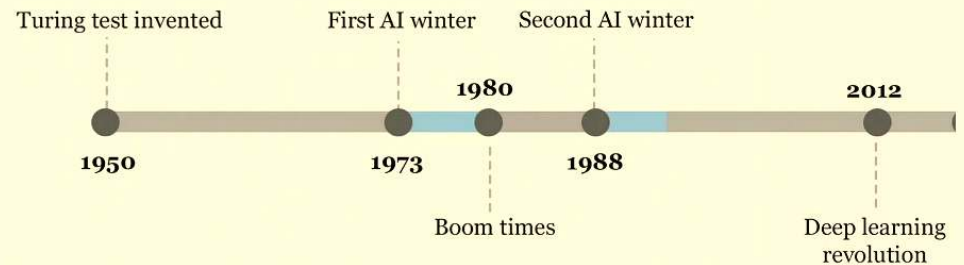
Alex Krizhevsky
University of Toronto
kriz@cs.utoronto.ca

Ilya Sutskever
University of Toronto
ilya@cs.utoronto.ca

Geoffrey E. Hinton
University of Toronto
hinton@cs.utoronto.ca

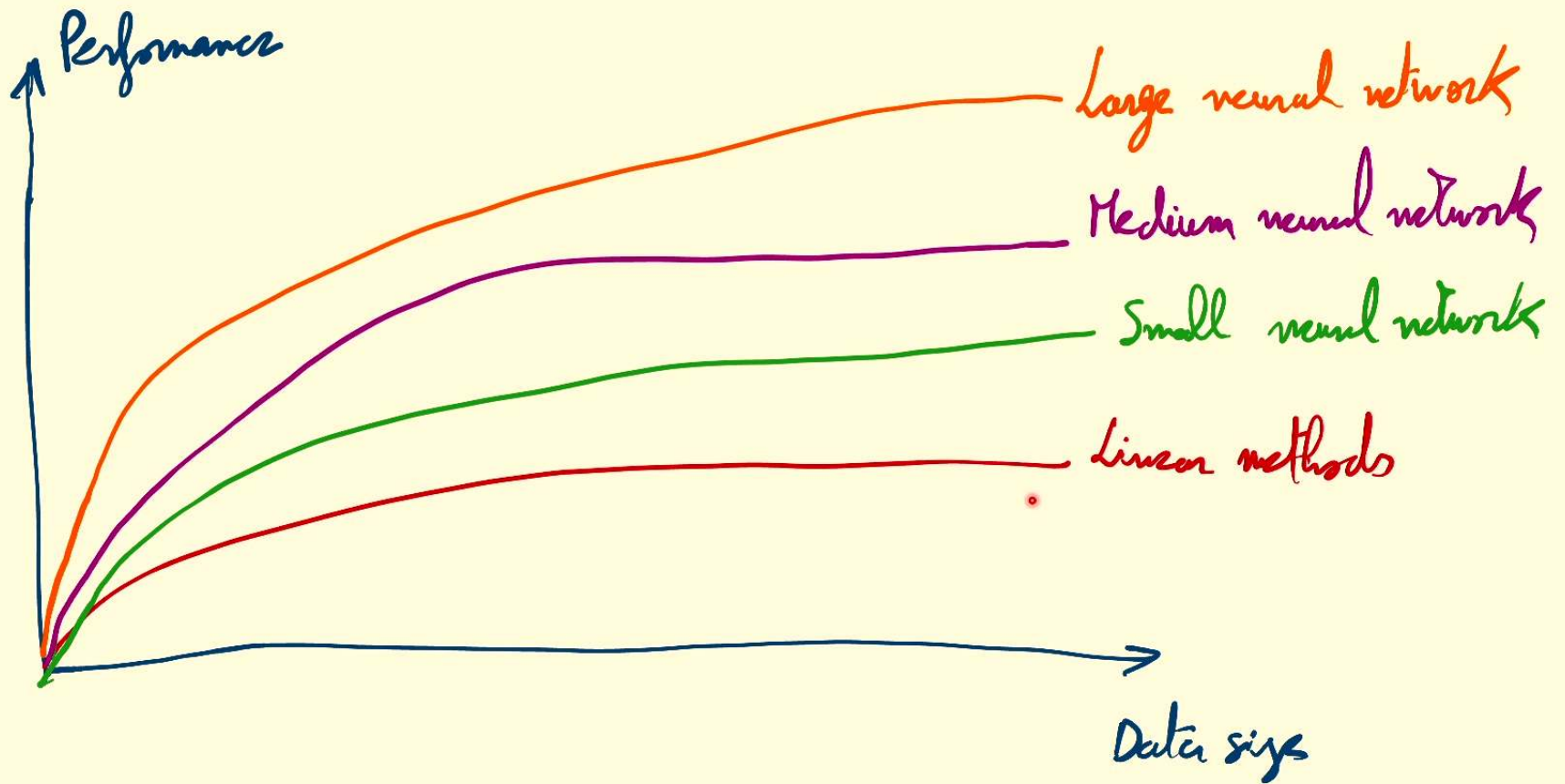
Abstract

We trained a large, deep convolutional neural network to classify the 1.2 million high-resolution images in the ImageNet ILSVRC-2010 contest into the 1000 different classes. On the test data, we achieved top-1 and top-5 error rates of 37.5% and 17.0% which is considerably better than the previous state-of-the-art. The neural network, which has 60 million parameters and 650,000 neurons, consists of five convolutional layers, some of which are followed by max-pooling layers, and three fully-connected layers with a final 1000-way softmax. To make training faster, we used non-saturating neurons and a very efficient GPU implementation of the convolution operation. To reduce overfitting in the fully-connected layers we employed a recently-developed regularization method called "dropout" that proved to be very effective. We also entered a variant of this model in the ILSVRC-2012 competition and achieved a winning top-5 test error rate of 15.3%, compared to 26.2% achieved by the second-best entry.



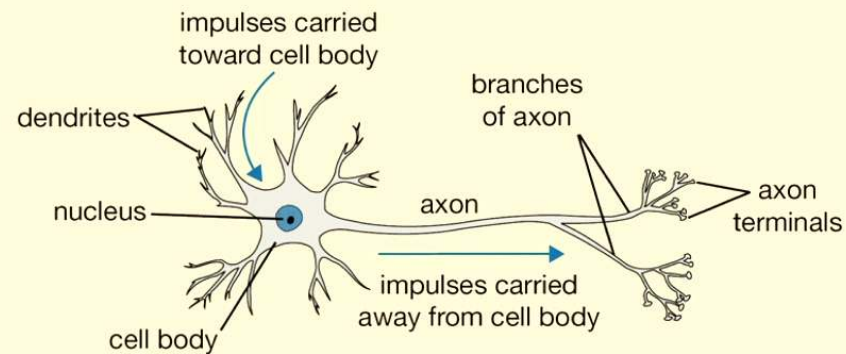
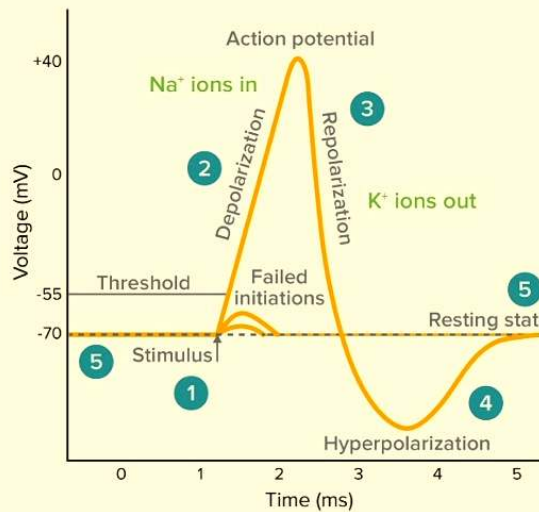
Credit: towardsdatascience.com

Why neural networks now?



Inspiration: the brain

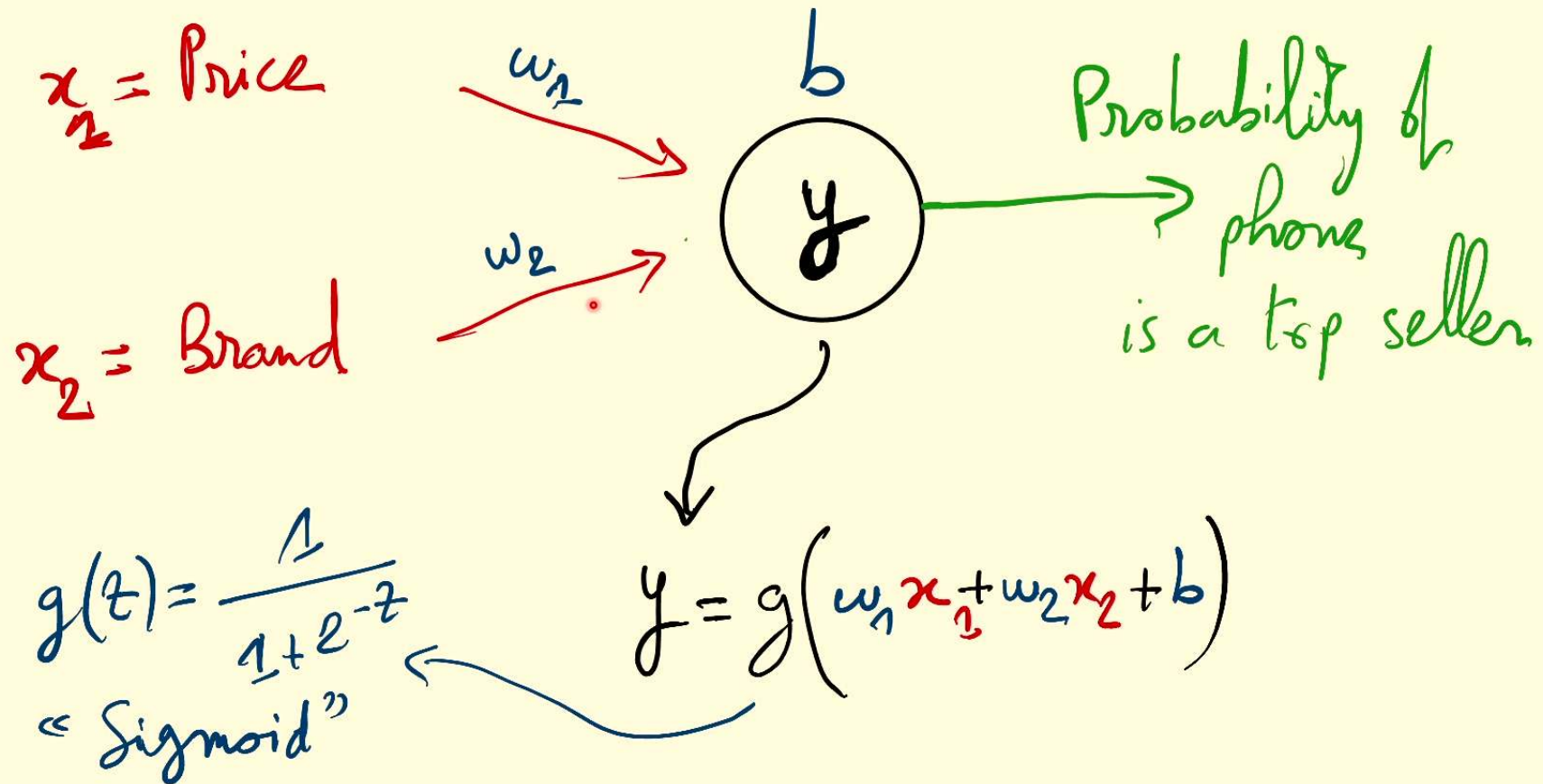
- Our brain has $\sim 10^{11}$ neurons, each of which communicates to other $\sim 10^4$ neurons



- Neurons receive input signals and accumulate voltage. After some threshold they will fire spiking responses.

credit: www.moleculardevices.com, <http://cs231n.github.io/neural-networks-1/>

Simplified model of an artificial neuron



Simplified model of an artificial neuron

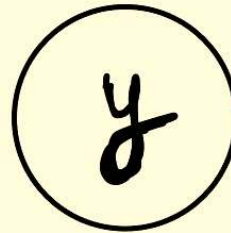
$x_2 = \text{Price}$

$$w_2 = \frac{-1}{1000}$$

$$b = 0$$

$x_2 = \text{Brand}$

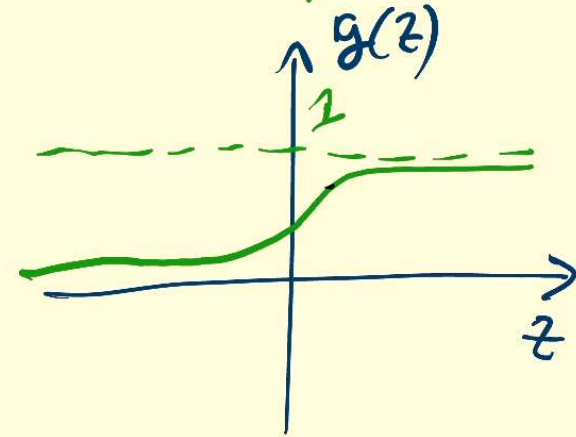
$$w_2 = 1$$



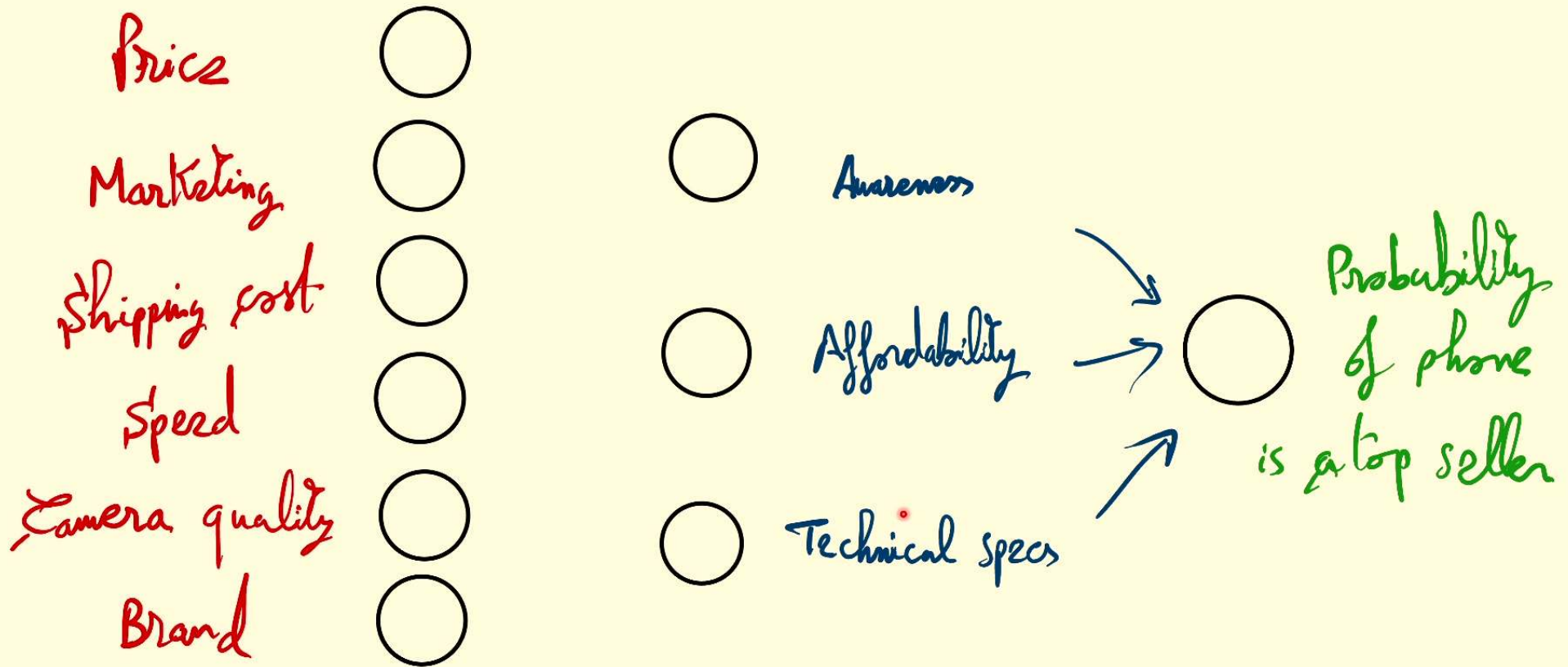
Probability of
phone
is a top seller.

① $x_1 = 500, x_2 = 1$

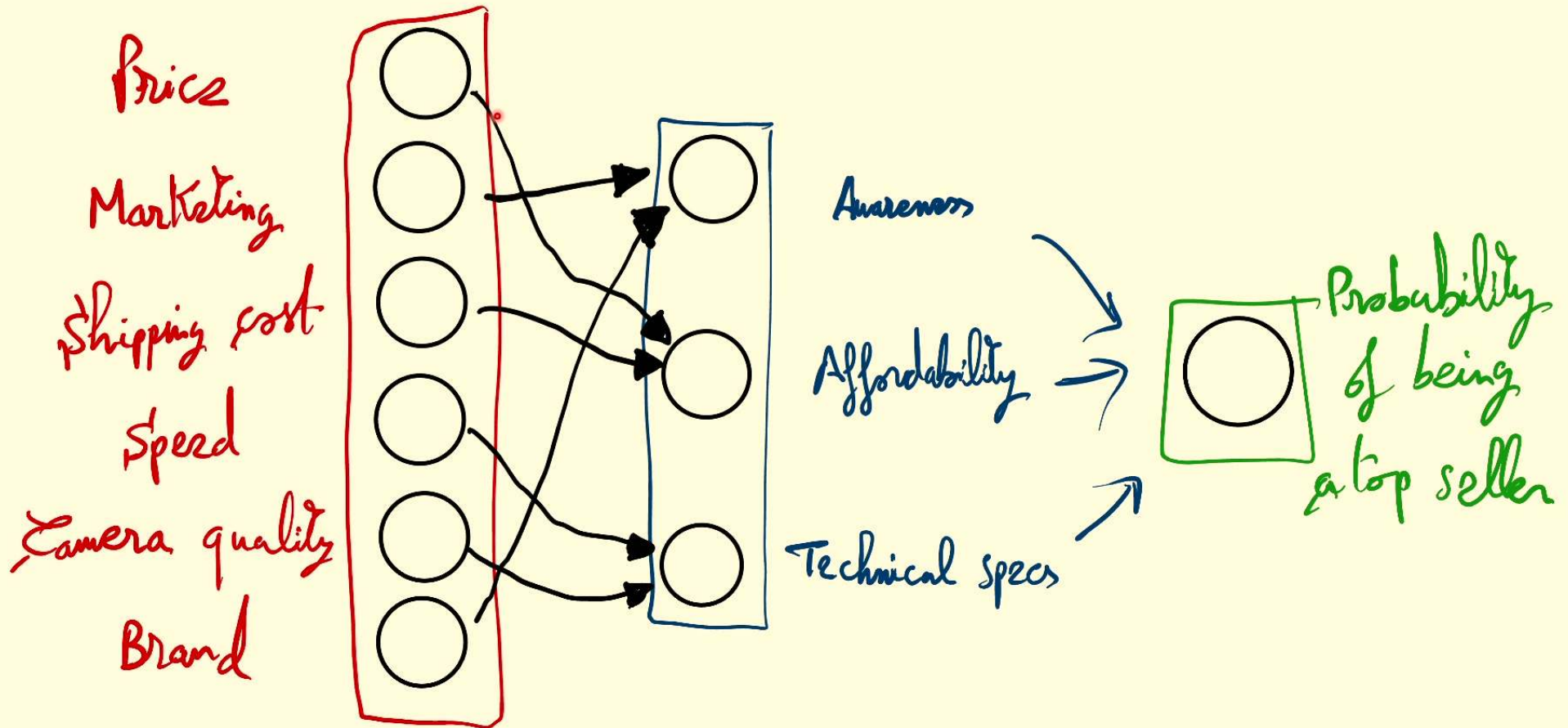
② $x_1 = 250, x_2 = 0$



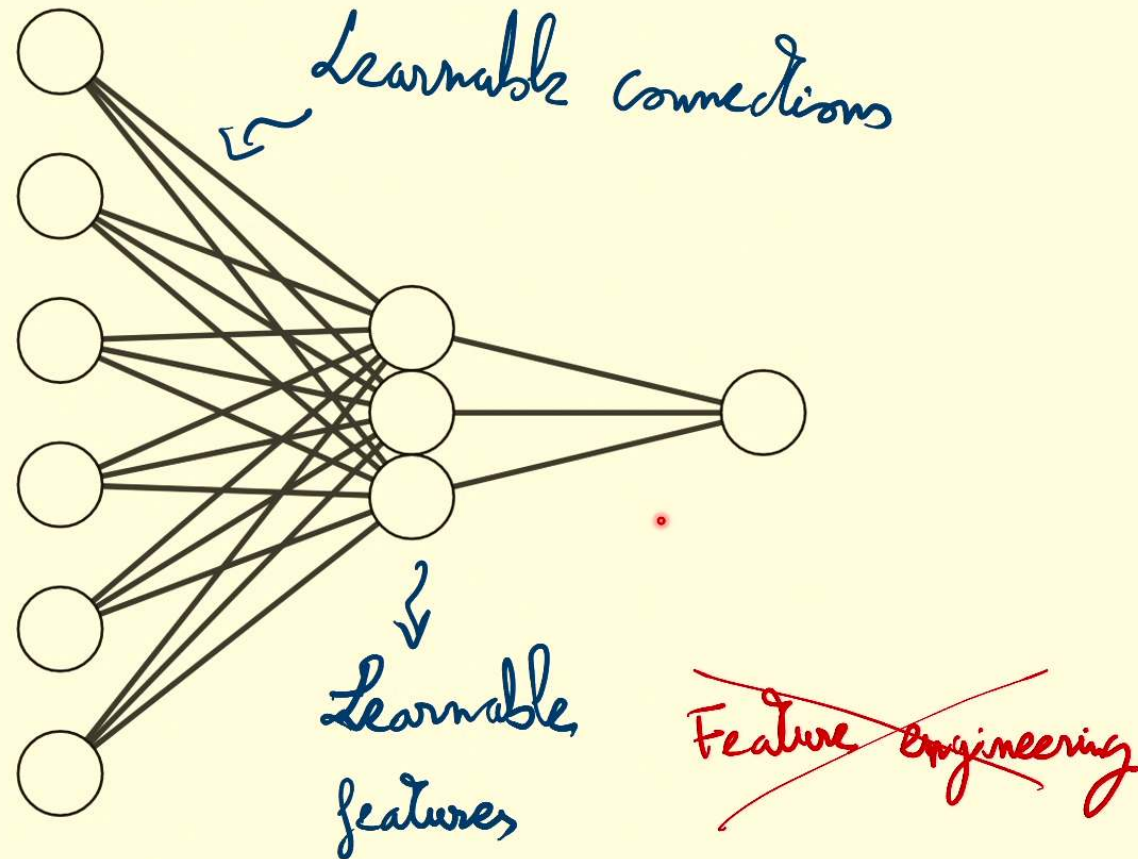
Simplified model of artificial neurons



Notion of a layer

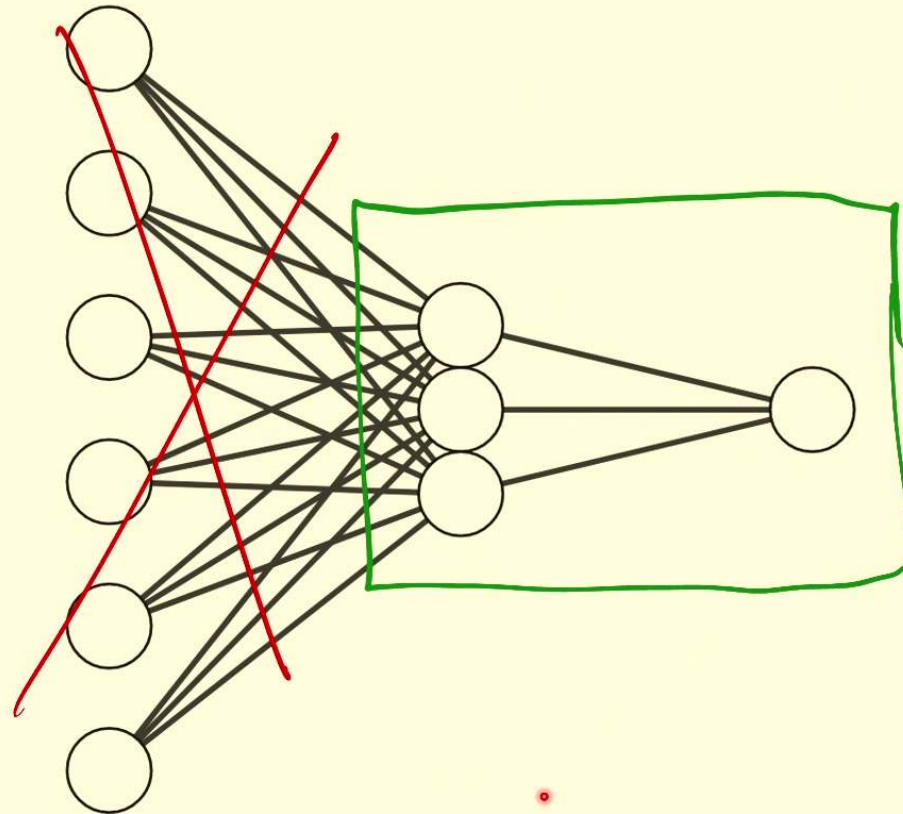


Feed-forward neural networks

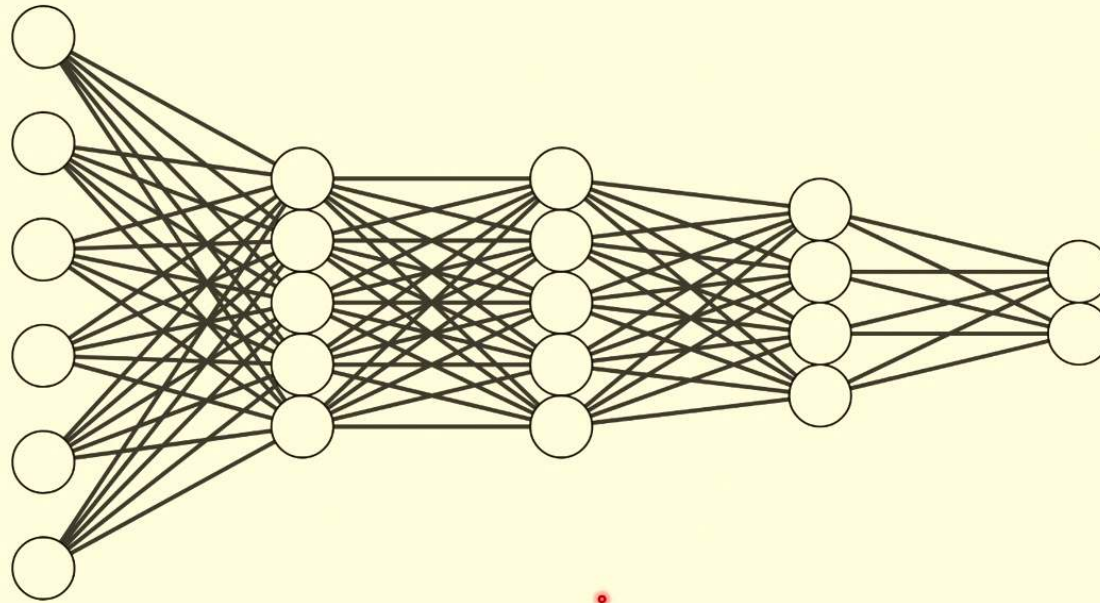


Credit: <https://alexlenail.me/NN-SVG/index.html>

Special case: logistic regression

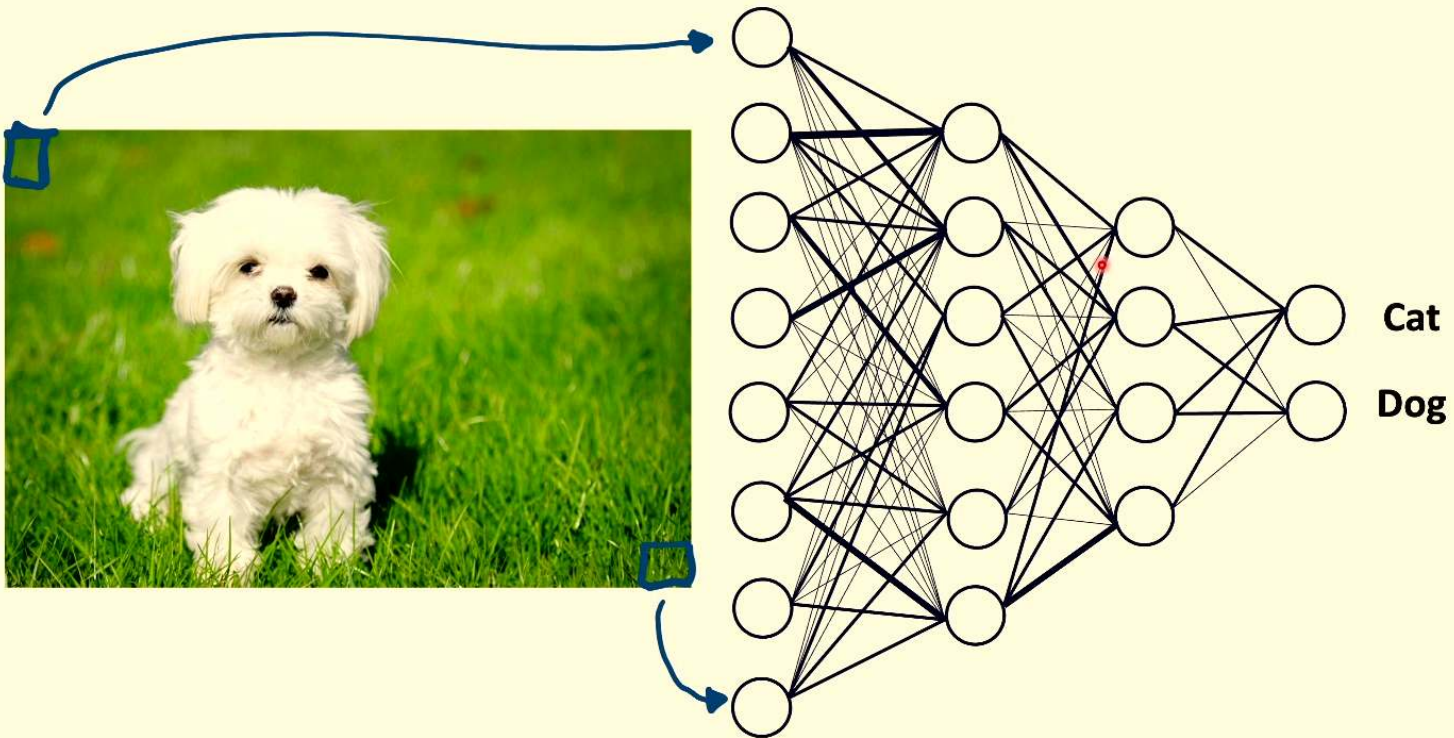


Deep feed-forward neural networks

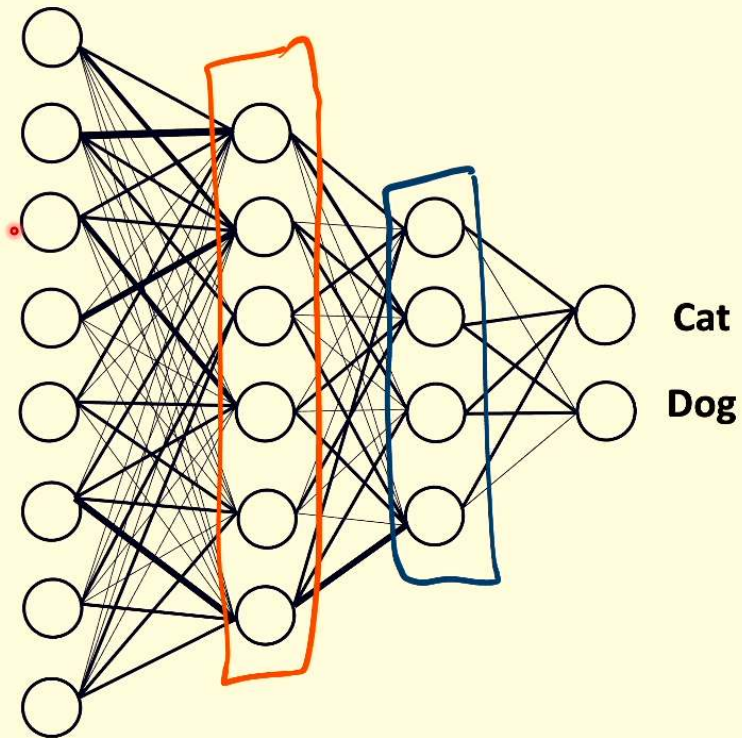
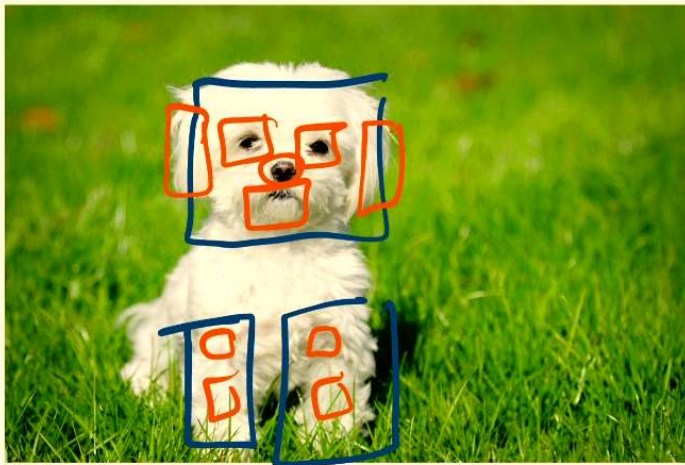


Neural networks can reach 100s of layers.

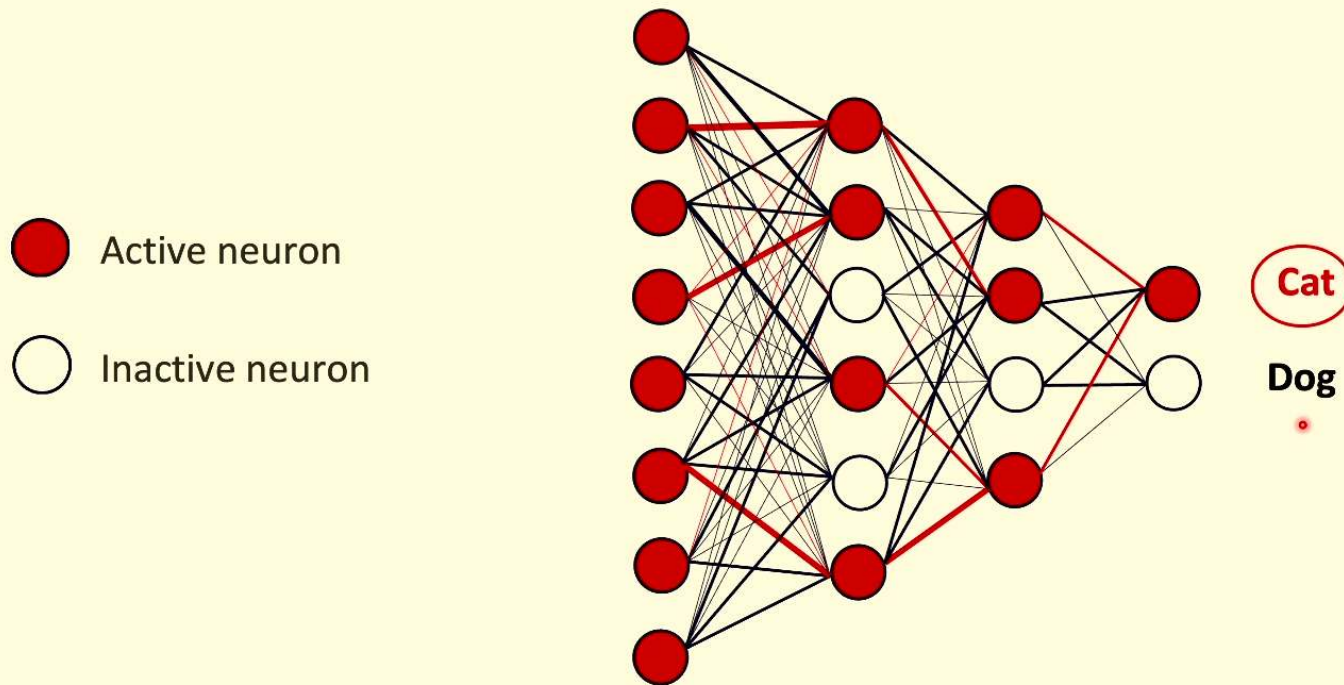
Choose a neural network



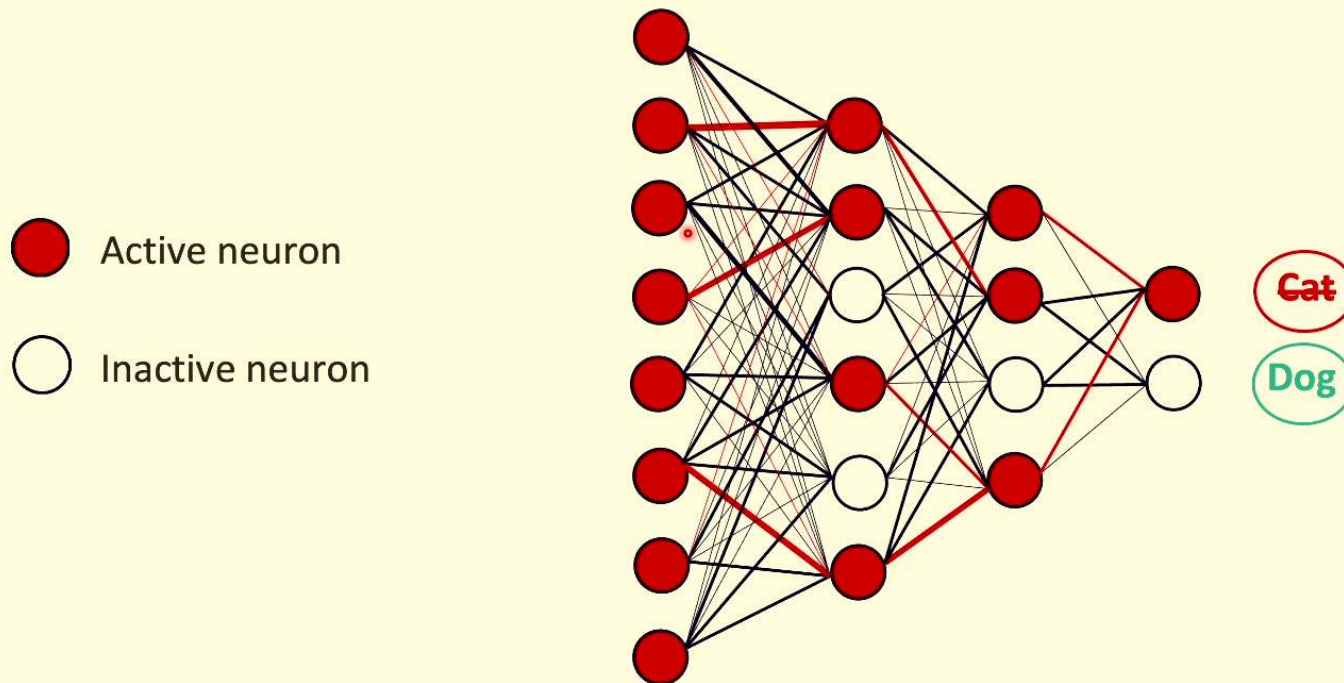
Expected goal of hidden layers: learning features



Neural network: **“It is a cat!”**



Neural network: **“It is a cat!”**

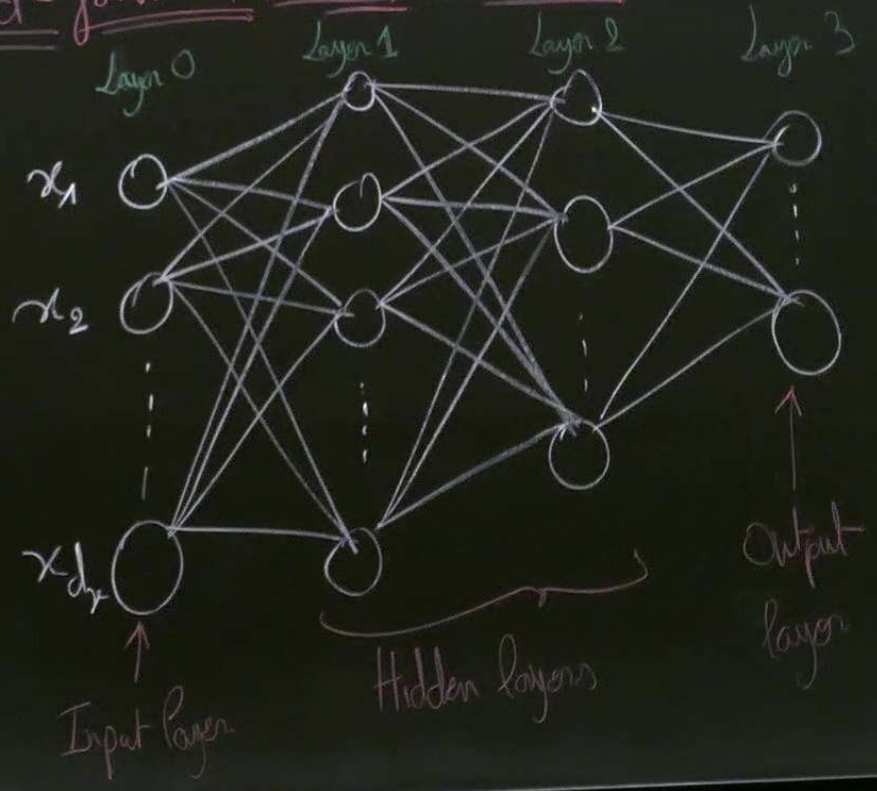


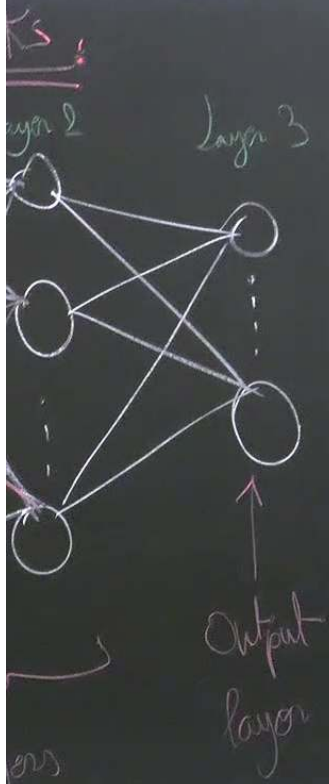
Let's switch to the blackboard to learn the math of Feed-forward NNs



FNNs)

Feed-forward Neural Networks



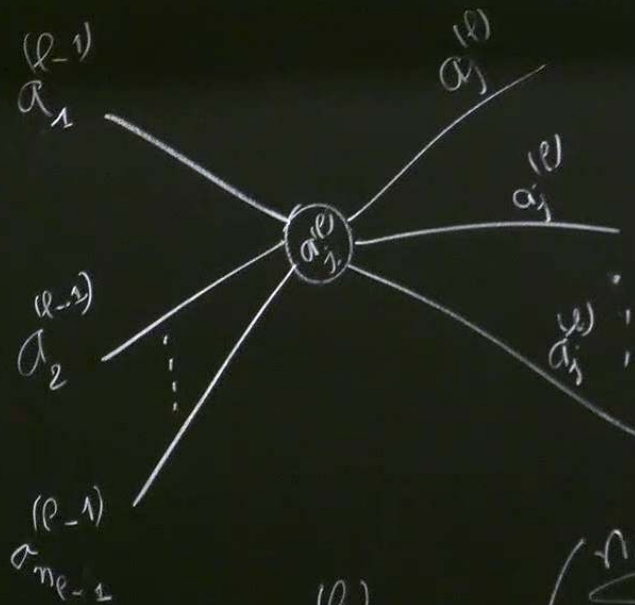


○ is called a "neuron"

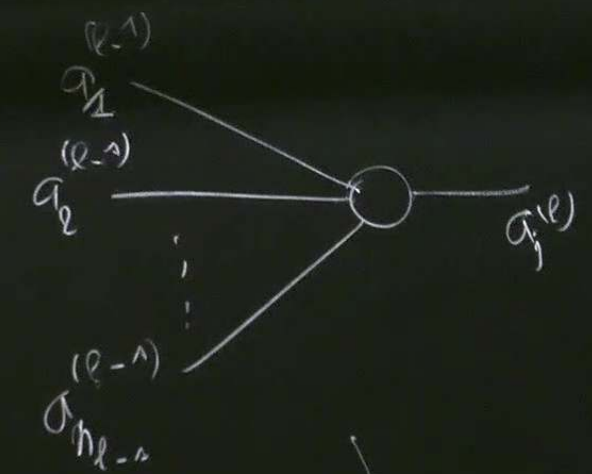
Notation:

- * $n_l \equiv$ "# of neurons in layer l "
- * $L \equiv$ largest value of l ($L=3$)
- * $a_j^{(l)} \equiv$ output from the j th neuron in layer l

- * $n_0 = d_x$
- * $n_L = d_y$
- * $a_j^{(0)} = x_j \quad (1 \leq j \leq d_x)$

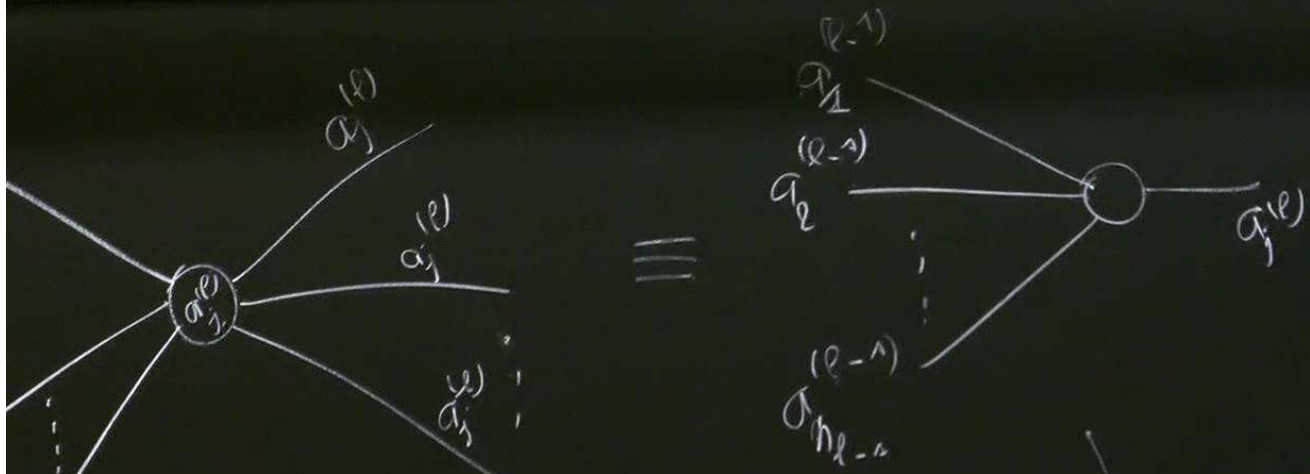


\equiv



$$a_j^{(l)} = g \left(\sum_{i=1}^{n_l-1} \dots \right)$$

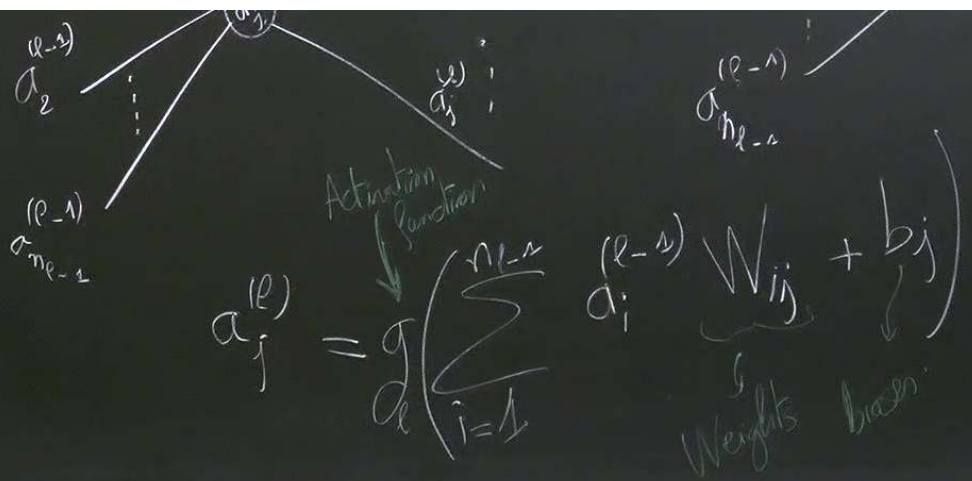
$$a_i^{(l-1)} (W_{ij} + b_j)$$



Activation function

$$a_j^{(l)} = g \left(\sum_{i=1}^{n_{l-1}} \dots \right)$$

$$a_i^{(l-1)} \left(\underbrace{W_{ij}}_{\text{Weights}} + \underbrace{b_j}_{\text{bias}} \right)$$



$$\vec{a}^{(l)} = (a_1^{(l)}, a_2^{(l)}, \dots, a_{n_l}^{(l)})$$

$$\vec{a}^{(l)} = f(\vec{x})$$

$$\vec{a}^{(l)} = g(\vec{a}^{(l-1)} W + \vec{b})$$

$$a_j^{(l)} = g \left(\sum_{i=1}^{n_{l-1}} a_i^{(l-1)} w_{ij}^{(l)} + b_j^{(l)} \right)$$

Weights bias

$$\vec{a}^{(l)} = (a_1^{(l)}, a_2^{(l)}, \dots, a_{n_l}^{(l)})$$

$$\vec{a}^{(l)} = f(\vec{x})$$

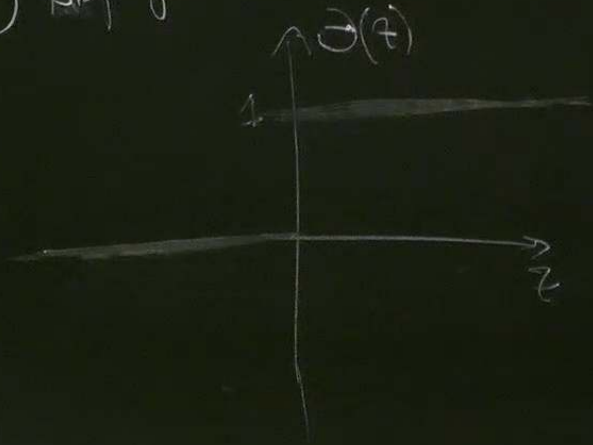
$$\vec{a}^{(l)} = g \left(\vec{a}^{(l-1)} W^{(l)} + \vec{b}^{(l)} \right)$$

$$W = n_{l-1} \times n_l$$

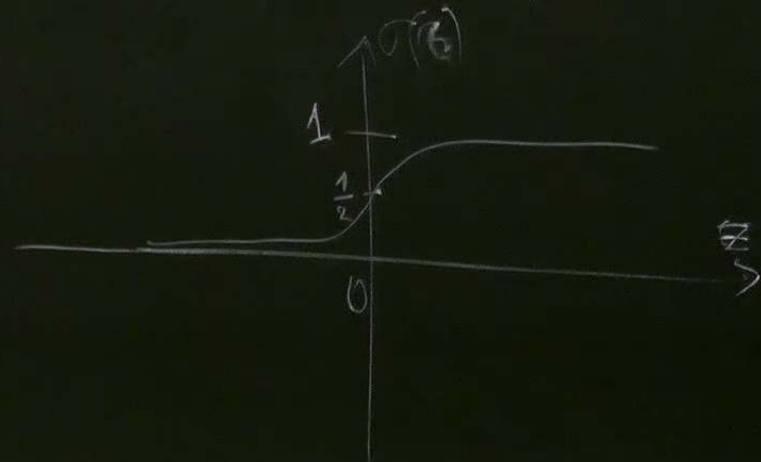
$$\vec{b} = n_l$$

Activation functions.

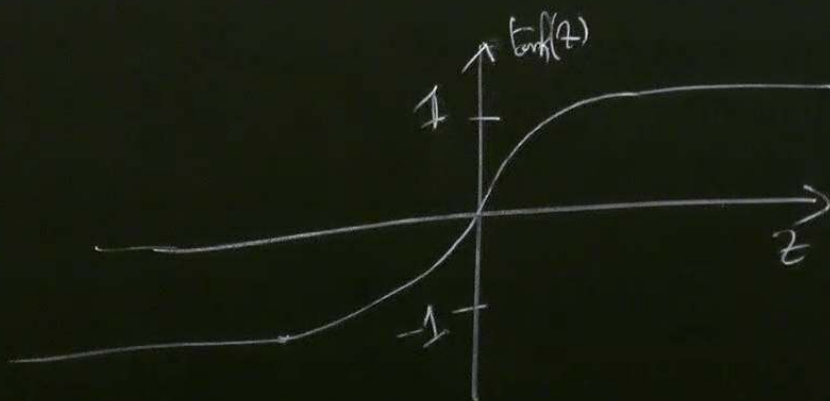
① Step function $\Theta(z)$



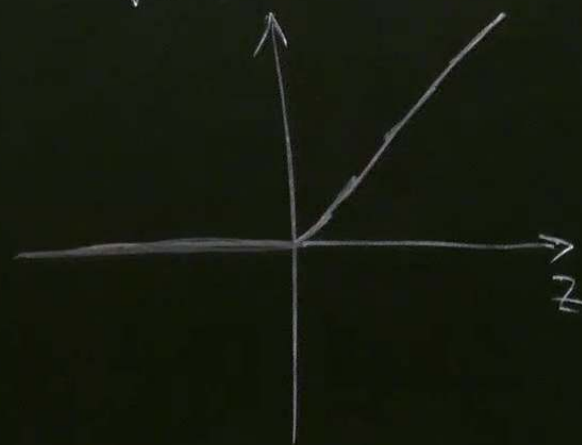
② Sigmoid $\sigma(z) = \frac{1}{1+e^{-z}}$

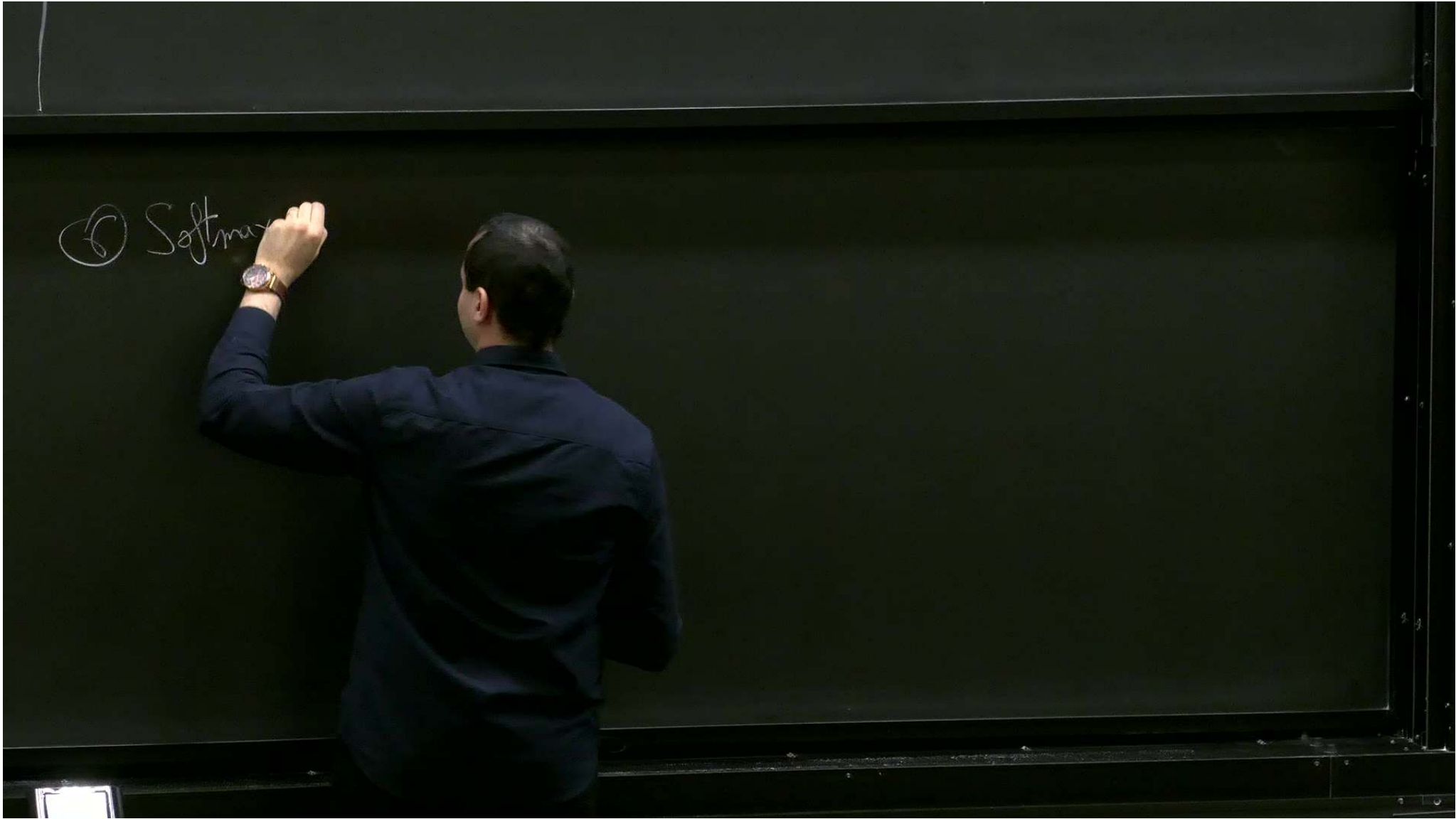


③ $\tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$



④ Rectified Linear unit
 $\text{ReLU} = \max(0, z)$





⑤ Softmax activation (Output layer)

$$P_i = \left[\text{Softmax} \left(\vec{z} \right) \right]_i = \frac{e^{z_i}}{\sum_{i=1}^n e^{z_i}}$$

(z_1, z_2, \dots, z_n)

$$\sum_i P_i = 1$$

One-hot encoding:

$$0, 1 \rightarrow \vec{y} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

Cost Functions

① Mean-Squared error (MSE)

$$J_{MSE} = \frac{1}{|D|} \sum_{\vec{x} \in D} \sum_{i=1}^{n_c} (a_i^{(L)}(\vec{x}) - y_i(\vec{x}))^2$$

② Cross Entropy (CE)

$$J_{CE} = \frac{1}{|D|} \sum_{\vec{x} \in D} \sum_{i=1}^{n_c} \left[-y_i(\vec{x}) \log(a_i^{(L)}(\vec{x})) - (1 - y_i(\vec{x})) \log(1 - a_i^{(L)}(\vec{x})) \right]$$

$$0 = \frac{\partial}{\partial a} \left[-y \log(a) - (1-y) \log(1-a) \right] \rightarrow a = y$$