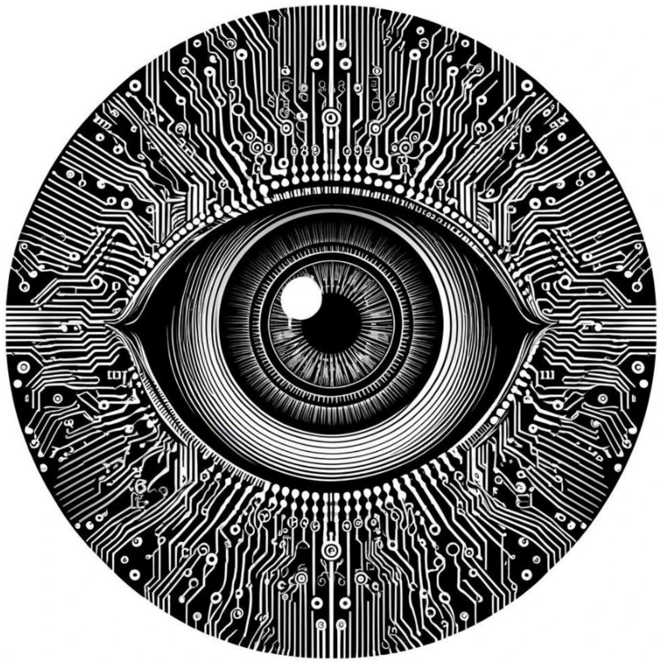
KI Service Zentrum
by Hasso-Plattner-Institut

Introduction to Neural Networks



Antonio Rueda-Toicen

SPONSORED BY THE

Federal Ministry
of Education
and Research

Learning goals

- Explain how neural networks learn during training
- Identify the components of input units, hidden layers, and output units
- Apply concepts of weights, gradients and loss
- Understand the purpose of validation and test data

What is an artificial neural network?

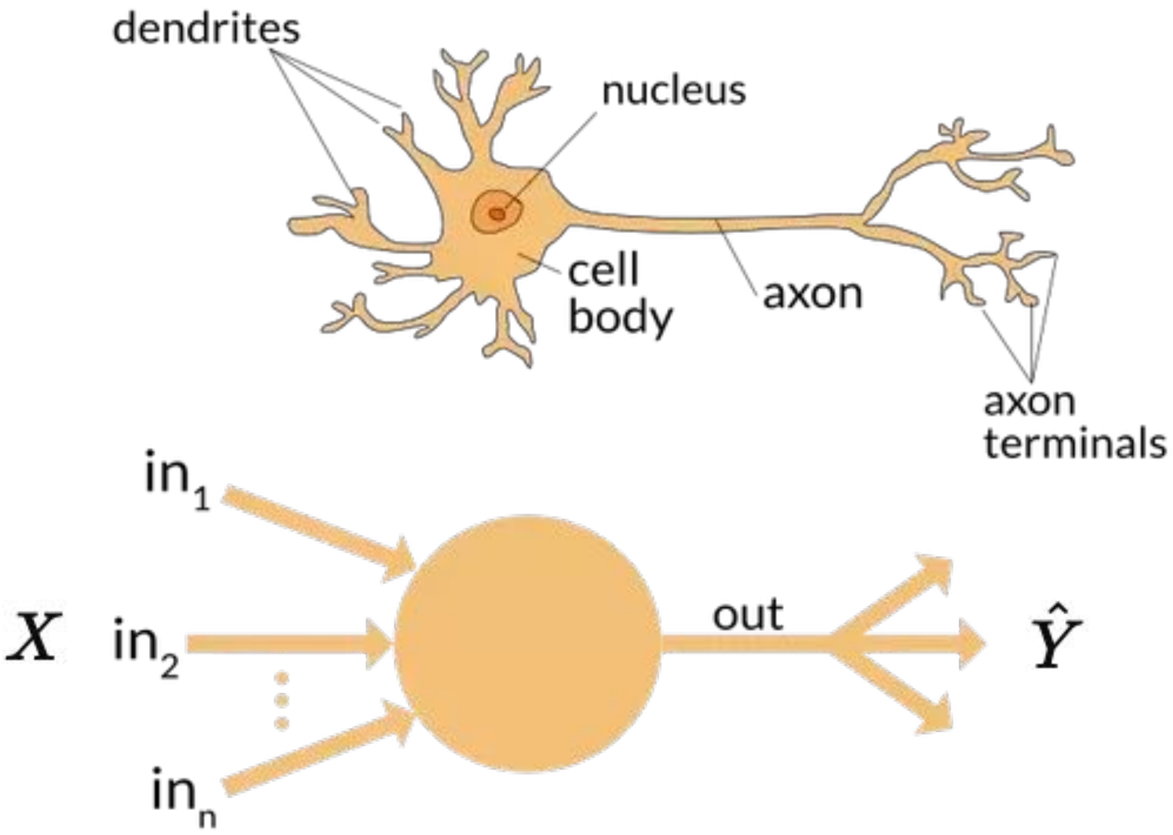
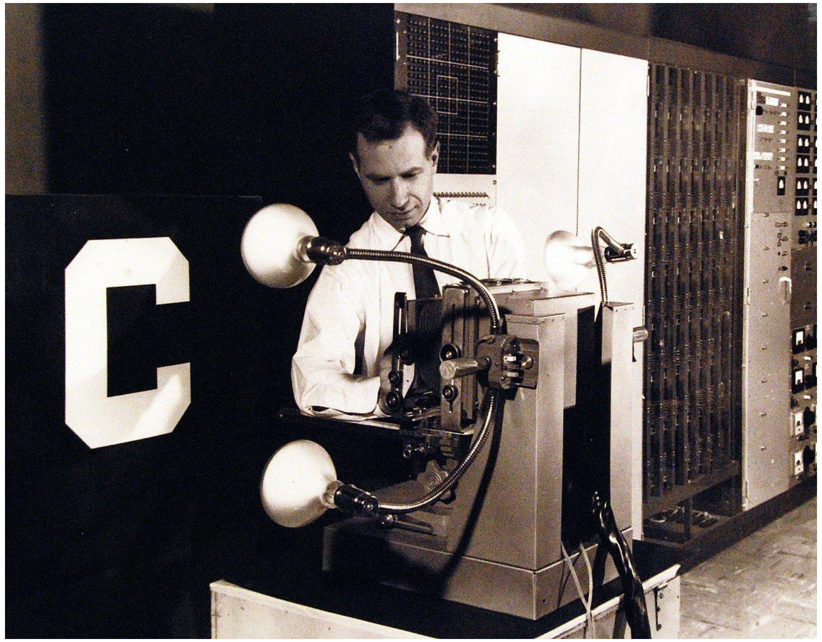


Image from source

- Artificial “neurons” are also called “processing units”
- We can think of them as functions that map X inputs to Ŷ outputs

The perceptron



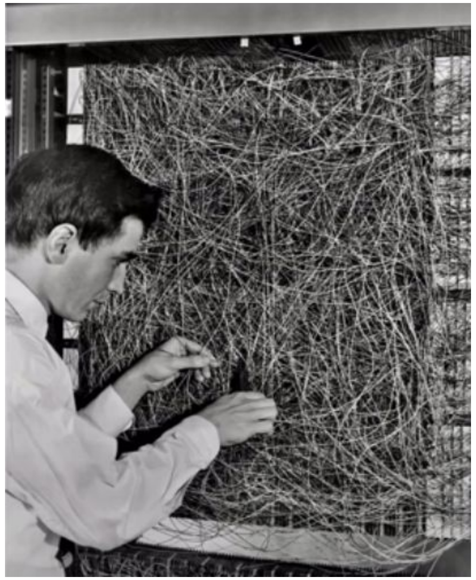
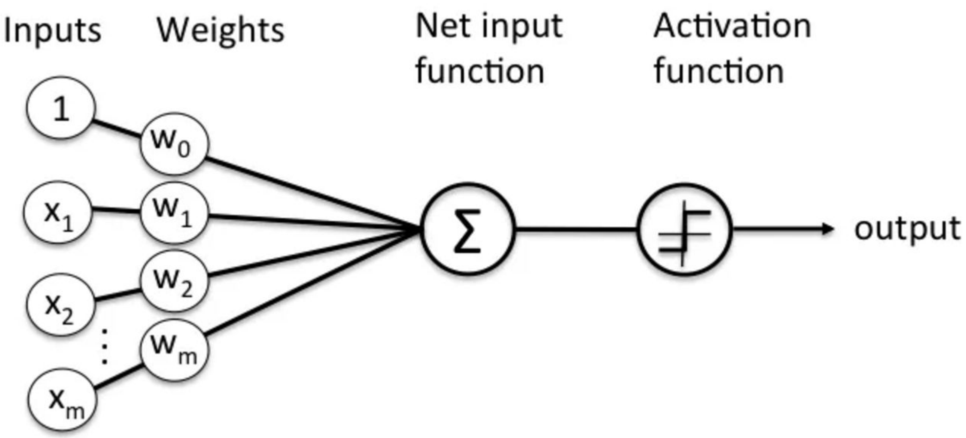


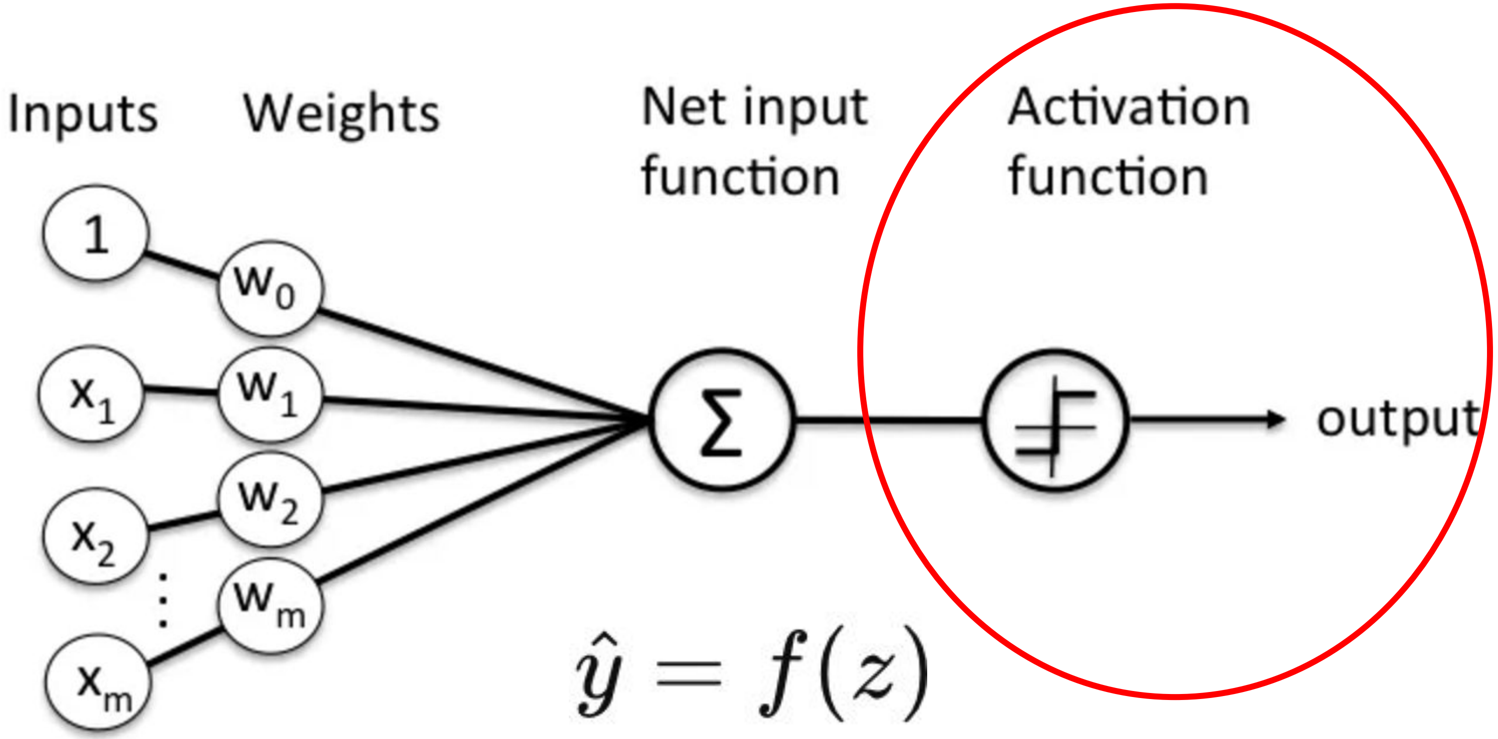
Image from source



$$\hat{y} = f\left(\sum_{j=0}^{m} w_j x_{ij}\right)$$

Images: Mark I Perceptron at Cornell University (1961) source

A modern fully connected layer

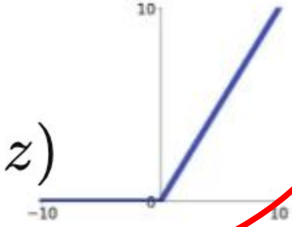


$$\hat{y} = f(z)$$

$$z = \sum_{i=0}^{m} w_i x_i$$

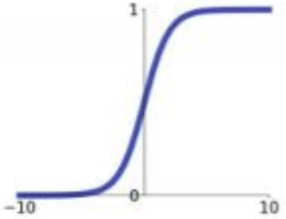
ReLU

$$f(z) = \max(0, z)$$



Sigmoid

$$f(z) = \frac{1}{1 + e^{-z}}$$



Making a perceptron learn the price of a vintage car

training

1964 Chevrolet Corvette Stingray

Image source

y = 83999 EUR (real price, target value)

ŷ = 84000 EUR (a possible prediction)

Absolute error = |ŷ - y | = 1 EUR

validation

Image source

testing

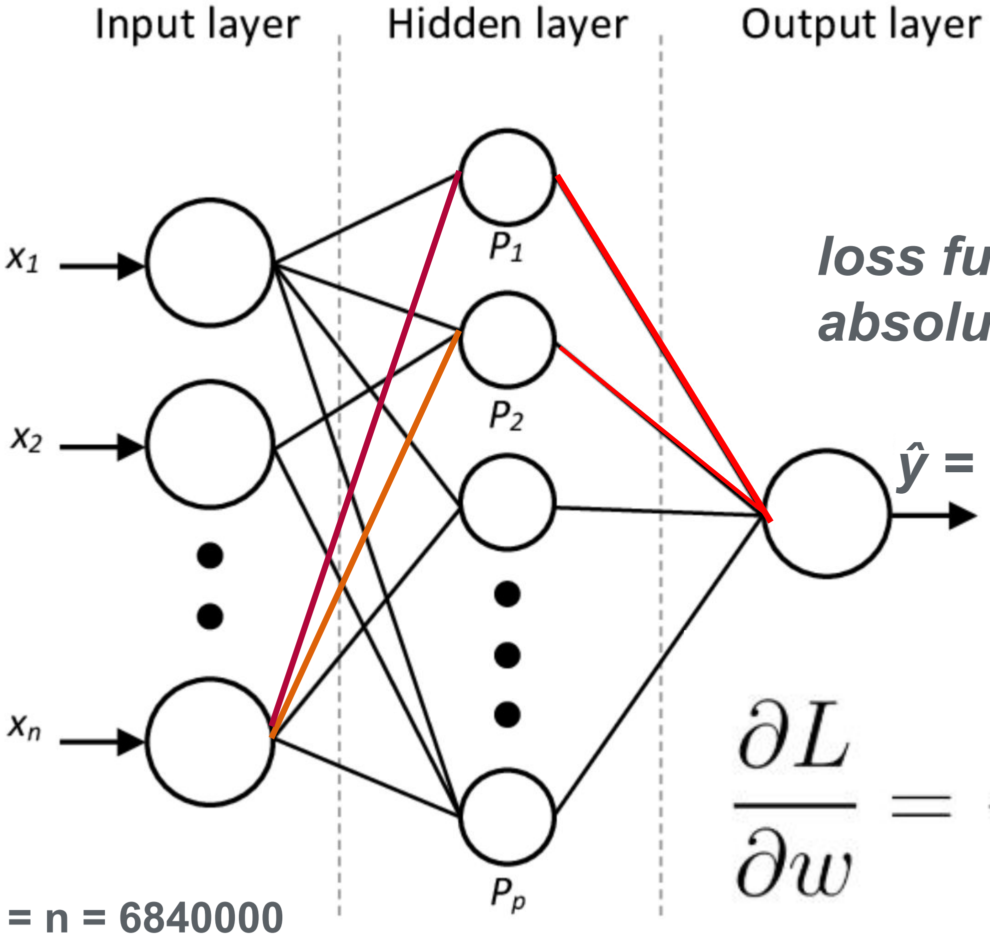
Image source

Training a fully connected network (multilayer perceptron)

y = 83999 EUR

H = 1900
W = 1200
C = 3 (RGB)

C * H * W = n = 6840000

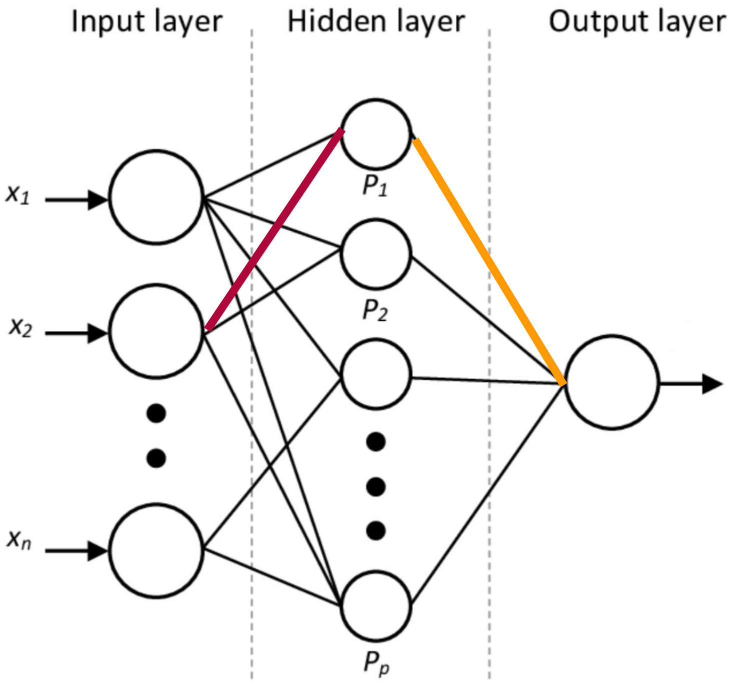


loss function =
absolute error = $|\hat{y} - y|$ = 12188

$\hat{y} = f(z)$ = 71811

$$\frac{\partial L}{\partial w} = -\operatorname{sign}(y - f(x))\frac{\partial f(x)}{\partial w}$$

Understanding network weights and their updates



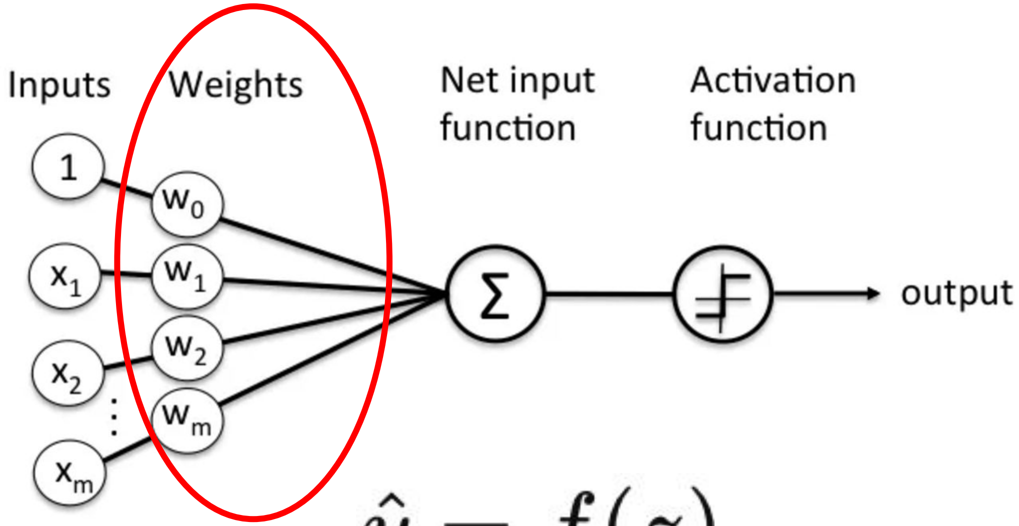
w_{ij} = weight from neuron i to neuron j

learning_rate = step size for updates, a constant like 0.01

L = loss function

$$w_{ij} = w_{ij} - (\text{learning rate} * \frac{dL}{dw_{ij}})$$

Modern perceptrons use loss functions as feedback



$$\hat{y} = f(z)$$

$$z = \sum_{i=0}^{m} w_i x_i$$

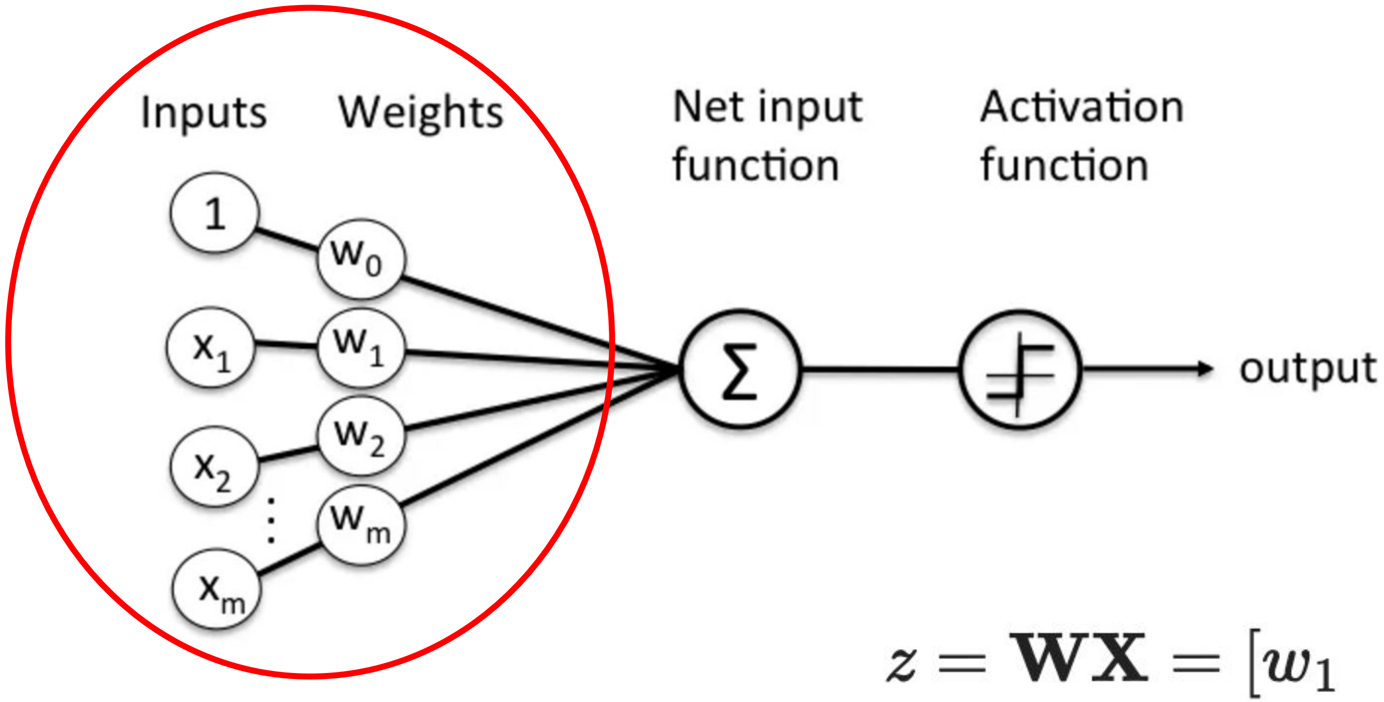
Regression

$$\text{Mean Absolute Error} = \frac{1}{N}\sum_{i=1}^{N} |y_i - \hat{y}_i|$$

Classification

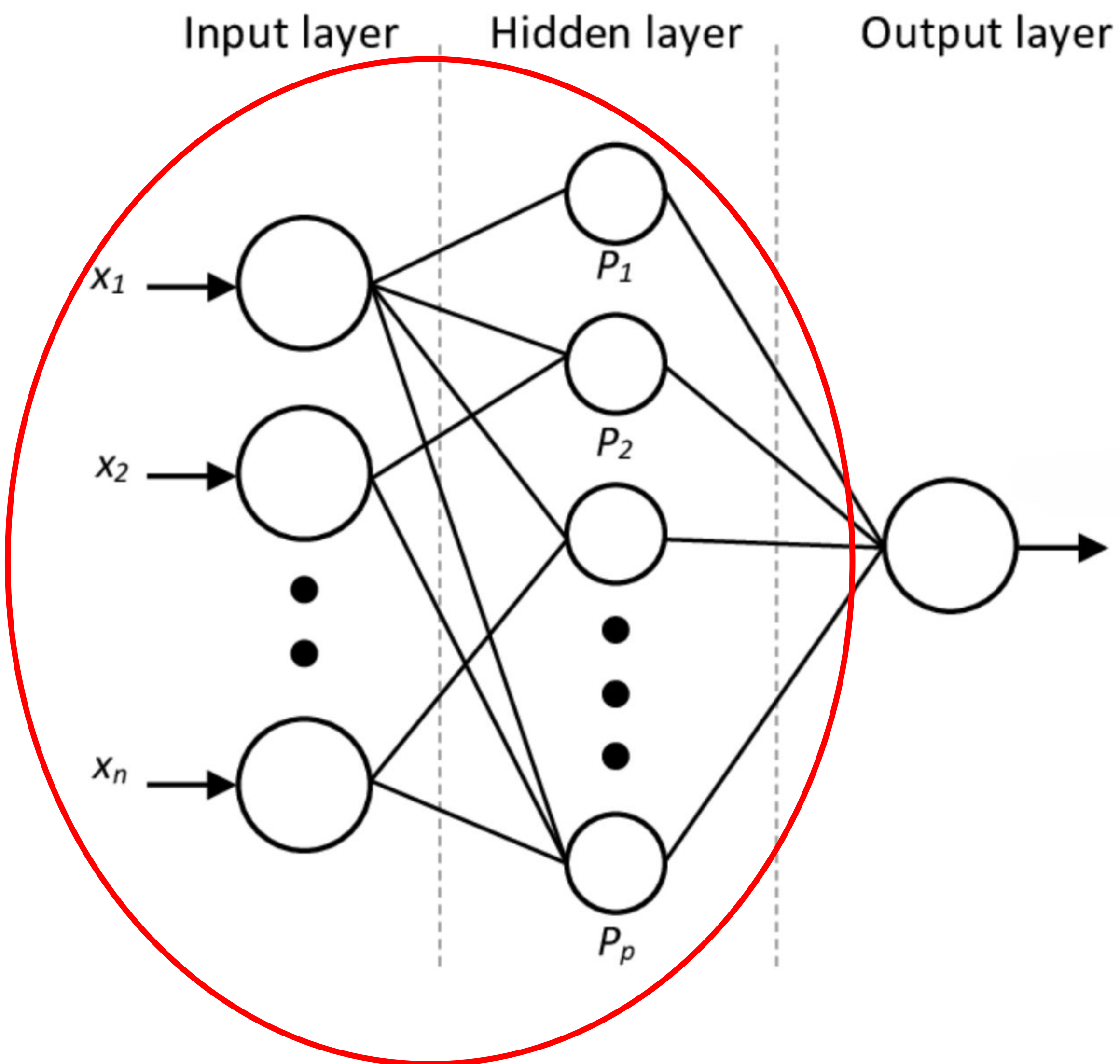
$$\text{Cross Entropy} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C} y_{i,c}\log(\hat{y}_{i,c})$$

Matrix multiplication: features times weights



$$z = \mathbf{WX} = \begin{bmatrix} w_1 & w_2 & \cdots & w_m \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{bmatrix} = \sum_{i=1}^{m} w_i x_i$$

Dimensions of the weight matrix define the output's

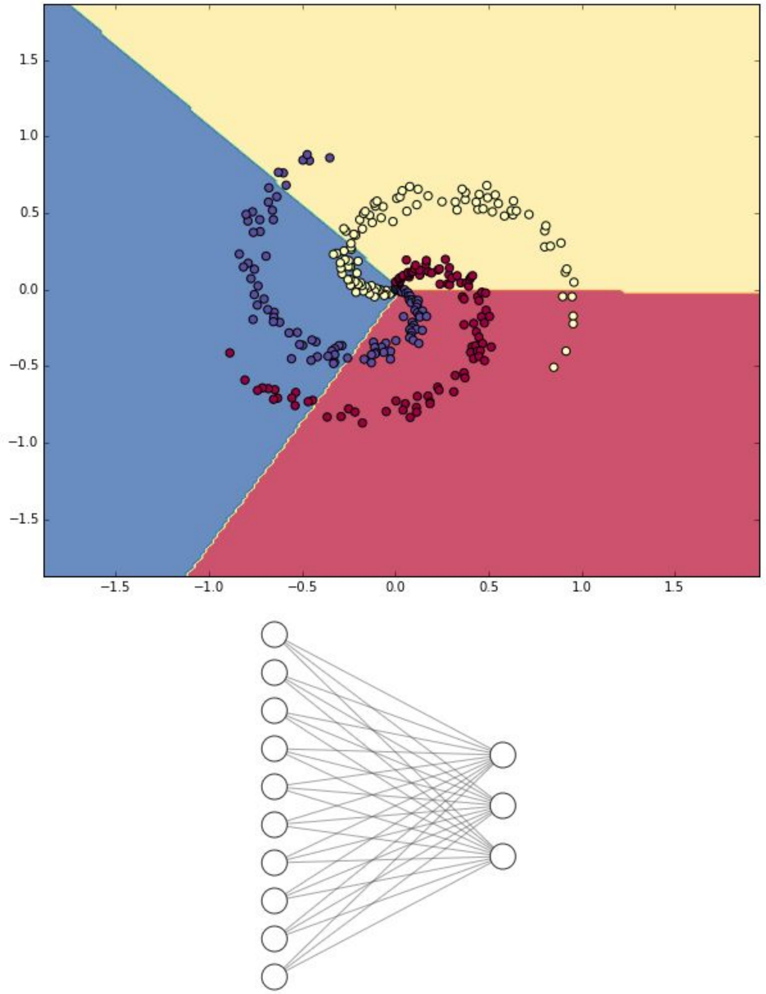


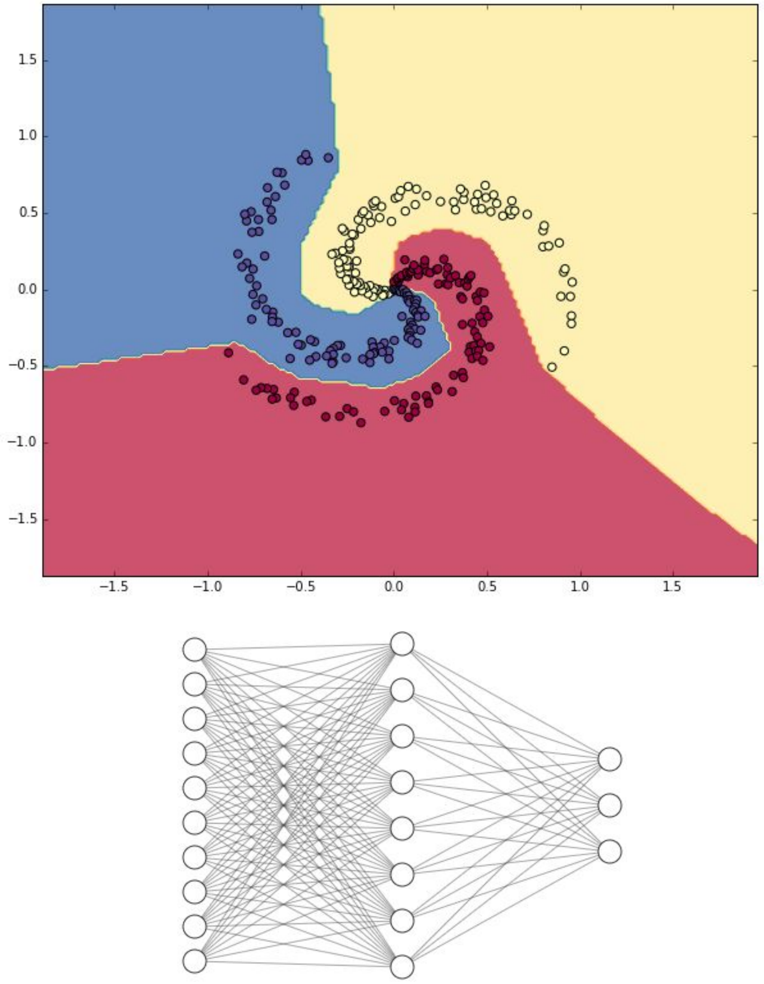
$$P = WX$$

$$\begin{bmatrix} p_1 \\ p_2 \\ \vdots \\ p_p \end{bmatrix} = \begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1n} \\ w_{21} & w_{22} & \cdots & w_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ w_{p1} & w_{p2} & \cdots & w_{pn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

For example: $p = 4 \quad n = 3$

Why do we add hidden layers?





Validating and testing the model

sample to decide when to stop training: validation

***y* = 82999**
Image source

sample to use as a final benchmark: test

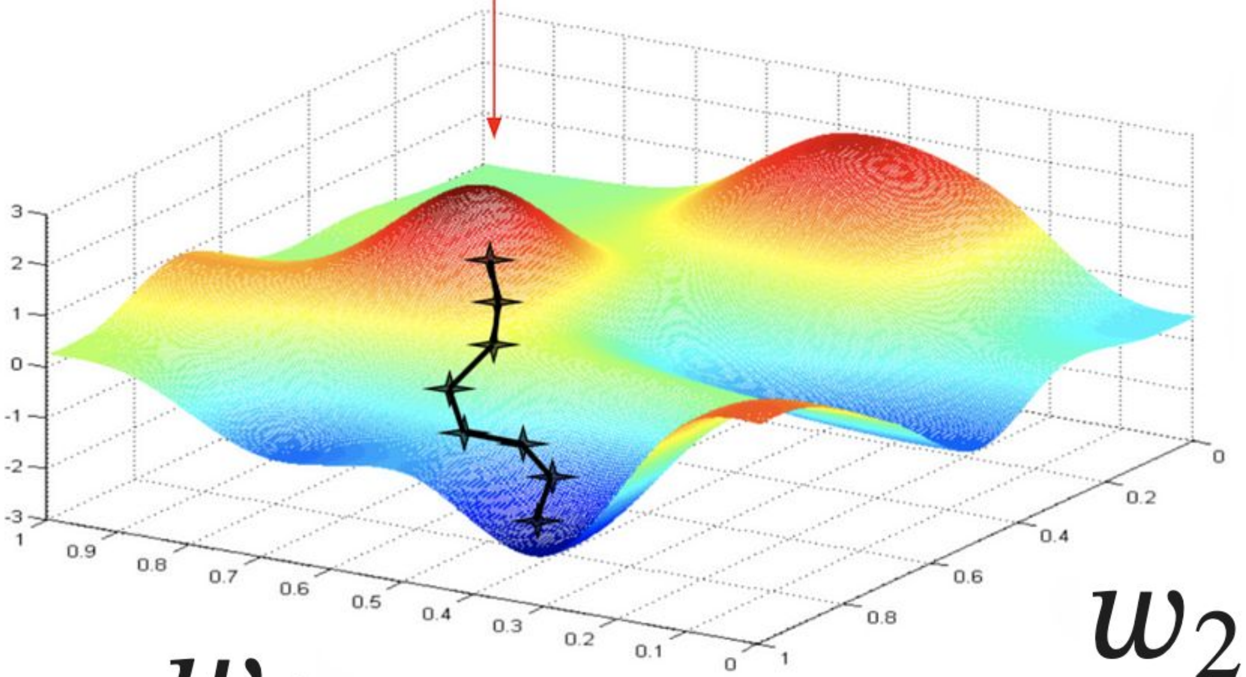
***y* = 82699**

Image source

Gradient descent

we start at a random position in the loss landscape

$L(w_1, w_2)$



w_1

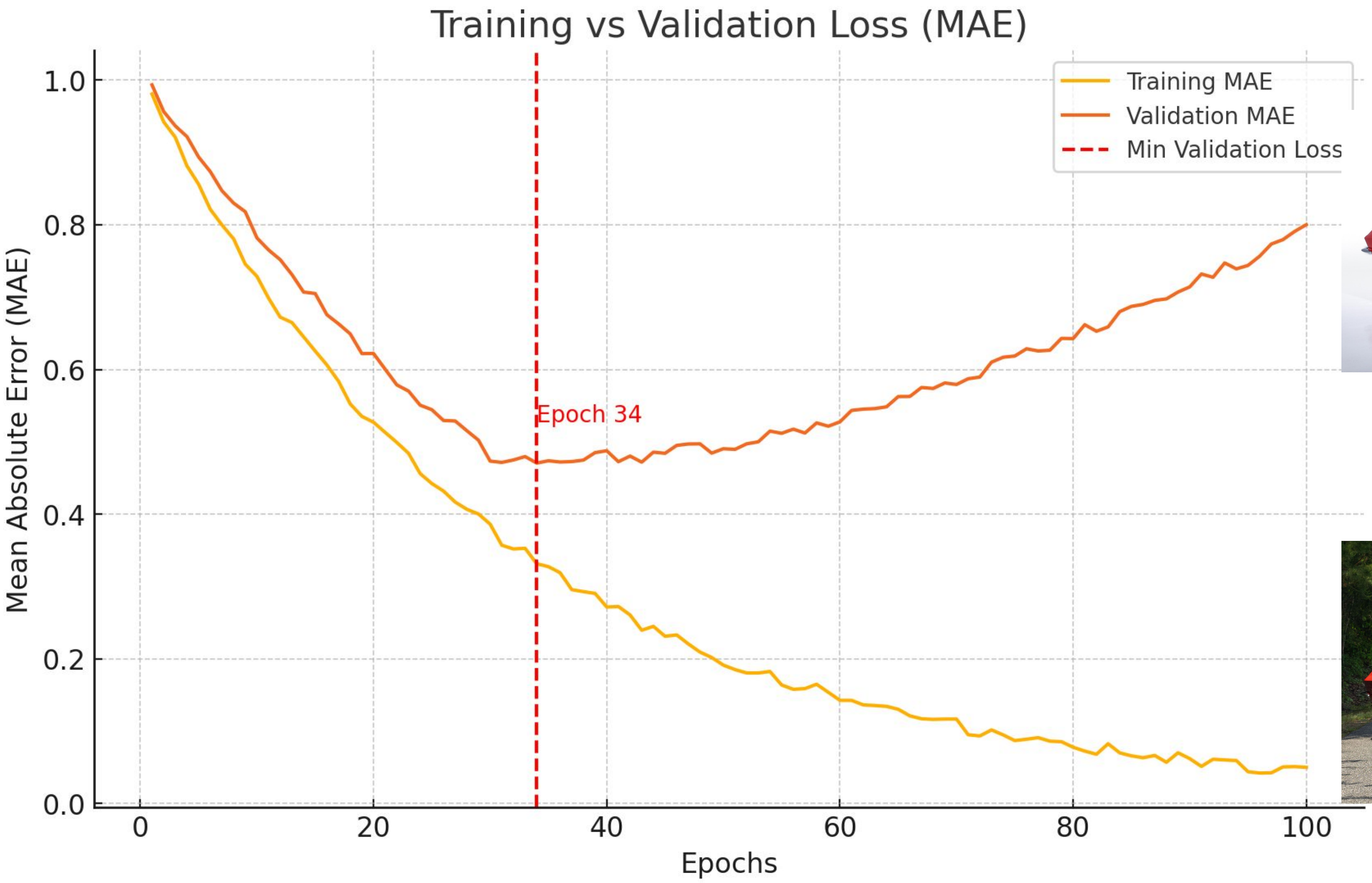
w_2

training image
role: update weights

$$w_{ij} = w_{ij} - (\text{learning rate} * \frac{dL}{dw_{ij}})$$

validation image
role: evaluate best weights

Training vs validation loss



The test set gives the benchmark of generalization

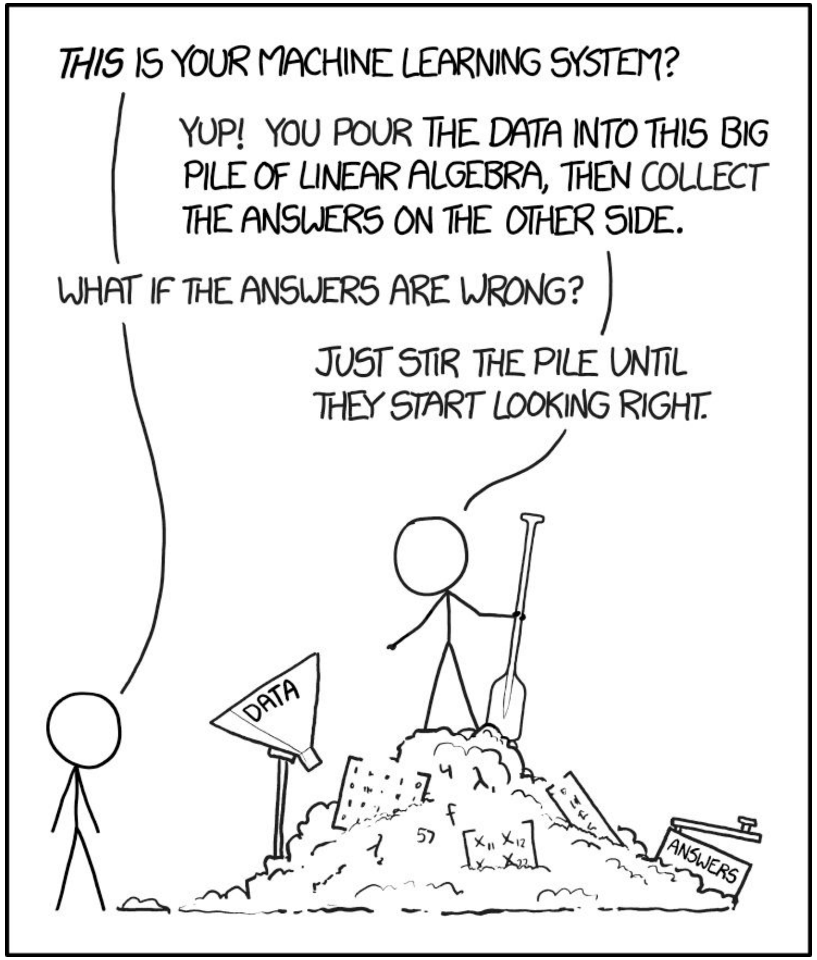


Image from https://xkcd.com/1838/

y = 82699
ŷ = 84015
test set error = |y - ŷ| = 1316

Summary

Neural networks are <u>simplified</u> mathematical models of biological neurons

- They are functions with trainable parameters (aka ‘weights’) that we stack in layers with variable number of processing units (aka ‘neurons’)
- The size of our weight matrices define the dimensions of our hidden and output layers

We use loss functions and gradients to update the weights of a network

- Loss functions provide feedback to the model
- The weight update is an iterative process (gradient descent)

Deciding on training settings (aka “hyperparameters”) is an explorative process

- We use validation sets to decide how and when to stop training. We use tests to provide final benchmarks

SPONSORED BY THE
Federal Ministry of Education and Research

Further reading

On the origin of neural networks

- https://nautil.us/the-man-who-tried-to-redeem-the-world-with-logic-2885/

Annotated history of modern AI and deep learning by Jurgen Schmidhüber

- https://people.idsia.ch/~juergen/deep-learning-history.html

Frank Rosenblatt's Perceptron

- https://news.cornell.edu/stories/2019/09/professors-perceptron-paved-way-ai-60-years-too-soon

SPONSORED BY THE

Federal Ministry
of Education
and Research