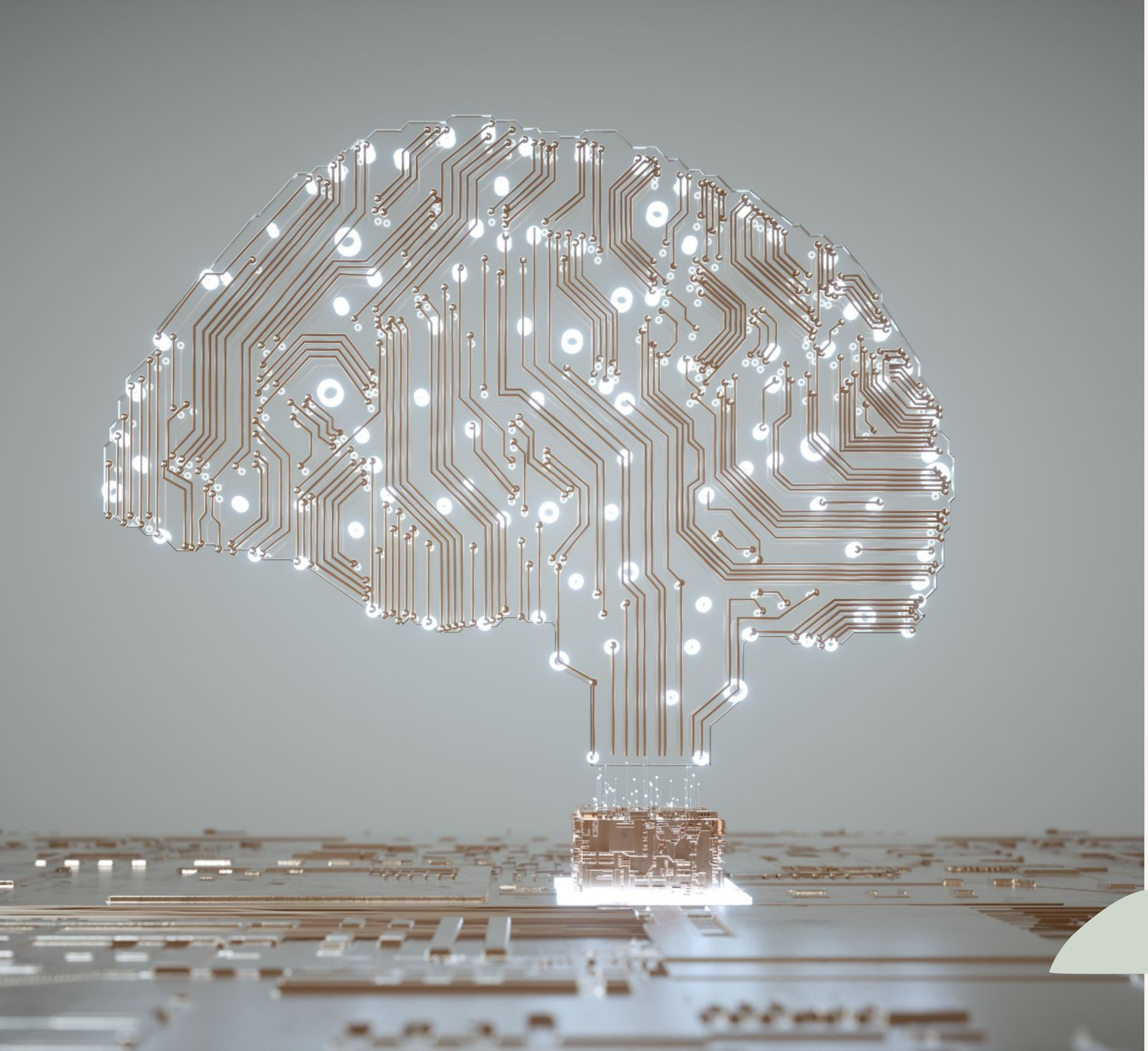


LOGISTIC REGRESSION *for Image Classification*

CREATIVE PROGRAMMING AND COMPUTING Course

Prof. Massimiliano Zanoni

Slides and Presentation by

Student Marco Muraro

POLITECNICO MILANO 1863

MAE
MASTER OF SCIENCE MUSIC AND ACOUSTIC ENGINEERING

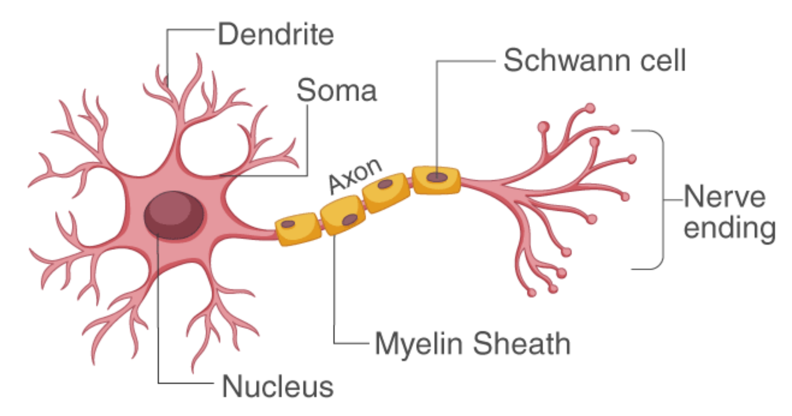
NEURON

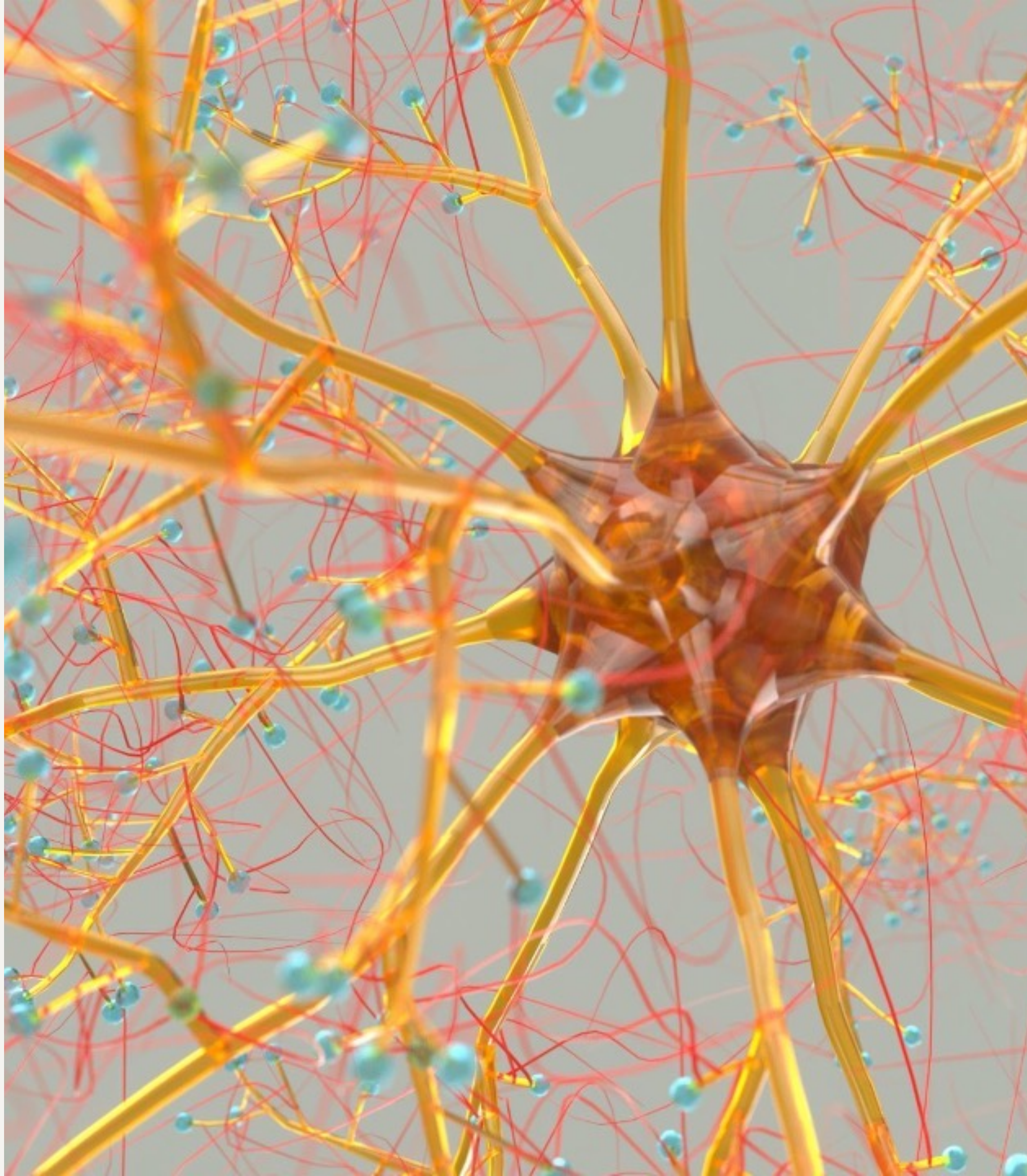
The **brain** is a **network of neurons**

Dendrites collect electrical stimuli coming from other neurons and accumulate input charges

Activation Process

The neuron fires a new electrical impulse when the total charge exceeds a certain threshold



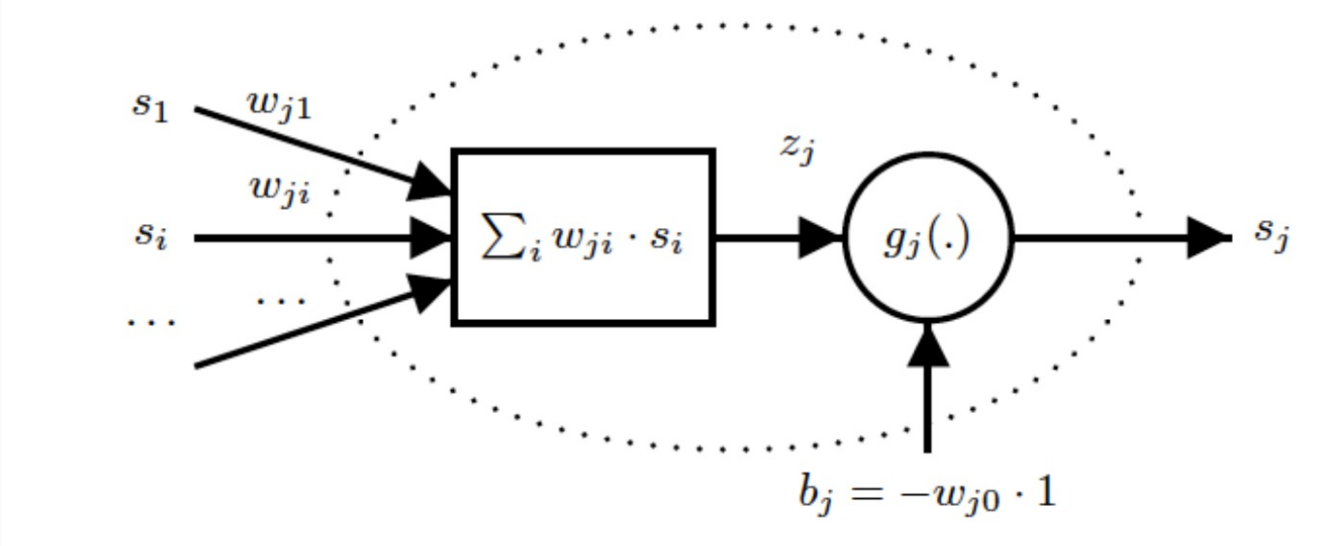


PERCEPTRON

Feed-Forward Model

$$s_j = g_j(z_j - b_j) = g_j\left(\sum_{i=1}^{N} w_{ij}s_i + w_{j0}s_0\right) \quad \rightarrow \quad \boldsymbol{s_j = g_j\left(\sum_{i=0}^{N} w_{ij}s_i\right)}$$

Output



Activation Value $z_j = \sum_{i=1}^{N} w_{ij}s_i$

Activation Function $g_j(\cdot)$

Activation Threshold $b_j = -w_{j0}s_0 = -w_{j0}$

BINARY CLASSIFICATION

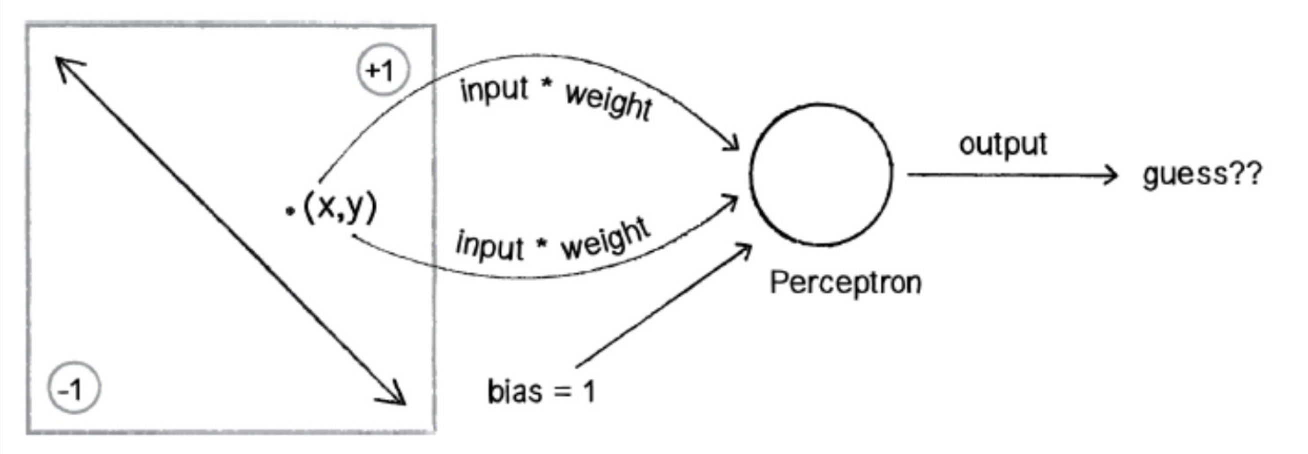
Linear Separation

Decision boundary is defined as a straight line

$$x_1 w_1 + x_2 w_2 + w_0 = 0$$

$$\Downarrow$$

$$x_2 = -\frac{w_1}{w_2} x_1 - \frac{w_0}{w_2}$$



The **learning process** consists in **learning the weights giving the best linear decision boundary**

The perceptron keeps learning by mistakes and updates the weights to find the best ones

The **output** is a **class** in the classification problem

POLITECNICO
MILANO 1863

BINARY CLASSIFICATION

Learning Process

- Start with a **random guess for the weights**
- The perceptron **guesses an answer** $$s_j = g_j\left(\sum_{i=1}^{N} w_{ij} s_i - b_j\right) = g_j\left(\sum_{i=0}^{N} w_{ij} s_i\right)$$
- Compute the **error function**

  $\epsilon = \hat{s}_j - s_j$ where $\hat{s}_j$ is the **desired output** and s_j is the **estimated output** (guess)

- **Adjust the weights**

  HEBB LEARNING RULE $w_i^{n+1} = w_i^n + \Delta w_i$ $\Delta w_i = \eta \cdot \hat{s}_j \cdot s_i$

- **Repeat the process** until **stop condition** η is the **learning rate**

  The **number of iterations** is denoted as the **number of epochs**

LOGISTIC REGRESSION

for Binary Classification

Logistic Regression is the process of modelling the probability of a discrete outcome given an input variable.

The most common application of logistic regression is **binary classification**:
the **binary outcome** is TRUE/FALSE, YES/NO, -1/1, 0/1, etc.

Given a feature vector x,
logistic regression aims to find

$$\hat{\boldsymbol{y}} = \boldsymbol{P}(\boldsymbol{y} = \mathbf{1}|\boldsymbol{x})$$

→

Learning Process

Learning the weights w and the bias b giving the best estimate $\hat{y}$ for the probability $P(y = 1|x)$

$$\hat{\boldsymbol{y}} = \boldsymbol{g}(\boldsymbol{w}^{\boldsymbol{T}}\boldsymbol{x} + \boldsymbol{b})$$

LOGISTIC REGRESSION

for Binary Classification

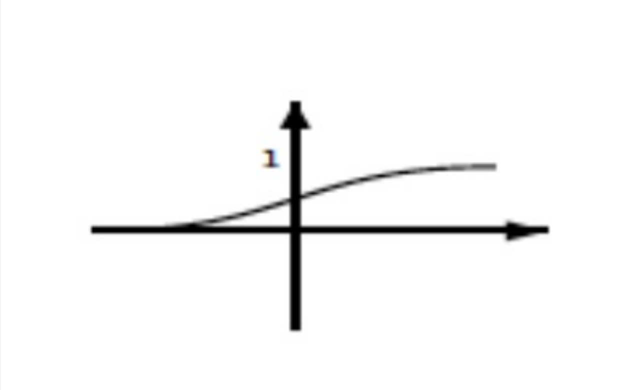
Activation function $g(\cdot)$ is applied

$$\hat{\boldsymbol{y}} = \boldsymbol{g}(\boldsymbol{w}^T\boldsymbol{x} + \boldsymbol{b})$$

We need a function mapping input data into a probability, i.e. within the range $[0,1]$

Sigmoid Function

$$\boldsymbol{g}(\boldsymbol{z}_j) = \frac{\mathbf{1}}{\mathbf{1} + \boldsymbol{e}^{\boldsymbol{K}\boldsymbol{z}_j}} \quad \boldsymbol{K} < \mathbf{0}$$



POLITECNICO
MILANO 1863

LOGISTIC REGRESSION

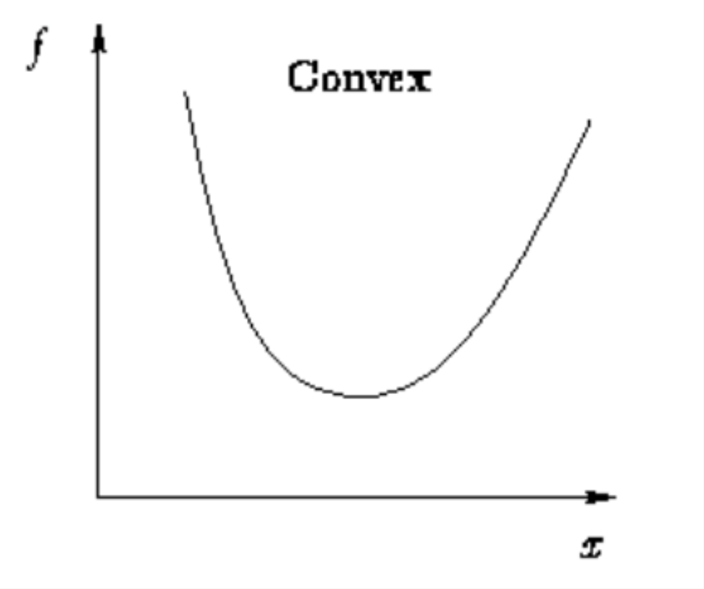
for Binary Classification

Loss Function

In order to learn the best weights w and the bias b, we need a **loss function** $L(\hat{y}, y)$ to be minimized

$$L(\hat{y}, y) = -\log(y\ log\hat{y} + (1-y)\log(1-\hat{y}))$$

for a **single feature vector**



POLITECNICO
MILANO 1863

LOGISTIC REGRESSION

for Binary Classification

Cost Function

For the **whole dataset** (multiple feature vectors)

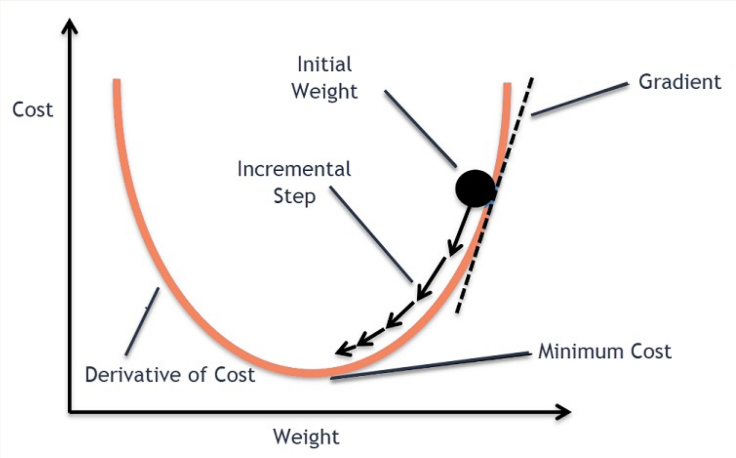
$$J(w,b) = \frac{1}{m}\sum_{I=1}^{m} L\left(\hat{y}^{(i)}, y^{(i)}\right) = -\frac{1}{m}\sum_{I=1}^{m}\left(y^{(i)}\, log\hat{y}^{(i)} + \left(1 - y^{(i)}\right)\log\left(1 - \hat{y}^{(i)}\right)\right)$$

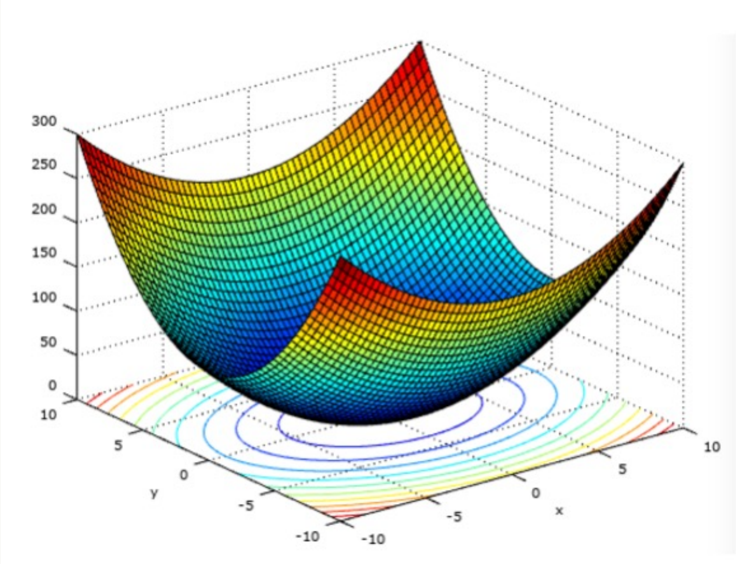
Therefore, weights w and bias b keep being adjusted **minimizing** $J(w,b)$ until the local minimum (global optimum) has been reached

$$w = w - \alpha\frac{dJ(w)}{dw} \qquad b = b - \alpha\frac{dJ(b)}{db}$$

α is the **learning rate**

The adjustment procedure is called **Back-Propagation** and the algorithm employed for optimization process is called **Gradient Descent**



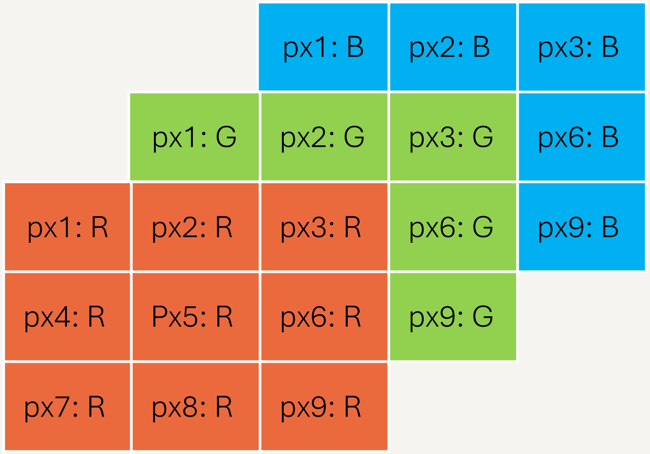


LOGISTIC REGRESSION

for Image Classification

POLITECNICO
MILANO 1863

LOGISTIC REGRESSION

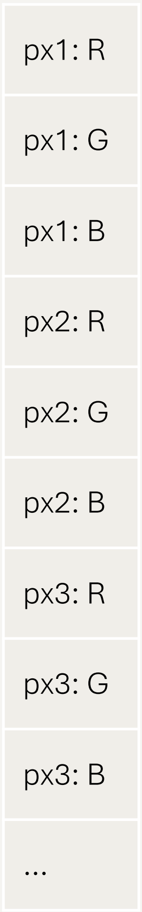


Dimensions: (width, height, channels)

(R, G, B) colour

Flattening Process

Feature Vector



POLITECNICO MILANO 1863

Let’s jump to the code!

References

- Lecture Slides by Prof. Massimiliano Zanoni
- Edgar, T.W. & Manz, D.O.. (2017). Research Methods for Cyber Security. Chapter 4.

POLITECNICO
MILANO 1863

MAE
MASTER OF SCIENCE
MUSIC AND ACOUSTIC
ENGINEERING

Prof. Massimiliano Zanoni

Slides and Presentation
by
Student Marco Muraro