

LOGISTIC REGRESSION *for Image Classification*

CREATIVE PROGRAMMING
AND COMPUTING
Course

Prof. Massimiliano Zanoni

Slides and Presentation
by
Student Marco Muraro



POLITECNICO
MILANO 1863



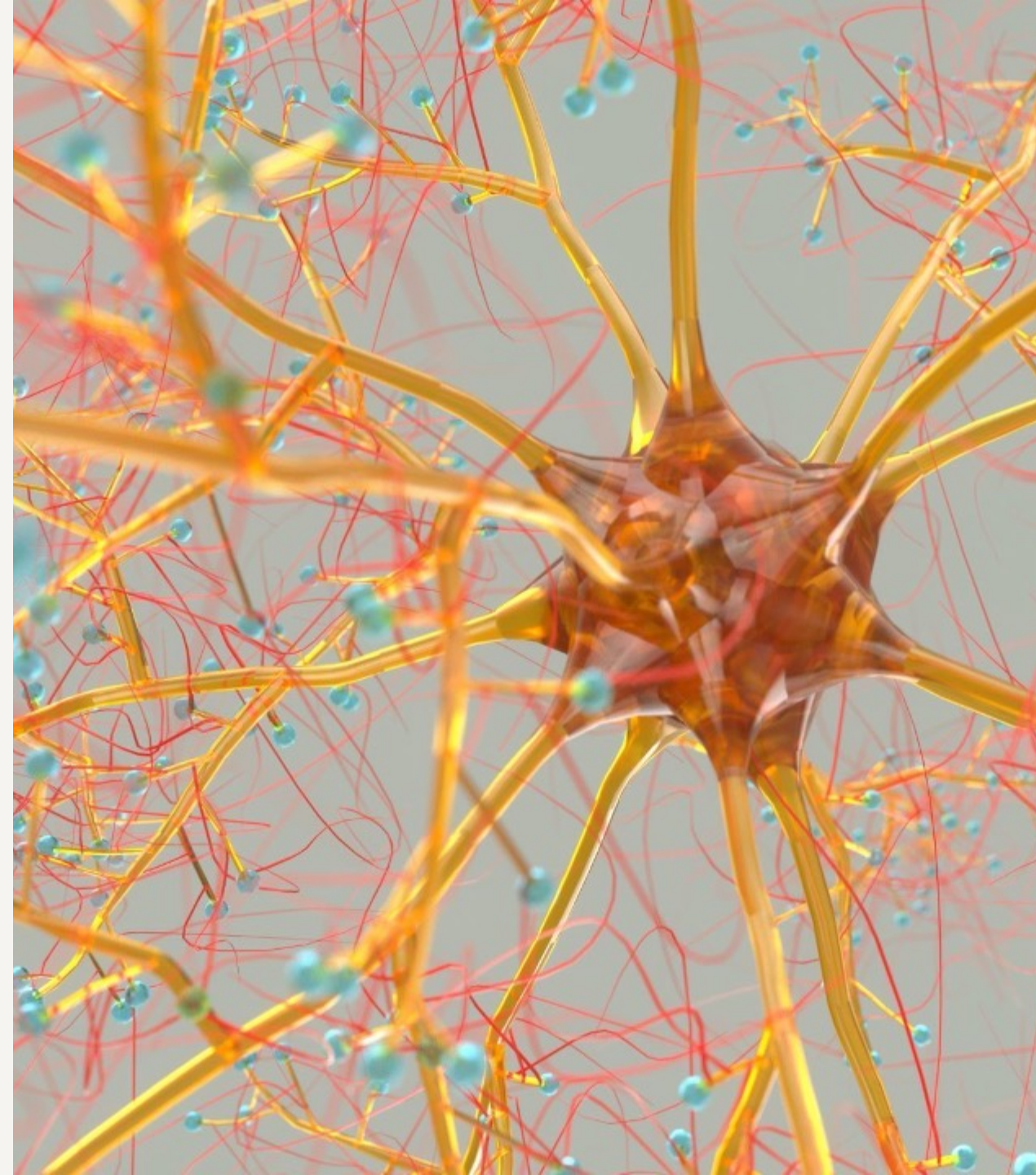
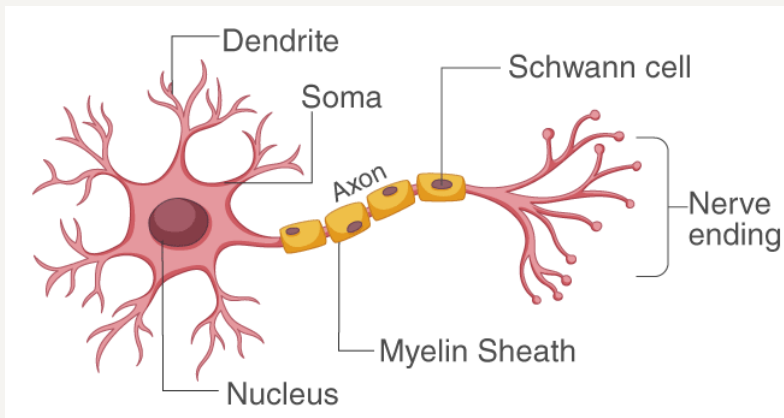
NEURON

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

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

Activation Process

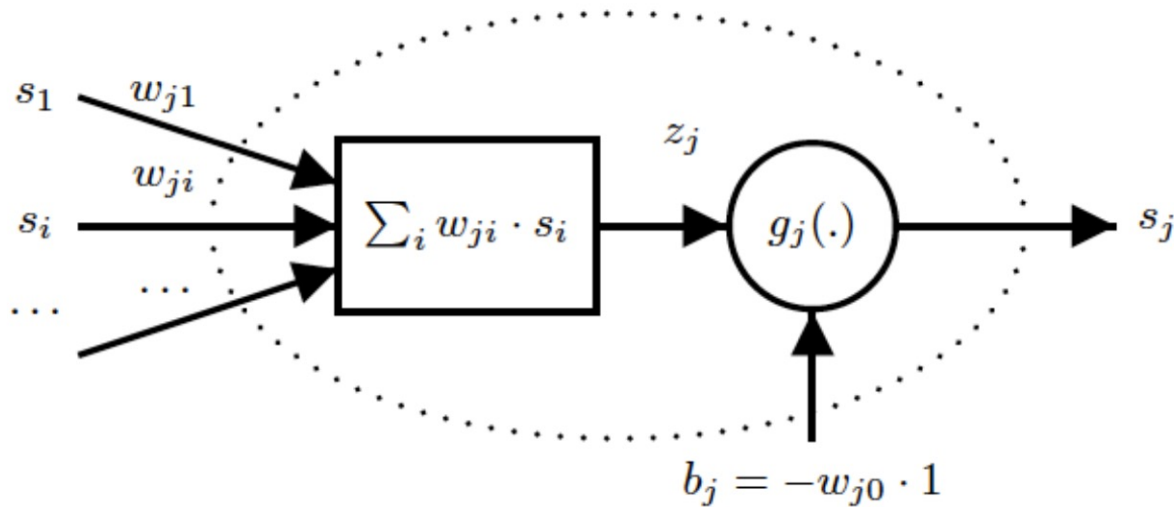
The neuron fires a new electrical impulse when the total charge exceeds a certain threshold



PERCEPTRON

Feed-Forward Model

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



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

Activation Function $g_j(\cdot)$

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

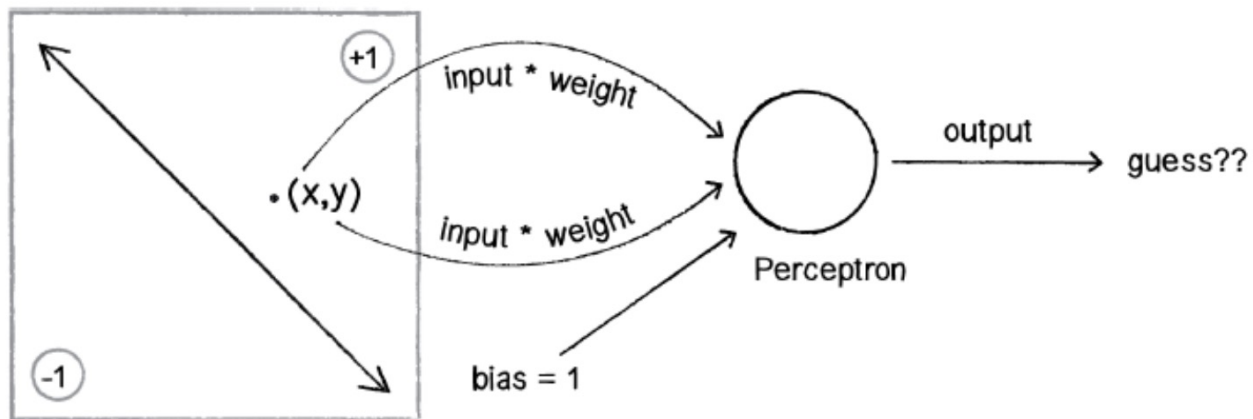
Linear Separation

Decision boundary is defined as a straight line

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



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



BINARY CLASSIFICATION

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

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

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



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**

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

- Adjust the weights

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

- 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 modeling 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.

Learning Process

Given a feature vector \mathbf{x} ,
logistic regression aims to find

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



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

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

LOGISTIC REGRESSION

for Binary Classification

Activation function $g(\cdot)$ is applied

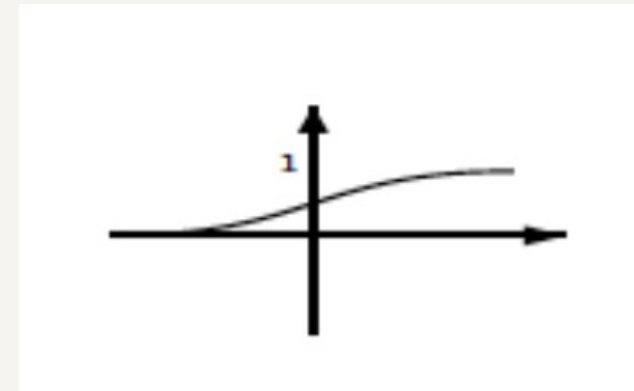
$$\hat{y} = g(\mathbf{w}^T \mathbf{x} + b)$$



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

Sigmoid Function

$$g(z_j) = \frac{1}{1 + e^{Kz_j}} \quad K < 0$$



LOGISTIC REGRESSION

for Binary Classification

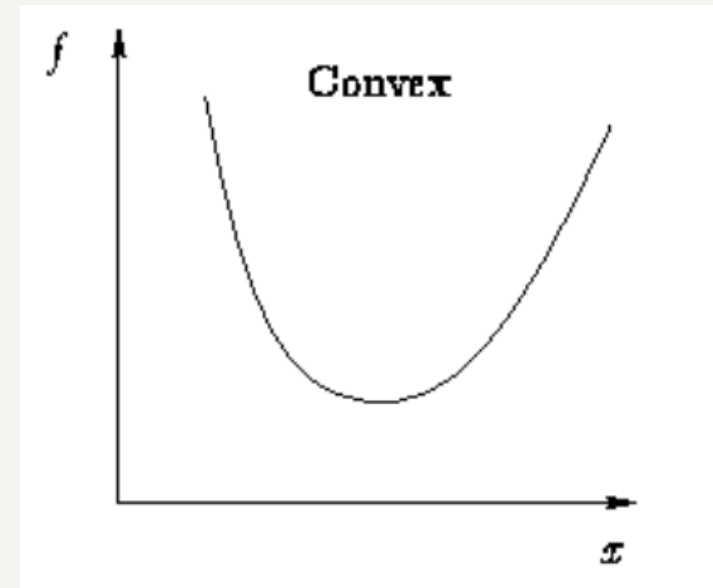
Loss Function

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**





LOGISTIC REGRESSION

for Binary Classification

Cost Function

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

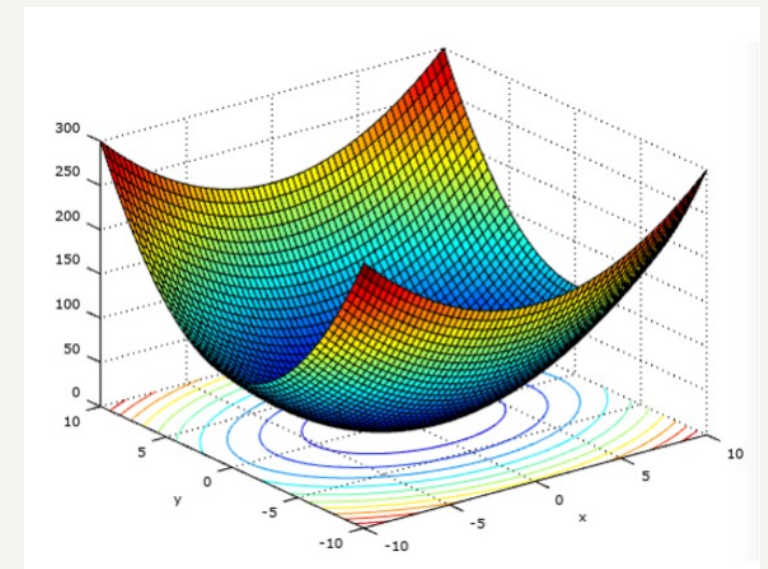
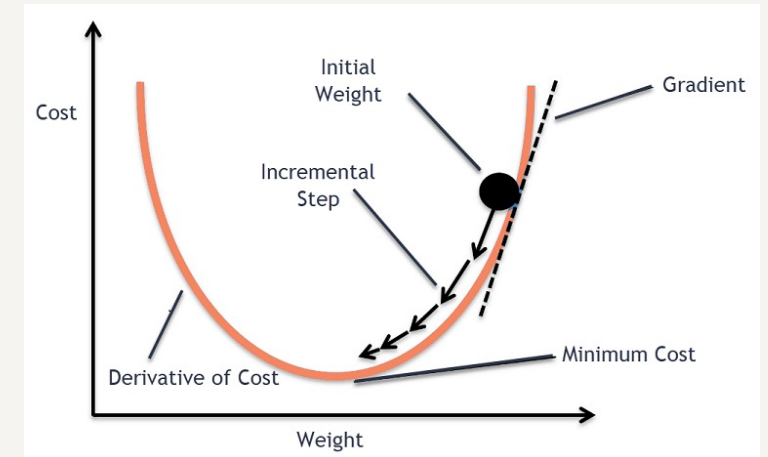
$$J(w, b) = \frac{1}{m} \sum_{i=1}^m L(\hat{y}^{(i)}, y^{(i)}) = -\frac{1}{m} \sum_{i=1}^m (y^{(i)} \log \hat{y}^{(i)} + (1 - y^{(i)}) \log(1 - \hat{y}^{(i)}))$$

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

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

α is the **learning rate**

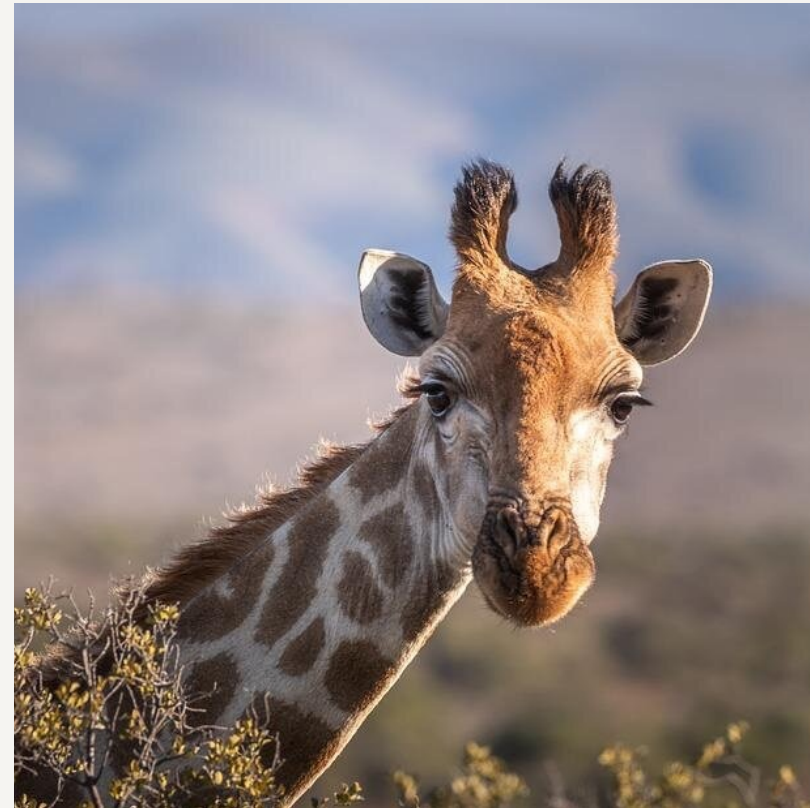
The adjustment procedure is called **Back-Propagation** and the algorithm employed for optimization process is called **Gradient Descent**



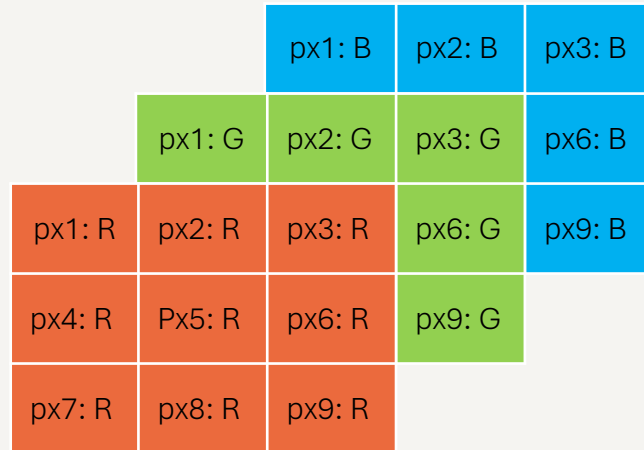


LOGISTIC REGRESSION

for Image Classification



LOGISTIC REGRESSION



px1	px2	px3
px4	px5	px6
px7	px8	px9

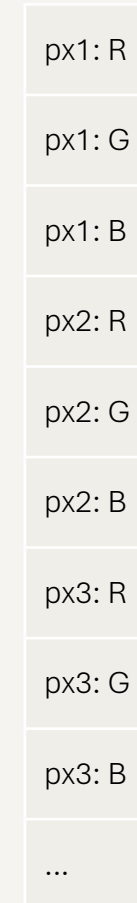
Dimensions: (width, height, channels)

↑
(R, G, B) colour

Flattening Process



Feature Vector



**Let's jump to
the code!**

THANK YOU!

LOGISTIC REGRESSION *for Image Classification*

CREATIVE PROGRAMMING AND COMPUTING
Course

References

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

Prof. Massimiliano Zanoni

Slides and Presentation
by
Student Marco Muraro



POLITECNICO
MILANO 1863

