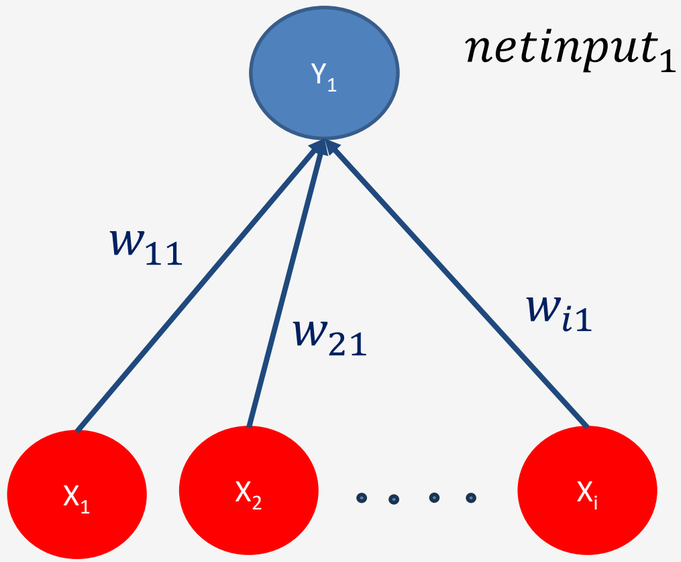
Natural and Artificial Cognition Lab Orazio Miglino
University of Naples Federico II

RETI NEURALI ARTIFICIALI

Teoria e implementazione

Il PRIMO PERCETTRONE
Attivazione del neurone



$$netinput_1 = \sum_{i=1}^{N} x_i w_{i1}$$

$$activation_1 = f(netinput_1)$$

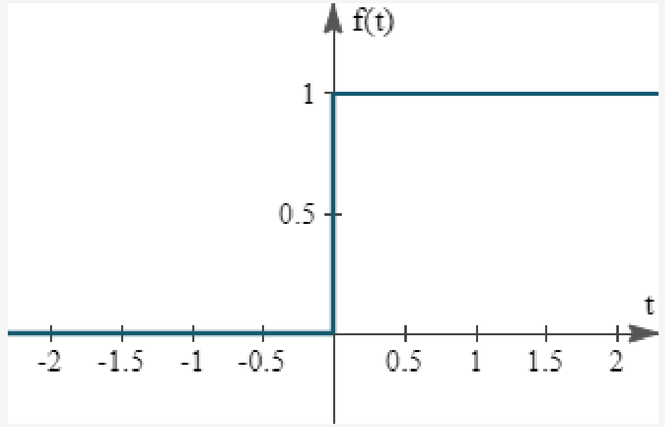
$$activation_1 = \begin{cases} 1 \; if \; netinput_1 \geq threshold \\ 0 \; if \; netinput_1 < threshold \end{cases}$$

$$\sum_{i=1}^{N} x_i w_{i1} = threshold$$

$$\sum_{i=1}^{N} x_i w_{i1} - threshold = 0$$

$$netinput_1 = \sum_{i=1}^{N} x_i w_{i1} - w_{01}$$

bias



ESERCITAZIONE

if **A** is TRUE and **B** is TRUE, then **A OR B** is TRUE

if **A** is TRUE and **B** is FALSE, then **A OR B** is TRUE

if **A** is FALSE and **B** is TRUE, then **A OR B** is TRUE

if **A** is FALSE and **B** is FALSE, then **A OR B** is FALSE

TRUE -> 1
FALSE -> 0

A	B	A *OR* B
1	1	1
1	0	1
0	1	1
0	0	0

ESERCITAZIONE

if **A** is TRUE and **B** is TRUE, then **A OR B** is TRUE

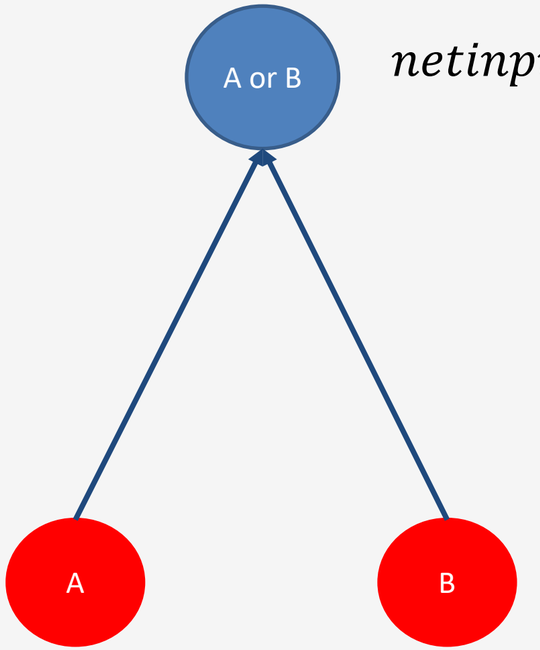
if **A** is TRUE and **B** is FALSE, then **A OR B** is TRUE

if **A** is FALSE and **B** is TRUE, then **A OR B** is TRUE

if **A** is FALSE and **B** is FALSE, then **A OR B** is FALSE

TRUE -> 1
FALSE -> 0

A	B	A *OR* B
1	1	1
1	0	1
0	1	1
0	0	0



$$netinput_1 = \sum_{i=1}^{N} x_i w_{i1} - w_{01}$$

ESERCITAZIONE

if **A** is TRUE and **B** is TRUE, then **A OR B** is TRUE

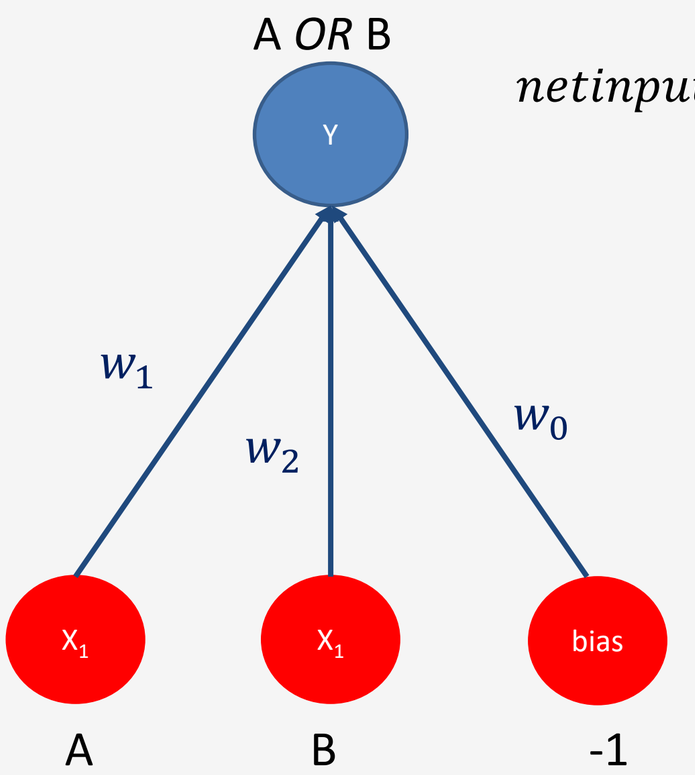
if **A** is TRUE and **B** is FALSE, then **A OR B** is TRUE

if **A** is FALSE and **B** is TRUE, then **A OR B** is TRUE

if **A** is FALSE and **B** is FALSE, then **A OR B** is FALSE

TRUE -> 1
FALSE -> 0

A	B	A *OR* B
1	1	1
1	0	1
0	1	1
0	0	0



$$netinput_1 = \sum_{i=1}^{N} x_i w_{i1} - w_{01}$$

ESERCITAZIONE

if **A** is TRUE and **B** is TRUE, then **A OR B** is TRUE

if **A** is TRUE and **B** is FALSE, then **A OR B** is TRUE

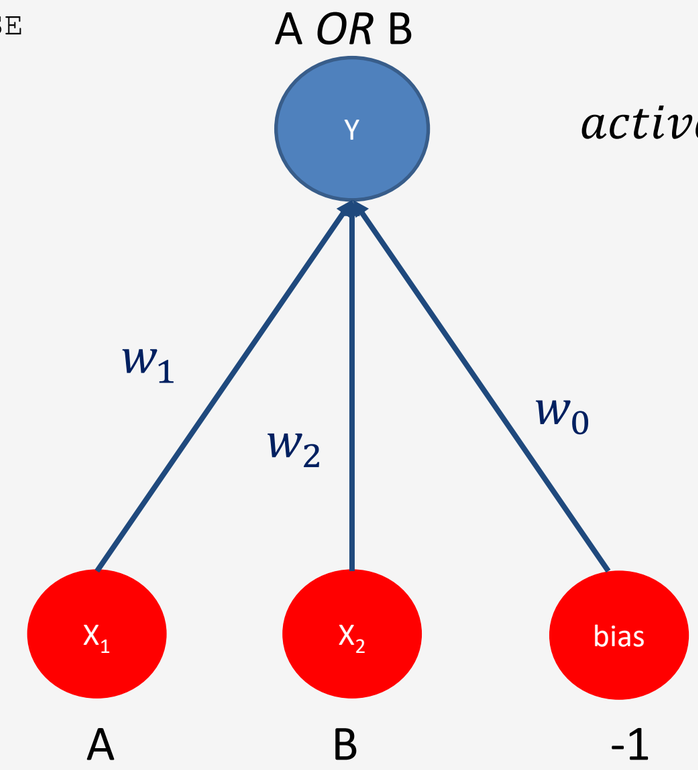
if **A** is FALSE and **B** is TRUE, then **A OR B** is TRUE

if **A** is FALSE and **B** is FALSE, then **A OR B** is FALSE

TRUE -> 1
FALSE -> 0

A	B	A *OR* B
1	1	1
1	0	1
0	1	1
0	0	0

$$netinput = x_1 \cdot w_1 + x_2 \cdot w_2 + bias \cdot w_0$$



$$activation = \begin{cases} 1 \; if \;\; netinput \geq 0 \\ 0 \; if \;\; netinput < 0 \end{cases}$$

$$\Delta w_i = x_i \cdot y \cdot \eta$$

$\eta = valore\ fisso\ tra\ 0\ e\ 1$
$w_0 = random$
$w_1 = random$
$w_2 = random$

ESERCITAZIONE

```
import random
lr = 0.1 #learning rate
bias = -1
weights = [random.random(),random.random(),random.random()] #weights generated in a list (3 weights in total for 2 neurons
```

Funzione che calcola l'attivazione del neurone e la regola di HEBB

```
#HEBB RULE
def Perceptron(input1, input2, output) :
   netinput = input1*weights[0]+input2*weights[1]+bias*weights[2]
   if netinput > 0 : #activation function (here Heaviside)
      outputP = 1
   else :
      outputP = 0
   if outputP != output:
      weights[0] += input1 * output * lr
      weights[1] += input2 * output * lr
      weights[2] += bias * output * lr
```

$$netinput = x_1 \cdot w_0 + x_2 \cdot w_1 + bias \cdot w_2$$

$$activation = \begin{cases} 1 \; if \;\; netinput \geq 0 \\ 0 \; if \;\; netinput < 0 \end{cases}$$

$$w_0^t = w_0^{t-1} + \; x_1 \cdot y \cdot \eta$$

$$w_1^t = w_1^{t-1} + \; x_2 \cdot y \cdot \eta$$

$$w_2^t = w_2^{t-1} + bias \cdot y \cdot \eta$$

ESERCITAZIONE

Attivazione della rete neurale presentando tutti gli esempi

```
for i in range(100) :
    Perceptron(1,1,1) #True or true
    Perceptron(1,0,1) #True or false
    Perceptron(0,1,1) #False or true
    Perceptron(0,0,0) #False or false
```

Epoche di apprendimento

A	B	A *OR* B
1	1	1
1	0	1
0	1	1
0	0	0

Testiamo il comportamento della rete neurale

```
x = int(input())
y = int(input())
netinput = x*weights[0] + y*weights[1] + bias*weights[2]
if netinput > 0: #activation function
    outputP = 1
else :
    outputP = 0
print(x, "or", y, "is : ", outputP)
```

ESERCITAZIONE

```
#DELTA RULE
def Perceptron(input1, input2, output) :
   netinput = input1*weights[0]+input2*weights[1]+bias*weights[2]
   if netinput > 0 : #activation function (here Heaviside)
      outputP = 1
   else :
      outputP = 0
   error = output - outputP
   if error != 0:
      weights[0] += error * input1 * lr
      weights[1] += error * input2 * lr
      weights[2] += error * bias * lr
```

$$w_0^t = w_0^{t-1} + \eta(d - y)x_1$$
$$w_1^t = w_1^{t-1} + \eta(d - y)x_2$$
$$w_2^t = w_2^{t-1} + \eta(d - y)bias$$