Facultatea de Electronică,
Telecomunicații și
Tehnologia Informației

# SISTEME INTELIGENTE DE SUPORT DECIZIONAL

Ș.l.dr.ing. Laura-Nicoleta IVANCIU

## Curs 8 – Rețele neuronale artificiale. Clasificatoare cu RNA.

# Cuprins

- RNA – definire, arhitectură
- Funcții de activare
- Instruirea RNA
- Tipuri de probleme rezolvabile cu RNA
- Clasificatoare cu RNA – clasificare liniară
- Instruire prin minimizarea erorii
- Studii de caz

RNA = calculator distribuit, masiv paralel, care achiziționează noi cunoștințe pe baza experienței anterioare și le face disponibile pentru utilizarea ulterioară (S.Haykin, 1994)

- se bazează pe extragerea unui model pe bază de exemple

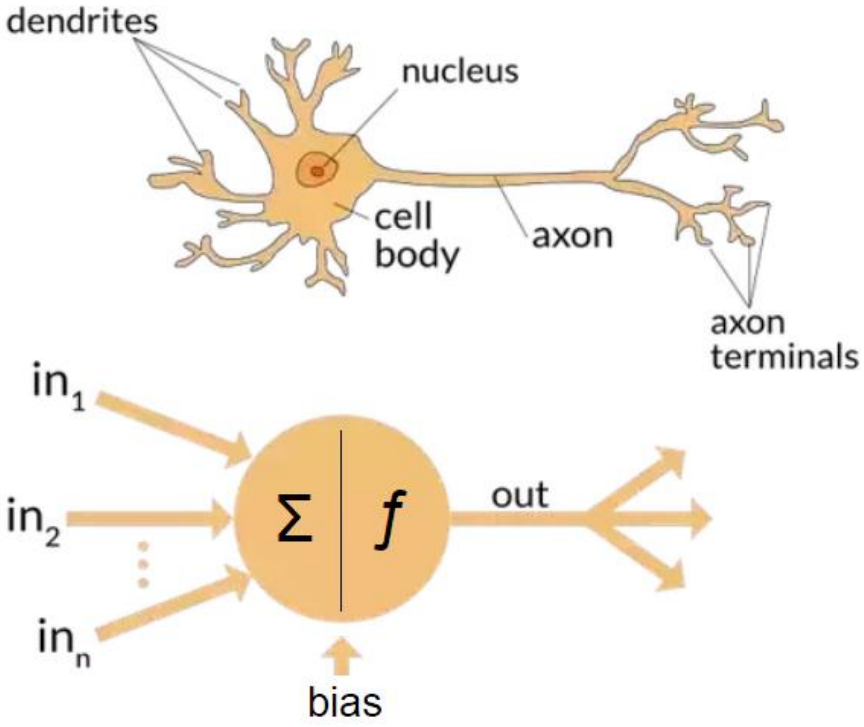
- neuronul artificial imită funcționarea neuronilor naturali

- învățare pe bază de suficient de multe exemple

# Asemănarea cu creierul uman

➢ cunoștințele sunt achiziționate de rețeaua neuronală printr-un proces de învățare

➢ cunoștințele sunt depozitate în conexiunile inter-neuronale (ponderi sinaptice)

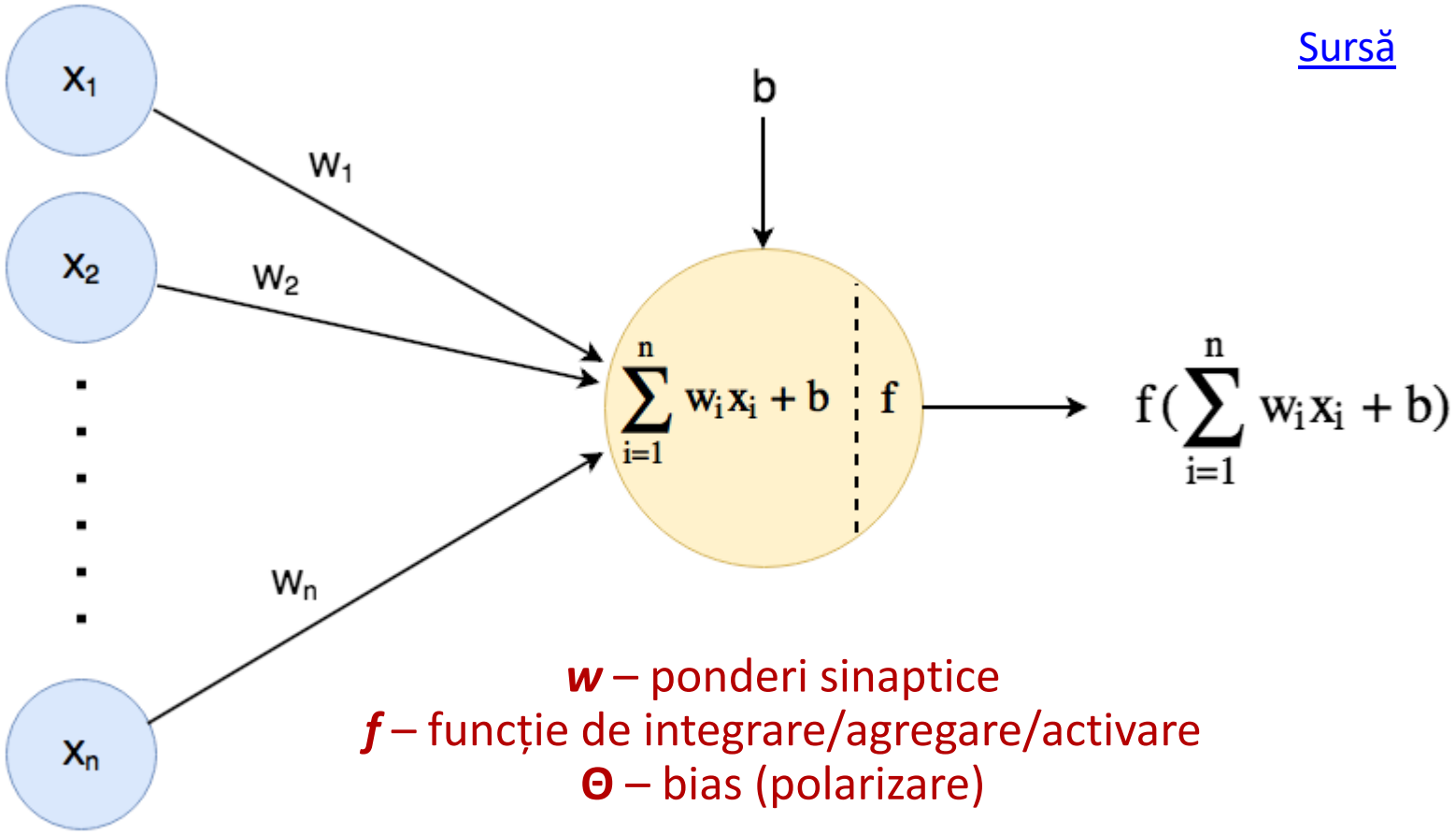



Sursă

# Neuronul artificial

## Modelul de bază McCulloch-Pitts (1943)

Sursă



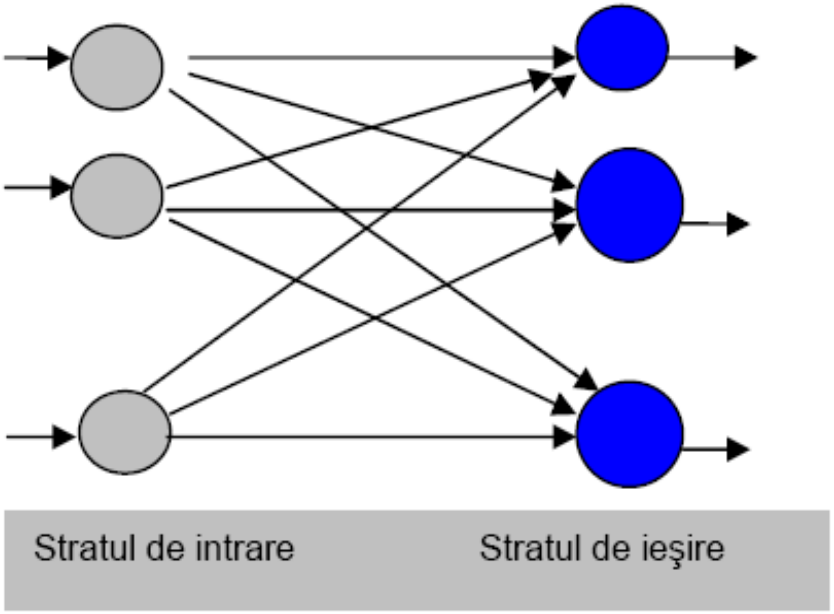

***w*** – ponderi sinaptice
***f*** – funcție de integrare/agregare/activare
**Θ** – bias (polarizare)

RNA este complet determinată prin:

- tipul unităților funcționale (neuroni)
- arhitectură (amplasare neuroni)
- algoritm de funcționare (transformare semnal intrare în semnal ieșire)
- algoritm de învățare/instruire (cum achiziționează rețeaua noi cunoștințe pe bază de exemple)

# Arhitectura RNA

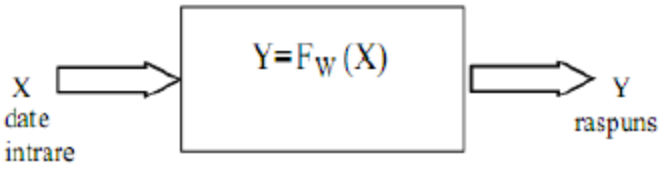
- rețele feed-forward (unidirecționale)




unistrat

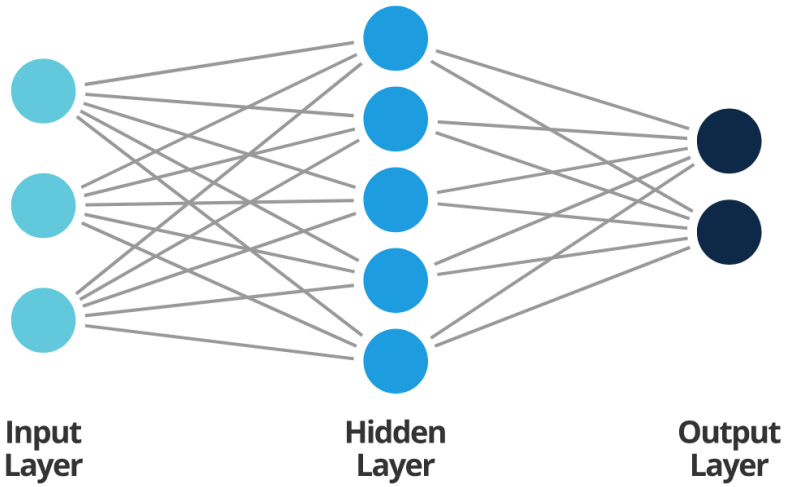
$$X = \begin{bmatrix} x_1 \\ x_2 \\ \dots \\ x_N \end{bmatrix} \quad W = \begin{bmatrix} w_{11} \dots w_{1M} \\ w_{21} \\ \dots \\ w_{N1} \dots w_{NM} \end{bmatrix} \quad B = \begin{bmatrix} b_1 \\ b_2 \\ \dots \\ b_M \end{bmatrix}$$

$$Y = f(W^{\tau} \cdot X + B)$$



# Arhitectura RNA

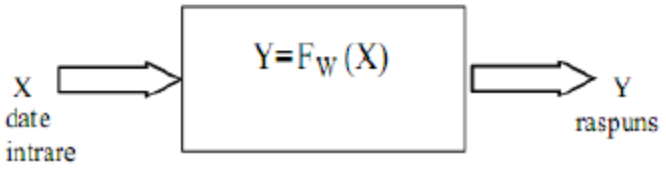
➢ rețele feed-forward (unidirecționale)




Sursă

$$X = \begin{bmatrix} x_1 \\ x_2 \\ \dots \\ x_N \end{bmatrix} \quad W = \begin{bmatrix} w_{11}...w_{1M} \\ w_{21} \\ \dots \\ w_{N1}...w_{NM} \end{bmatrix} \quad B = \begin{bmatrix} b_1 \\ b_2 \\ \dots \\ b_M \end{bmatrix}$$

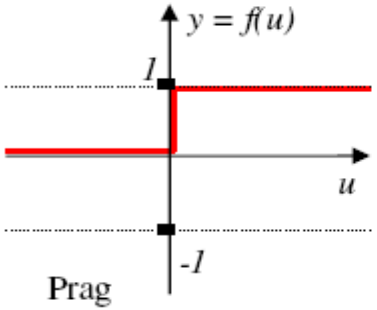
$$Y = f(W^{\tau} \cdot X + B)$$




multistrat

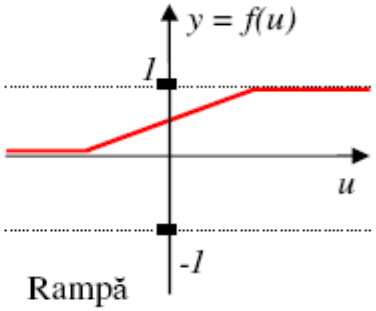
- mai lente decât cele unistrat
- pot implementa funcții mai complexe

# Funcții de activare



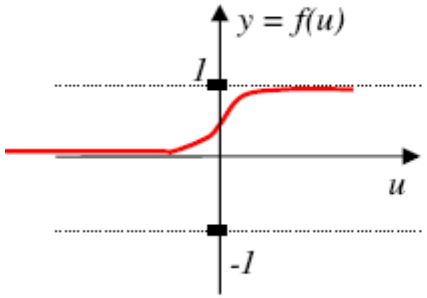

Prag

$$f(u) = \begin{cases} 0, u \le 0 \\ 1, u > 0 \end{cases}$$



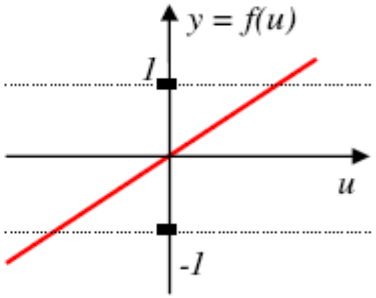

Rampă

$$f(u) = \begin{cases} 0, u \le 0 \\ u/k, 0 < u \le k \\ 1, k < u \end{cases}$$




Sigmoidală

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




Liniară

$$f(u) = ku$$

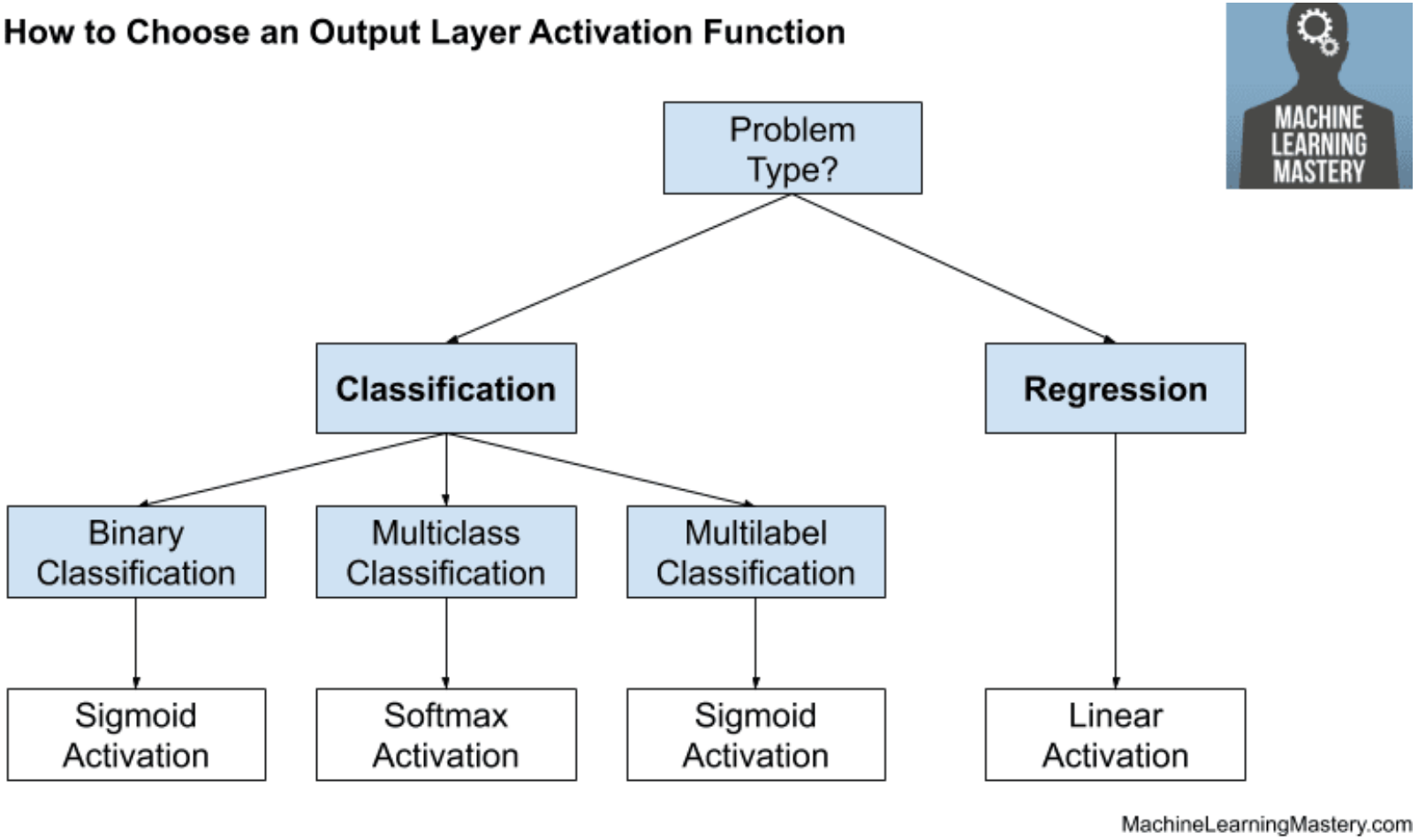
$$ReLU: f(u) = \max(u, 0)$$

Rectified Linear Unit (ReLU)

Grafic?

$$\text{Softmax: } f(s) = \frac{e_i^s}{\sum_{i=1}^{n} e_i^s}$$

# Funcții de activare




Sursă

Instruirea RNA = procesul adaptării ponderilor și a polarizărilor

Algoritm de instruire = modul în care se modifică ponderile/polarizările

$$w_{kj}(n+1) = w_{kj}(n) + \Delta w_{kj}(n)$$

unde: k – neuron de la care "pleacă" ponderea,
j – neuron spre care "vine" ponderea,
n – momentul de timp

$$b_k(n+1) = b_k(n) + \Delta b_k(n)$$

$\Delta w_{kj}(n)$, $\Delta b_k(n)$ - algoritmi de instruire

# Reguli și algoritmi de instruire

**Regulă de instruire** = formulă matematică care specifică cum se modifică parametrii RNA, pentru a atinge obiectivul dorit

- ✓ instruire prin minimizarea erorii (perceptron learning rule)
- ✓ instruire bazată pe gradient (gradient descent)
- ✓ instruire hebbiană
- ✓ instruire competitivă

**Algoritm de instruire** = proces iterativ bazat pe o regulă de instruire

Etape:

1. inițializare ponderi și polarizări
2. calcul ieșire neuroni
3. calcul cantități $\Delta w$, $\Delta b$ cu care se modifică fiecare pondere, polarizare
4. modificare ponderi și polarizări
5. salt la etapa 2 – recalcularea ieșirilor neuronilor

# Tipuri de probleme rezolvabile cu RNA

- aproximare de funcții
  - interpolare perechi intrare-ieșire
  - domeniu de ieșire continuu

*Ex:* modelarea directă/inversă a unui sistem necunoscut, predicție

- clasificare
  - gruparea vectorilor de intrare în clase
  - domeniu de ieșire discret

*Ex:* recunoașterea formelor, operații de telecom (decodare, demodulare, regenerare semnale), decizie

- optimizare
  - găsirea punctului de minim/maxim al unei funcții
  - set de ponderi pentru care diferența dintre ieșirea dorită și ieșirea rețelei este minimă, în sens statistic

*Ex:* problema comis-voiajorului

# Aproximare de funcții (function fitting)

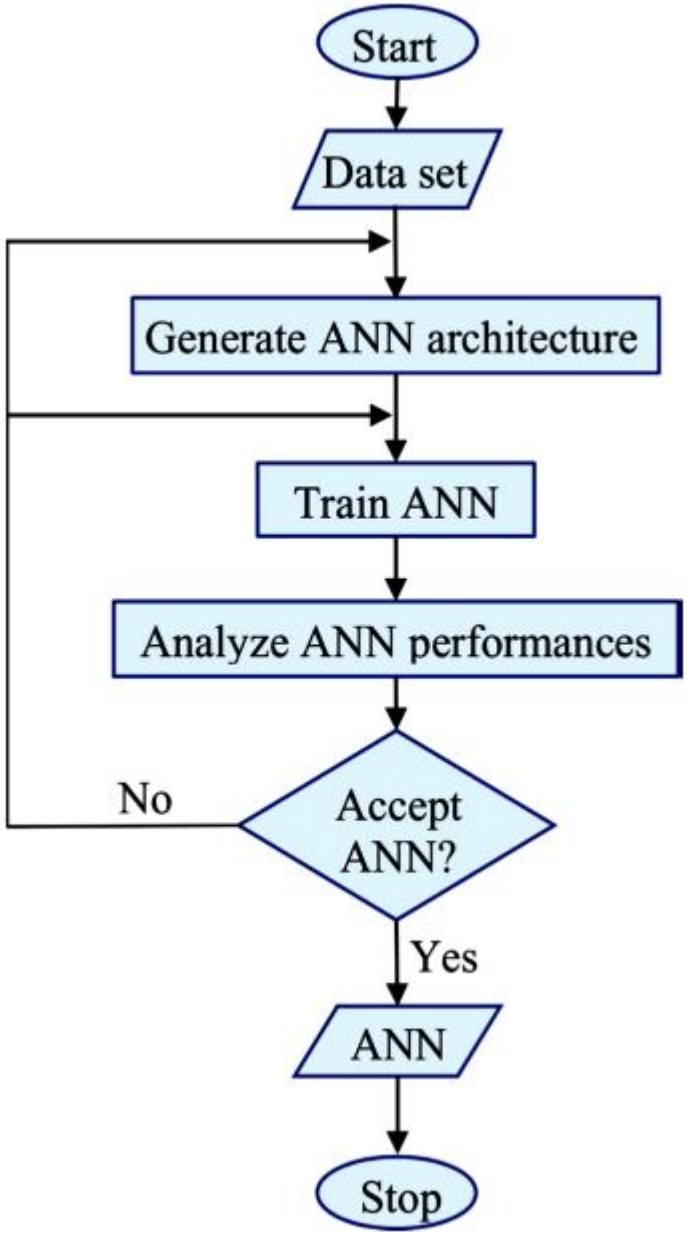
- Obiectiv: dezvoltarea unui sistem (model) care să ofere o aproximare suficient de bună a unei funcții cunoscute

- Etape:
  - pregătire date
  - selectare arhitectură rețea
  - antrenarea rețelei
  - testare (validare)
  - utilizarea propriu-zisă

# Aproximare de funcții

- Antrenare supervizată
- Perechi de date intrare-ieșire

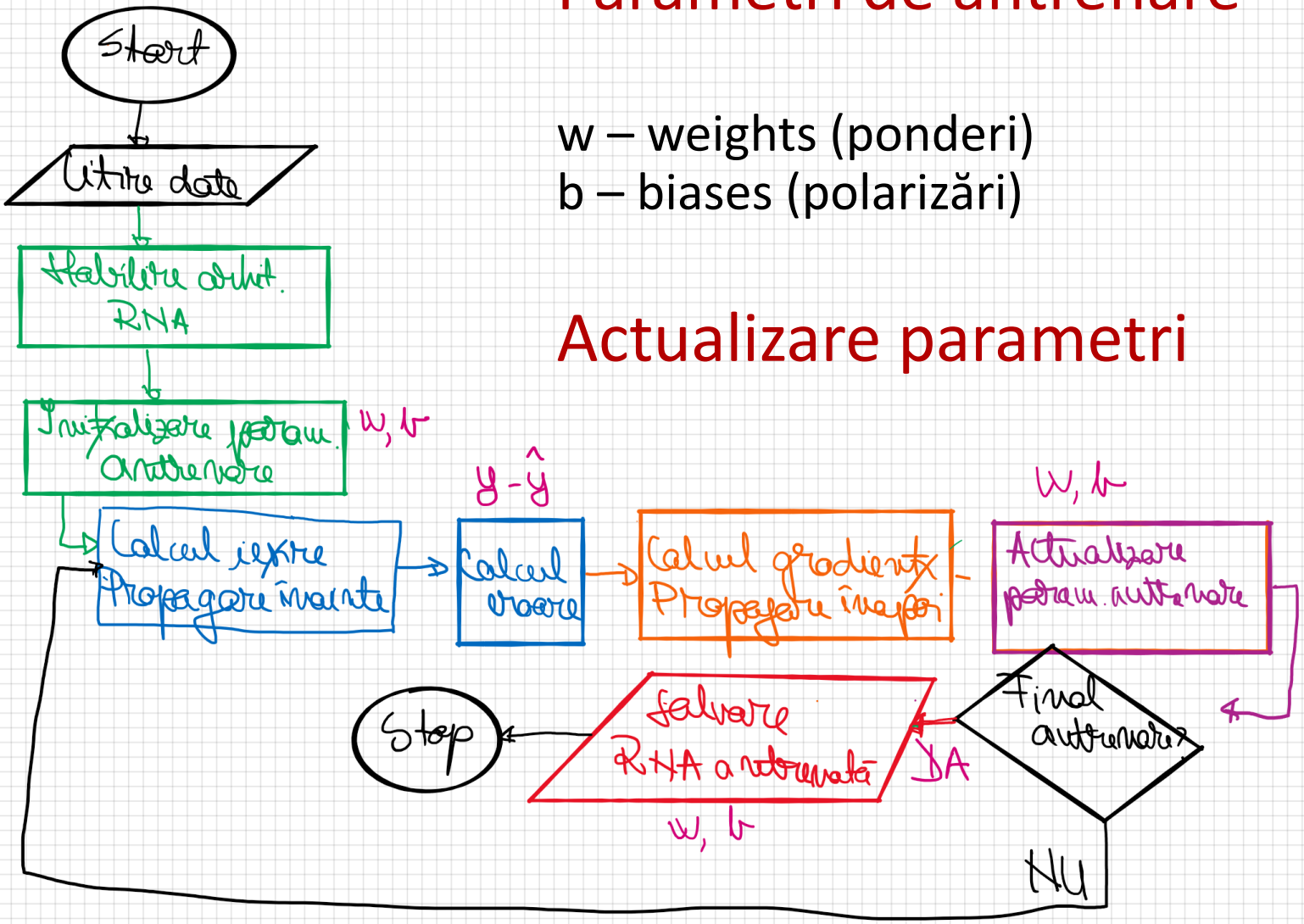
Exemple:
- aprox. preț case
- aprox. % grăsime din organism
- aprox. preț acțiuni, valută, petrol



# Parametri de antrenare

w – weights (ponderi)
b – biases (polarizări)

# Actualizare parametri



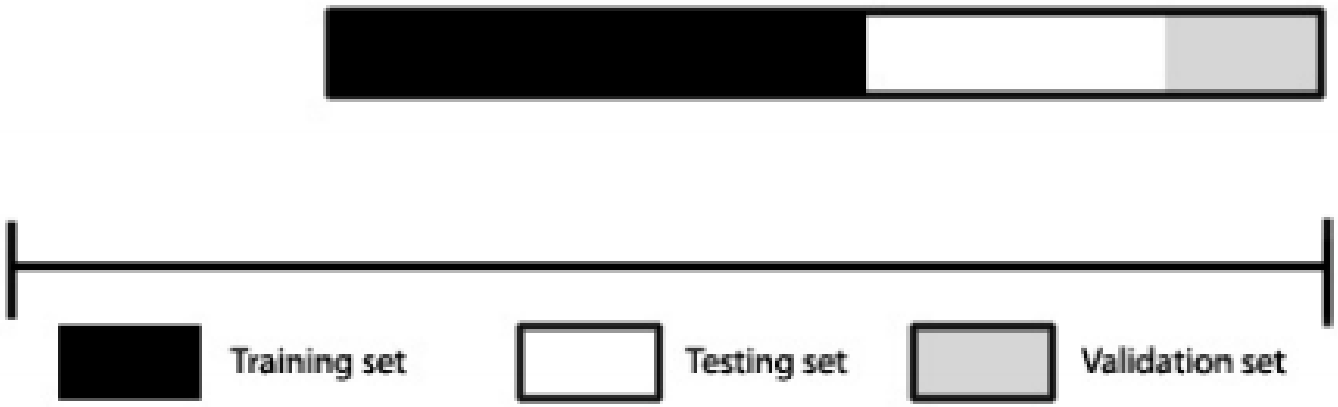
# Considerente de proiectare

## Alegerea setului de date de intrare

antrenare – 50% - minim *5*nr.ponderi* – previne supra-potrivirea

validare – 20%-40%

testare – 10% - 30%



# Considerente de proiectare

## Număr de neuroni

Rețea cu 3 straturi:

$$p_{hidden} = \sqrt{n_{in} * m_{out}}$$

$m_{out} > 1$ - rezultate inferioare față de $m_{out} = 1$

Observație: pentru aplicații de clasificare, $m_{out}$ = *nr. clase*

# Considerente de proiectare

## Inițializare ponderi

Valori random, într-un interval simetric față de 0
Ex: [-0.5; 0.5]

Rule of thumb: $\left[-\frac{1}{\#\ ponderi}; \frac{1}{\#\ ponderi}\right]$

#ponderi – numărul de conexiuni (ponderi) care intră într-un nod

Valori mai mici pentru nodurile în care intră mai multe conexiuni

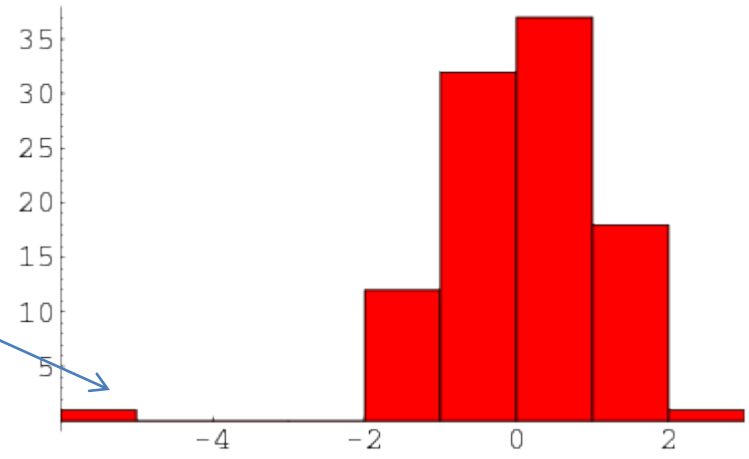
# Considerente de proiectare

## Pregătirea setului de date (data preparation)

Tipuri de date:

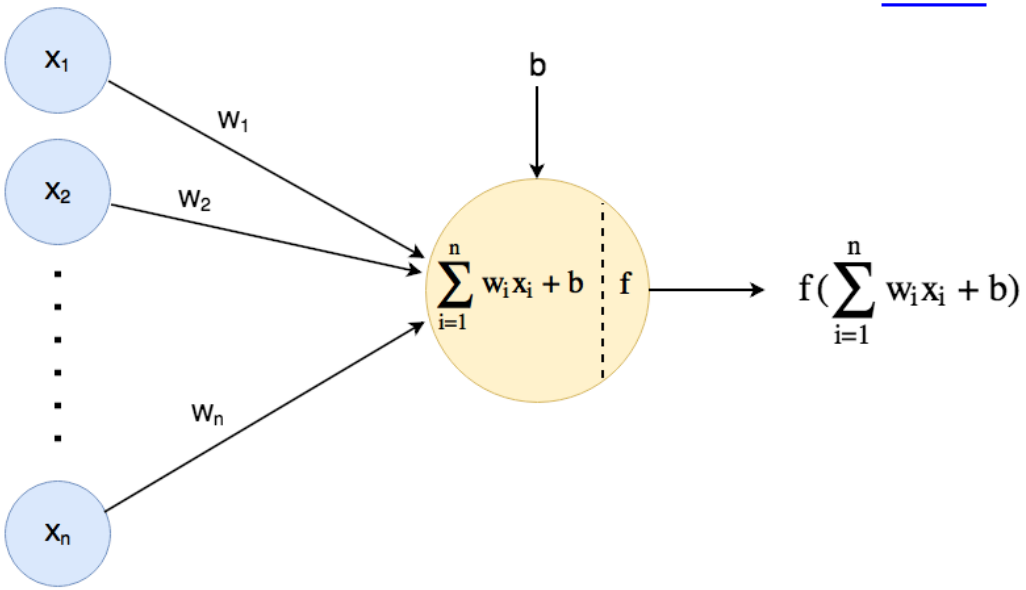
Simbolice: A, da/nu, mic/mare/mediu
Numerice discrete: -2, 5, 34
Numerice continue: -3.421, 98.02

- eliminare excepții vizibile (outliers)
- eliminare date redundante
- transformare și codare



# Clasificare liniară
## Perceptronul

Sursă



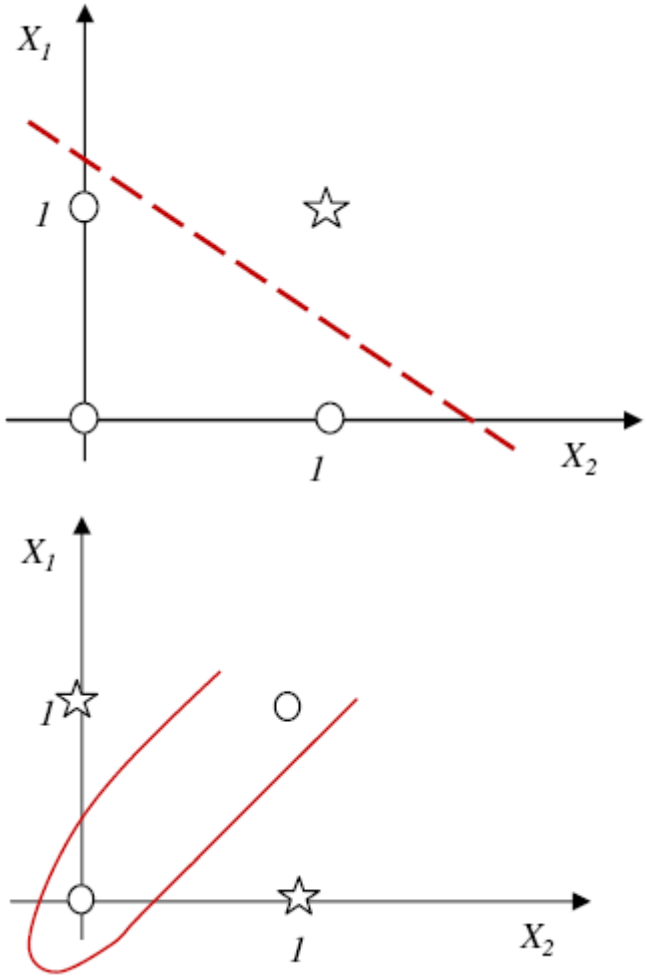

- clasificator binar (2 clase)
- clase liniar separabile
- hiperplan de separație

$$y = f(u) = \begin{cases} 0, u \le 0 \\ 1, u > 0 \end{cases}$$

funcție de activare binară (treaptă unitate)

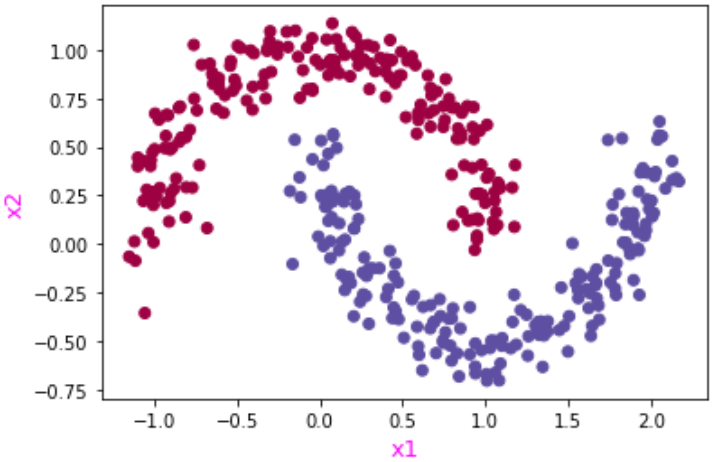
# Clasificare liniară
## Separabilitate

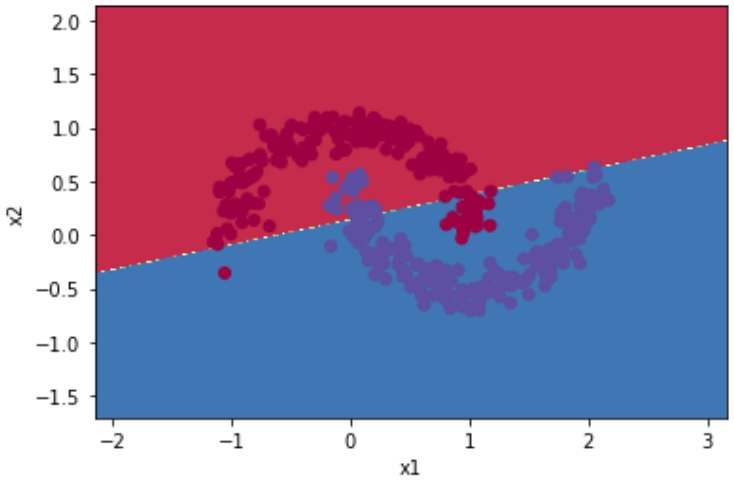
| X1 | X2 | AND | OR | XOR |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 1 |
| 1 | 0 | 0 | 1 | 1 |
| 1 | 1 | 1 | 1 | 0 |

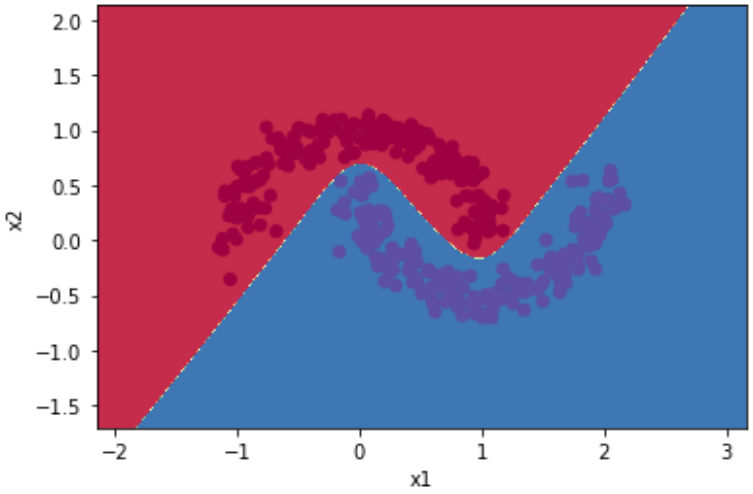



*XOR nu se poate implementa cu perceptron!*

# Clasificare liniară
## Separabilitate

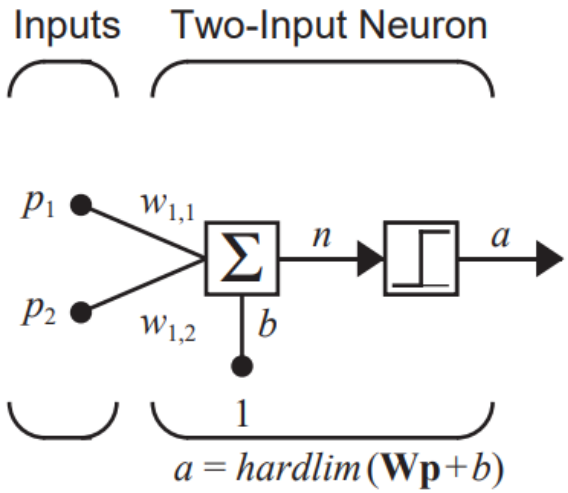


# Clasificare liniară

## Instruire prin minimizarea erorii (perceptron learning rule)




$$a = hardlim(n)$$
$$= hardlim(Wp + b)$$
$$= hardlim(w_{11}p_1 + w_{12}p_2 + b)$$

Granița de decizie (decision boundary): n = 0

Exemplu:

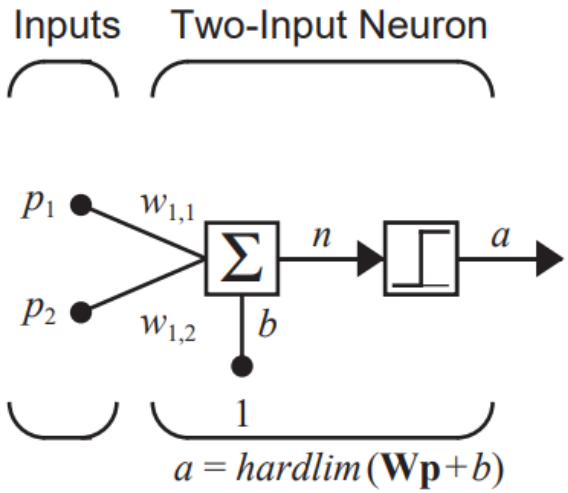
$$w_{11} = 1, w_{12} = 1, b = -1$$

Granița de decizie este dată de ecuația: $w_{11}p_1 + w_{12}p_2 + b = p_1 + p_2 - 1 = 0$

Pentru reprezentare grafică, se iau în considerare intersecțiile cu cele 2 axe.

# Clasificare liniară

## Instruire prin minimizarea erorii (perceptron learning rule)




$$a = hardlim(n)$$
$$= hardlim(Wp + b)$$
$$= hardlim(w_{11}p_1 + w_{12}p_2 + b)$$

$$W = \begin{bmatrix} w_{11} \\ w_{12} \end{bmatrix}$$

Instruire = modificare ponderi și polarizări

$e = t - a$, t = target (valoare dorită/cunoscută)

$$W^{new} = W^{old} + e * p^T$$

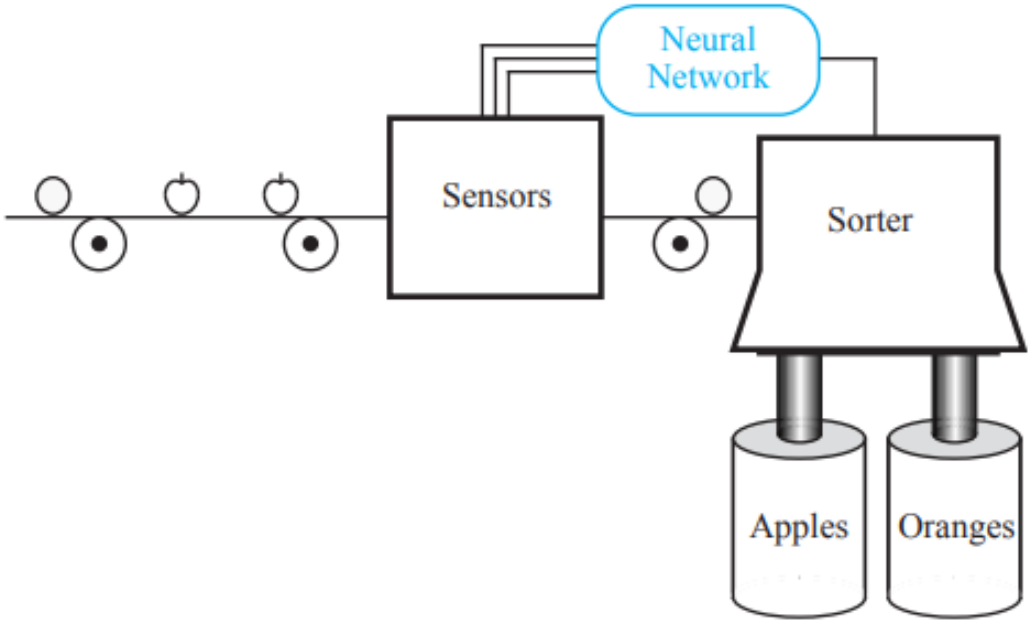
$$b^{new} = b^{old} + e$$

# Clasificare liniară

## Instruire prin minimizarea erorii (perceptron learning rule)

Sursă

Exemplu:




$$\left\{\mathbf{p}_1 = \begin{bmatrix}1\\-1\\-1\end{bmatrix}, t_1 = \begin{bmatrix}0\end{bmatrix}\right\} \qquad \left\{\mathbf{p}_2 = \begin{bmatrix}1\\1\\-1\end{bmatrix}, t_2 = \begin{bmatrix}1\end{bmatrix}\right\}$$

# Clasificare liniară

## Instruire prin minimizarea erorii (perceptron learning rule)

1. Inițializare ponderi, polarizări

$$\mathbf{W} = \begin{bmatrix} 0.5 & -1 & -0.5 \end{bmatrix}, \quad b = 0.5$$

$$\left\{ \mathbf{p}_1 = \begin{bmatrix} 1 \\ -1 \\ -1 \end{bmatrix}, t_1 = \begin{bmatrix} 0 \end{bmatrix} \right\} \qquad \left\{ \mathbf{p}_2 = \begin{bmatrix} 1 \\ 1 \\ -1 \end{bmatrix}, t_2 = \begin{bmatrix} 1 \end{bmatrix} \right\}$$

2. Iterația 1: [Sursă]

- aplicare vector p1 la intrarea rețelei:

$$a = hardlim(\mathbf{W}\mathbf{p}_1 + b) = hardlim\left( \begin{bmatrix} 0.5 & -1 & -0.5 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \\ -1 \end{bmatrix} + 0.5 \right)$$

$$= hardlim(2.5) = 1$$

- calcul eroare $e = t_1 - a = 0 - 1 = -1$
- actualizare ponderi și polarizări

$$\mathbf{W}^{new} = \mathbf{W}^{old} + e\mathbf{p}^T = \begin{bmatrix} 0.5 & -1 & -0.5 \end{bmatrix} + (-1) \begin{bmatrix} 1 & -1 & -1 \end{bmatrix}$$

$$= \begin{bmatrix} -0.5 & 0 & 0.5 \end{bmatrix}.$$

$$b^{new} = b^{old} + e = 0.5 + (-1) = -0.5$$

# Clasificare liniară
## Instruire prin minimizarea erorii (perceptron learning rule)

3. Iterația 2:

$$a = hardlim\,(\mathbf{W}\mathbf{p}_2 + b) = hardlim\,(\begin{bmatrix}-0.5 & 0 & 0.5\end{bmatrix}\begin{bmatrix}1\\1\\-1\end{bmatrix} + (-0.5))$$

Sursă

$$= hardlim\,(-1.5) = 0$$

$$e = t_2 - a = 1 - 0 = 1$$

$$\mathbf{W}^{new} = \mathbf{W}^{old} + e\mathbf{p}^T = \begin{bmatrix}-0.5 & 0 & 0.5\end{bmatrix} + 1\begin{bmatrix}1 & 1 & -1\end{bmatrix} = \begin{bmatrix}0.5 & 1 & -0.5\end{bmatrix}$$

$$b^{new} = b^{old} + e = -0.5 + 1 = 0.5$$

# Clasificare liniară
## Instruire prin minimizarea erorii (perceptron learning rule)

4. Iterația 3:

$$a = hardlim(\mathbf{W}\mathbf{p}_1 + b) = hardlim\left(\begin{bmatrix}0.5 & 1 & -0.5\end{bmatrix}\begin{bmatrix}1\\-1\\-1\end{bmatrix} + 0.5\right)$$

$$= hardlim(0.5) = 1$$

$$e = t_1 - a = 0 - 1 = -1$$

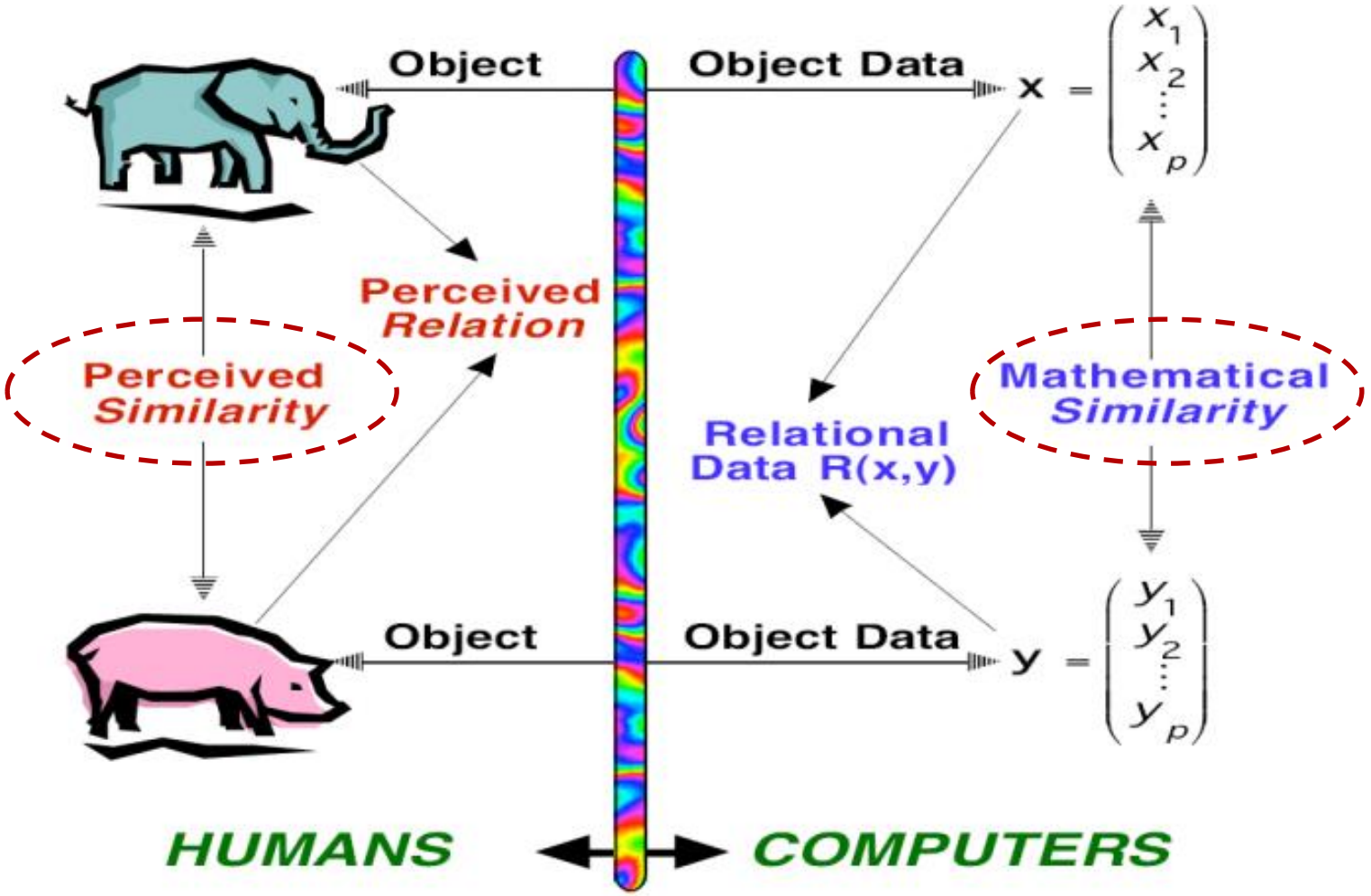
$$\mathbf{W}^{new} = \mathbf{W}^{old} + e\mathbf{p}^T = \begin{bmatrix}0.5 & 1 & -0.5\end{bmatrix} + (-1)\begin{bmatrix}1 & -1 & -1\end{bmatrix}$$

$$= \begin{bmatrix}-0.5 & 2 & 0.5\end{bmatrix}$$

$$b^{new} = b^{old} + e = 0.5 + (-1) = -0.5$$

Sursă

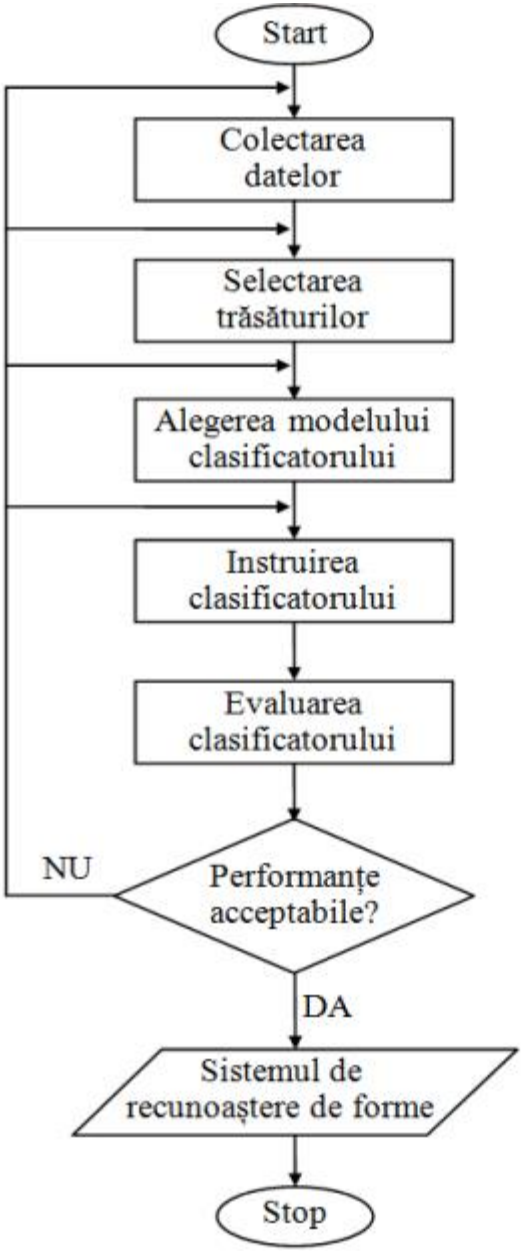
# Recunoaștere forme human vs. computer



# Recunoaștere forme (pattern recognition)

Etape:

1. extragere trăsături
2. stabilire prototip clasă
3. alocare obiect - clasă



# Clasificarea crabilor cu RNA

- formularea problemei
- pregătirea datelor
- proiectarea clasificatorului cu RNA
- testare

# Clasificarea crabilor cu RNA

## Formularea problemei

Să se proiecteze un clasificator cu RNA, care să identifice sexul unui crab, folosind caracteristicile sale fizice:

*species*
*frontal lip*
*rear width*
*length*
*width*
*depth*

Intrare: cele 6 caracteristici
Ieșire dorită(target): sexul crabului (M/F)

# Clasificarea crabilor cu RNA

## Pregătirea datelor

- Transformarea valorilor non-numerice (sex M/F) în valori numerice

F: [1 0]
M: [0 1]

sau F = 1
M = 2

sau

F: [0 0]
M: [0 1]

# Clasificarea crabilor cu RNA

## Proiectarea clasificatorului

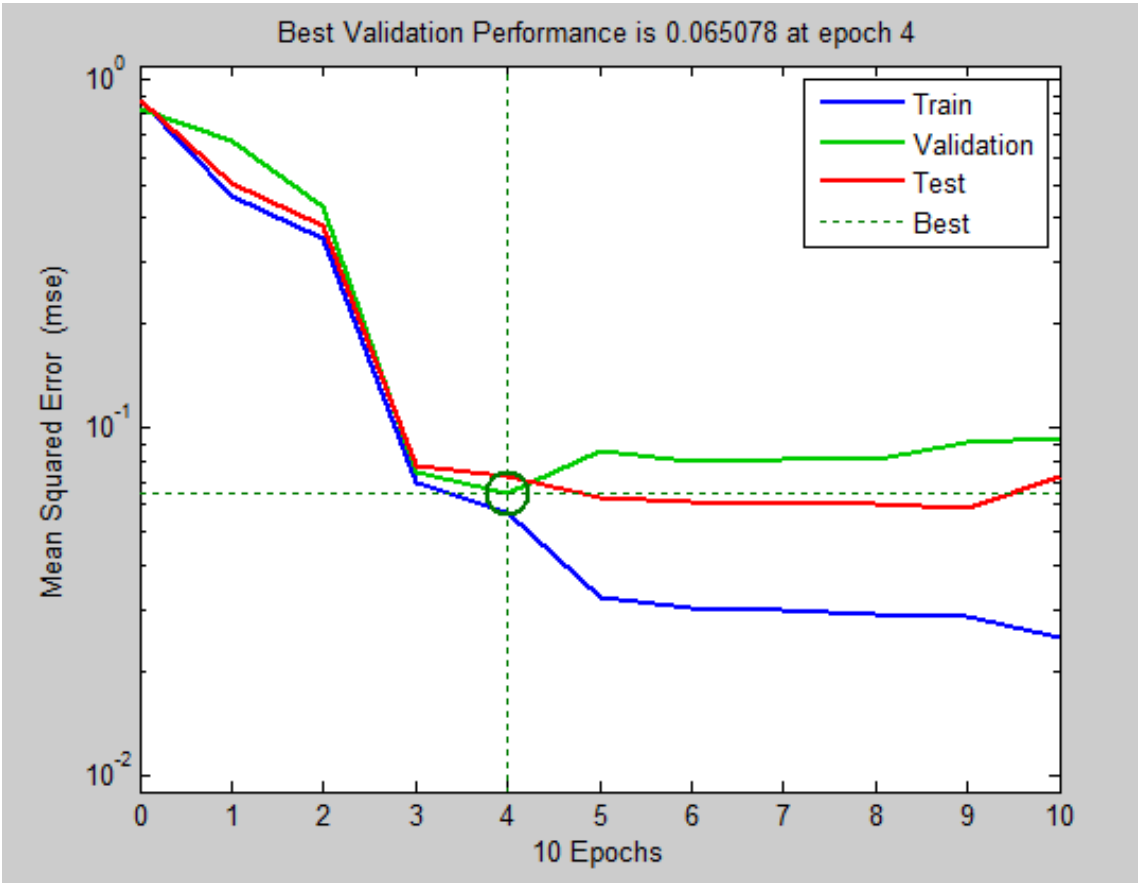
Inițializare ponderi cu valori aleatoare

Rețea feed-forward cu 20 de neuroni pe stratul ascuns

Antrenare – 10 epoci

# Clasificarea crabilor cu RNA

## Testare - grafic



# Clasificarea crabilor cu RNA

## Testare - matricea de clasificare

```
f_f  % Female crabs classified as Female
f_m % Female crabs classified as Male
m_m  % Male crabs classified as Male
m_f  % Male crabs classified as Female
cm = [f_f    f_m;
       m_f    m_m]  % classification matrix
```

Total testing samples: 40

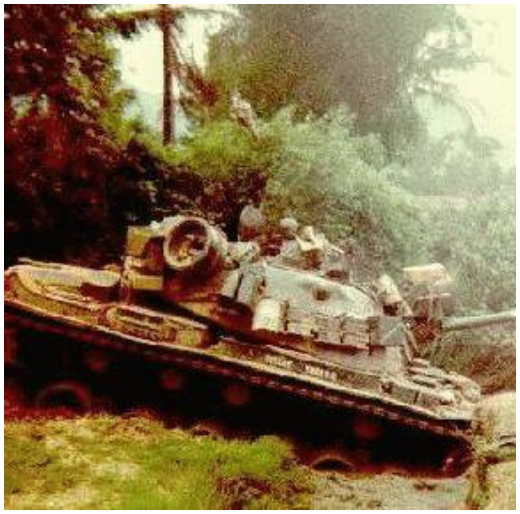
cm =

```
   18    0
    1   21
```

Percentage Correct classification   : 97.500000%
Percentage Incorrect classification : 2.500000%

# Pentagonul și Rețelele Neuronale

Problema:
Anii ’80, Pentagon, boom tehnologic, protecție armată
Planul:
Detecție tancuri din imagini
Soluția:
Prelucrare de imagini folosind RNA



# Pentagonul și Rețelele Neuronale

Implementare:
100 poze copaci + tancuri
100 poze copaci fără tancuri
Antrenare rețea cu 50 poze copaci + tancuri, 50 poze copaci fără tancuri

Validare:
Detecție corectă pentru restul de 50+50 poze

Testare:
Poze noi, rezultate complet aleatoare și incorecte

*Care să fie problema?*

# Pentagonul și Rețelele Neuronale

# Pentagonul și Rețelele Neuronale

cer senin

cer noros

*“The military was now the proud owner of a multi-million dollar mainframe computer that could tell you if it was sunny or not.”*

## Sumar

- RNA – definire, arhitectură ✓
- Funcții de activare ✓
- Instruirea RNA ✓
- Tipuri de probleme rezolvabile cu RNA ✓
- Clasificatoare cu RNA – clasificare liniară ✓
- Instruire prin minimizarea erorii ✓
- Studii de caz ✓

În episodul următor: Entități de Inteligență Artificială pentru sisteme de decizie. Chatbots și robo-advisors.