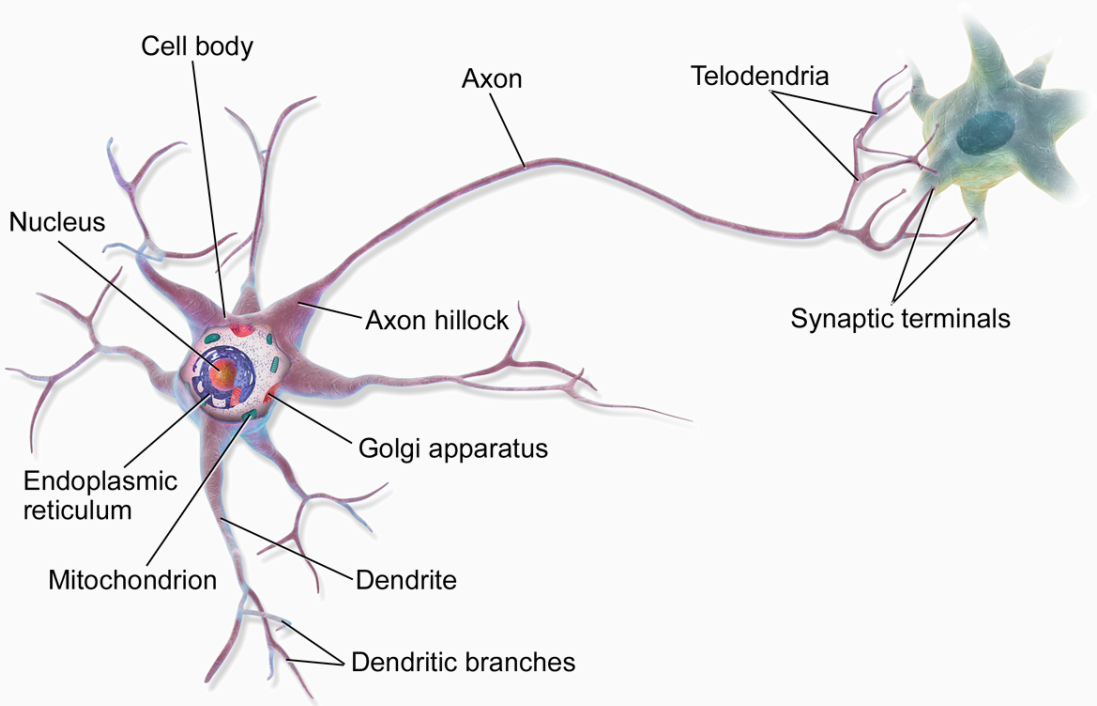
A perceptron

Neuron



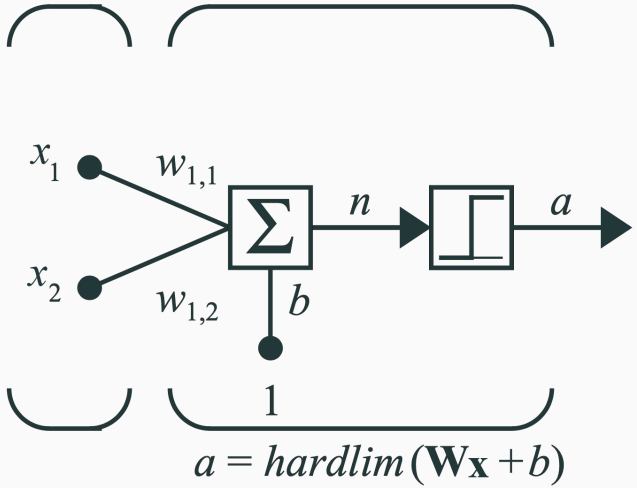
- absztrakt neuron modell: $a = f\left(\sum_i x_i w_i + b\right)$
- x_i: a neuron bemenetei
- w_i: súlyzók
- b: bias
- f: aktivációs függvény
- a: a neuron kimenete

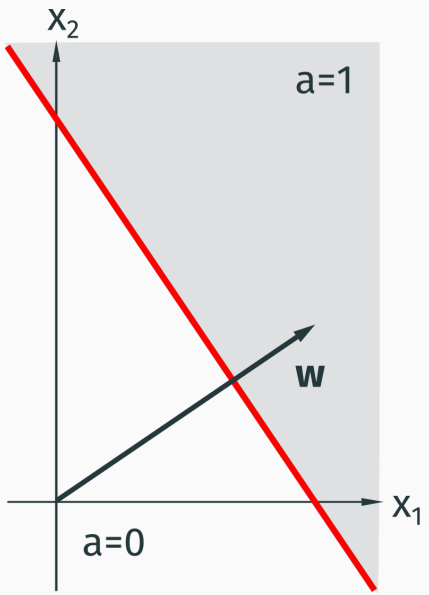
A perceptron

- a perceptron egy olyan neuron, amely **hardlim** aktivációs függvényt* használ

$$f : \mathbb{R} \to \{0,1\}, \quad f(x) = \begin{cases} 1, & \text{ha} \quad x \geq 0 \\ 0 & \text{különben} \end{cases}$$

- $a = hardlim(\mathbf{w}^T\mathbf{x} + b)$
- a perceptron egy **lineáris** és **bináris** osztályozó
- döntési felület: azon bemenetek halmaza, amelyre $\mathbf{w}^T\mathbf{x} + b = 0$
- bináris osztályozási problémák: spam?, rák?, hitel?, stb



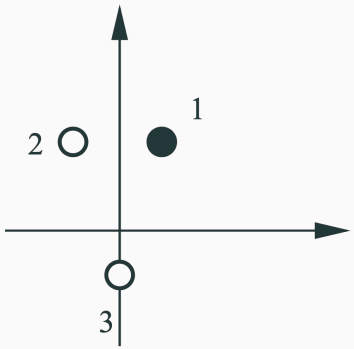


*vagy valamilyen más lépcsőfüggvényt

A perceptron tanítása

- tanítsunk meg egy perceptronnak a következő leképzéseket:

$$\left\{\mathbf{p}_1 = \begin{bmatrix}1\\2\end{bmatrix}, t_1 = 1\right\}, \left\{\mathbf{p}_2 = \begin{bmatrix}-1\\2\end{bmatrix}, t_1 = 0\right\}, \left\{\mathbf{p}_3 = \begin{bmatrix}0\\-1\end{bmatrix}, t_1 = 0\right\}$$



- tanítás: a súlyzóvektor értékeinek beállítása

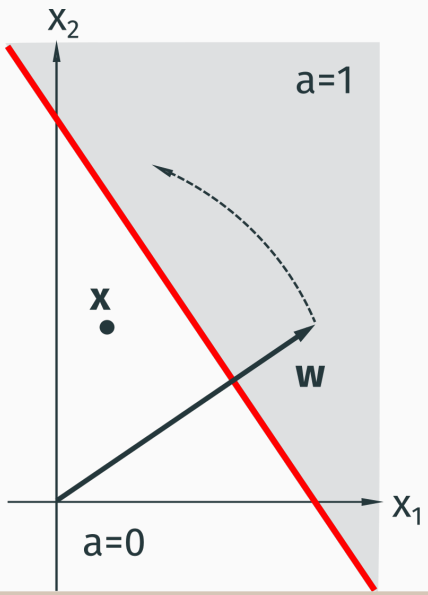
Perceptron tanítási szabály (perceptron learning rule)

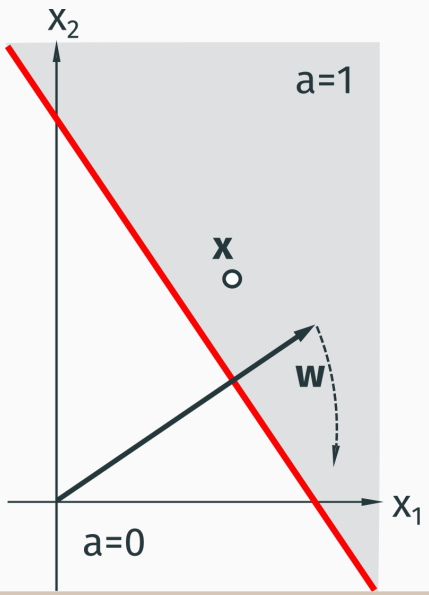
Jelölés:

- a jelölés egyszerűsítéséért: $w_0 \leftarrow b$ és $x_0 \leftarrow 1$
- $\mathbf{x}$ egy bemenet és d a hozzátartozó helyes osztályozás
- $y = f(\mathbf{w}^T\mathbf{x})$ a perceptron kimenete
- $e = (d - y)$ a perceptron hibája az $\mathbf{x}$ bemenetre

Tanítási szabály:

- ha $e = 0$, akkor $\mathbf{w}$ változatlan
- ha $e = 1$, akkor $\mathbf{w} \leftarrow \mathbf{w} + \mathbf{x}$
- ha $e = -1$, akkor $\mathbf{w} \leftarrow \mathbf{w} - \mathbf{x}$



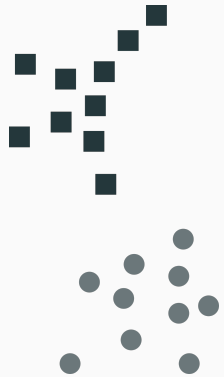


Tanítási algoritmus

- tanítás: a $\mathbf{w}$ iteratív módon történő beállítása
- tanítási halmaz: $T = \{(\mathbf{x}^1, d^1), (\mathbf{x}^2, d^2), \ldots, (\mathbf{x}^N, d^N)\}$
- bizonyítottan konvergál, ha T lineárisan szétválasztható

```
function w = PerceptronLearning(x, d)
  [N, n] = size(x);
  lr = 0.01;
  w = randn(n,1);
  E = 1;
  while E ~= 0
  E = 0;
    for i = 1:N
      yi = hardlim(x(i,:)*w);
      ei = d(i)-yi;
      w = w + lr*ei*x(i,:)';
      E = E + ei^2;
    end
  end
end
```

A perceptron korlátai



(a) lineárisan szétválasztható

(b) lineárisan nem szétválasztható

- hány megoldás van? melyiket preferáljuk?
- mi az a legegyszerűbb logikai függvény, amit a perceptron nem tud megtanulni?
- több osztály? puha margó (soft margin)?

Példa

- WIN32 hívások alapján döntsük el, hogy egy processz rosszindulatú-e
- nyers tanítási halmaz:

```
CoCreateInstance,CoInitializeEx,CreatePipe,CreateProcess,Cre,...,malicious
CoInitializeEx,Connect,CreateRemoteThread,CreateService,Crea,...,benign
AdjustTokenPrivileges,CoCreateInstance,CoInitializeEx,Connec,...,malicious
Bind,CoInitializeEx,CreateMutex,CreateRemoteThread,CreateThr,...,benign
AdjustTokenPrivileges,Bind,CoInitializeEx,CreateMutex,Create,...,benign
AdjustTokenPrivileges,Connect,CreatePipe,CreateProcess,Creat,...,benign
AdjustTokenPrivileges,CoCreateInstance,Connect,CreatePipe,Cr,...,malicious
AdjustTokenPrivileges,CoCreateInstance,CreateService,CryptAc,...,malicious
...
```

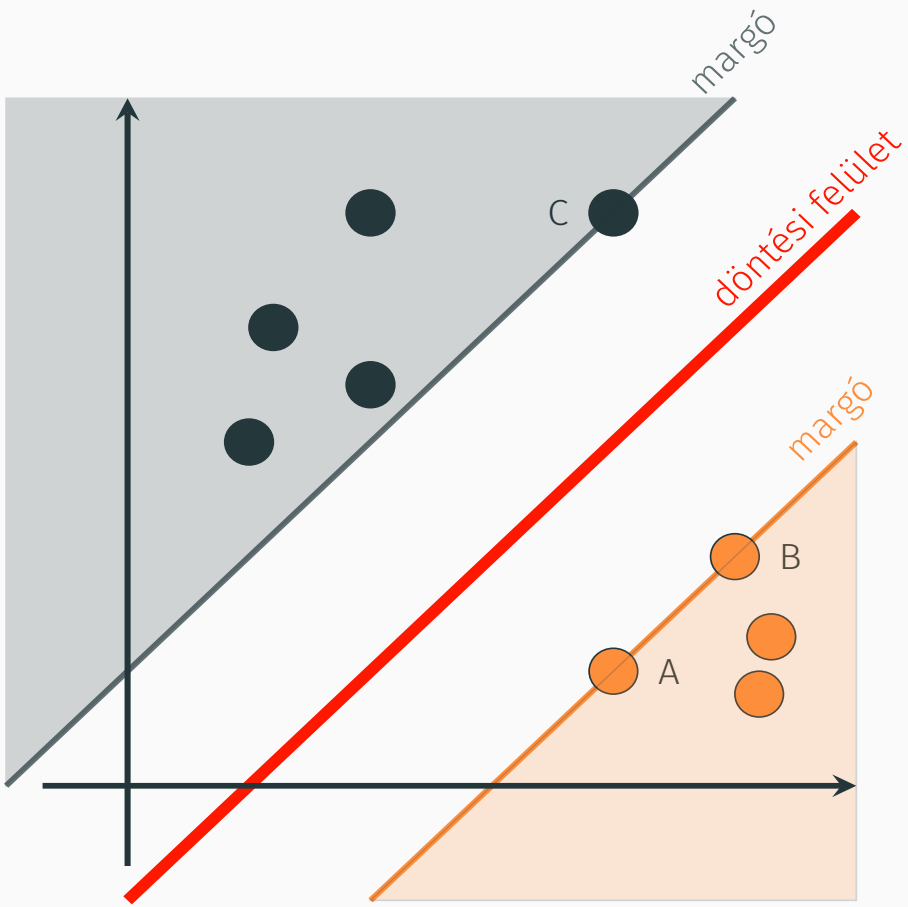
- összesen $n = 167$ különböző függvény, $\{f_1, f_2, \ldots f_n\}$
- kimenet kódolása: malicious — 1, benign — 0
- $F = \{f_k\}$ bemenet kódolása: 1, ha $f_i \in F$, 0 különben
- **tesztelési halmaz**
- a perceptron viselkedését egy **konfúziós mátrixban** összesítjük:

		actual class	
		benign	malicious
pred class	benign	#true positives	#false positives
	malicious	#false negatives	#true negatives

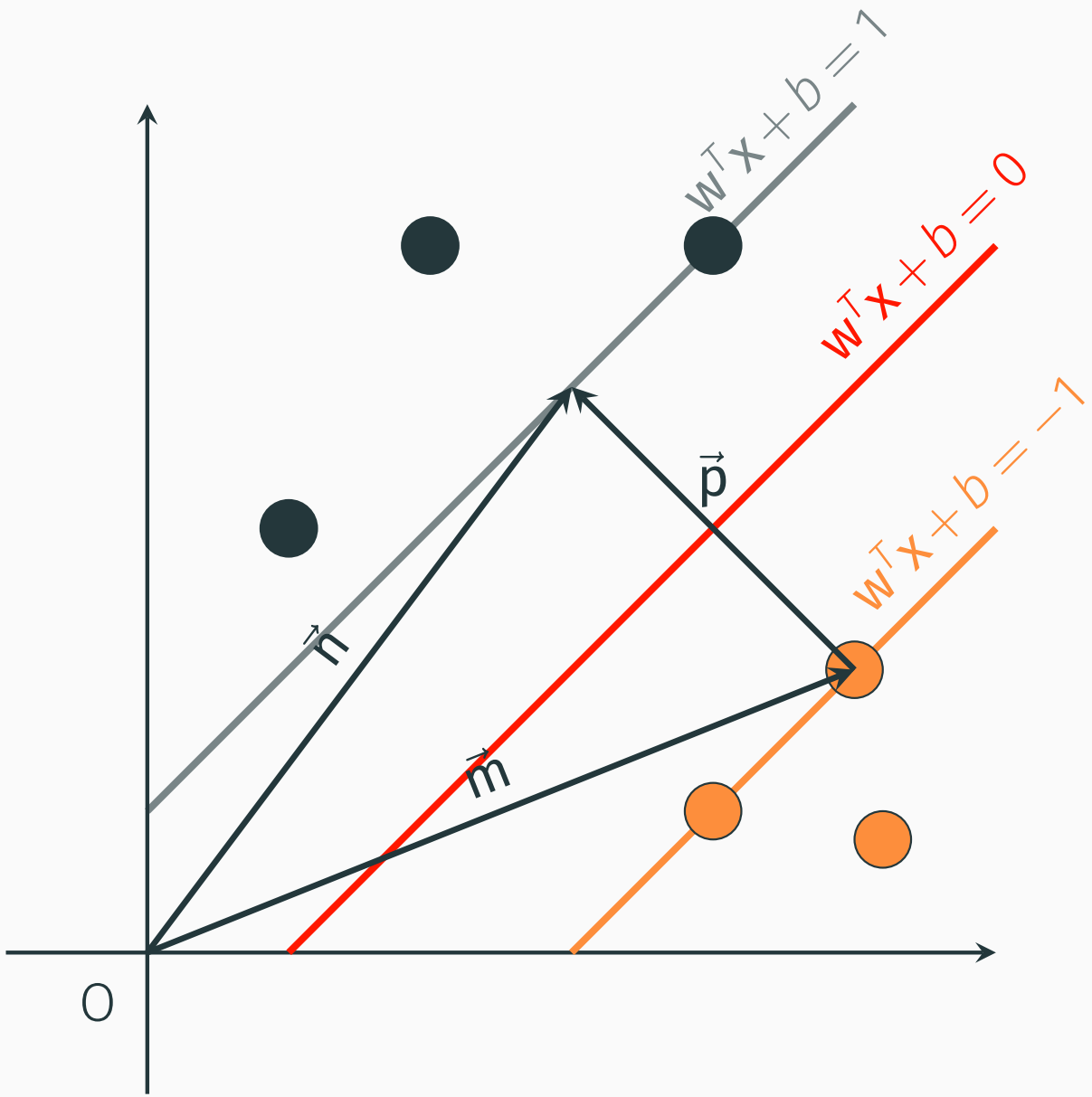
Support vector machine (SVM)

Szupport vektor gép

- maximális margó (maximal margin)



A margó méretének kiszámítása



- $||p|| = \frac{2}{||w||}$

Maximális margó kiszámítása

- $\{\mathbf{x}_i, y_i\}_{i=1..N}$ tanítási halmaz, $y_i = \pm 1$

- $\mathbf{w}^T\mathbf{x}_i + b \geq 1$, ha $y_i = 1$
- $\mathbf{w}^T\mathbf{x}_i + b \leq -1$, ha $y_i = -1$
- $\frac{2}{||\mathbf{w}||} \rightarrow \mathbf{max} \Leftrightarrow ||\mathbf{w}|| \rightarrow \mathbf{min}$

- optimizációs probléma: $\underset{\mathbf{w}}{\text{argmin}}||\mathbf{w}||$, úgy hogy $y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1$
- implementációk: `quadprog` (Matlab), `qp` (Octave)

Könyvészet

- Christopher Bishop: Pattern Recognition and Machine Learning, 2006
- Simon Haykin: Neural Networks and Learning Machines, 2009
- Alexandre Kowalczyk: Support Vector Machines Succintly
- Martin Hagan: Neural Network Design 2nd ed, 2014