

Adatbányászat: Osztályozás További módszerek

5. fejezet

Tan, Steinbach, Kumar
Bevezetés az adatbányászatba
előadás-fóliák
fordította
Ispány Márton

Osztályozási szabályok

- „Ha...akkor...” szabályok összességével osztályozzuk a rekordokat
- Szabály: $(Feltétel) \rightarrow y$
 - ahol
 - ◆ *Feltétel* attributumok konjunkciója
 - ◆ y osztály címke
 - *Baloldal*: a szabály feltétele, előzménye
 - *Jobboldal*: a szabály következménye
 - Példák osztályozási szabályokra:
 - ◆ $(Vértípus=Meleg) \wedge (Tojásrakás=Igen) \rightarrow Madarak$
 - ◆ $(Adózott\ jövedelem < 50K) \wedge (Visszatérítés=Igen) \rightarrow Adóelkerülés=Nem$

Példa osztályozási szabályra

Name	Blood Type	Give Birth	Can Fly	Live in Water	Class
human	warm	yes	no	no	mammals
python	cold	no	no	no	reptiles
salmon	cold	no	no	yes	fishes
whale	warm	yes	no	yes	mammals
frog	cold	no	no	sometimes	amphibians
komodo	cold	no	no	no	reptiles
bat	warm	yes	yes	no	mammals
pigeon	warm	no	yes	no	birds
cat	warm	yes	no	no	mammals
leopard shark	cold	yes	no	yes	fishes
turtle	cold	no	no	sometimes	reptiles
penguin	warm	no	no	sometimes	birds
porcupine	warm	yes	no	no	mammals
eel	cold	no	no	yes	fishes
salamander	cold	no	no	sometimes	amphibians
gila monster	cold	no	no	no	reptiles
platypus	warm	no	no	no	mammals
owl	warm	no	yes	no	birds
dolphin	warm	yes	no	yes	mammals
eagle	warm	no	yes	no	birds

R1: (Élveszülő = nem) \wedge (Tud repülni = igen) \rightarrow Madár

R2: (Élveszülő = nem) \wedge (Vízben él = igen) \rightarrow Hal

R3: (Élveszülő = igen) \wedge (Vér = meleg) \rightarrow Emlős

R4: (Élveszülő = nem) \wedge (Tud repülni = nem) \rightarrow Hüllő

R5: (Vízben él = néha) \rightarrow Kétéltű

Osztályozási szabályok alkalmazása

- Az r szabály **lefed**i az x esetet ha az eset attributumai kielégítik a szabály feltételeit.

R1: (Élveszülő = nem) \wedge (Tud repülni = igen) \rightarrow Madár

R2: (Élveszülő = nem) \wedge (Vízben él = igen) \rightarrow Hal

R3: (Élveszülő = igen) \wedge (Vér = meleg) \rightarrow Emlős

R4: (Élveszülő = nem) \wedge (Tud repülni = nem) \rightarrow Hüllő

R5: (Vízben él = néha) \rightarrow Kétéltű

Name	Blood Type	Give Birth	Can Fly	Live in Water	Class
hawk	warm	no	yes	no	?
grizzly bear	warm	yes	no	no	?

Az R1 szabály lefed i a sólyom \Rightarrow madár szabályt

Az R3 szabály lefed i a grizzly \Rightarrow emlős szabályt

Lefedettség és pontosság

- Egy szabály lefedettsége:
 - Azon rekordok aránya, amelyek kielégítik a szabály feltételét.
- Egy szabály pontossága:
 - Azon rekordok aránya, amelyek egyaránt kielégítik a szabály feltételét és következményét.

Tid	Vissza- térítés	Családi állapot	Jöve- delem	Osztály
1	Igen	Nőtlen	125K	Nem
2	Nem	Házás	100K	Nem
3	Nem	Nőtlen	70K	Nem
4	Igen	Házás	120K	Nem
5	Nem	Elvált	95K	Igen
6	Nem	Házás	60K	Nem
7	Igen	Elvált	220K	Nem
8	Nem	Nőtlen	85K	Igen
9	Nem	Házás	75K	Nem
10	Nem	Nőtlen	90K	Igen

(Állapot=Nőtlen) → Nem

Lefedettség = 40%, Pontosság = 50%

Hogy működnek az osztályozási szabályok?

R1: (Élveszülő = nem) \wedge (Tud repülni = igen) \rightarrow Madár

R2: (Élveszülő = nem) \wedge (Vízben él = igen) \rightarrow Hal

R3: (Élveszülő = igen) \wedge (Vér = meleg) \rightarrow Emlős

R4: (Élveszülő = nem) \wedge (Tud repülni = nem) \rightarrow Hüllő

R5: (Vízben él = néha) \rightarrow Kételtű

Name	Blood Type	Give Birth	Can Fly	Live in Water	Class
lemur	warm	yes	no	no	?
turtle	cold	no	no	sometimes	?
dogfish shark	cold	yes	no	yes	?

A lemur kiváltja az R3 szabályt, így emlősnek osztályozzuk.

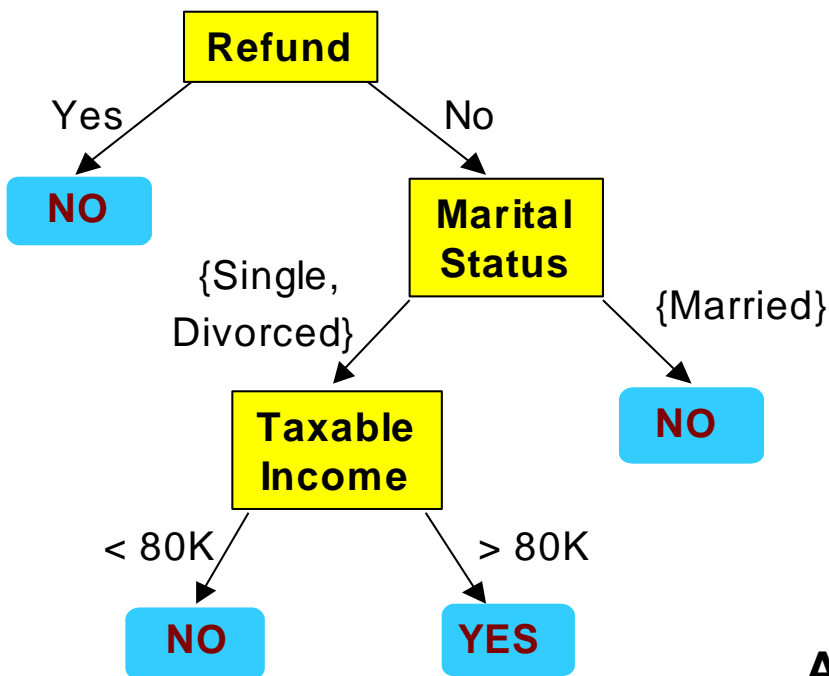
A teknős egyaránt kiváltja az R4 és R5 szabályokat.

A kutyahal cápa egyik szabályt sem váltja ki.

Osztályozási szabályok jellemzése

- Teljesen kizáró szabályok
 - Egy osztályozó teljesen kizáró szabályokból áll, ha a szabályok függetlenek egymástól (a feltételek metszete üres).
 - Minden rekordot legfeljebb egy szabály fed le.
- Kimerítő szabályok
 - Egy osztályozó kimerítő lefedés, ha az attributum értékek minden lehetséges kombinációját tartalmazza a feltételekben.
 - Minden rekordot lefed legalább egy szabály.

Döntési fáktól a szabályokig



Classification Rules

(Refund=Yes) ==> No

(Refund=No, Marital Status={Single, Divorced}, Taxable Income<80K) ==> No

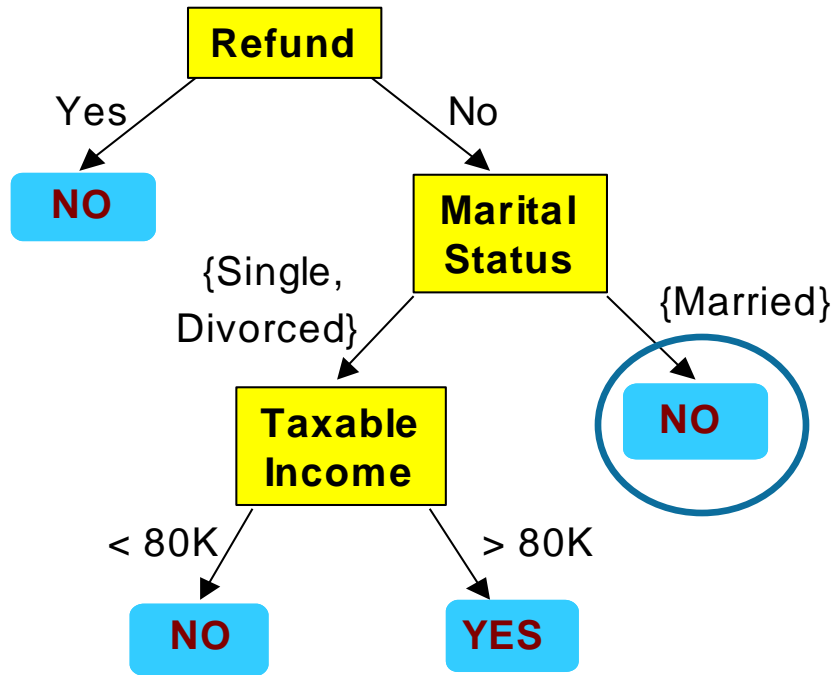
(Refund=No, Marital Status={Single, Divorced}, Taxable Income>80K) ==> Yes

(Refund=No, Marital Status={Married}) ==> No

A szabályok teljesen kizáróak és kimerítőek.

A szabály halmaz pontosan annyi információt tartalmaz mint a fa.

Szabályok egyszerűsítése



Tid	Visszatérítés	Családi állapot	Jövedelem	Csalás
1	Igen	Nőtlen	125K	Nem
2	Nem	Házas	100K	Nem
3	Nem	Nőtlen	70K	Nem
4	Igen	Házas	120K	Nem
5	Nem	Elvált	95K	Igen
6	Nem	Házas	60K	Nem
7	Igen	Elvált	220K	Nem
8	Nem	Nőtlen	85K	Igen
9	Nem	Házas	75K	Nem
10	Nem	Nőtlen	90K	Igen

Kezdeti szabály: $(\text{Visszatérítés}=\text{Nem}) \wedge (\text{Állapot}=\text{Házas}) \rightarrow \text{Nem}$

Egyszerűsített szabály: $(\text{Állapot}=\text{Házas}) \rightarrow \text{Nem}$

Az egyszerűsítés hatása

- A szabályok már nem lesznek teljesen kizáróak.
 - Egy rekord egynél több szabályt is kiválthat.
 - Megoldás?
 - ◆ Szabályok rendezése
 - ◆ Rendezetlen szabályok – használjunk szavazási sémákat
- A szabályok már nem lesznek kimerítőek.
 - Egy rekord egyetlen szabályt sem vált ki.
 - Megoldás?
 - ◆ Használjunk egy alapértelmezett osztályt.

Rendezett szabály halmazok

- A szabályokat prioritásuk szerint sorba rendezzük.
 - Egy rendezett szabályhalmazt döntési listának nevezünk.
- Egy teszt rekordot inputként kap az osztályozó.
 - A legelső osztályhoz rendeljük, amelyet kivált.
 - Ha egyetlen szabályt sem vált ki, akkor az alapértelmezett osztályba kerül.

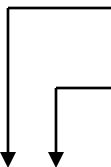
R1: (Élveszülő = nem) \wedge (Tud repülni = igen) \rightarrow Madár

R2: (Élveszülő = nem) \wedge (Vízben él = igen) \rightarrow Hal

R3: (Élveszülő = igen) \wedge (Vér = meleg) \rightarrow Emlős

R4: (Élveszülő = nem) \wedge (Tud repülni = nem) \rightarrow Hüllő

R5: (Vízben él = sometimes) \rightarrow Kétéltű



Name	Blood Type	Give Birth	Can Fly	Live in Water	Class
turtle	cold	no	no	sometimes	?

Szabály rendező sémák

- Szabály alapú rendezés
 - Az egyedi szabályokat minőségük alapján rendezzük.
- Osztály alapú rendezés
 - Az egy osztályhoz tartozó szabályok együtt fordulnak elő.

Rule-based Ordering

(Refund=Yes) ==> No

(Refund=No, Marital Status={Single,Divorced},
Taxable Income<80K) ==> No

(Refund=No, Marital Status={Single,Divorced},
Taxable Income>80K) ==> Yes

(Refund=No, Marital Status={Married}) ==> No

Class-based Ordering

(Refund=Yes) ==> No

(Refund=No, Marital Status={Single,Divorced},
Taxable Income<80K) ==> No

(Refund=No, Marital Status={Married}) ==> No

(Refund=No, Marital Status={Single,Divorced},
Taxable Income>80K) ==> Yes

Osztályozási szabályok építése

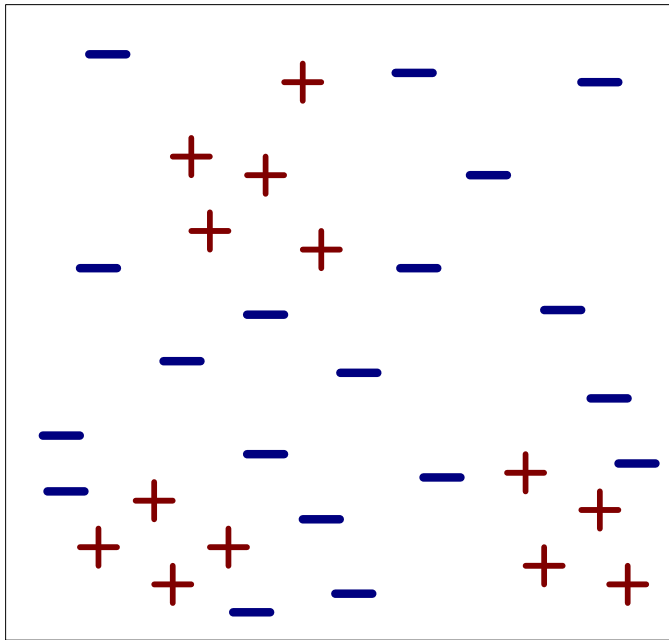
- Közvetlen módszerek:
 - ◆ Szabály kinyerés közvetlenül az adatokból.
 - ◆ Példák: RIPPER, CN2, Holte 1R módszere.

- Közvetett módszerek:
 - ◆ Szabály kinyerés más osztályozási módszerekből (pl. döntési fák, neurális hálók stb.).
 - ◆ Példa: C4.5 szabályok.

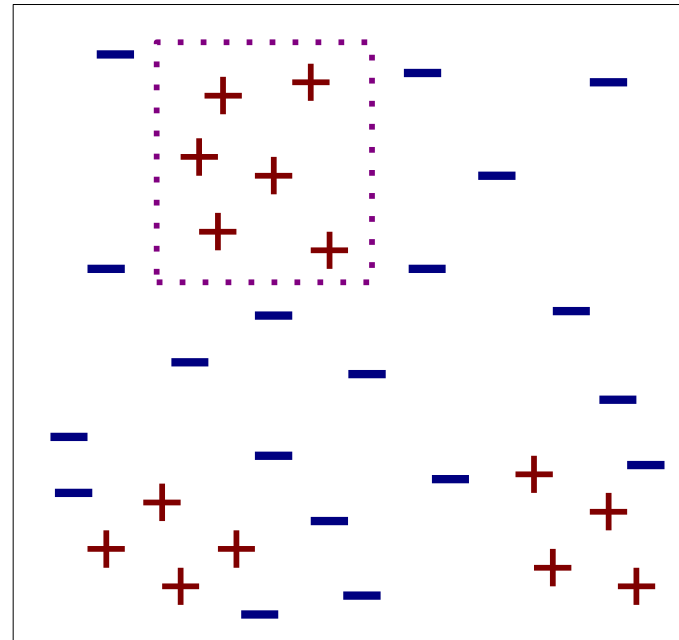
Közvetlen módszer: Szekvenciális lefedés

1. Induljunk ki az üres szabályból.
2. Hozzunk létre egy szabályt a Learn-One-Rule függvény segítségével.
3. Távolítsuk el azokat a tanító rekordokat, amelyeket lefed a szabály.
4. Ismételjük a (2) és (3) lépést ameddig a megállási kritérium nem teljesül.

Példa szekvenciális lefedésre

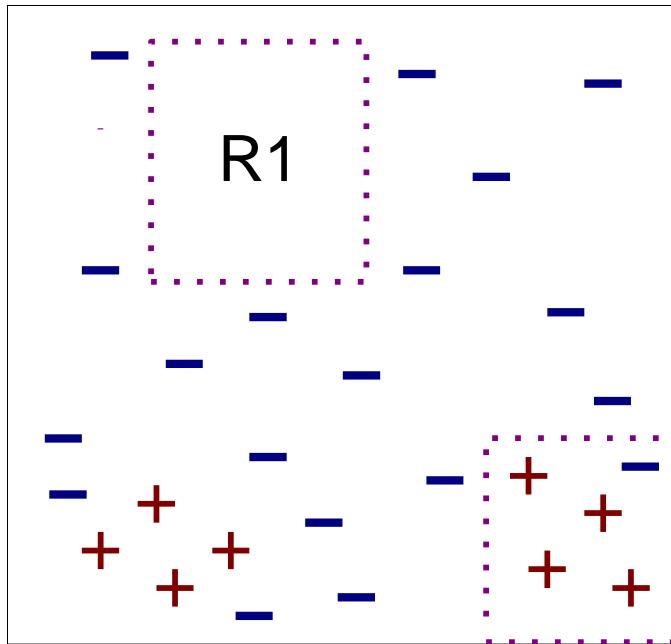


(i) Original Data

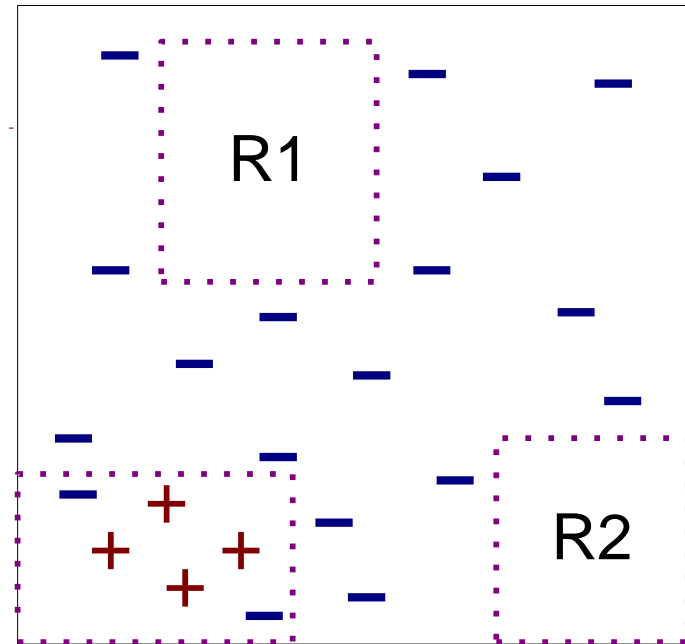


(ii) Step 1

Példa szekvenciális lefedésre



(iii) Step 2



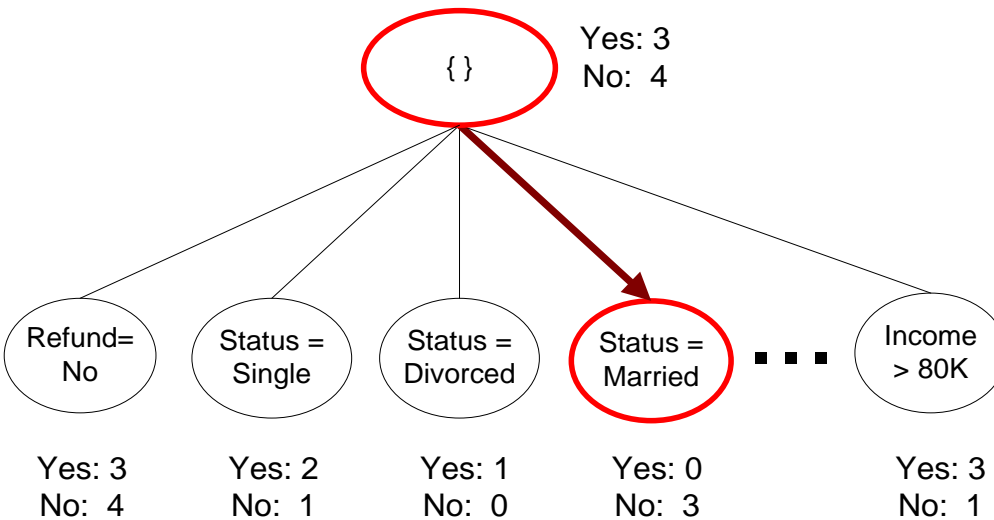
(iv) Step 3

A szekvenciális lefedés szempontjai

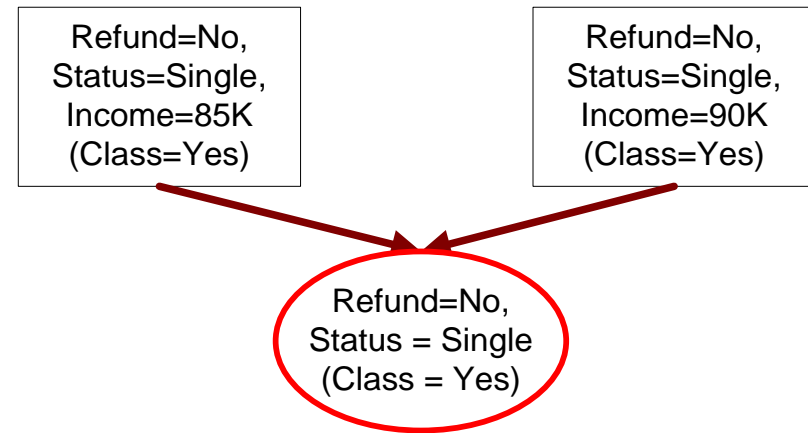
- Szabály építés
- Eset kizárás
- Szabály kiértékelés
- Leállási kritérium
- Szabály tisztítás

Szabály építés

- Két általános stratégia



(a) General-to-specific



(b) Specific-to-general

Példák szabály építésre

- CN2 algoritmus:

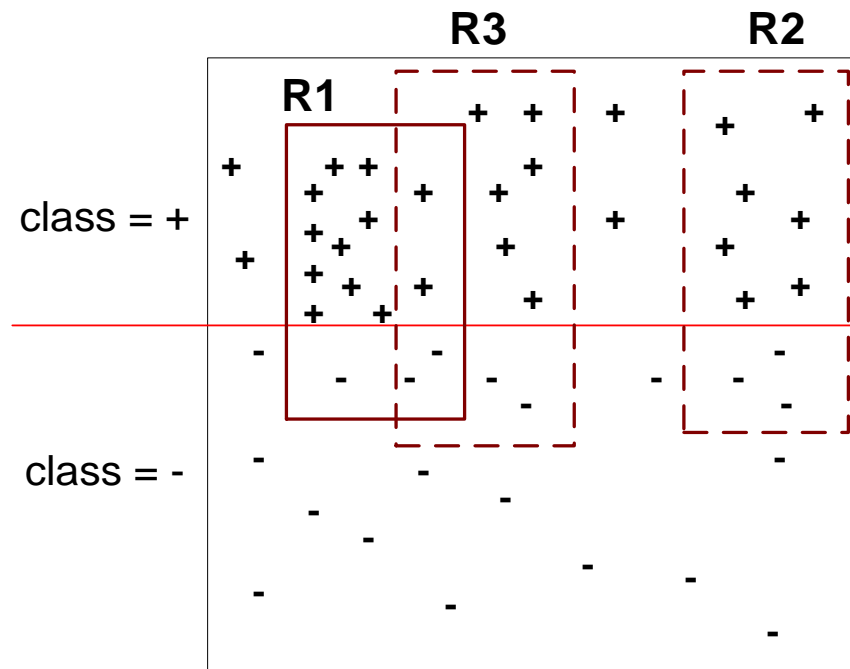
- Induljunk ki az üres szabályból: $\{\}$.
- Bővítsük úgy, hogy közben az entrópiát minimalizáljuk: $\{A\}$, $\{A,B\}$, ...
- Határozzuk meg a szabály következményét a szabály által lefedett esetek többségi osztályát véve.

- RIPPER algoritmus:

- Induljunk ki az üres szabályból : $\{\} \Rightarrow$ osztály.
- Bővítsük úgy, hogy a FOIL-féle információ nyereséget maximalizáljuk:
 - ◆ R_0 : $\{\} \Rightarrow$ osztály (kezdeti szabály)
 - ◆ R_1 : $\{A\} \Rightarrow$ osztály (szabály a bővítés után)
 - ◆ $\text{Nyeresség}(R_0, R_1) = t [\log (p_1/(p_1+n_1)) - \log (p_0/(p_0 + n_0))]$,
 - ◆ ahol t : R_0 és R_1 által lefedett pozitív esetek száma,
 p_0 : R_0 által lefedett pozitív esetek száma,
 n_0 : R_0 által lefedett negatív esetek száma,
 p_1 : R_1 által lefedett pozitív esetek száma,
 n_1 : R_1 által lefedett negatív esetek száma.

Eset kizárás

- Miért van szükség eset kizárásra?
 - Különben a következő szabály megegyezik az előzővel.
- Miért töröljük pozitív eseteket?
 - Biztosítsuk a következő szabály különbözőségét.
- Miért töröljük negatív eseteket?
 - Megelőzzük a szabály pontosságának alulbecslését.
 - Hasonlítsuk össze az R2 és R3 szabályokat az ábrán.



Szabály kiértékelés

- Mérőszámok:

- Pontosság = $\frac{n_c}{n}$

- Laplace = $\frac{n_c + 1}{n + k}$

- M-becslés = $\frac{n_c + kp}{n + k}$

n : a szabály által lefedett esetek száma

n_c : a szabály által lefedett pozitív esetek száma

k : osztályok száma

p : a pozitív eset apriori valószínűsége

Leállási feltétel és szabály tisztítás

- Leállási feltétel
 - Számoljuk ki a nyereséget.
 - Ha a nyereség nem szignifikáns, akkor dobjuk el az új szabályt.
- Szabály tisztítás
 - Hasonló döntési fák utó-tisztításához.
 - Hiba csökkentés tisztítással:
 - ◆ Hagyjunk el a szabályból egy kifejezést.
 - ◆ Hasonlítsuk össze a tisztítás előtti és utáni hibát az ellenőrző adatállományon.
 - ◆ Ha a hiba javul, akkor tisztítsunk a kifejezés elhagyásával.

A közvetlen módszer vázlatja

- Építsünk egy egyszerű szabályt.
- Távolítsunk el eseteket a szabály alapján.
- Egyszerűsítsük a szabályt (ha szükséges).
- Adjuk hozzá a szabályt az aktuális szabály halmazhoz.
- Ismételjük a fenti lépéseket.

Közvetlen módszer: RIPPER

- Bináris feladat esetén válasszuk pozitív osztálynak az egyik és negatív osztálynak a másik osztályt.
 - Tanítsunk szabályokat a pozitív osztályra.
 - Legyen a negatív osztály az alapértelmezett osztály.
- Több osztályos feladat esetén:
 - Rendezzük az osztályokat növekvő osztály–gyakoriság szerint (azoknak az eseteknek az aránya, melyek egy osztályhoz tartoznak).
 - Először tanítsunk egy szabály halmazt a legkisebb osztályra, kezeljük a maradékot negatív osztályként.
 - Ismételjük meg a következő legkisebb osztállyal mint pozitív osztály.

Közvetlen módszer: RIPPER

- Szabály építés:
 - Induljunk ki az üres szabályból.
 - Bővítsük addig míg a FOIL információ nyereség javul.
 - Álljunk meg amikor a szabály tovább már nem fedile a negatív eseteket.
 - Közvetlenül tisztítsuk a szabályt járulékos hiba tisztítással.
 - A tisztítás mérőszáma: $v = (p-n)/(p+n)$
 - ◆ p: a szabály által lefedett pozitív esetek száma az ellenőrző adatállományban,
 - ◆ n: a szabály által lefedett negatív esetek száma az ellenőrző adatállományban.
 - Tisztítási módszer: töröljük a feltételek olyan véges sorozatát, amely maximalizálja v-t.

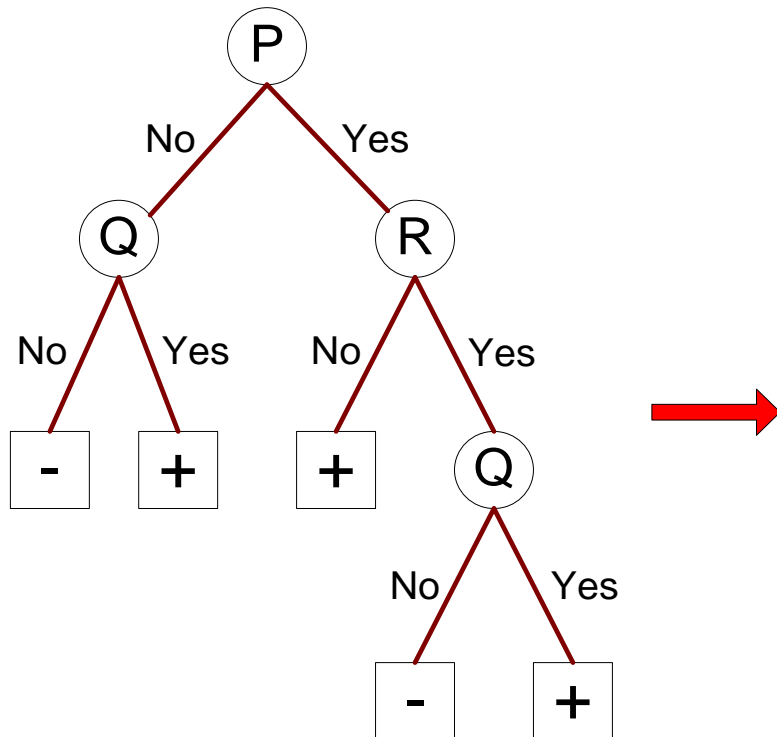
Közvetlen módszer: RIPPER

- Szabály halmaz építése:
 - Használjunk szekvenciálisan lefedő algoritmust.
 - ◆ Keressük meg azt a legjobb szabályt, amely lefedi a pozitív esetek aktuális halmazát.
 - ◆ Elimináljuk a szabály által lefedett pozitív és negatív eseteket.
 - Mindig mikor egy szabállyal bővítjük a szabály halmazt számoljuk ki az új leíró hosszt.
 - ◆ Álljunk le az új szabály hozzáadásával, ha annak leíró hossza d bittel nagyobb mint az eddig kapott legkisebb leíró hossz.

Közvetlen módszer: RIPPER

- Optimalizáljuk a szabályhalmazt:
 - Az R szabályhalmaz minden r szabályára
 - ◆ Tekintsünk 2 alternatív szabályt:
 - Helyettesítő szabály (r^*): építsünk új szabályt előlről.
 - Módosított szabály (r'): bővítsünk az r kiterjesztésével.
 - ◆ Hasonlítsuk össze az r szabályhalmazt az r^* és r' szabályhalmazokkal.
 - ◆ Válasszuk azt a a szabályhalmazt, amely minimális lesz az MDL elv alapján.
 - Ismételjük a szabály generálást és optimalizálást a fennmaradó pozitív esetekre.

Közvetett módszerek



Rule Set

r1: (P=No,Q=No) ==> -

r2: (P=No,Q=Yes) ==> +

r3: (P=Yes,R=No) ==> +

r4: (P=Yes,R=Yes,Q=No) ==> -

r5: (P=Yes,R=Yes,Q=Yes) ==> +

Közvetett módszerek: C4.5 szabályok

- Nyerjük ki szabályokat egy tisztítatlan (teljes) döntési fából.
- Minden $r: A \rightarrow y$ szabályra
 - Tekintsünk egy $r': A' \rightarrow y$ alternatív szabályt, ahol A' -t úgy kapjuk, hogy A -ból törölünk egy kifejezést.
 - Hasonlítsuk össze az r és az összes r' pesszimista hiba rátáját.
 - Tisztítsunk amennyiben egy r' -nek kisebb a pesszimista hiba rátája.
 - Ismételjük amíg már nem tudjuk javítani az általánosítási hibát.

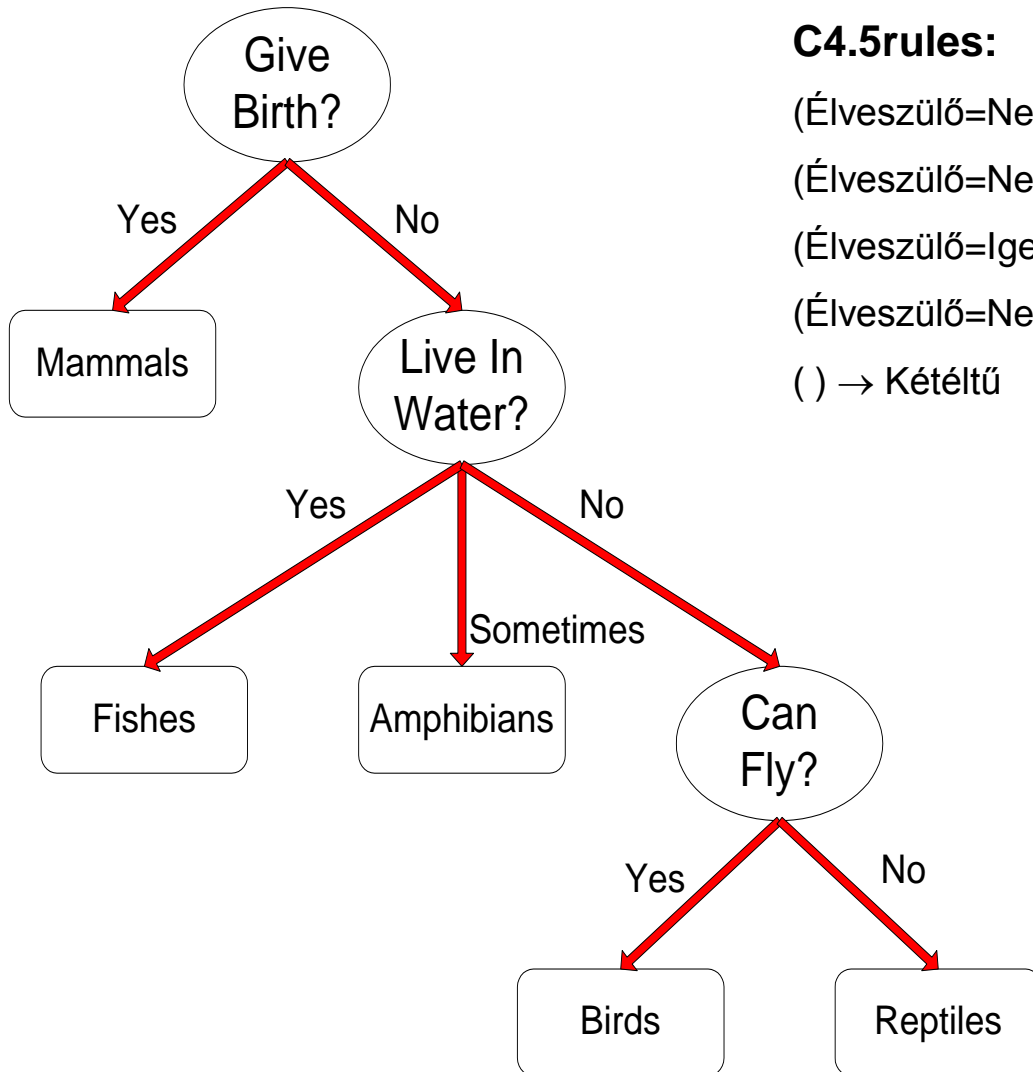
Közvetett módszer: C4.5 szabályok

- A szabályok rendezése helyett rendezzük szabályok részhalmazait (**osztály rendezés**).
 - Minden részhalmaz szabályoknak egy olyan összessége, melynek következménye ugyanaz (osztály).
 - Számoljuk ki minden részhalmaz leíró hosszát.
 - ◆ Leíró hossz = $L(\text{error}) + g L(\text{model})$,
 - ◆ g egy olyan paraméter, amely figyelembe veszi a szabályhalmazban lévő redundáns attribútumokat (alapérték = 0.5).

Példa

Name	Give Birth	Lay Eggs	Can Fly	Live in Water	Have Legs	Class
human	yes	no	no	no	yes	mammals
python	no	yes	no	no	no	reptiles
salmon	no	yes	no	yes	no	fishes
whale	yes	no	no	yes	no	mammals
frog	no	yes	no	sometimes	yes	amphibians
komodo	no	yes	no	no	yes	reptiles
bat	yes	no	yes	no	yes	mammals
pigeon	no	yes	yes	no	yes	birds
cat	yes	no	no	no	yes	mammals
leopard shark	yes	no	no	yes	no	fishes
turtle	no	yes	no	sometimes	yes	reptiles
penguin	no	yes	no	sometimes	yes	birds
porcupine	yes	no	no	no	yes	mammals
eel	no	yes	no	yes	no	fishes
salamander	no	yes	no	sometimes	yes	amphibians
gila monster	no	yes	no	no	yes	reptiles
platypus	no	yes	no	no	yes	mammals
owl	no	yes	yes	no	yes	birds
dolphin	yes	no	no	yes	no	mammals
eagle	no	yes	yes	no	yes	birds

Összevetés: C4.5, C4.5 szabályok, RIPPER



C4.5rules:

(Élveszülő=Nem, Tud úszni=Igen) → Madár

(Élveszülő=Nem, Vízben él=Igen) → Hal

(Élveszülő=Igen) → Emlős

(Élveszülő=Nem, Tud repülni=Nem, Vízben él=Nem) → Hüllő

() → Kétéltű

RIPPER:

(Vízben él=Igen) → Hal

(Van lába=Nem) → Hüllő

(Élveszülő=Nem, Tud repülni=Nem, Vízben él=Nem) → Hüllő

(Tud repülni=Igen, Élveszülő=Nem) → Madár

() → Emlős

Összevetés: C4.5, C4.5 szabályok, RIPPER

C4.5 és C4.5 szabályok:

		Jósolt osztály				
		Kétéltű	Hal	Hüllő	Madár	Emlős
Valódi osztály	Kétéltű	2	0	0	0	0
	Hal	0	2	0	0	1
	Hüllő	1	0	3	0	0
	Madár	1	0	0	3	0
	Emlős	0	0	1	0	6

RIPPER:

		Jósolt osztály				
		Kétéltű	Hal	Hüllő	Madár	Emlős
Valódi osztály	Kétéltű	0	0	0	0	2
	Hal	0	3	0	0	0
	Hüllő	0	0	3	0	1
	Madár	0	0	1	2	1
	Emlős	0	2	1	0	4

Osztályozási szabályok előnyei

- Legalább annyira kifejezőek mint a döntési fák.
- Könnyen interpretálhatóak.
- Könnyen generálhatóak.
- Gyorsan osztályozhatóak általuk az új esetek.
- Hatékonyságuk összevethető a döntési fákéval.

Eset alapú osztályozók

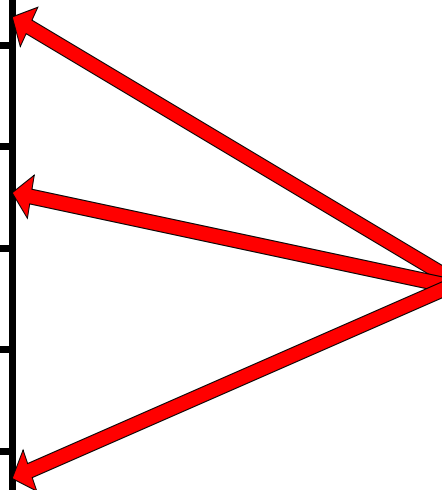
Set of Stored Cases

Atr1	AtrN	Class
			A
			B
			B
			C
			A
			C
			B

- Letároljuk a tanító rekordokat
- A tanító rekordokat használjuk az új esetek osztályainak előrejelzésére

Unseen Case

Atr1	AtrN



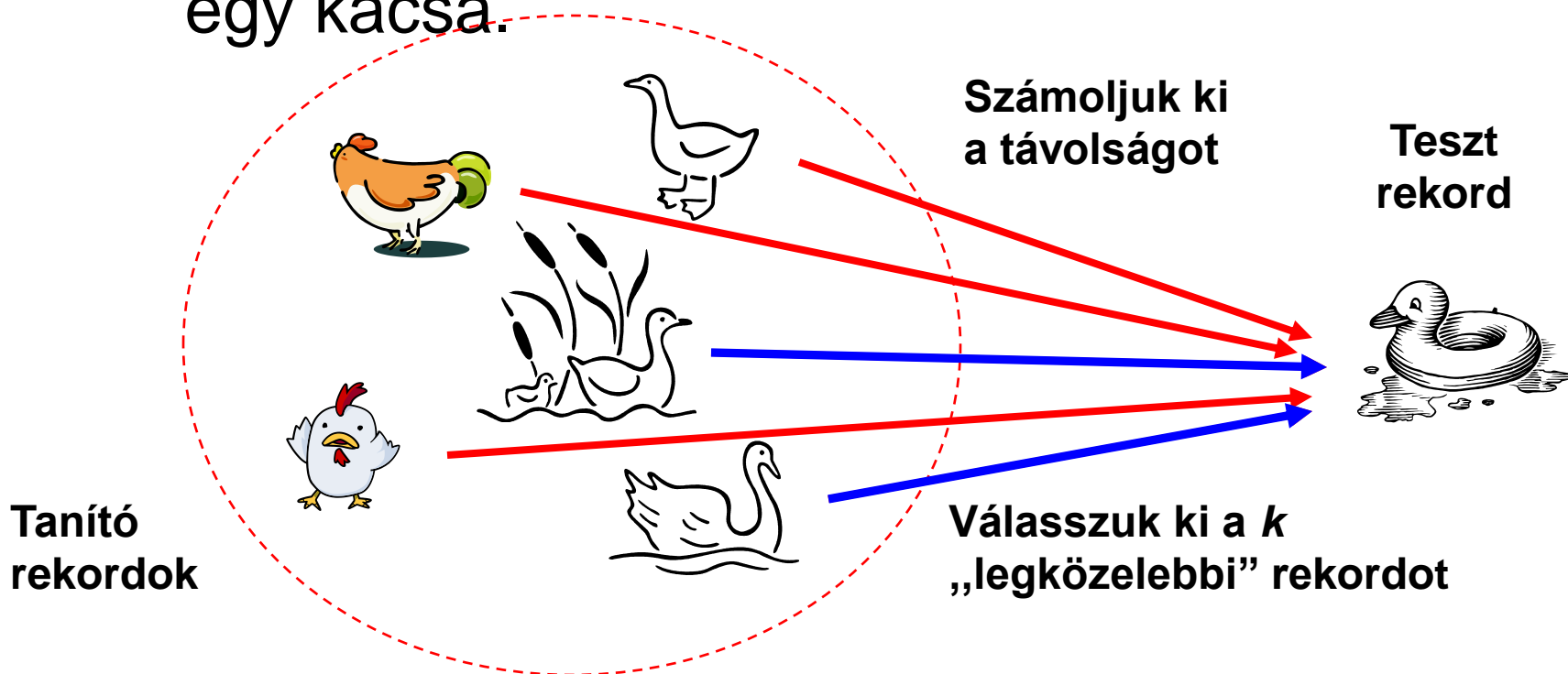
Eset alapú osztályozók

- Példák:
 - Rote tanuló algoritmus
 - ◆ A teljes tanító adatállományt memorizálja, és csak akkor hajtja végre az osztályozást, ha az új rekord attributum értékei pontosan illeszkednek egy tanító esetre.
 - Legközelebbi szomszéd
 - ◆ Használjuk a k „legközelebbi” pontot (legközelebbi szomszédok) az osztályozás végrehajtására.

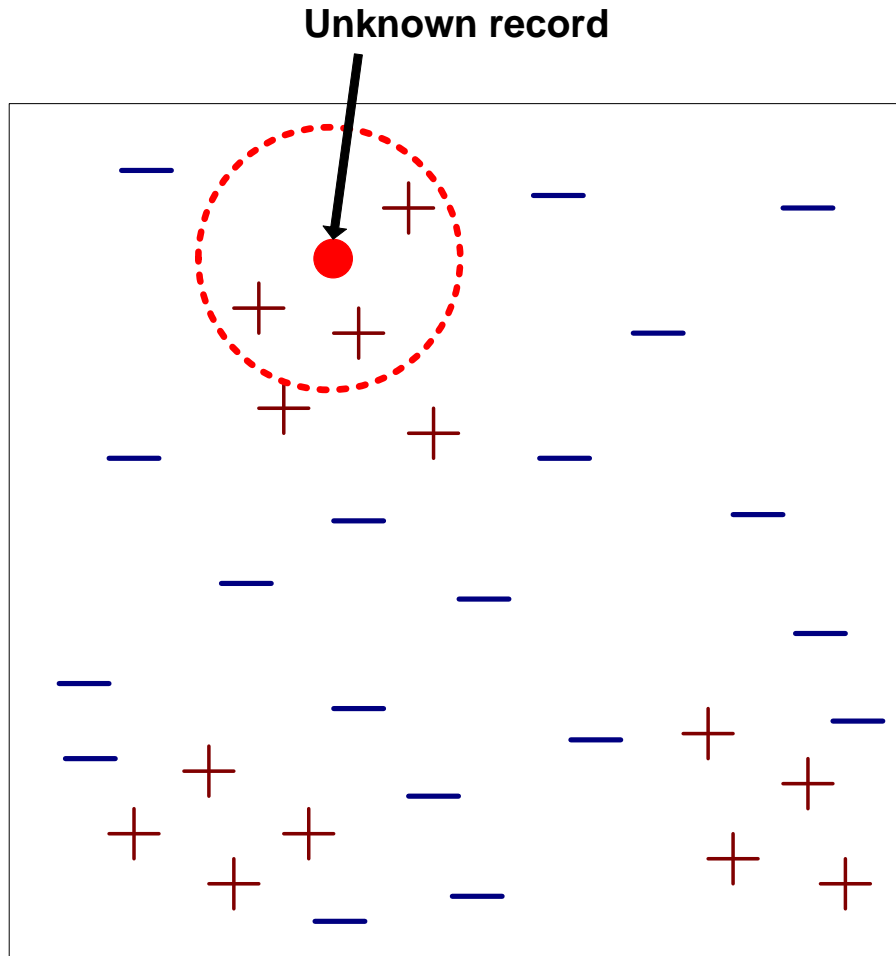
Legközelebbi szomszéd osztályozók

- Alapgondolat:

- Ha valami úgy totyog mint egy kacska, úgy hápog mint egy kacska, akkor az valószínűleg egy kacska.

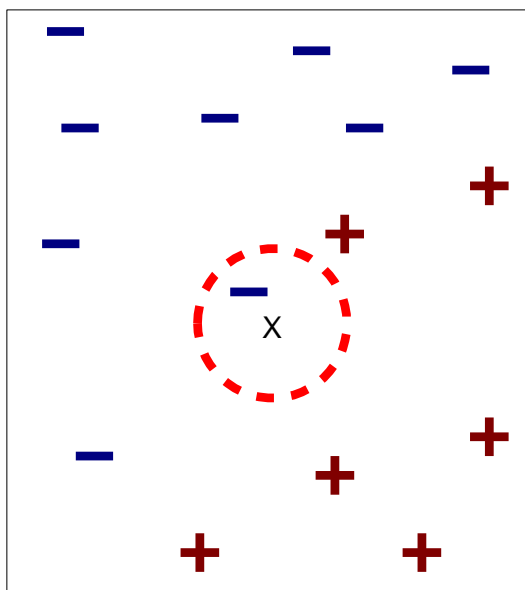


Legközelebbi szomszéd osztályozók

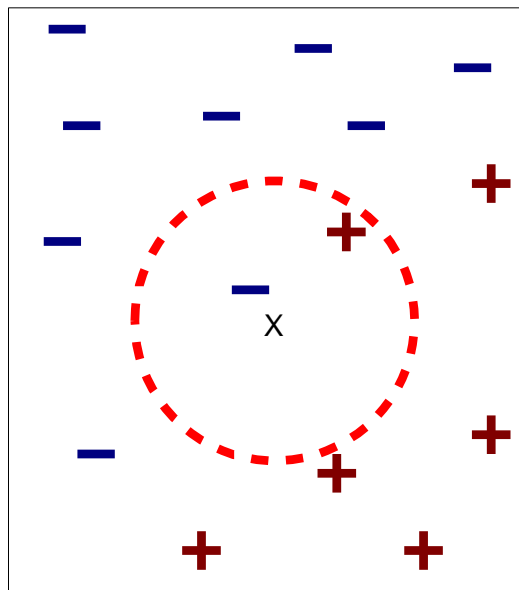


- Három dolog szükséges
 - Rekordok egy halmaza
 - A rekordok közötti távolság számolására szolgáló metrika
 - A k szám, a meghatározandó legközelebbi szomszédok száma
- Egy új rekord osztályozása:
 - Számoljuk ki a távolságot a többi tanító rekordtól.
 - Határozzuk meg a k legközelebbi szomszédot.
 - Hsználjuk a legközelebbi szomszédok osztálycímkeit az új rekord besorolására (pl. többségi szavazást véve).

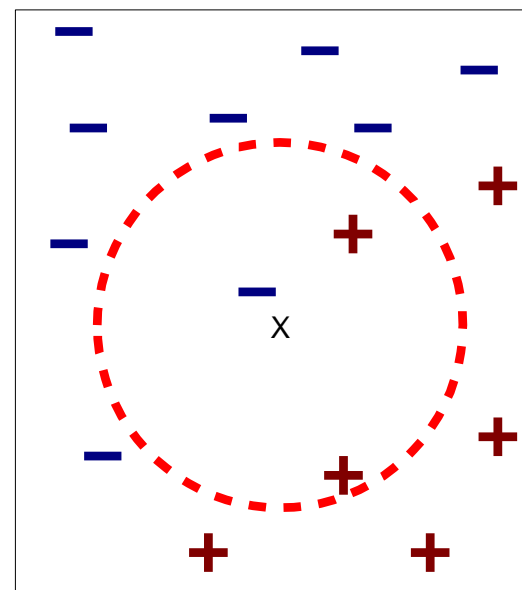
A legközelebbi szomszéd definíciója



(a) 1-nearest neighbor



(b) 2-nearest neighbor

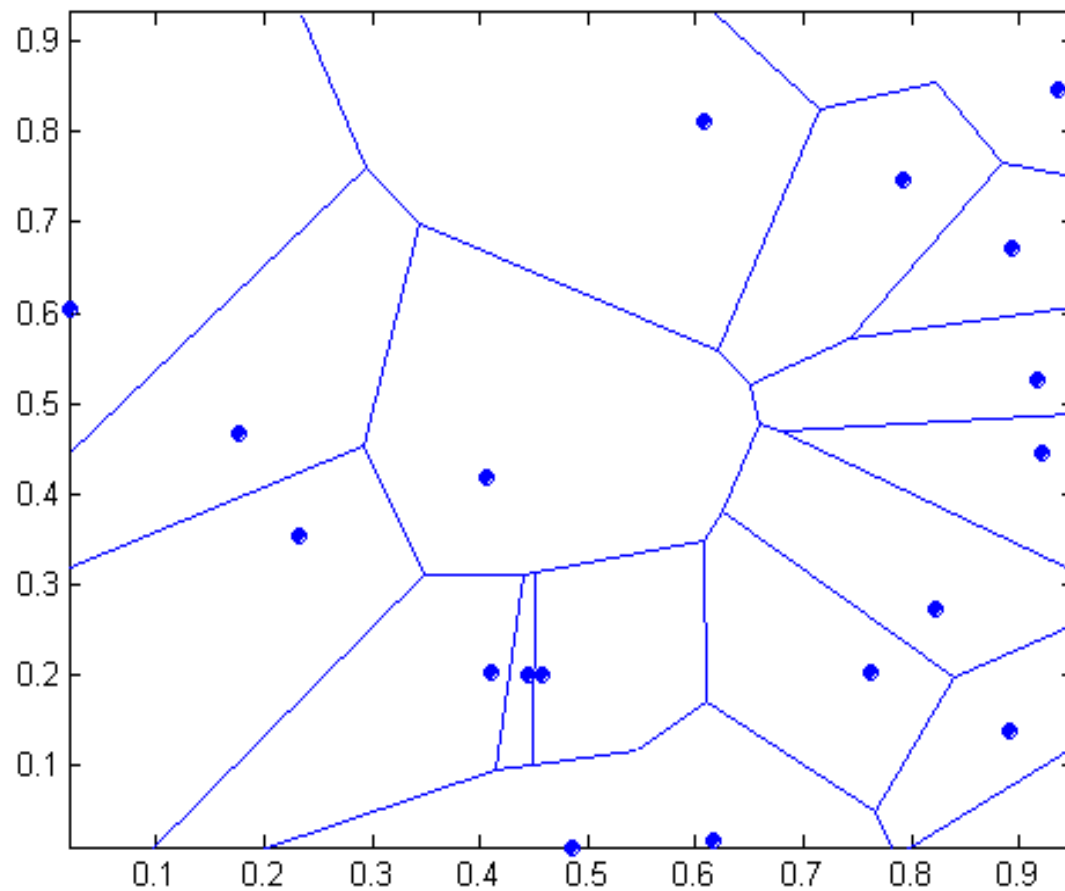


(c) 3-nearest neighbor

Az x rekord k legközelebbi szomszédja azok a rekordok, melyek távolsága x -től a k legkisebb távolság.

1 legközelebbi szomszéd

Voronoi diagram



Legközelebbi szomszéd osztályozók

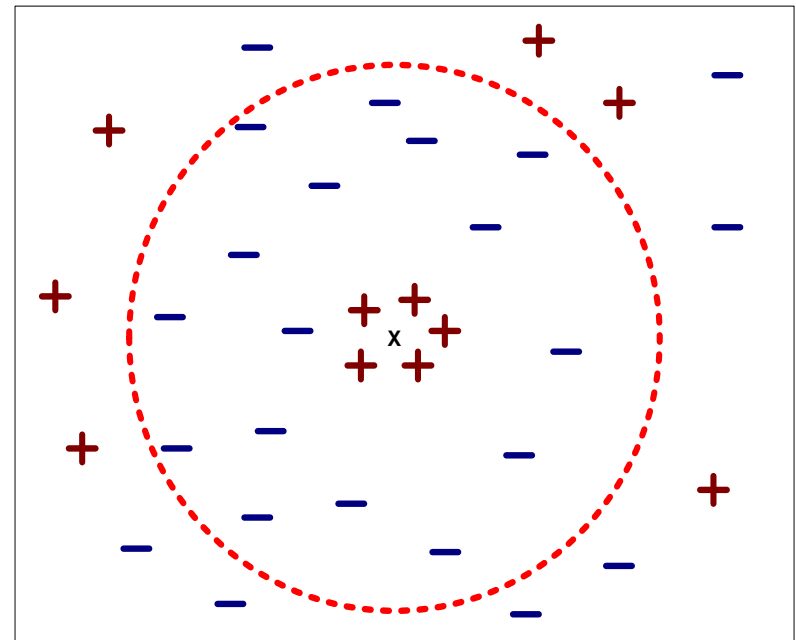
- Számoljuk ki két pont távolságát:
 - Euklideszi távolság

$$d(p, q) = \sqrt{\sum_i (p_i - q_i)^2}$$

- A legközelebbi szomszédok alapján határozzuk meg az osztályt:
 - Vegyünk a többségi osztályt a k szomszéd közül.
 - Súlyozzuk a szavazatokat a távolságnak megfelelően.
 - ◆ súly: $w = 1/d^2$

Legközelebbi szomszéd osztályozók

- A k érték megválasztása:
 - Ha k túl kicsi, akkor a módszer érzékeny a hibás rekordokra.
 - Ha k túl nagy, akkor a szomszédság más osztálybeli pontokat is tartalmazhat.



Legközelebbi szomszéd osztályozók

- Skálázási szempontok
 - Az attributumokat átskálázhatjuk így előzve meg azt, hogy egy attributum dominálja a távolságot.
 - Példa:
 - ◆ Egy személy magassága 1.5m és 1.8m között van.
 - ◆ Egy személy súlya 90lb és 300lb között van.
 - ◆ Egy személy bevétele \$10K és \$1M között van.

Legközelebbi szomszéd osztályozók

- Problémák az euklideszi távolsággal:
 - Sok dimenziós adatok
 - ◆ **dimenzió probléma**
 - A természetes szemlélettel ellenkező eredményt is adhat.

1 1 1 1 1 1 1 1 1 1 0

0 1 1 1 1 1 1 1 1 1 1

$d = 1.4142$

vagy

1 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 1

$d = 1.4142$

- ◆ Megoldás: Normalizáljuk a vektorokat!

Legközelebbi szomszéd osztályozók

- A legközelebbi szomszéd osztályozók lusta tanuló algoritmusok.
 - Nem építenek explicit modelleket.
 - Mások mint a mohó tanító algoritmusok, ld. döntési fák és osztályozási szabályok.
 - Az új rekordok osztályozása viszonylag költséges.

Példa: PEBLS

- PEBLS: Párhuzamos példa alapú tanuló rendszer (Parallel Exemplar-Based Learning System, Cost & Salzberg)
 - Egyaránt működik folytonos és kategórikus változókkal.
 - ◆ Kategórikus változóknál két érték távolságát a módosított értékek differenciája metrikával (MVDM) számoljuk.
 - Minden rekordhoz egy súlyt rendel.
 - A legközelebbi szomszédok száma: $k = 1$.

Példa: PEBLS

Tid	Visszatérítés	Családi állapot	Jövedelem	Csalás
1	Igen	Nőtlen	125K	Nem
2	Nem	Házias	100K	Nem
3	Nem	Nőtlen	70K	Nem
4	Igen	Házias	120K	Nem
5	Nem	Elvált	95K	Igen
6	Nem	Házias	60K	Nem
7	Igen	Elvált	220K	Nem
8	Nem	Nőtlen	85K	Igen
9	Nem	Házias	75K	Nem
10	Nem	Nőtlen	90K	Igen

Két kategórikus érték távolsága:

$d(\text{Nőtlen}, \text{Házias})$

$$= |2/4 - 0/4| + |2/4 - 4/4| = 1$$

$d(\text{Nőtlen}, \text{Elvált})$

$$= |2/4 - 1/2| + |2/4 - 1/2| = 0$$

$d(\text{Házias}, \text{Elvált})$

$$= |0/4 - 1/2| + |4/4 - 1/2| = 1$$

$d(\text{Visszatérítés=Igen}, \text{Visszatérítés=Nem})$

$$= |0/3 - 3/7| + |3/3 - 4/7| = 6/7$$

Osz-tály	Családi állapot		
	Nőtlen	Házias	Elvált
Igen	2	0	1
Nem	2	4	1

Osz-tály	Visszatérítés	
	Igen	Nem
Igen	0	3
Nem	3	4

$$d(V_1, V_2) = \sum_i \left| \frac{n_{1i}}{n_1} - \frac{n_{2i}}{n_2} \right|$$

Példa: PEBLS

<i>Tid</i>	Vissza- térítés	Családi állapot	Jöve- delem	Csalás
X	Igen	Nőtlen	125K	Nem
Y	Nem	Házass	100K	Nem

Az X és Y rekordok közötti távolság:

$$\Delta(X, Y) = w_X w_Y \sum_{i=1}^d d(X_i, Y_i)^2$$

ahol: $w_X = \frac{\text{Azon esetek száma, ahol } X \text{ - t használjuk}}{\text{Azon esetek száma, ahol } X \text{ helyesen prediktál}}$

$w_X \cong 1$ ha X az esetek többségében pontos előrejelzést ad

$w_X > 1$ ha X nem ad megbízható előrejelzést

Bayes osztályozó

- Egy valószínűségszámítási módszer osztályozási problémák megoldására.
- Feltételes valószínűség:

$$P (C | A) = \frac{P (A , C)}{P (A)}$$

$$P (A | C) = \frac{P (A , C)}{P (C)}$$

- Bayes tétel:

$$P (C | A) = \frac{P (A | C) P (C)}{P (A)}$$

Példa a Bayes tételre

- Adottak:
 - Az orvosok tudják, hogy az agyhártyagyulladás az esetek 50%-ban nyakfájást okoz.
 - Annak valószínűsége, hogy egy páciensnek agyhártyagyulladása van $1/50000$.
 - Annak valószínűsége, hogy egy páciensnek nyakfájása van $1/20$.
- Ha egy páciensnek nyakfájása van, akkor mi annak a valószínűsége, hogy agyhártyagyulladásban szenved?

$$P(M | S) = \frac{P(S | M) P(M)}{P(S)} = \frac{0.5 \times 1 / 50000}{1 / 20} = 0.0002$$

Bayes osztályozók

- Tekintsük valószínűségi változónak az összes attributumot és a cél (osztály) változót.
- Legyen adott egy rekord az (A_1, A_2, \dots, A_n) attributumértékekkel
 - A cél a C osztályozó változó előrejelzése.
 - Azt az értékét keressük C -nek, amely maximalizálja $P(C | A_1, A_2, \dots, A_n)$ -t.
- Tudjuk-e közvetlenül becsülni $P(C | A_1, A_2, \dots, A_n)$ -t az adatokból?

Bayes osztályozók

- Megközelítés:
 - Számoljuk ki a $P(C | A_1, A_2, \dots, A_n)$ poszteriori valószínűséget minden C értékre a Bayes tétellel.

$$P(C | A_1 A_2 \dots A_n) = \frac{P(A_1 A_2 \dots A_n | C) P(C)}{P(A_1 A_2 \dots A_n)}$$

- Válasszuk azt a C értéket, amely maximalizálja $P(C | A_1, A_2, \dots, A_n)$ -t.
 - Ekvivalens annak a C értéknek megtalálásával, mely maximalizálja $P(A_1, A_2, \dots, A_n | C) P(C)$ -t.
- Hogyan becsüljük $P(A_1, A_2, \dots, A_n | C)$ -t?

Bayes osztályozók

- Tételezzünk fel függetlenséget az A_i attributumok között ha az osztály adott:
 - $P(A_1, A_2, \dots, A_n | C) = P(A_1 | C_j) P(A_2 | C_j) \dots P(A_n | C_j)$
 - Az $P(A_i | C_j)$ valószínűséget becsülhetjük minden A_i és C_j esetén.
 - Egy új rekord a C_j osztályba kerül ha a $P(C_j) \prod P(A_i | C_j)$ maximális.

Hogyan becsüljük valószínűséget?

Tid	Visszatérítés	Családi állapot	Adóköteles jövedelem	Csalás
1	Igen	Nőtlen	125K	Nem
2	Nem	Házás	100K	Nem
3	Nem	Nőtlen	70K	Nem
4	Igen	Házás	120K	Nem
5	Nem	Elvált	95K	Igen
6	Nem	Házás	60K	Nem
7	Igen	Elvált	220K	Nem
8	Nem	Nőtlen	85K	Igen
9	Nem	Házás	75K	Nem
10	Nem	Nőtlen	90K	Igen

- Osztály: $P(C) = N_C/N$

- Pl. $P(\text{Nem}) = 7/10$
 $P(\text{Igen}) = 3/10$

- Diszkrét attributumokra:

$$P(A_i | C_k) = |A_{ik}| / N_{C_k}$$

- ahol $|A_{ik}|$ azon esetek száma, ahol az A_i attributumérték fordult elő és a C_k osztályba tartoznak.
- Példák:

$$P(\text{Állapot}=\text{Házás}|\text{Nem}) = 4/7$$
$$P(\text{Visszatérítés}=\text{Igen}|\text{Igen})=0$$

Hogyan becsüljük valószínűséget?

- Folytonos attributumokra:
 - **Diszkrétizáljunk** résztartományokra osztva:
 - ◆ egy sorrendi attributum értéket rendelünk részenként,
 - ◆ megsérti a függetlenségi feltételezést.
 - **Bináris vágás:** $(A < v)$ vagy $(A > v)$
 - ◆ válasszuk a két ág egyikét mint új attributumot.
 - **Valószínűségi sűrűségbecslés:**
 - ◆ Tegyük fel, hogy az attributum normális eloszlású.
 - ◆ Használjuk az adatokat az eloszlás paramétereinek becslésére (pl. átlag és szórás).
 - ◆ Ha ismert a valószínűségi eloszlás, akkor használhatjuk a $P(A_i|c)$ feltételes valószínűség becslésére.

Hogyan becsüljük valószínűséget?

Tid	Vissza- térítés	Családi állapot	Adóköteles jövedelem	Csalás
1	Igen	Nőtlen	125K	Nem
2	Nem	Házás	100K	Nem
3	Nem	Nőtlen	70K	Nem
4	Igen	Házás	120K	Nem
5	Nem	Elvált	95K	Igen
6	Nem	Házás	60K	Nem
7	Igen	Elvált	220K	Nem
8	Nem	Nőtlen	85K	Igen
9	Nem	Házás	75K	Nem
10	Nem	Nőtlen	90K	Igen

- Normális eloszlás:

$$P(A_i | c_j) = \frac{1}{\sqrt{2\pi\sigma_{ij}^2}} e^{-\frac{(A_i - \mu_{ij})^2}{2\sigma_{ij}^2}}$$

- minden (A_i, c_j) párra

- (Jövedelem, Osztály=Nem):

- Ha Osztály=Nem

◆ minta átlag = 110

◆ minta variancia = 2975

$$P(\text{Jövedelem} = 120 | \text{Nem}) = \frac{1}{\sqrt{2\pi(54.54)}} e^{-\frac{(120 - 110)^2}{2(2975)}} = 0.0072$$

Példa naív Bayes osztályozóra

Adott az alábbi teszt rekord:

$X = (\text{Visszatérítés} = \text{Nem}, \text{Házasság} = \text{Házas}, \text{Jövedelem} = 120\text{K})$

naive Bayes Classifier:

$$P(\text{Refund}=\text{Yes}|\text{No}) = 3/7$$

$$P(\text{Refund}=\text{No}|\text{No}) = 4/7$$

$$P(\text{Refund}=\text{Yes}|\text{Yes}) = 0$$

$$P(\text{Refund}=\text{No}|\text{Yes}) = 1$$

$$P(\text{Marital Status}=\text{Single}|\text{No}) = 2/7$$

$$P(\text{Marital Status}=\text{Divorced}|\text{No}) = 1/7$$

$$P(\text{Marital Status}=\text{Married}|\text{No}) = 4/7$$

$$P(\text{Marital Status}=\text{Single}|\text{Yes}) = 2/7$$

$$P(\text{Marital Status}=\text{Divorced}|\text{Yes}) = 1/7$$

$$P(\text{Marital Status}=\text{Married}|\text{Yes}) = 0$$

For taxable income:

If class=No: sample mean=110

sample variance=2975

If class=Yes: sample mean=90

sample variance=25

- $$P(X|\text{Osztály}=\text{Nem}) = P(\text{Vtér}=\text{Nem}|\text{Osztály}=\text{Nem}) \\ \times P(\text{Házasság}|\text{Osztály}=\text{Nem}) \\ \times P(\text{Jöv}=\text{120K}|\text{Osztály}=\text{Nem}) \\ = 4/7 \times 4/7 \times 0.0072 = 0.0024$$

- $$P(X|\text{Osztály}=\text{Igen}) = P(\text{Vtér}=\text{Nem}|\text{Osztály}=\text{Igen}) \\ \times P(\text{Házasság}|\text{Osztály}=\text{Igen}) \\ \times P(\text{Jöv}=\text{120K}|\text{Osztály}=\text{Igen}) \\ = 1 \times 0 \times 1.2 \times 10^{-9} = 0$$

Mivel $P(X|\text{Nem})P(\text{Nem}) > P(X|\text{Igen})P(\text{Igen})$

ezért $P(\text{Nem}|X) > P(\text{Igen}|X)$

$\Rightarrow \text{Osztály} = \text{Nem}$

Bayes osztályozó

- Ha a feltételes valószínűségek egyike 0, akkor az egész kifejezés 0.
- Valószínűségi becslés:

$$\text{Eredeti} \quad : P (A_i | C) = \frac{N_{ic}}{N_c}$$

$$\text{Laplace} \quad : P (A_i | C) = \frac{N_{ic} + 1}{N_c + c}$$

$$\text{m - estimate} \quad : P (A_i | C) = \frac{N_{ic} + mp}{N_c + m}$$

c: osztályok száma

p: prior valószínűség

m: paraméter

Példa naív Bayes osztályozóra

Name	Give Birth	Can Fly	Live in Water	Have Legs	Class
human	yes	no	no	yes	mammals
python	no	no	no	no	non-mammals
salmon	no	no	yes	no	non-mammals
whale	yes	no	yes	no	mammals
frog	no	no	sometimes	yes	non-mammals
komodo	no	no	no	yes	non-mammals
bat	yes	yes	no	yes	mammals
pigeon	no	yes	no	yes	non-mammals
cat	yes	no	no	yes	mammals
leopard shark	yes	no	yes	no	non-mammals
turtle	no	no	sometimes	yes	non-mammals
penguin	no	no	sometimes	yes	non-mammals
porcupine	yes	no	no	yes	mammals
eel	no	no	yes	no	non-mammals
salamander	no	no	sometimes	yes	non-mammals
gila monster	no	no	no	yes	non-mammals
platypus	no	no	no	yes	mammals
owl	no	yes	no	yes	non-mammals
dolphin	yes	no	yes	no	mammals
eagle	no	yes	no	yes	non-mammals

A: attributumok

M: emlősök

N: nem emlősök

$$P(A | M) = \frac{6}{7} \times \frac{6}{7} \times \frac{2}{7} \times \frac{2}{7} = 0.06$$

$$P(A | N) = \frac{1}{13} \times \frac{10}{13} \times \frac{3}{13} \times \frac{4}{13} = 0.0042$$

$$P(A | M)P(M) = 0.06 \times \frac{7}{20} = 0.021$$

$$P(A | N)P(N) = 0.0042 \times \frac{13}{20} = 0.0027$$

P(A|M)P(M) > P(A|N)P(N)

=> emlős

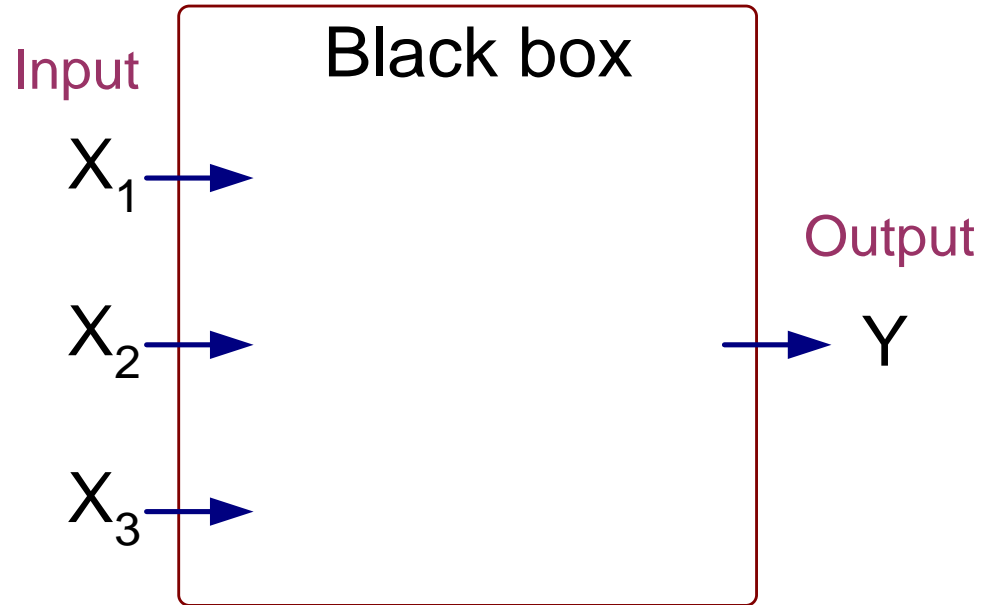
Give Birth	Can Fly	Live in Water	Have Legs	Class
yes	no	yes	no	?

Összegzés: Naív Bayes

- Robusztus izolált hibás pontokra.
- Kezeli a hiányzó értékeket a valószínűségek becslésénél ezen esetek figyelmen kívül hagyásával.
- Robusztus az irreleváns attributumokra.
- A függetlenségi feltétel nem teljesül egyes attributumokra.
 - Használjunk más módszereket, Bayes hálók (Bayesian Belief Networks, BBN).

Mesterséges neurális hálók (ANN)

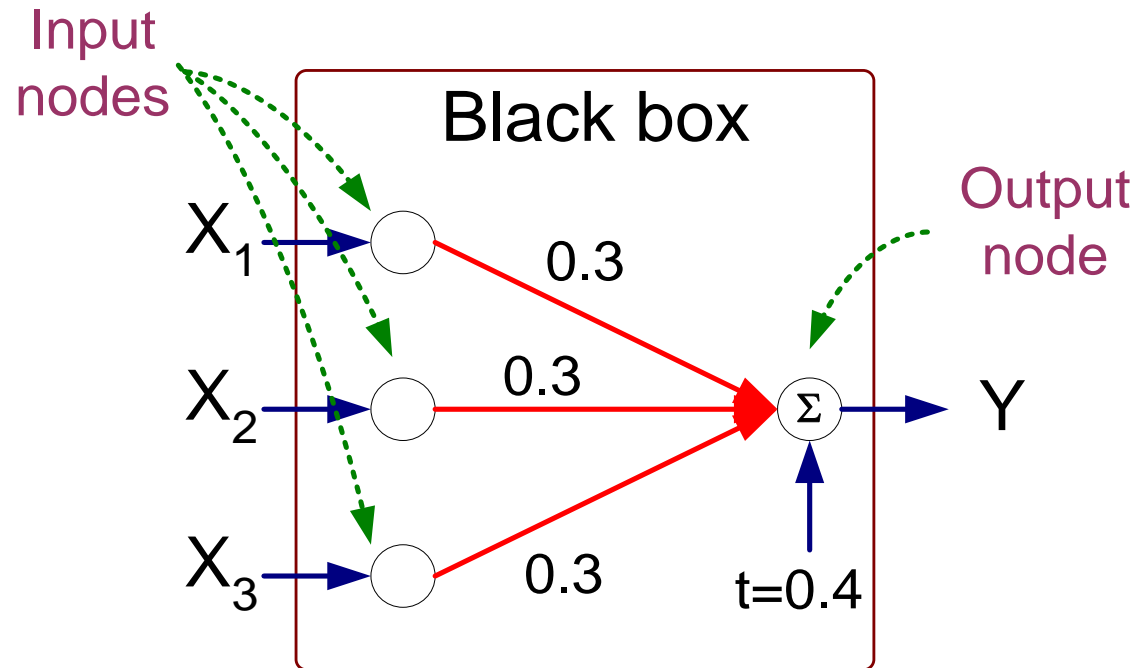
X_1	X_2	X_3	Y
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1
0	0	1	0
0	1	0	0
0	1	1	1
0	0	0	0



Az Y output 1 ha a három input közül legalább kettő 1.

Mesterséges neurális hálók (ANN)

X_1	X_2	X_3	Y
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1
0	0	1	0
0	1	0	0
0	1	1	1
0	0	0	0

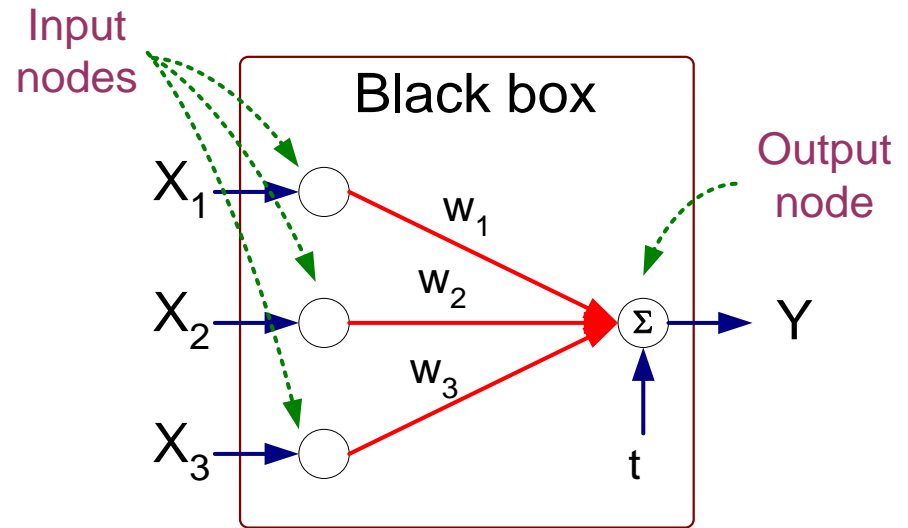


$$Y = I(0.3 X_1 + 0.3 X_2 + 0.3 X_3 - 0.4 > 0)$$

$$\text{ahol } I(z) = \begin{cases} 1 & \text{ha } z \text{ igaz} \\ 0 & \text{egyébként} \end{cases}$$

Mesterséges neurális hálók (MNH)

- A modell egymással összekötött csúcsok és súlyozott élek együttese.
- Az output csúcs összegzi az hozzátartozó input értékeket az éleken lévő súlyok szerint.
- Vessük össze az output csúcsban kapott értéket egy t küszöb számmal.

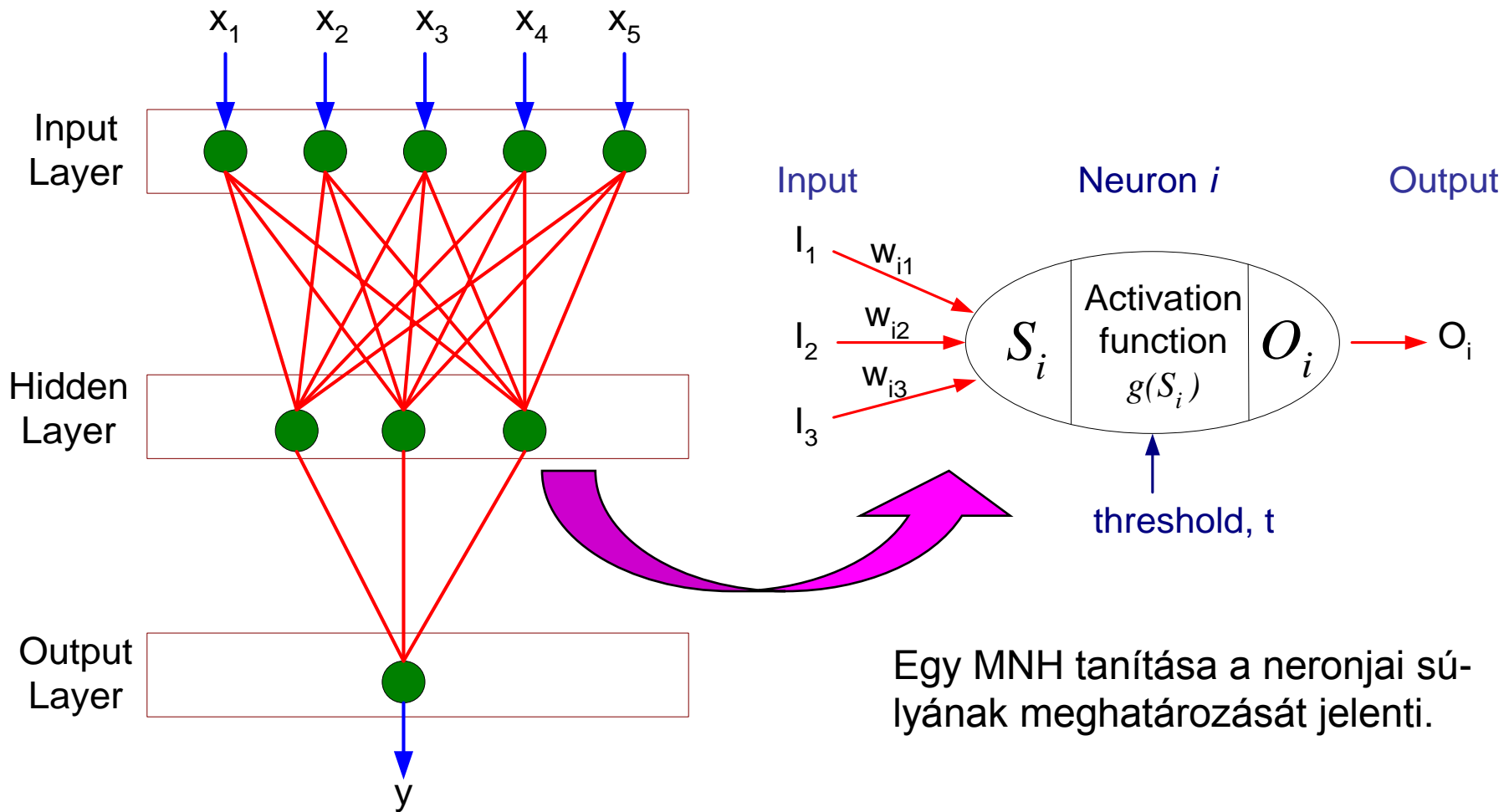


Perceptron modell

$$Y = I \left(\sum_i w_i X_i - t \right) \quad \text{vagy}$$

$$Y = \text{sign} \left(\sum_i w_i X_i - t \right)$$

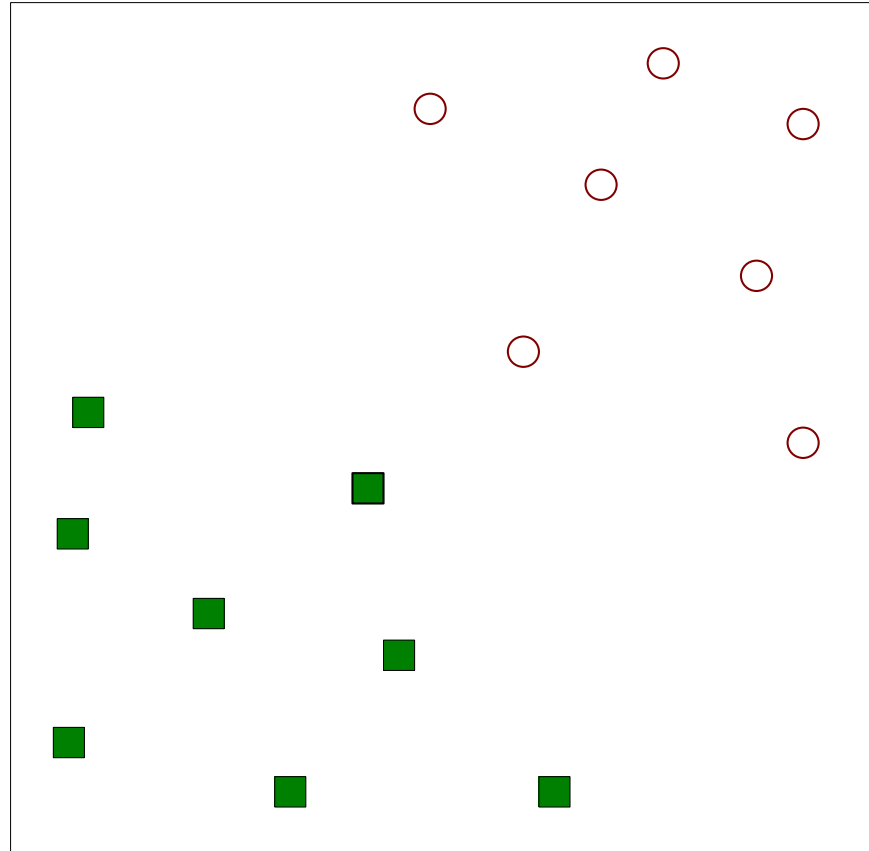
A MNH általános szerkezete



Algoritmus MNH tanítására

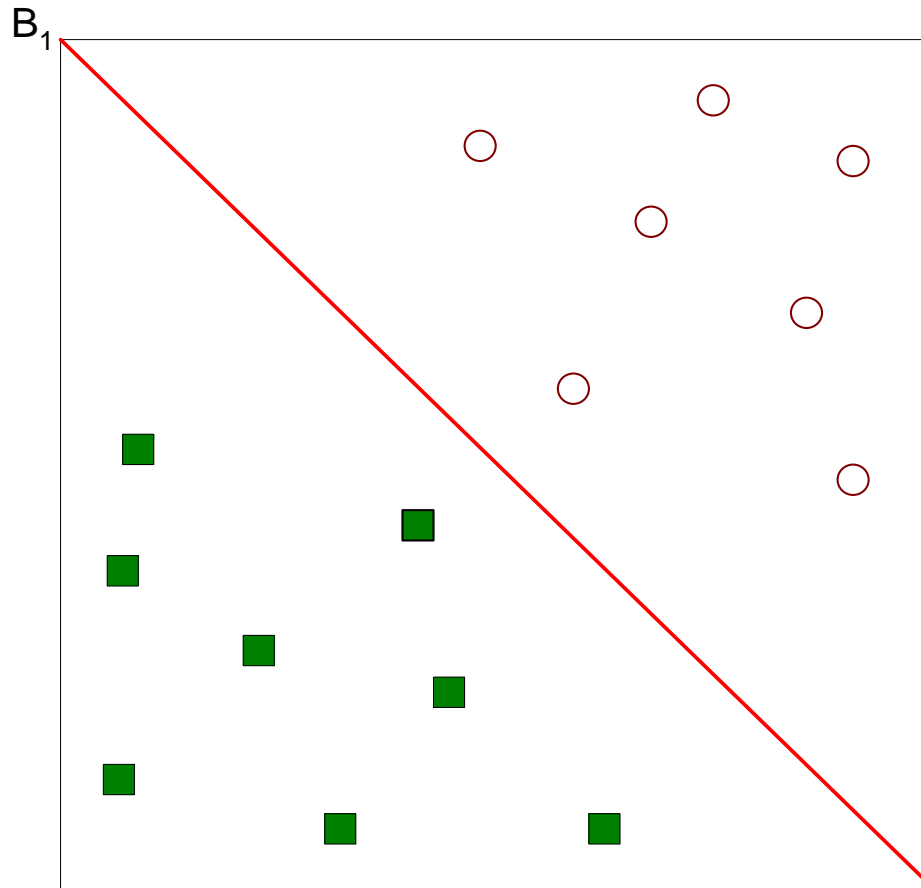
- Inicializáljuk a (w_0, w_1, \dots, w_k) súlyokat.
- Módosítsuk úgy a súlyokat, hogy az MNH outputja minél jobban egyezzen meg a tanító esetek osztály címkéivel.
 - Célfüggvény:
$$E = \sum_i [Y_i - f(w_i, X_i)]^2$$
 - Határozzuk meg azon w_i súlyokat, amelyek minimalizálják a fenti célfüggvényt.
 - ◆ Pl. hiba visszacsatolás algoritmus.

Támasz vektorgépek (SVM)



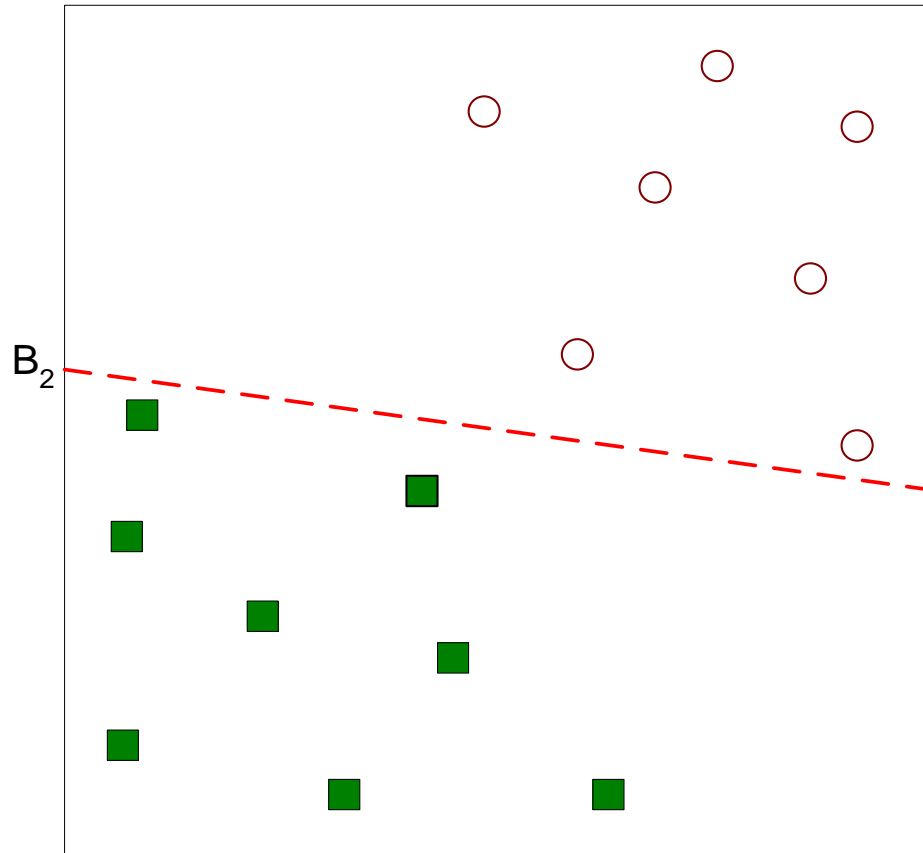
- Keressünk olyan hipersíkot (döntési határ), amely elválasztja az adatokat.

Támasz vektorgépek (SVM)



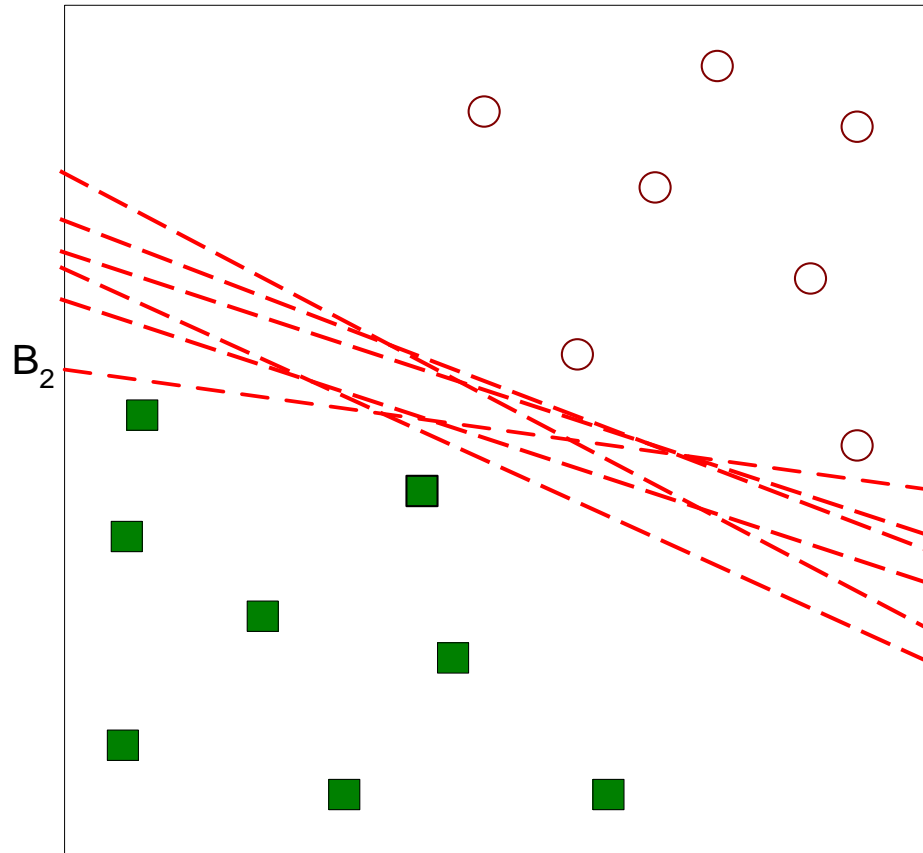
- Egy lehetséges megoldás.

Támasz vektorgépek (SVM)



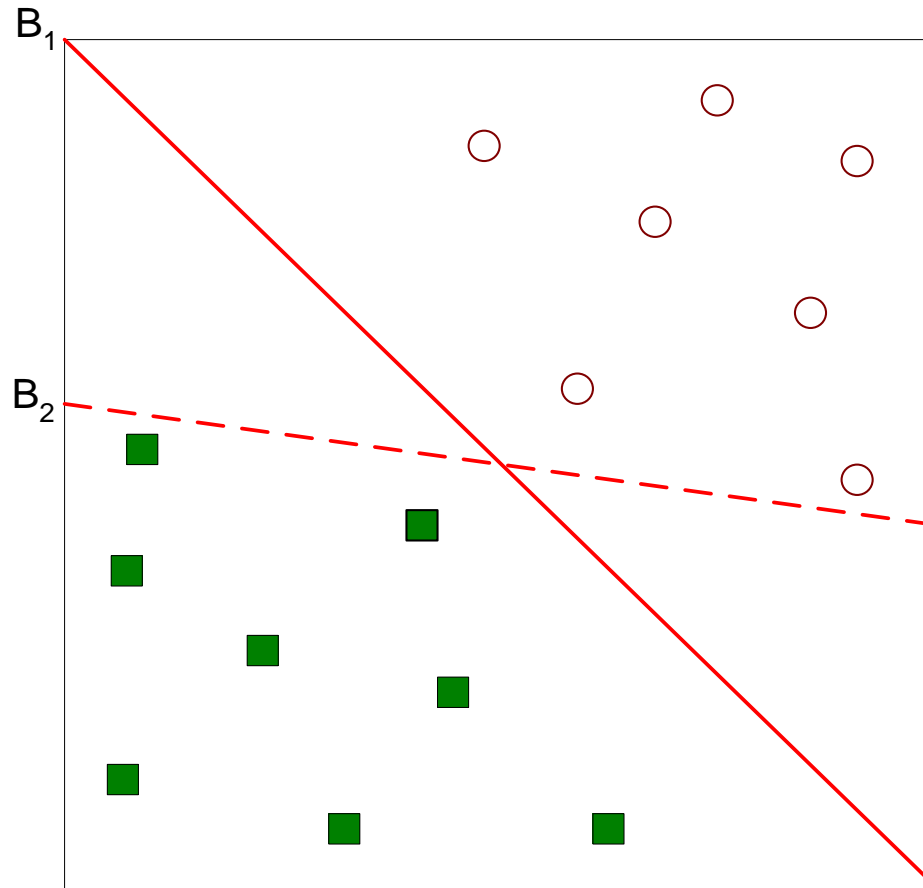
- Egy másik lehetséges megoldás.

Támasz vektorgépek (SVM)



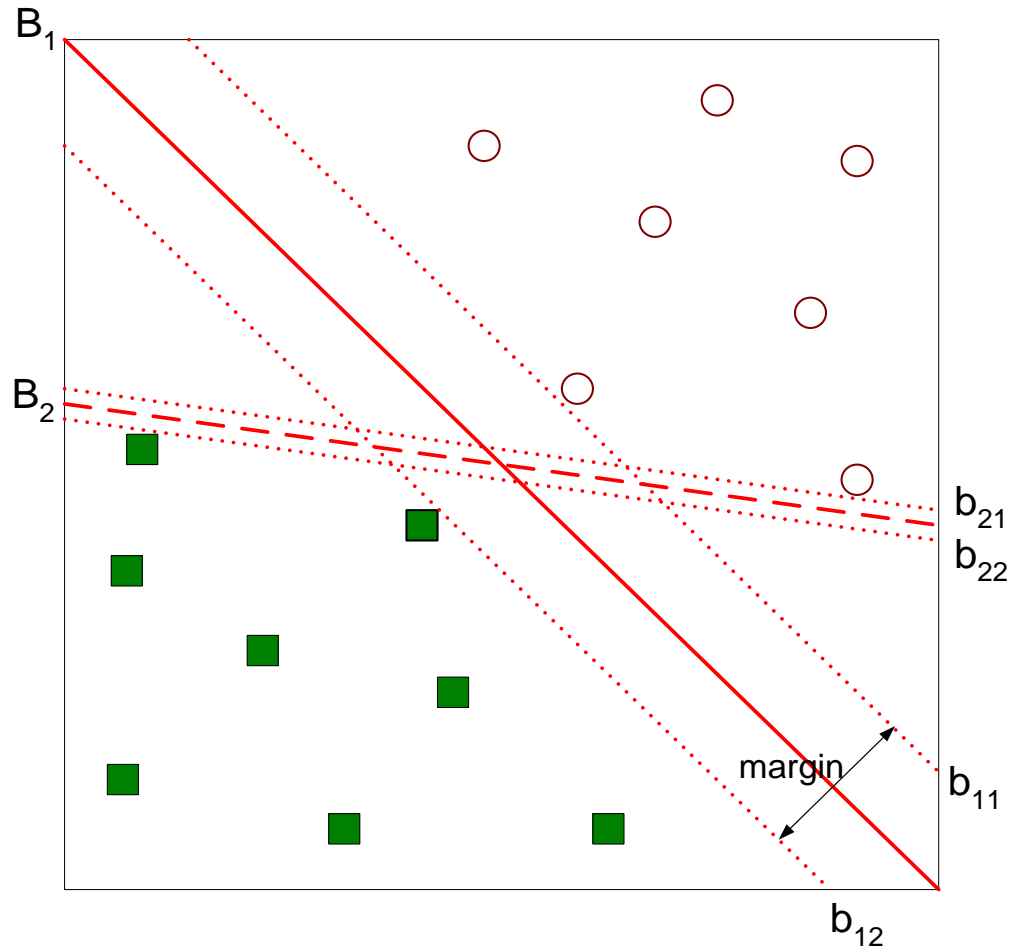
- További lehetséges megoldások.

Támasz vektorgépek (SVM)



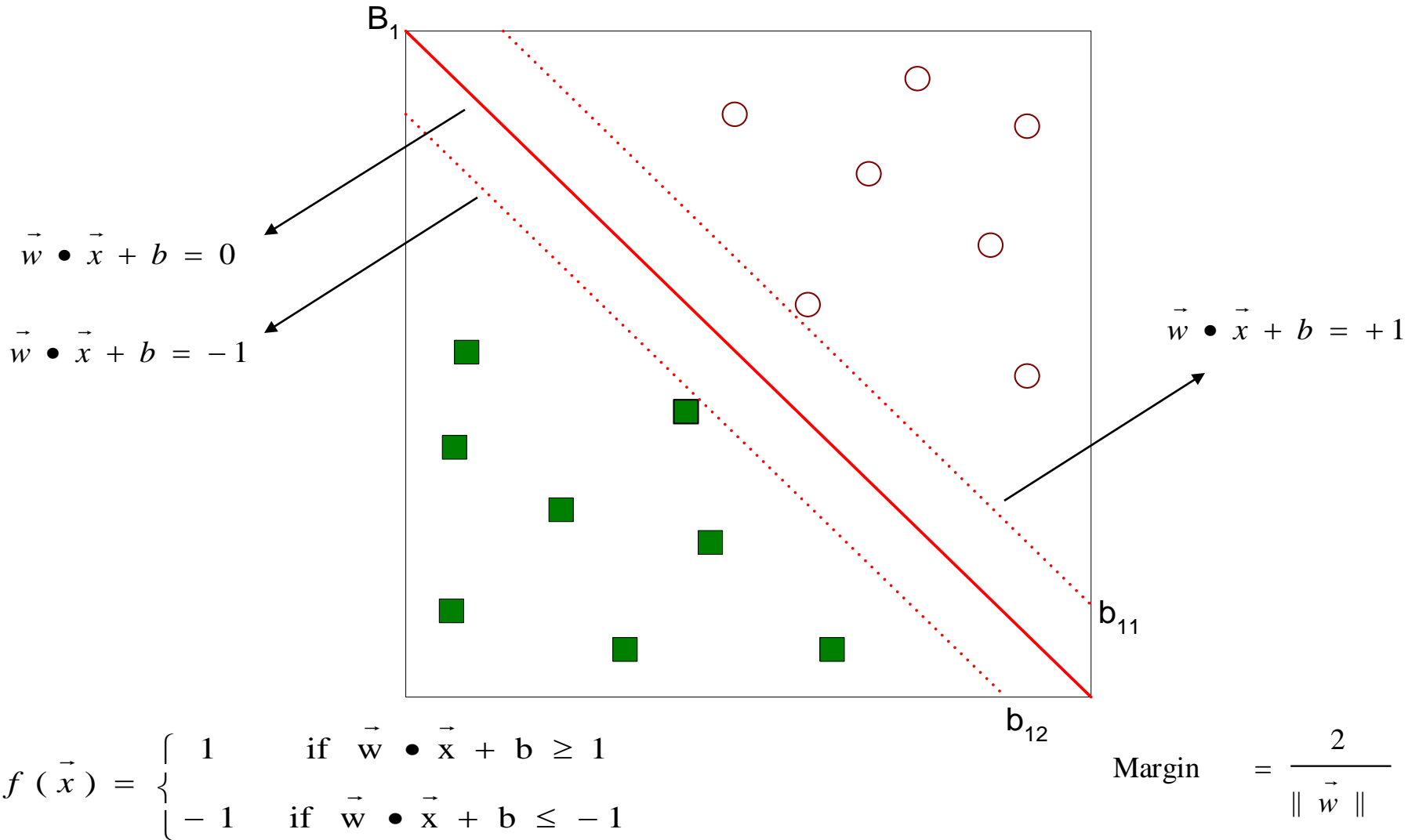
- Melyik a jobb? B_1 vagy B_2 ?
- Hogyan definiálhatjuk a jobb fogalmát?

Támasz vektorgépek (SVM)



- Keressük azt a hipersíkot, mely **maximalizálja** a margót
=> B_1 jobb mint B_2 .

Támasz vektorgépek (SVM)



$$f(\vec{x}) = \begin{cases} 1 & \text{if } \vec{w} \cdot \vec{x} + b \geq 1 \\ -1 & \text{if } \vec{w} \cdot \vec{x} + b \leq -1 \end{cases}$$

$$\text{Margin} = \frac{2}{\|\vec{w}\|}$$

Támasz vektorgépek (SVM)

- Maximalizálni akarjuk:

$$\text{Margin} = \frac{2}{\|\vec{w}\|}$$

- Ez ekvivalens minimalizálni: $L(w) = \frac{\|\vec{w}\|^2}{2}$

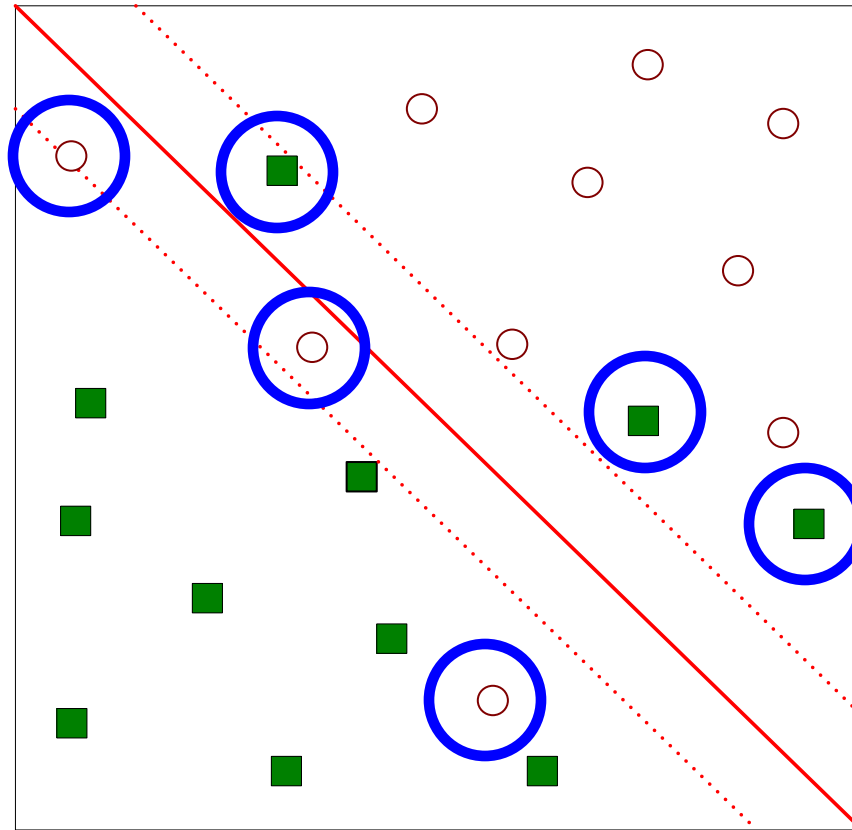
- De eleget kell tenni a következő kényszereknek:

$$f(\vec{x}_i) = \begin{cases} 1 & \text{if } \vec{w} \cdot \vec{x}_i + b \geq 1 \\ -1 & \text{if } \vec{w} \cdot \vec{x}_i + b \leq -1 \end{cases}$$

- ◆ Ez kényszerfeltétel melletti optimalizációs feladat.
 - Numerikus módszerek (pl. kvadratikus programozás).

Támasz vektorgépek (SVM)

- Mi van ha a feladat nem lineárisan szeperálható?



Támasz vektorgépek (SVM)

- Mi van ha a feladat nem lineárisan szeparálható?
 - Vezessünk be lötyögő változókat

- ◆ Minimalizálni kell:

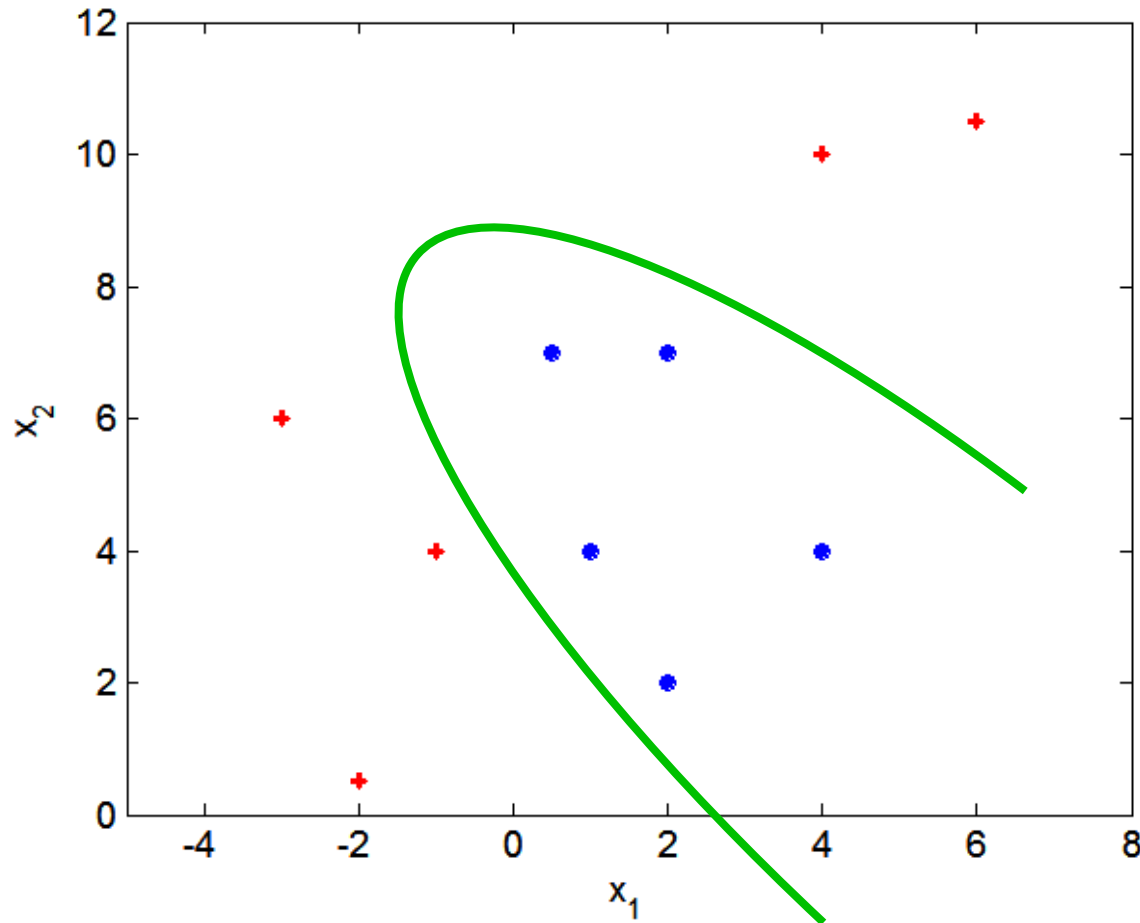
$$L(w) = \frac{\|\vec{w}\|^2}{2} + C \left(\sum_{i=1}^N \xi_i^k \right)$$

- ◆ Kényszerfeltételek:

$$f(\vec{x}_i) = \begin{cases} 1 & \text{if } \vec{w} \cdot \vec{x}_i + b \geq 1 - \xi_i \\ -1 & \text{if } \vec{w} \cdot \vec{x}_i + b \leq -1 + \xi_i \end{cases}$$

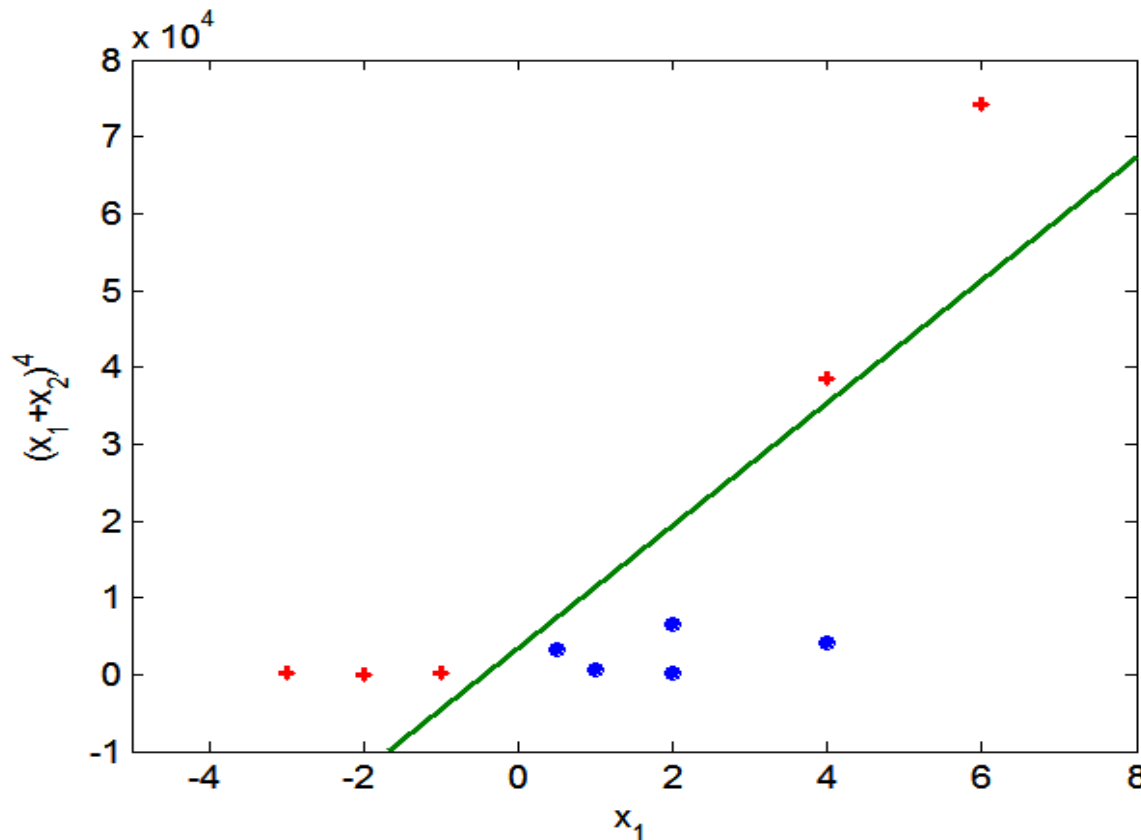
Nemlineáris támasz vektorgépek

- Mi van ha a döntési határ nem lineáris?



Nemlineáris támasz vektorgépek

- Transzformáljuk az adatokat egy magasabb dimenziójú térbe (kernel trükk).



Osztályozás regresszió útján

- Ahelyett hogy egy rekord **osztályát** jeleznénk előre próbáljuk meg **előrejelezni az osztály valószínűségét** amely már egy folytonos mennyiség
- Egy **folytonos mennyiség előrejelzését regressziós** feladatnak nevezzük
- Általános megközelítés: találjunk egy olyan folytonos függvényt, amely jól modellezi (illeszkedik) a folytonos pontfelhőre.

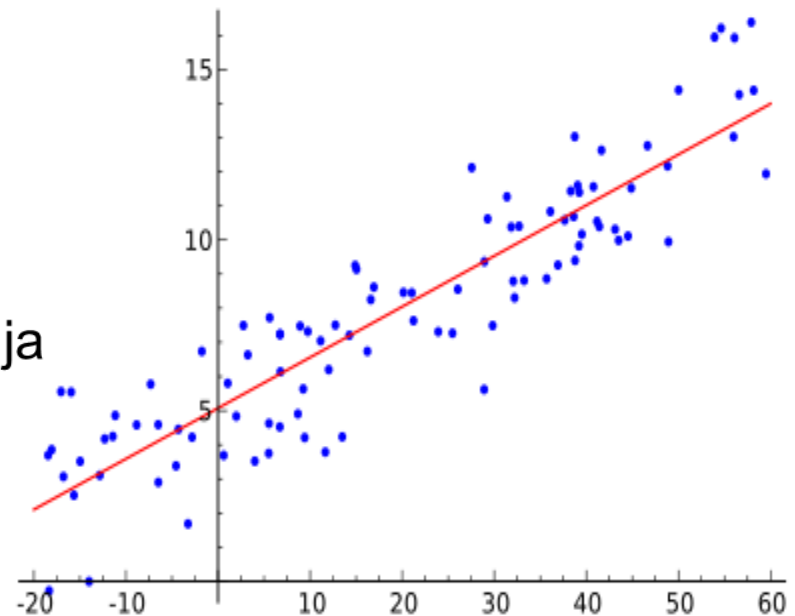
Példa: Lineáris regresszió

- Egy adott adatállomány, i.e., $\{(x_1, y_1), \dots, (x_n, y_n)\}$, esetén találjunk egy olyan lineáris függvényt, amely adott x_i vektor esetén az y_i értéket úgy jelzi előre mint $y'_i = w^T x_i$

- Találjunk egy olyan w súlyvektort, amely minimalizálja a négyzetösszeg hibát:

$$\sum_i (y'_i - y_i)^2$$

- A probléma megoldására számos módszer ismert.



Osztályozás regresszió útján

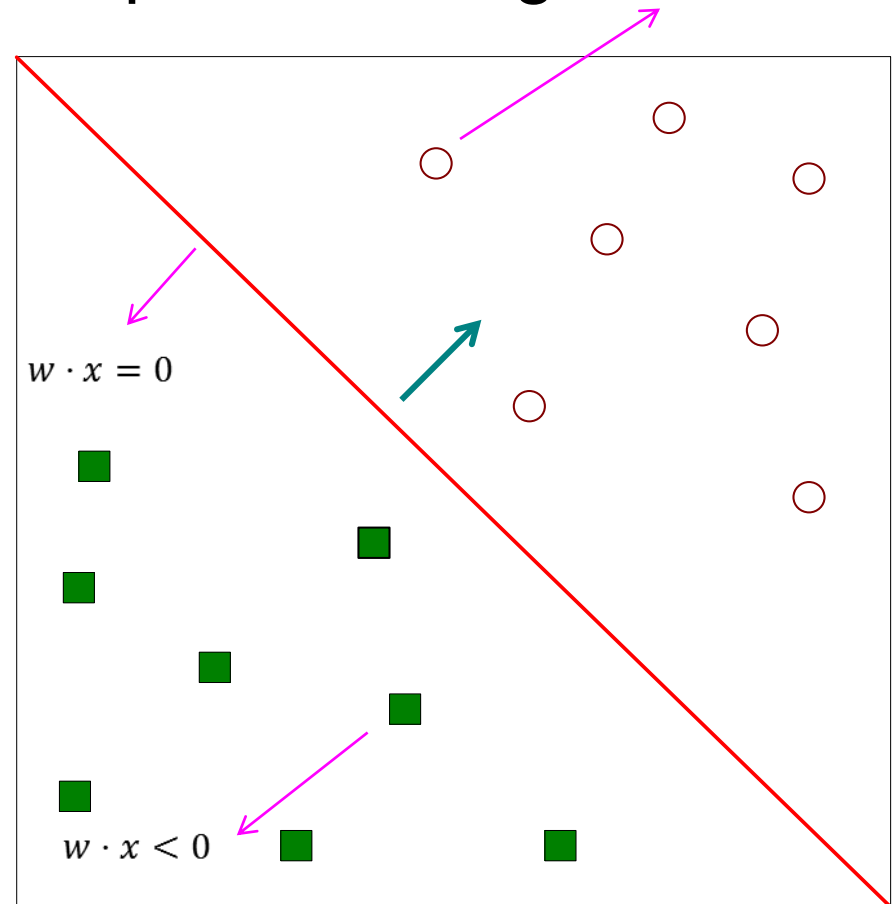
- Feltételezzük a lineáris szeperálhatóságot $w \cdot x > 0$

Pozitív osztály esetén minél **nagyobb** $w \cdot x$, a pont annál távolabb van az osztályozási határtól és annál **biztosabb** a **pozitív osztályba** való tartozás

- Definiáljuk $P(C_+|x)$ -t mint **növekvő** függvényét $w \cdot x$ -nek

Negatív osztály esetén minél **kisebb** $w \cdot x$, a pont annál távolabb van az osztályozási határtól és annál **biztosabb** a **negatív osztályba** való tartozás

- Definiáljuk $P(C_-|x)$ -t mint **csökkenő** függvényét $w \cdot x$ -nek



Logisztikus regresszió

A logisztikus függvény

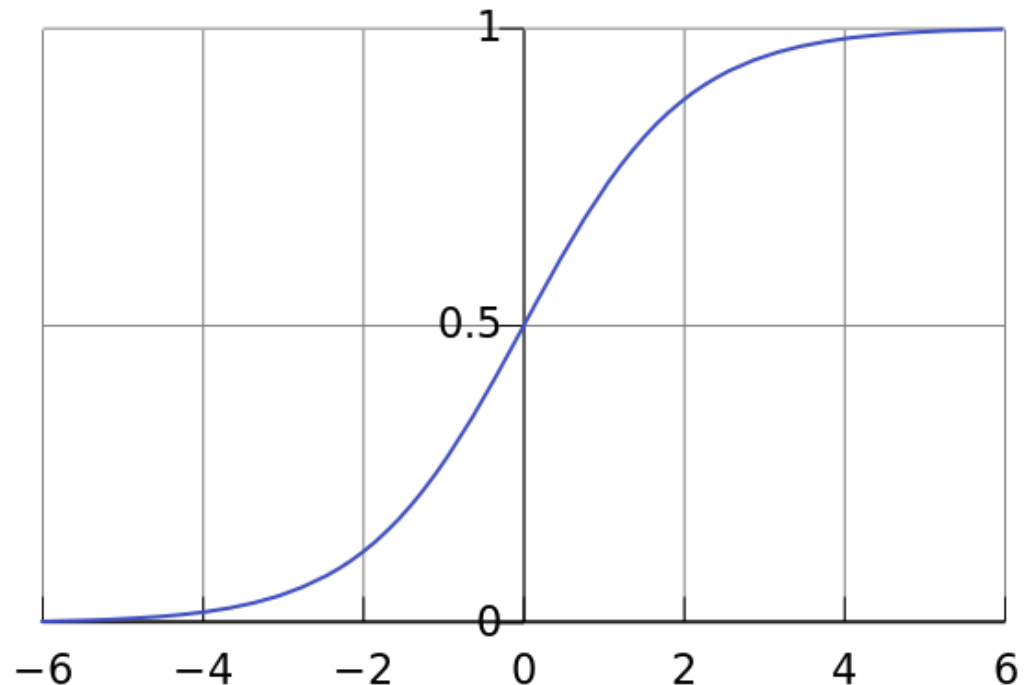
$$f(t) = \frac{1}{1 + e^{-t}}$$

$$P(C_+|x) = \frac{1}{1 + e^{-w \cdot x}}$$

$$P(C_-|x) = \frac{e^{-w \cdot x}}{1 + e^{-w \cdot x}}$$

$$\log \frac{P(C_+|x)}{P(C_-|x)} = w \cdot x$$

Lineáris regresszió a **log-odds hánydoson**



Logisztikus Regresszió: Találjunk egy olyan w vektort amely **maximalizálja** a megfigyelt adatok valószínűségét.

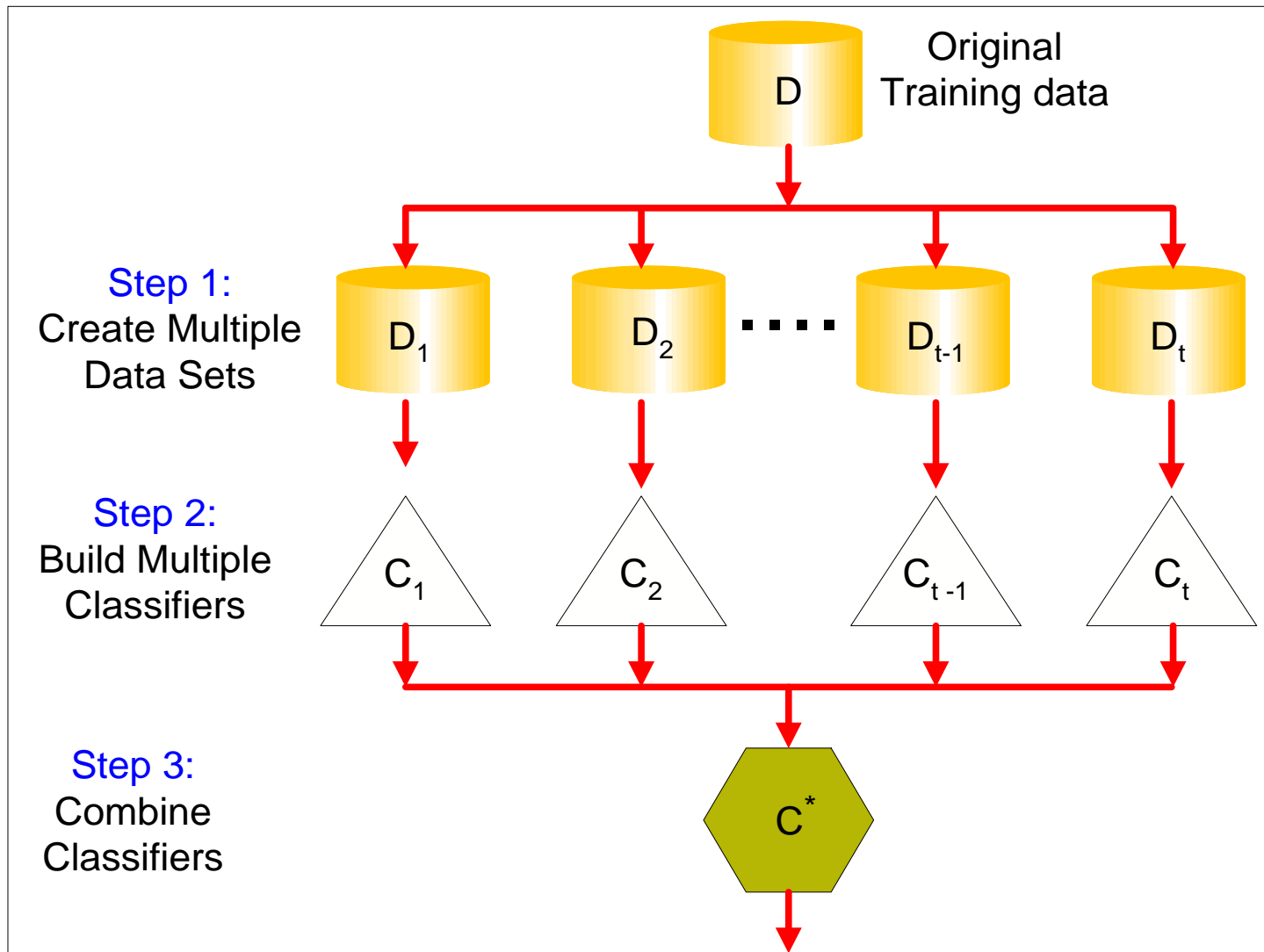
Logisztikus regresszió

- Előállítja az **osztályhoz** való tartozás **valószínűségének becslését** amely gyakran hasznos és egy pontosabb leírását adja a döntés megbízhatóságának.
- A **súlyok** hasznosak lehetnek a **jellemzők fontosságának** megértésében.
- Viszonylag nagy méretű adatállományokon is működik.
- Gyors az alkalmazásokon mivel az osztályok becslése csak a súlyvektortól függ.

Együttes módszerek

- Osztályozók egy halmazát hozzuk létre a tanító állományon.
- Egy új rekord osztályát úgy jelezzük előre, hogy a sok osztályozó által kapott előrejelzéseket összesítjük.

Általános ötlet



Miért működhet?

- Tegyük fel, hogy adott 25 egyszerű osztályozónk.
 - Minden osztályozó hibája $\varepsilon = 0.35$.
 - Tegyük fel, hogy az osztályozók függetlenek.
 - Annak valószínűsége, hogy az együttes osztályozó hibás döntést hoz:

$$\sum_{i=13}^{25} \binom{25}{i} \varepsilon^i (1 - \varepsilon)^{25-i} = 0.06$$

Példák együttes módszerekre

- Hogyan hozhatjuk létre osztályozók együttesét?
 - Bagging (bootstrap aggregating)
 - Boosting (gyorsítás)

Bagging

- Visszatevéses mintavétel

Original Data	1	2	3	4	5	6	7	8	9	10
Bagging (Round 1)	7	8	10	8	2	5	10	10	5	9
Bagging (Round 2)	1	4	9	1	2	3	2	7	3	2
Bagging (Round 3)	1	8	5	10	5	5	9	6	3	7

- Minden bootstrap mintán építünk fel egy osztályozót.
- Minden egyes rekordot $(1 - 1/n)^n$ valószínűséggel választunk ki.

Boosting (gyorsítás)

- Egy olyan iteratív eljárás, amely a tanító rekordok eloszlását adaptívan változtatva a korábban tévesen osztályozott rekordokra fókuszál.
 - Kezdetben mind az összes N rekord egyenlő súlyt kap.
 - A bagging-gel szemben a súlyok változhatnak egy iterációs ciklus befejeztével.

Boosting (gyorsítás)

- A rosszul osztályozott rekordoknak nőni fog a súlya.
- A helyesen osztályozott rekordoknak csökkenni fog a súlya.

Original Data	1	2	3	4	5	6	7	8	9	10
Boosting (Round 1)	7	3	2	8	7	9	4	10	6	3
Boosting (Round 2)	5	4	9	4	2	5	1	7	4	2
Boosting (Round 3)	4	4	8	10	4	5	4	6	3	4

- A 4. rekordot nehéz osztályozni
- A súlya nő, ezért nagyobb eséllyel választjuk ki ismét a következő körökben.

Példa: AdaBoost

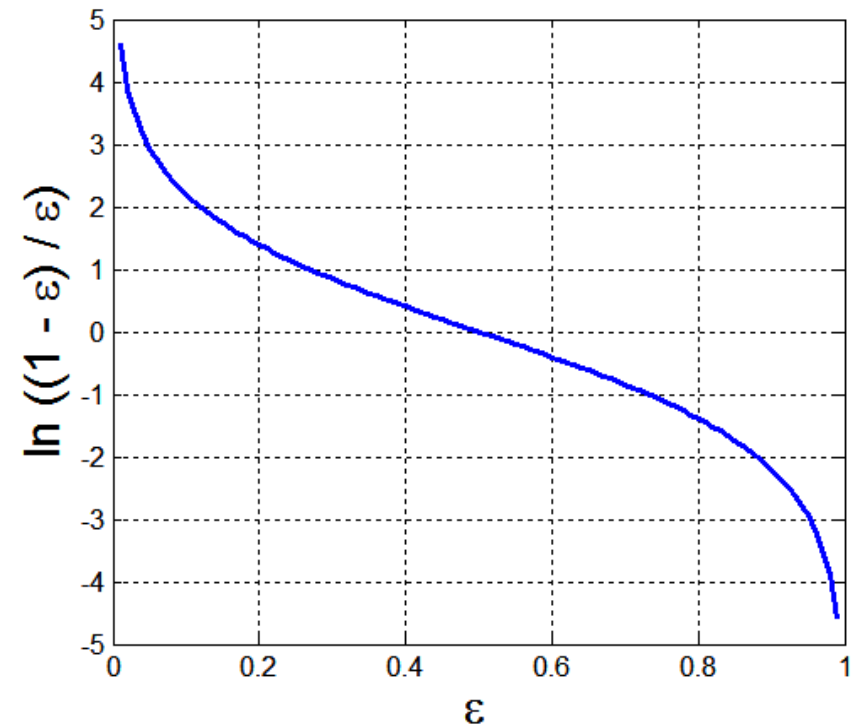
- Alap osztályozók: C_1, C_2, \dots, C_T

- Hiba ráta:

$$\varepsilon_i = \frac{1}{N} \sum_{j=1}^N w_j \delta(C_i(x_j) \neq y_j)$$

- Az osztályozó fontossága:

$$\alpha_i = \frac{1}{2} \ln \left(\frac{1 - \varepsilon_i}{\varepsilon_i} \right)$$



Példa: AdaBoost

- A súlyok frissítése:

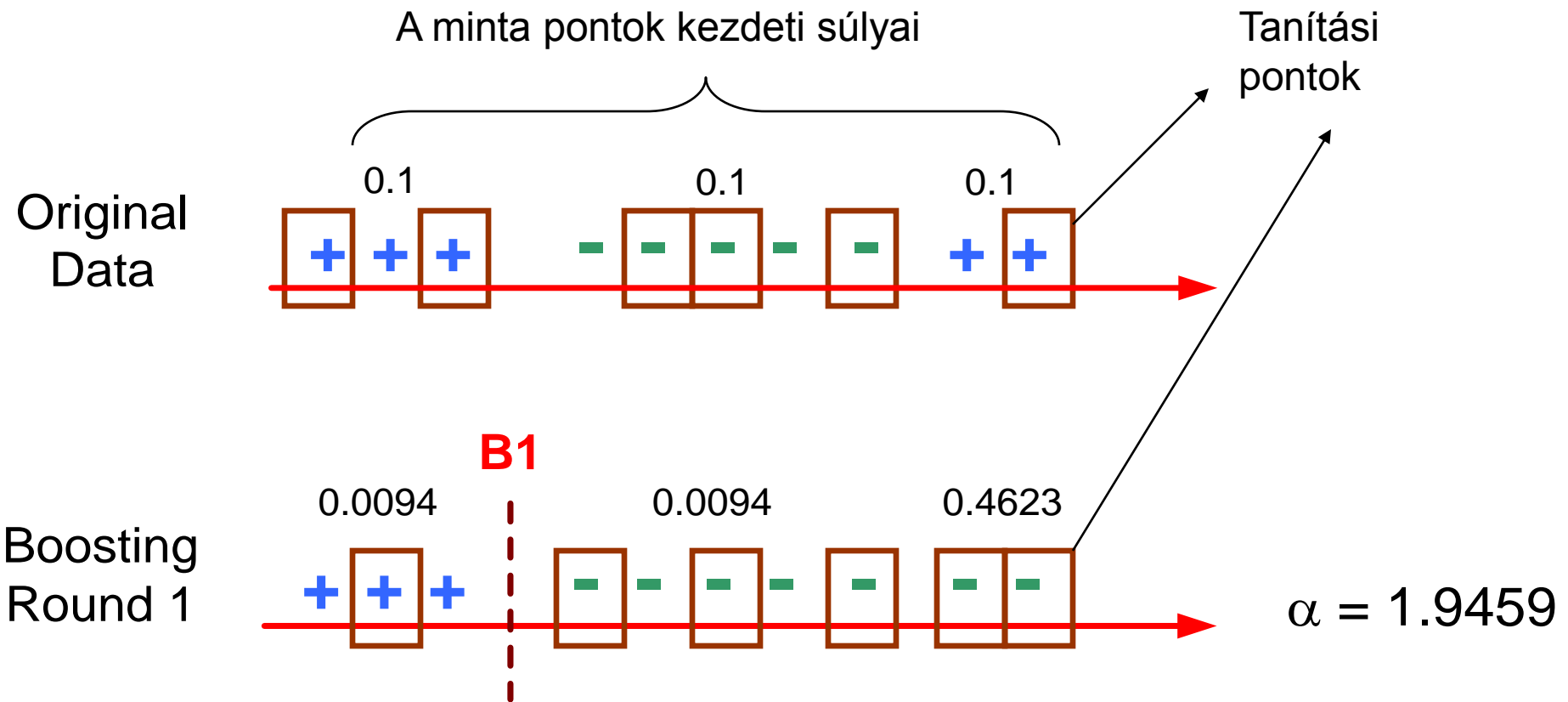
$$w_i^{(j+1)} = \frac{w_i^{(j)}}{Z_j} \begin{cases} \exp^{-\alpha_j} & \text{ha } C_j(x_i) = y_i, \\ \exp^{\alpha_j} & \text{ha } C_j(x_i) \neq y_i, \end{cases}$$

ahol Z_j egy normalizáló tényező

- Ha bármelyik közbenső körben a hiba 50% fölé megy, akkor a súlyok visszaállnak $1/n$ -re, és a mintavételi folyamat megismétlődik.
- Osztályozás:

$$C^*(x) = \arg \max_y \sum_{j=1}^T \alpha_j \delta(C_j(x) = y)$$

Az AdaBoost szemléltetése



Az AdaBoost szemléltetése

