

# Inteligență Artificială

Universitatea Politehnica Bucuresti  
Anul universitar 2021-2022

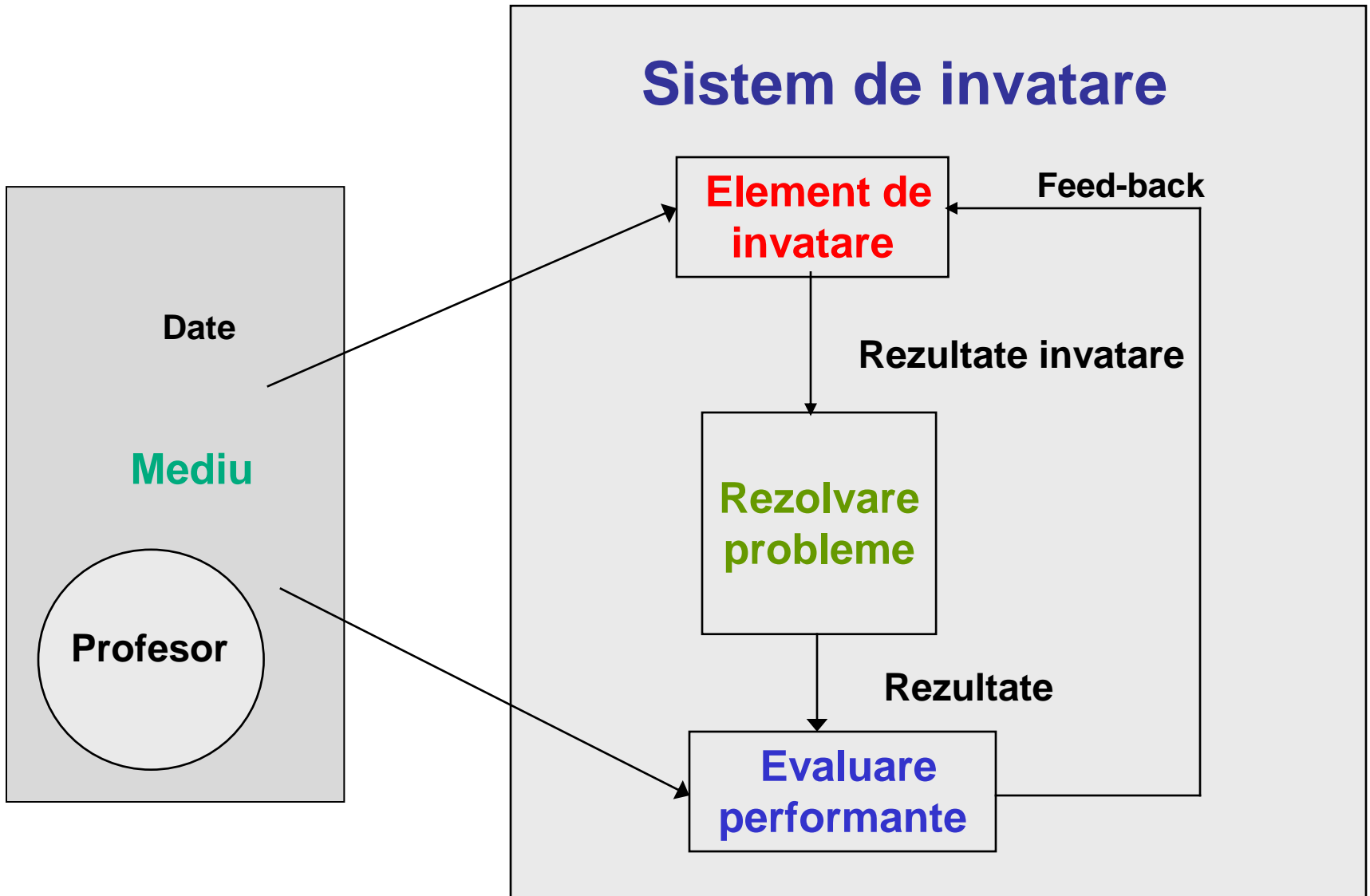
Adina Magda Florea

# Curs 7

- Elemente de învățare automată
- Arbori de decizie

# 1. Definiții

- Învățarea este procesul prin care un sistem își îmbunătățește performanțele (Herbert Simon)
- Învățarea este achiziția cunoștințelor explicite
- Învățarea este achiziția deprinderilor de rezolvare a problemelor
- Învățarea este formarea teoriilor, formarea ipotezelor și inferența inductivă
- ML – observă cantități mari de date și caută șabloane pentru predicție



# Invatare automata

## De ce sa invete?

- Taskuri definite prin exemple
- Relatii / corelatii in cantitati mari de date
- Mediu in schimbare
- Date diferite, cu zgomot
- Cantitate de cunostinte prea mare pentru a fi reprezentate explicit

# Denumiri utilizate

- Instanță
- Concept
- Concept vizat (target concept)
- Clasă de ipoteze
- Mulțimea de învățare (Training set)
- Mulțimea de test (Test set)

# Denumiri utilizate

- $T$  – vector de intrare, vector sablon, vector de caracteristici, esantioane, exemple, instanțe
- $x_i$  – caracteristici, attribute, variabile de intrare, componente
- $x_i$  – pot fi valori reale, valori numerice intregi, valori simbolice, valori booleene
- $f(X_i)$ 
  - valori reale:  $h$  – functie de esantionare
  - valori simbolice:  $h$  – clasificator
  - booleene: 1 – instanta pozitiva, 0 – instanta negativa

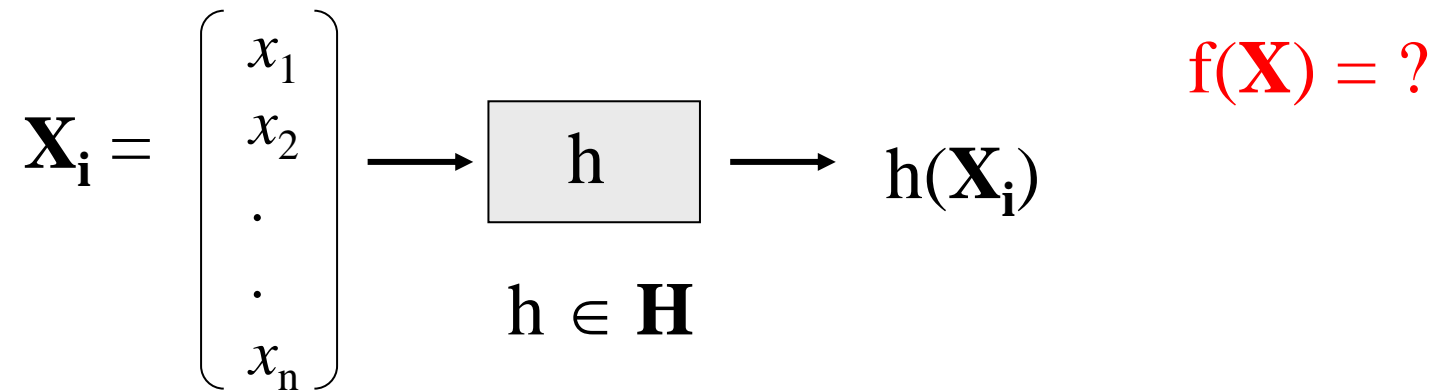
## 2. Tipuri de invatare

- **Învățare supervizată** – determinarea ipotezei de invatare pe baza unor date etichetate
  - Simbolica sau non-simbolica
- **Învățare nesupervizată** - determinarea ipotezei de invatare / a unei structuri pe baza unor date neetichetate
  - Simbolica sau non-simbolica
- ***Învățarea prin recompensa***



# Model simplu

$T = \{\mathbf{X}_1, \mathbf{X}_2, \dots, \mathbf{X}_m\}$  – **multime de invatare**



■ **Invatare supervizata** – se cunosc  $f(\mathbf{X}_1), \dots, f(\mathbf{X}_m)$

Gasim **h** a.i.  $h(\mathbf{X}_i) = f(\mathbf{X}_i)$ ,  $i=1, m \rightarrow h(\mathbf{X}_i) = f(\mathbf{X}_i)$ ,  $\forall i$

Clasificare – **f** - valori discrete – grupeaza exemple

Regresie – **f** – valori continue, estimeaza sau prezice valori

# Metode de invatare supervizata

## Clasificare

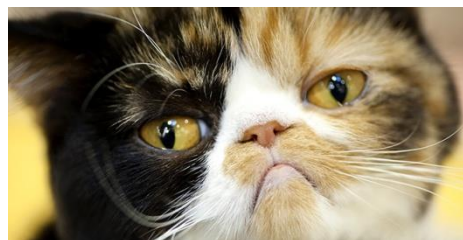
- Arbori de decizie
- Clasificare probabilistica (Naïve Bayes)
- K-Nearest neighbours (k-NN pt clasificare)
- SVM
- Retele neurale
- Invățare mulțime (ensemble learning): păduri aleatoare, ADA boost

# Metode de invatare supervizata

## **Regresie**

- Regresie liniara
- SVR
- Arbori de regresie
- K-Nearest neighbours (k-NN pt regresie)

# Clasificare vs regresie



## ■ Caine / Pisica

Clasificare

- Dan 10.000 lei
- George 3.000 lei
- Vlad 15.000 lei
- Maria 5.000 lei

credit de 30.000 lei  
credit de 10.000 lei  
credit de 50.000 lei  
cat credit?

Regresie

# Clasificare vs regresie

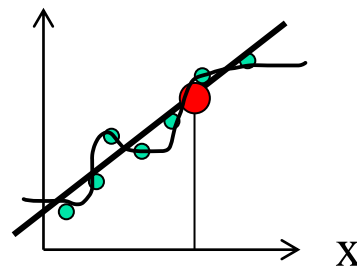
- Reprezentam ipotezele de invatare sub o forma care sa permita ca iesirea algoritmului sa fie o eticheta (clasa): arbori, retele neurale, etc.

Clasificare

- Reprezentam ipoteza de invatare ca o functie liniara

$$h(x) = a_0 + a_1 x$$

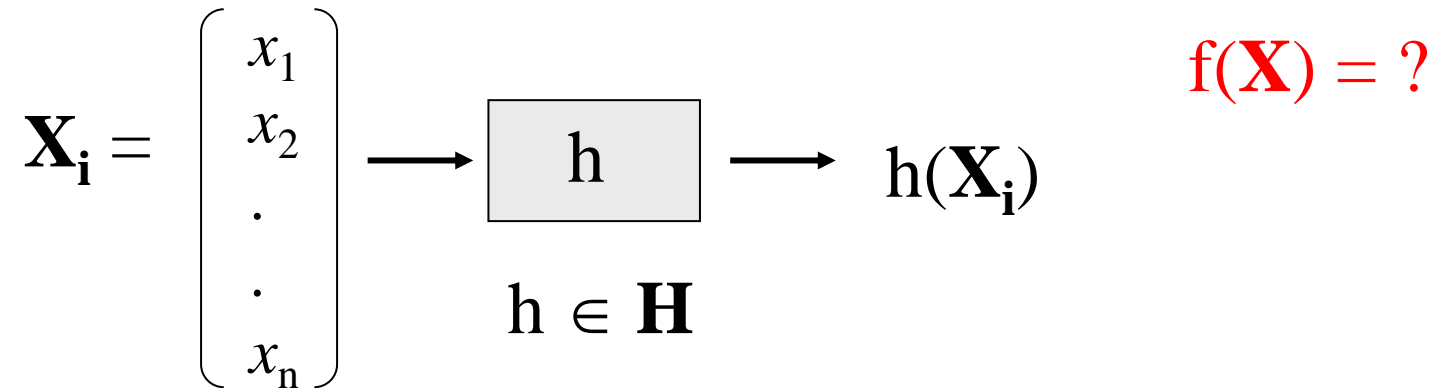
- Cat este valoarea y pt punctul rosu?



Regresie

# Model simplu

$T = \{\mathbf{X}_1, \mathbf{X}_2, \dots, \mathbf{X}_m\}$  – **multime de invatare**



- **Invatare ne-supervizata** – NU se cunosc  $f(\mathbf{X}_1), \dots, f(\mathbf{X}_m)$

Imparte  $T$  in submultimi – clase

Se poate vedea tot ca invatarea unei functii – val  $f$  = numele submultimii careia ii apartine  $\mathbf{X}_i$

- **Invatare prin recompensa** – Se cunosc recompensele pentru valorile  $h(\mathbf{X}_1), \dots, h(\mathbf{X}_m)$ , Nu se cunosc valorile lui  $f$

# Metode de invatare nesupervizata

- Clustering
  - k-means clustering (grupare), k-means ierarhic (grupare ierarhica)
- Retele neurale
  - Retele neurale cu auto-organizare
  - Autoencoders
  - Deep belief networks

# Metode de invatare nesupervizata

- Expectation maximization (EM)
  - HMM – algoritmul Baum-Welch
  - Retele Bayesiene – estimarea parametrilor in cazul datelor incomplete
  - Soft clustering – clasele se pot suprapune
- Principal Component Analysis (PCA)



# Caracteristici importante

- **Regimuri de invatare:**

- Batch
- Incremental

- **Zgomot:**

- zgomot intrari (de ex valorile atributelor)
- zgomot iesiri (alterare iesiri)

- **Ipoteza favorita a inductiei** (Inductive bias)

- O multime de presupuneri (explicite sau implicite) pe care algoritmul de invatare se bazeaza pentru a obtine un model (a generaliza) din setul de invatare

# Inductive bias

- Spatiu de ipoteze restrictionat (Restricted Bias)
  - Functii liniare sau polinomiale de ordin mic
- Ipoteza preferata (Preference Bias)
  - Occam's razor
  - Nerest neighbours
  - Independenta conditionala maxima
  - Margine maxima
  - CNN, Deep RL agents

# Occam's Razor

## Principiul lamei lui Occam (*lex parsimoniae* )

- prefera explicatiile simple celor complexe
- selecteaza ipoteza care implica cele mai putine presupuneri, intre ipoteze similare/egal probabile
- Wiliam of Occam, 1285 – 1347
- filozof englez

*"non sunt multiplicanda entia praeter necessitatem"*



# Conditii pentru o invatare "corecta"

**Problema:** identifica personaje de film  
"bune" sau "rele" dupa modul in care arata

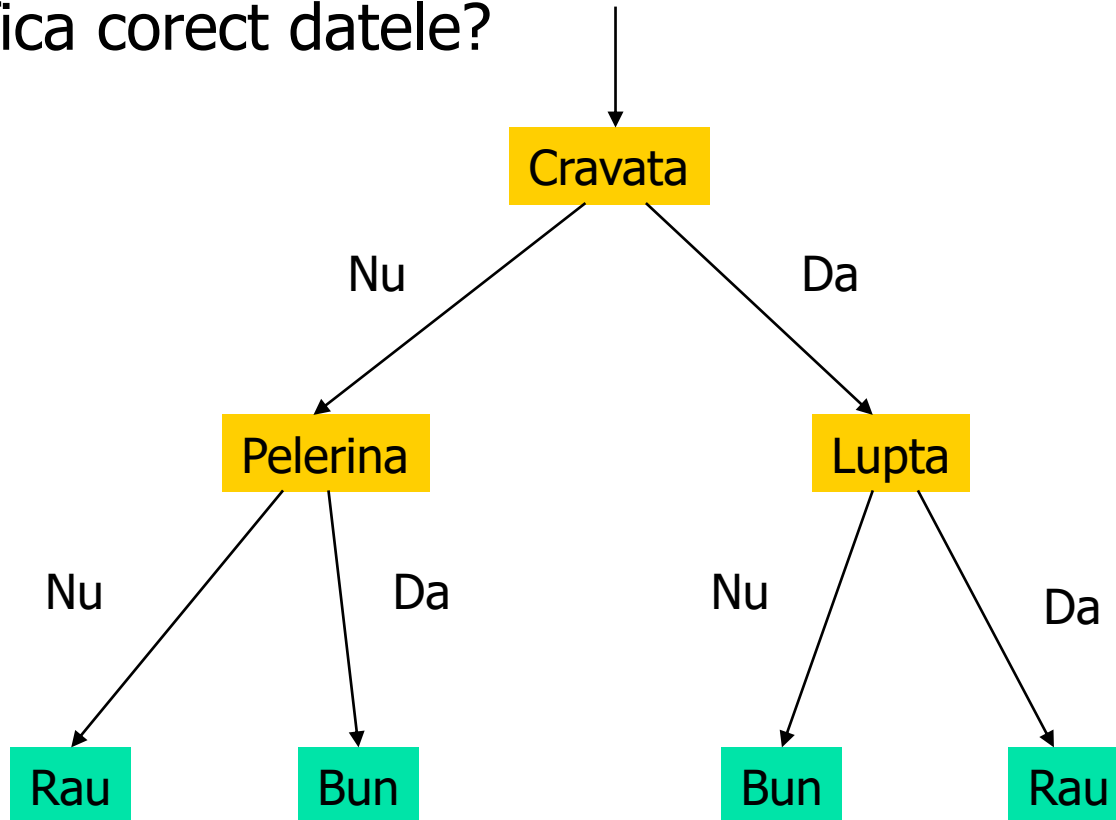


# Conditii pentru o invatare "corecta"

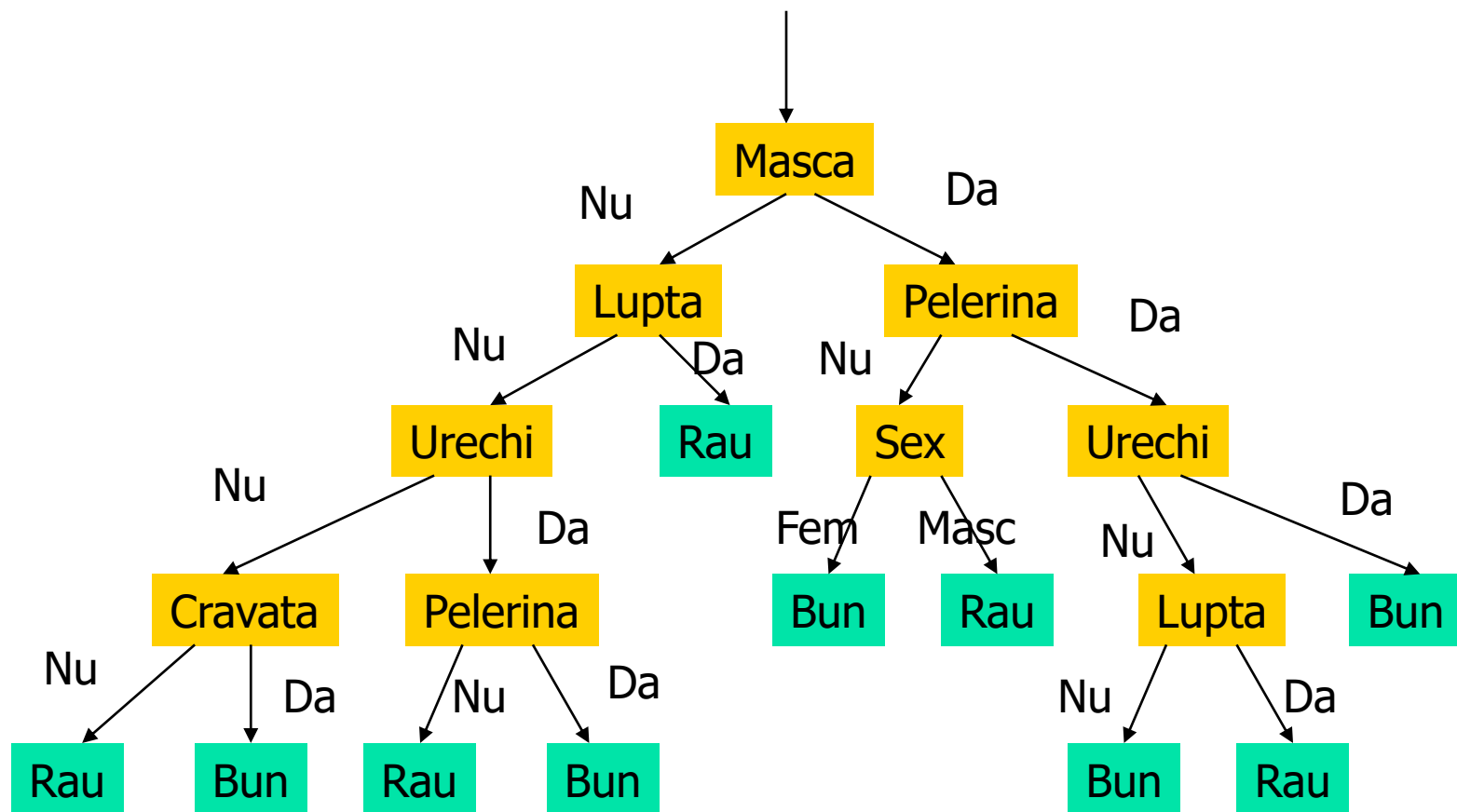
Atribute / Instante	Sex	Masca	Pelerina	Cravata	Urechi	Lupta	Clasa
	Set de invatare						
Batman	Masc	Da	Da	Nu	Da	Nu	Bun
Robin	Masc	Da	Da	Nu	Nu	Nu	Bun
Alfred	Masc	Nu	Nu	Da	Nu	Nu	Bun
Penguin	Masc	Nu	Nu	Da	Nu	Da	Rau
Catwoman	Fem	Da	Nu	Nu	Da	Nu	Rau
Joker	Masc	Nu	Nu	Nu	Nu	Nu	Rau
	Date de test						
Batgirl	Fem	Da	Da	Nu	Da	Nu	??
Fred	Masc	Da	Nu	Nu	Nu	Nu	??

# Conditii pentru o invatare "corecta"

Clasifica corect datele?



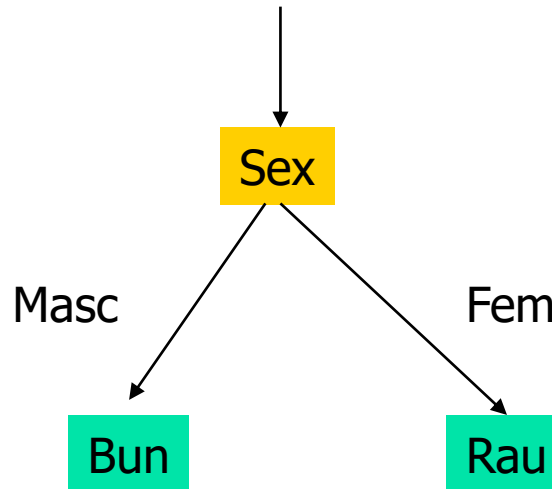
# Conditii pentru o invatare "corecta"



Clasifica corect datele?

# Conditii pentru o invatare "corecta"

Clasifica corect datele?



Aleg prima varianta (cf. lamei lui Occam)

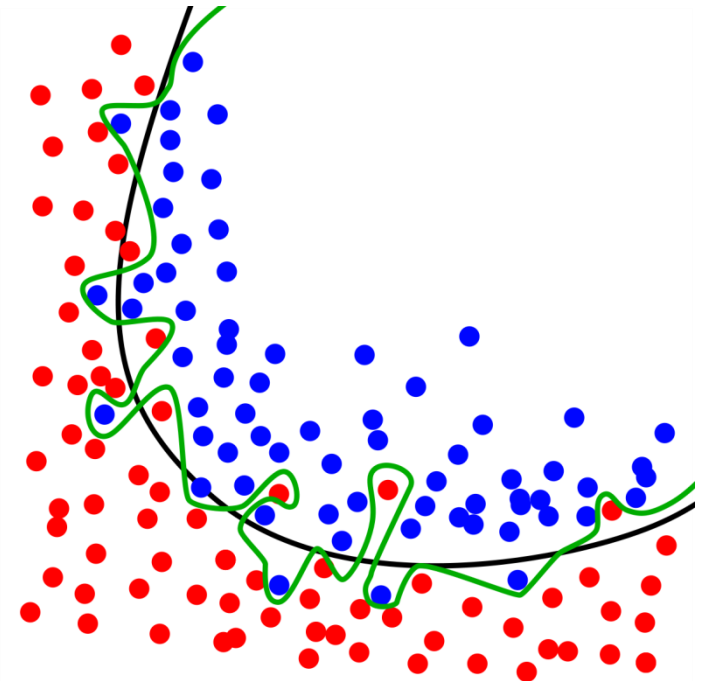


# Conditii pentru o invatare "corecta"

- Clasificatoarele trebuie sa fie suficient de "expresive" pentru a fi in concordanta cu setul de invatare
- Dar clasificatoarele care au o complexitate prea mare pot duce la fenomenul de "overfit" (overfitting)

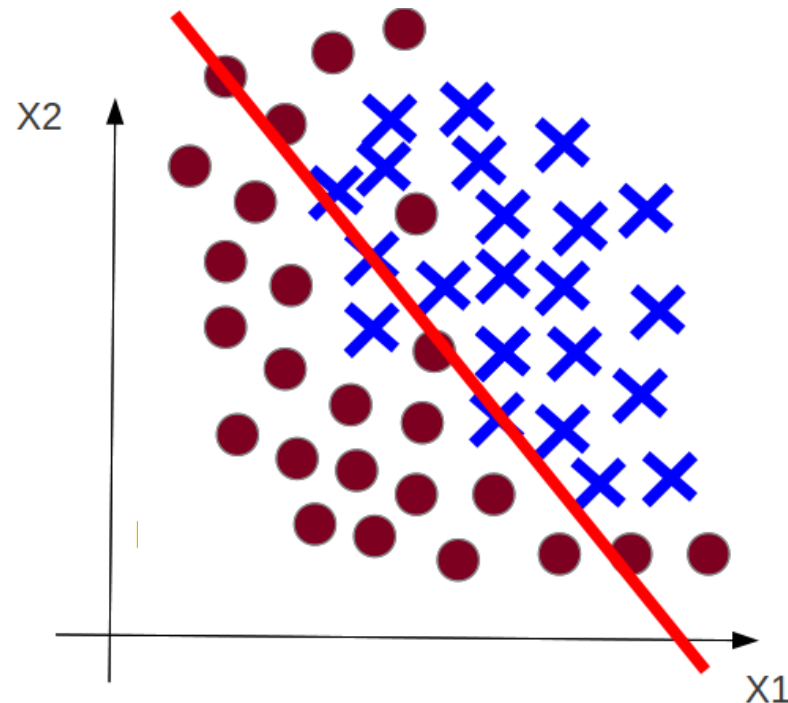
# Caracteristici importante

- **Overfitting** = modelul include zgomot sau sabloane de date nerelevante
- Nu mai poate generaliza
- Validare încrucișată
- Regularizare



# Caracteristici importante

- **Underfitting** = modelul nu se potrivește nici datelor de învățare și nici nu poate generaliza la date noi



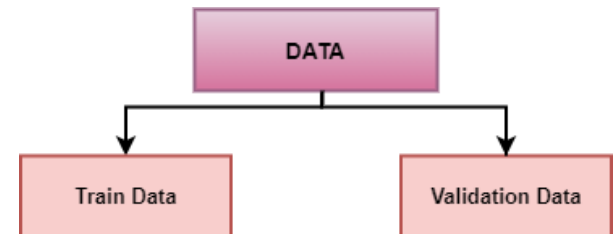
# Caracteristici importante

- **Regularizare**
- Evită overfitting
- Se poate aplica aproape oricărui model ML
- Simplifică modelele cu complexitate prea mare prin adăugarea de termeni de penalizare în funcția obiectiv

# Caracteristici importante

## Cross-validation

- Antrenarea se face pe subseturi de date de învățare și evaluare pe restul
- Metode:
  - Hold out
  - K-fold cross validation
  - Leave-P-out



### 3. Metrici de performanță

#### Regresie

- **Mean Absolute Error (MAE)** = media diferenței absolute între valori corecte și valori prezise
- **Root Mean Square Error (RMSE)** = rădăcina pătrată a mediei diferențelor pătratelor între valori corecte și valori prezise

# Metri de performanță pt clasificare

## Clasificare

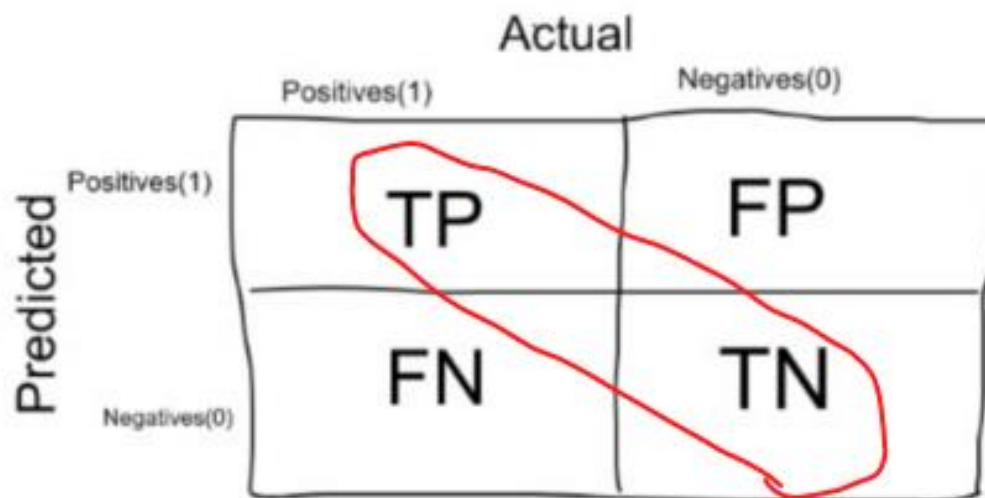
- **Confusion matrix** – corectitudinea și acuratețea modelului

		Actual Values	
		Positive (1)	Negative (0)
Predicted Values	Positive (1)	TP	FP
	Negative (0)	FN	TN

# Metrici de performanță pt clasificare

- **Accuracy**– utilă dacă clasele sunt distribuite uniform

		Actual	
		Positives(1)	Negatives(0)
Predicted	Positives(1)	TP	FP
	Negatives(0)	FN	TN



$$\text{Accuracy} = \frac{TP + TN}{TP + FP + FN + TN}$$



# Metrici de performanță pt clasificare

## ■ Precision (hit)

		Actual	
		Positives(1)	Negatives(0)
Predicted	Positives(1)	TP	FP
	Negatives(0)	FN	TN

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

Precision

# Metrici de performanță pt clasificare

## ■ Recall sau Sensitivity (misses)

		Actual	
		Positives(1)	Negatives(0)
Predicted	Positives(1)	TP	FP
	Negatives(0)	FN	TN

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

Recall or Sensitivity

# 4 Arbori de decizie

- ID3 – in jur de 1960s
- C4.5 (Quinlan):
- Permite attribute numerice
- Trateaza cazurile valorilor lipsa
- Trateaza cazul valorilor cu zgomot
- C4.5 – unul din cei mai cunoscuti algoritmi de invatare
- Ultima versiune de cercetare: C4.8, implementata in Weka
- Versiunea comerciala: C5.0

# Invatarea inductiva prin AD

- Vede invatarea ca achizitia cunostintelor structurate
- Reprezentarea cunostintelor = **arbori de decizie** (AD)
- Problema de invatare = **clasificare**
- **Invatare supervizata**
- Strategie = invatare batch (ne-incrementala)
- AD se construiesc pornind de la radacina spre frunze  
= *Top Down Induction of Decision Tree*

# ID3 (Quinlan)

- Univers de obiecte  $U$  descrise in termenii unei colectii de attribute  $\{A\}$
- Fiecare **atribut** masoara o caracteristica importanta a unui obiect  $o \in U$
- Domeniul de valori attribute  $D_A =$  discret, simbolic (ulterior extins)
- Fiecare obiect apartine unui **clase** dintr-o multime de clase mutual exclusive  $\{C\}$
- Se da **setul de invatare** (SI)
- **Problema** = obtinerea unor **reguli de clasificare** / construirea unui **AD** care clasifica corect nu numai  $\forall o \in SI$  dar si  $\forall o \in U$

# ID3 (Quinlan)

- Structura iterativa – fereastra din SI
- S-au gasit AD corecti in cateva iteratii pt 30 000 obiecte cu 50 attribute
- Empiric s-a aratat ca iterativ se obtin arbori mai buni decat daca s-ar construi din tot SI
- Utilizare AD
- Reguli de decizie

# ID3 (Quinlan)

## Metoda de constructie

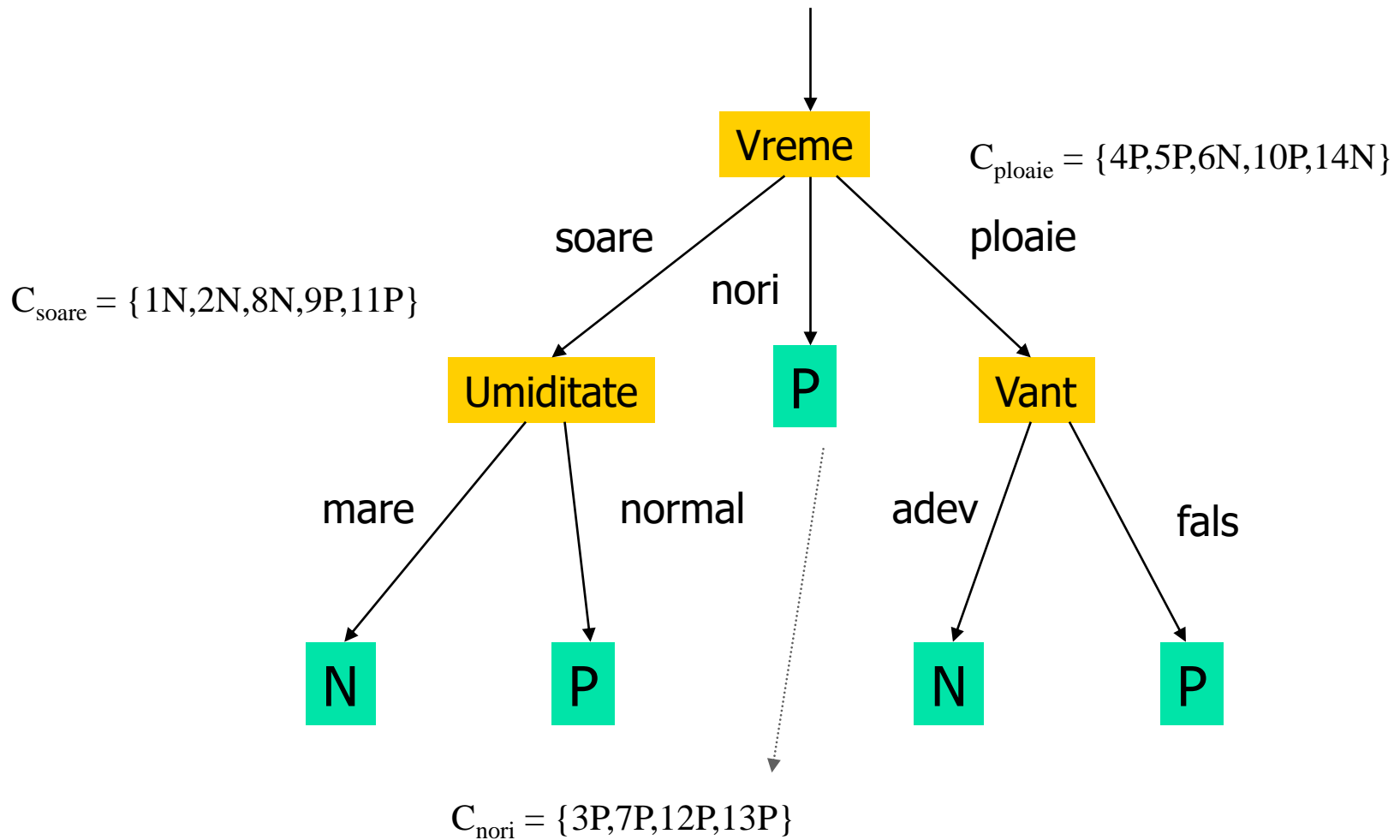
- $C$  = multimea de obiecte / ex inv. din SI
- $A^k$  – atribut test cu valori  $a^k_1, \dots, a^k_v$
- "divide-and-conquer"
- Impartirea/expandarea AD se opreste cand toate obiectele din  $C$  apartin unei aceleiasi clase
- Se termina intotdeauna (in cazul cel mai nefavorabil, cate un obiect in fiecare clasa)

# ID3 – Exemplu

No.	Atribute				Clasa
	Vreme	Temperatura	Umiditate	Vant	
1	soare	cald	mare	fals	N
2	soare	cald	mare	adev	N
3	nori	cald	mare	fals	P
4	ploaie	placut	mare	fals	P
5	ploaie	racoare	normal	fals	P
6	ploaie	racoare	normal	adev	N
7	nori	racoare	normal	adev	P
8	soare	placut	mare	fals	N
9	soare	racoare	normal	fals	P
10	ploaie	placut	normal	fals	P
11	soare	placut	normal	adev	P
12	nori	placut	mare	adev	P
13	nori	cald	normal	fals	P
14	ploaie	placut	mare	adev	N



# ID3 – Exemplu



# ID3 – Arbore minim

- Din acelasi SI se pot contrui diferiti AD
- Cum se poate obtine cel mai mic arbore (lama lui Occam) ?
- = Cum selectez atributul din radacina unui arbore?

# ID3 – Cum selectez A?

- Multimea de exemple  $C$  cu un numar de ex pozitive  $p \in P$  si exemple negative  $n \in N$

Se presupune ca:

- (1) Orice AD corect va clasifica obiectele proportional cu reprezentarea lor in  $C$

Un obiect (exemplu)  $o \in C$  va fi clasificat:

- $\in P$  cu probabilitatea  $p/(p+n)$
  - $\in N$  cu probabilitatea  $n/(p+n)$
- (2) Cand un AD este utilizat pentru a clasifica obiecte, acesta intoarce o clasa  $\Rightarrow$   
  
AD poate fi vazut ca o sursa a unui mesaj 'P' sau 'N' avand informatia necesara pentru a genera acest mesaj

# Teoria informatiei ofera criteriul

- Pentru un univers de mesaje

$$M = \{m_1, m_2, \dots, m_n\}$$

si o probabilitate  $p(m_i)$  de aparitie a fiecarui mesaj, **continutul informational**  $I(M)$  al mesajelor din  $M$  se defineste astfel:

$$I(M) = \sum_{i=1}^n -p(m_i) \log_2(p(m_i))$$

# Selectia testului (atributului)

- Fie multimea de obiecte  $C$  cu  $p \in P$  si  $n \in N$
- **Continutul de informatie**  $I(AD_{p,n})$  este

$$I(AD_{p,n}) = -\frac{p}{p+n} \log_2 \frac{p}{p+n} - \frac{n}{p+n} \log_2 \frac{n}{p+n}$$

- Selecteaza  $A$  in radacina; valori  $A \in \{a_1, \dots, a_v\}$
- Fie  $C_i$  cu  $p_i \in P$  si  $n_i \in N$ ,  $i=1, v$
- Continutul de informatie pentru fiecare  $C_i$  este  
 $I(AD_{p_i, n_i})$ ,  $i=1, v$

# Selectia testului (atributului)

- Dupa selectarea lui A in radacina, **cantitatea de informatie** necesara pentru a termina constructia arborelui este suma ponderata a continutului de informatie din toti subarborii

$$E(A) = \sum_{i=1}^v \frac{p_i + n_i}{p + n} I(AD_{pi,ni})$$

- unde ponderea ramurii  $i$  este fractiunea de obiecte din  $C$  care apartin lui  $C_i$ ;
- $v$  este numarul de valori ale lui  $A$

# Selectia testului (atributului)

- **Castigul informational** al unui atribut A obtinut prin selectia acestuia ca radacina a arborelui de decizie este:

$$G(A) = I(AD_{p,n}) - E(A)$$

- Se selecteaza A cu castig informational maxim
- Recursiv pentru a forma AD corespunzatori multimilor  $C_1 \dots C_v$

# Calcul G(A) pt Ex

- 14 exemple,  $9 \in P$ ,  $5 \in N$

- $I(AD_{p,n}) = 0.940$  bits

$$-\frac{9}{14} \log_2 \frac{9}{14} - \frac{5}{14} \log_2 \frac{5}{14}$$

- vreme

- soare -  $2 \in P$ ,  $3 \in N \Rightarrow I(AD_{p1,n1}) = 0.971$

- nori -  $4 \in P$ ,  $0 \in N \Rightarrow I(AD_{p2,n2}) = ?$

- ploaie -  $3 \in P$ ,  $2 \in N \Rightarrow I(AD_{p3,n3}) = ?$

- $E(vreme) = 0.694$  bits

$$\frac{5}{14} I(AD_{p1,n1}) + \frac{4}{14} I(AD_{p2,n2}) + \frac{5}{14} I(AD_{p3,n3})$$

- $G(vreme) = 0.940 - 0.694 = 0.246$  bits

- $G(temperatura) = 0.029$  bits

- $G(umiditate) = 0.151$

- $G(vant) = 0.048$  bits

$$C_{\text{soare}} = \{1N, 2N, 8N, 9P, 11P\}$$

$$C_{\text{nori}} = \{3P, 7P, 12P, 13P\}$$

$$C_{\text{ploaie}} = \{4P, 5P, 6N, 10P, 14N\}$$



# Generalizare la mai multe clase

- **Continutul de informatie**

$$I(Arb) = \sum_{i=1}^v -p(Cl = C_i) * \log_2 p(Cl = C_i)$$

- **Cantitatea de informatie necesara pentru a termina constructia arborelui**

$$E(A) = \sum_{i=1}^v \frac{|C_i|}{|C|} I(C_i)$$

- **Castigul informational**

$$G(A) = I(Arb) - E(A)$$

# Algoritm ID3

## functie **ind-arbore** (set-invatare, atribute, default)

1. **daca** set-invatare = vid **atunci intoarce** frunza etichetata cu *default* sau "Failure" Caz 1 – ex inv lipsa
2. **daca** toate exemplele din set-invatare sunt in aceeaasi clasa **atunci intoarce** o frunza etichetata cu acea clasa Bine = recunoaste
3. **daca** atribute este vida **atunci intoarce** o frunza etichetata cu disjunctia tuturor claselor din set-invatare Caz 2 – atr inadecvate
4. selecteaza un atribut A, creaza nod pt A si eticheteaza nodul cu A
5. sterge A din atribute → atribute1
6. **m** = cea mai frecventa clasa (set-invatare)
7. **pentru** fiecare valoare V a lui A **repeta**
  - fie  $\text{partitie}_V$  multimea exemplilor din set-invatare, cu valoarea V pentru A
  - creaza  $\text{nod}_V = \text{ind-arbore}(\text{partitie}_V, \text{atribute1}, m)$
  - creaza legatura nod A –  $\text{nod}_V$  etichetata cu V

**sfarsit**

# Complexitate

- In fiecare nod cu exceptia frunzelor trebuie aflat  $G$  (castig informational) pt fiecare atribut  $A$
- $G$  depinde de valorile  $p_i$  si  $n_i$  pentru fiecare valoare  $a_i$  a lui  $A \Rightarrow$  fiecare obiect din  $C$  trebuie investigat (clasa, valoare  $A$ )  $\Rightarrow$
- $O(|C| * |A|)$  ,  $|A|$  - nr attribute
- Pentru fiecare iteratie, **complexitatea ID3**
- $O(|C| * |A| * |AD|)$  , unde  $|AD|$  - numar de noduri interne  $AD$

# Cazuri speciale

**Caz 1.** Nu exista obiecte  $o \in C$  pentru care  $A=A_j$

(exemple de invatare lipsa)

- ID3 eticheteaza frunzele cu "*null*" sau "*Failure*" – deci nu clasifica in aceste noduri
- **Solutie**
  - Generalizeaza si se atribuie frunzei clasa cu cea mai mare frecventa de aparitie in C (cea mai frecventa)

# Cazuri speciale: Zgomot

## **Caz 2. Informatia din SI este afectata de zgomot** (attribute inadecvate)

- Zgomot
  - valori de atribut ale obiectelor din C afectate de zgomot
  - clasificare incorecta a obiectelor din C
- Erorile din C (zgomotele) pot duce la 2 probleme:
  - AD cu complexitate mai mare decat este necesar (a)
  - attribute inadecvate (b)

# Cazuri speciale: Zgomot

## Modificari necesare ID3 pt a trata zgomotul

- (a) Trebuie **sa decida daca testarea unor attribute** suplimentare va creste sau nu acuratetea predictiva a AD
- (b) Trebuie sa poata **lucra cu attribute inadecvate**

## Cum se realizeaza (a)

### Solutie

- $G(A) > \text{prag } \alpha$  absolut sau relativ
- $\alpha$  suficient de mare pt a elimina attribute nerelevante - dar elimina si attribute relevante pt cazul fara zgomot

# Cazuri speciale: Zgomot attribute

## Cum se realizeaza (b – attribute inadecvate)

- Trebuie produsa o eticheta pt  $C_i$  dar obiectele nu sunt in aceeasi clasa

### Solutia 1

- Se utilizeaza notiunea de apartenenta la o clasa cu o anumita probabilitate, de ex.  $p_i/(p_i+n_i)$

### Solutia 2

- Eticheteaza cu clasa cea mai numeroasa: P daca  $p_i > n_i$ , N daca  $p_i < n_i$ , oricare (P sau N) daca  $p_i = n_i$

# Cazuri speciale: Extinderi C4.5

## Caz 3. Valori necunoscute de atribute

### 3.1 Valori de atribute lipsa in setul de invatare

#### Solutia 1

- Atribuire valoarea cu cea mai mare frecventa

#### Solutia 2

- Foloseste probabilitati pt a determia distributia de probabilitate a valorilor lui A in C in functie de apartenenta la o clasa

$$prob(A = A_i | clasa = P) = \frac{prob(A = A_i \wedge clasa = P)}{prob(clasa = P)}$$

si alege valoarea cu cea mai mare probabilitate



# Cazuri speciale: attribute lipsa SI

## Solutia 3

- Construiești un AD pt a învăța valorile atributelor lipsa
- $C' \subset C$ ,  $C'$  cu valori pt  $A$
- În  $C'$  clasa este privită ca un atribut cu valori  $P$  sau  $N$
- Valorile lui  $A$  – clasele de învățat
- Obține  $AD'$  utilizat apoi pentru a clasifica obiectele din  $C-C' \Rightarrow$  determin valoarea atributului  $A$
- Solutia 3 > Solutia 2 > Solutia 1

# Cazuri speciale: Extinderi C4.5

## Caz 4. Atribute cu multe valori

- $A_1 \dots A_n$  - f multe valori simbolice sau valori numerice / continue, sau chiar valori aleatoare
- Castig mare  $\rightarrow$  arbore cu 1 nivel

## Solutia 1 (valori numerice)

- Partitionez in intervale  $(A_i + A_{i+1})/2$ , fiecare interval o valoare

# Cazuri speciale: A cu multe valori

## Exemplificare Solutia 1 (valori numerice)

- Partitionez in intervale  $(A_i + A_{i+1})/2$ , fiecare interval o valoare
- Ordonez valorile atributului din setul de invatare

64	65	68	69	70	71	72	72	75	75	80	81	83	85
da	nu	da	da	da	nu	nu	da	da	da	nu	da	da	nu
64.5	66.5			70.5		72	73.5		77.5	80.5		84	

# Cazuri speciale: Extinderi C4.5

## Solutia 2 (valori numerice)

Pt fiecare  $A_i$ ,  $i=1,m$  imparte obiectele in  $(-\infty, A_i]$  si

$(A_i, +\infty) \Rightarrow$  partitie  $P_i$

- Pentru fiecare  $P_i$  calculeaza castigul informational si selecteaza partitia cu castig informational maxim

# Cazuri speciale: A cu multe valori

## Solutia 3 (valori simbolice)

- Utilizeaza **Informatia de separare** = cantitatea de informatie necesara pentru a determina valoarea unui atribut A intr-un set de invatare C
- Fie  $P_{A,C}$  *distributia de probabilitate* a valorilor lui A

$$P_{AC} = (\frac{|A_1|}{|C|}, \dots, \frac{|A_v|}{|C|})$$

- **Informatia de separare**

$$ISep(A) = -\sum_{i=1}^v \frac{p_i + n_i}{p + n} \log_2 \frac{p_i + n_i}{p + n}$$

# Cazuri speciale: A cu multe valori

$$GR(A) = \frac{G(A)}{ISep(A)} \quad \text{sa fie cat mai mare}$$

- Dar  $GR$  poate sa nu fie intotdeauna definita ( $ISep=0$ ) sau sa tinda sa favorizeze  $ISep$  mici
- Selecteaza dintre attributele cu  $G(A)$  mare (peste medie) acelea cu  $GR(A)$  cel mai bun
- $G(\text{vreme}) = 0.246 \text{ bits}$  ←
- $G(\text{temperatura}) = 0.029 \text{ bits}$
- $G(\text{umiditate}) = 0.151 \text{ bits}$  ←
- $G(\text{vant}) = 0.048 \text{ bits}$
- $ISep(\text{vreme}) = 1.578, \quad ISep(\text{umiditate}) = 1 \text{ bit}$
- $GR(\text{vreme}) = 0.246/1.578=0.156$  ←
- $GR(\text{umiditate}) = 0.151/1.0 = 0.151$

# Cazuri speciale: A cu multe valori

- Utilizeaza diverse metode de invatare pentru a forma clase din aceste valori sau a grupa aceste valori in grupuri (cluster), clase care sa devina valori discrete (sau sa produca mai putine valori) pentru atribut

## 5. Random forests

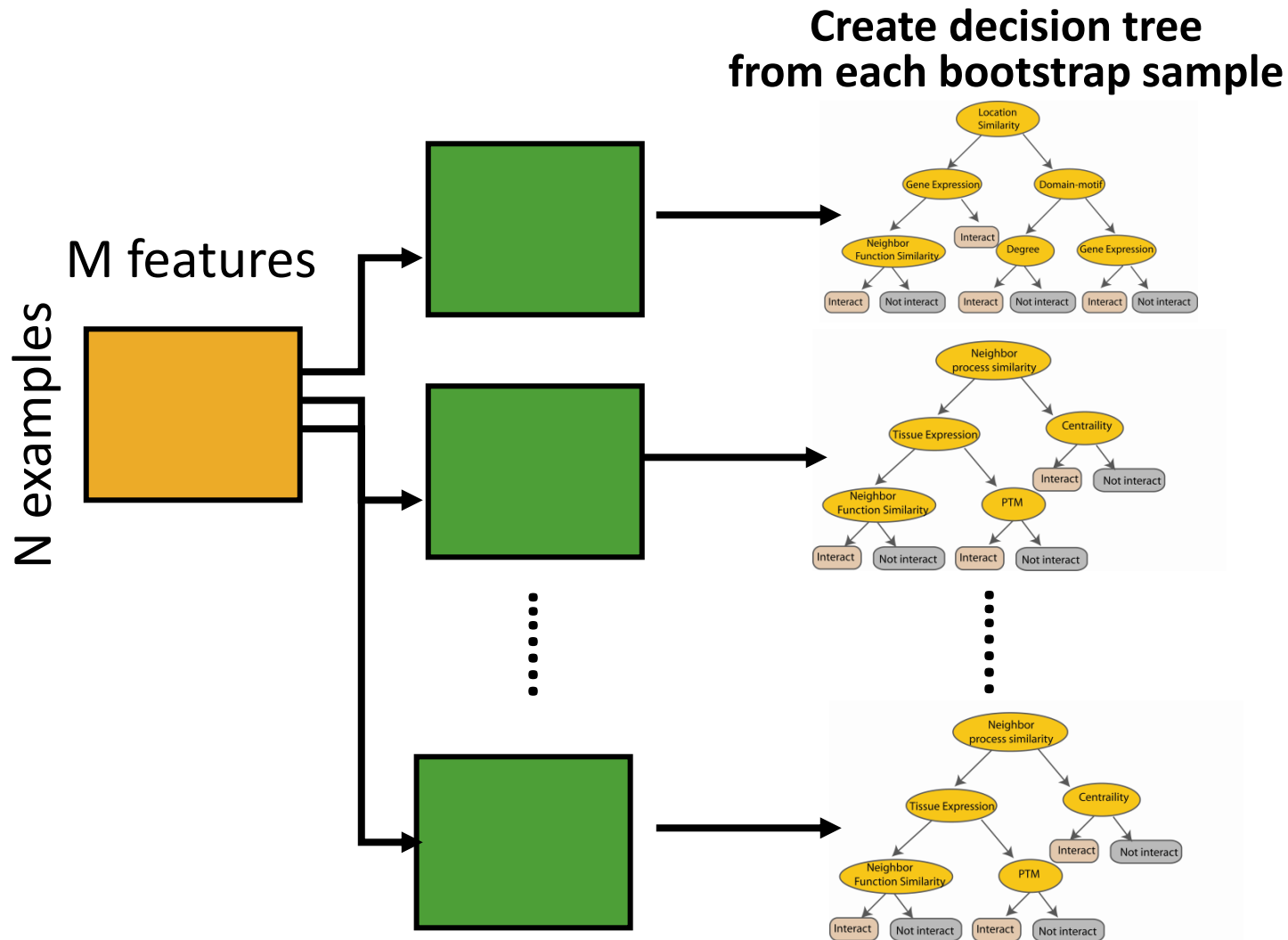
- Ipoteza metodei: combina mai multe modele de invatare a conceptului pentru a creste acurateta invatarii
- **Bagging** – obtine o medie a unor modele de invatare (fiecare poate fi afectat de zgomot sau de impartire favorizanta - biased)
- **Idee** - Selectam aleator din multimea de invatare o submultime de exemple si tot aleator selectam o submultime de attribute, apoi construim, pentru fiecare selectie cate un arbore (**random selection of attributes** and **random selection of examples**)
- Obtinem o **padure de arbori**

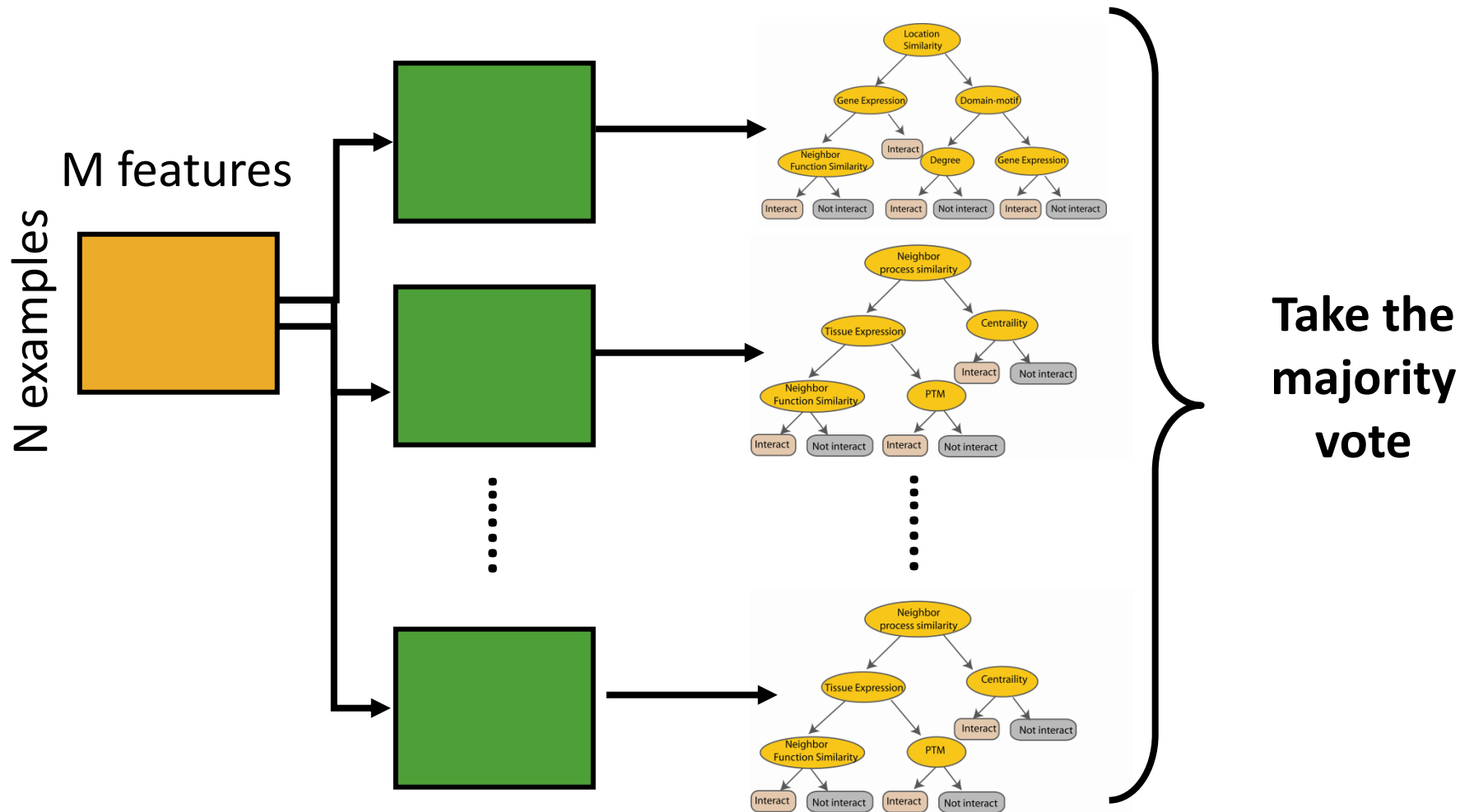


# Random forests

- $N$  – numărul de exemple de învățare,  $M$  numărul de atribute
- Se vor selecta în fiecare arbore  $m$  atribute ( $m < M$ ) și  $n$  ( $n < N$ ) exemple, aleator, din setul de învățare (bootstrap sample)
- Restul exemplurilor se folosesc pentru evaluare
- Fiecare arbore este complet construit

# Random forests





# Random forests

- Pentru o padure de arbori, si o instanta necunoscuta se clasifica instanta cu fiecare arbore
- Fiecare arbore aduce un vot, se considera clasificarea data de votul majoritar

## **Avantajele metodei**

- Acuratete mare de invatare
- Reduce influenta unui model favorizant (biased)
- Mai putin sensibil la zgomote
- Totusi are tendinta la overfitting pentru anumite seturi de date