

Curs 3-4:

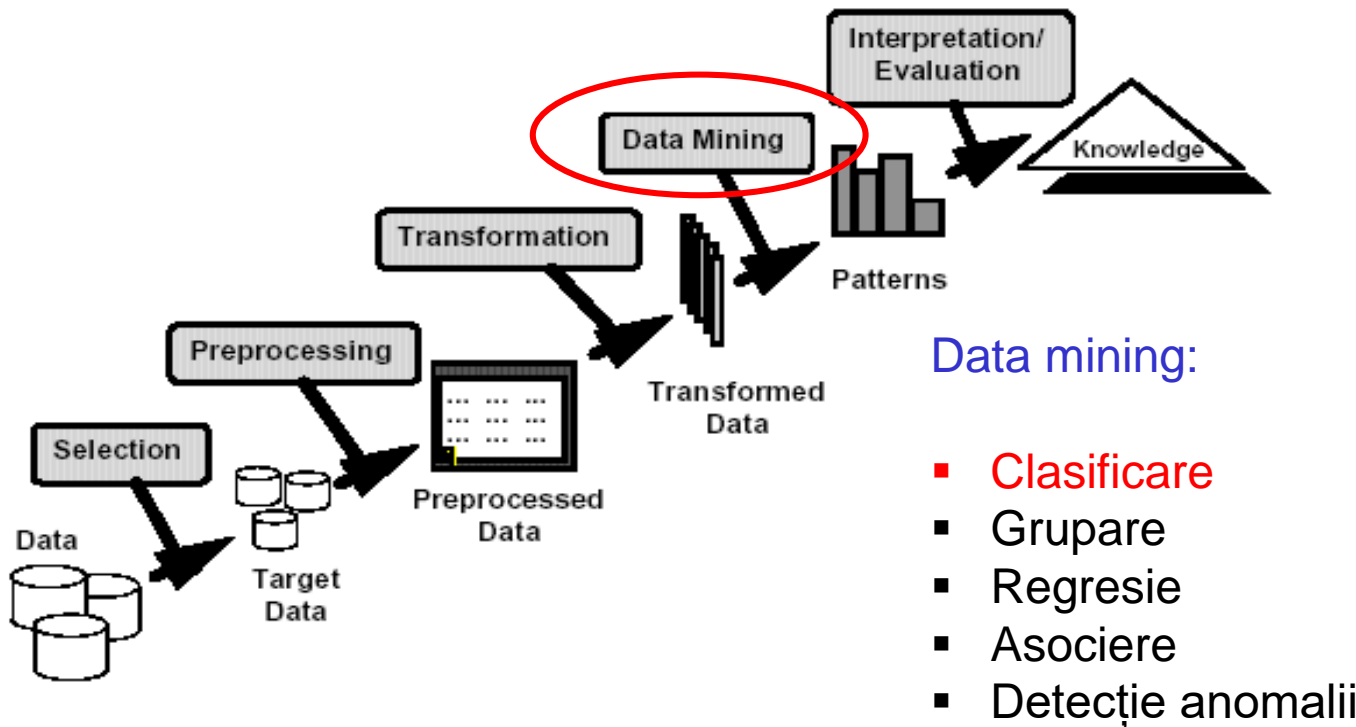
Clasificarea datelor – modele de clasificare (I)

Structura

- Motivare
- Concepte de bază în clasificare
- Măsuri de performanță
- Clasificatori
 - Bazați pe clasa majoritară (ZeroR)
 - Reguli simple de clasificare (OneR)
 - Construirea arborilor de decizie
 - Extragerea regulilor de clasificare
 - Clasificatori bazați pe instanțe (kNN)

Motivare

Reminder: procesul de descoperire a cunoștințelor



Motivare

Reminder: exemple de probleme de clasificare

- Clasificarea celulelor tumorale in benigne sau maligne (diagnoză medicală)
- Clasificarea tranzacțiilor efectuate cu cărți de credit ca fiind legitime sau frauduloase
- Clasificarea știrilor pe domenii: finanțe, meteo, divertisment, sport, etc (clasificare documente)
- Clasificarea e-mail-urilor ca spam sau utile (spam filtering)

Motivare

- **Diagnoza medicală** = prezicerea prezenței/absenței unei boli pe baza informațiilor disponibile într-o înregistrare medicală

Exemplu de set de date (breast-cancer-wisconsin - arff format –Lab 1)

```
@relation wisconsin-breast-cancer
@attribute Clump_Thickness integer [1,10]
@attribute Cell_Size_Uniformity integer [1,10]
@attribute Cell_Shape_Uniformity integer [1,10]
@attribute Marginal_Adhesion integer [1,10]
@attribute Single_Epi_Cell_Size integer [1,10]
@attribute Bare_Nuclei integer [1,10]
@attribute Bland_Chromatin integer [1,10]
@attribute Normal_Nucleoli integer [1,10]
@attribute Mitoses integer [1,10]
@attribute Class { benign, malignant}
@data
5,1,1,1,2,1,3,1,1,benign
5,4,4,5,7,10,3,2,1,benign
3,1,1,1,2,2,3,1,1,benign
8,10,10,8,7,10,9,7,1,malignant
1,1,1,1,2,10,3,1,1,benign
```

Concepte de bază

Ce se cunoaște?

- O colecție de înregistrări (instanțe) pentru care se cunoaște clasa căreia îi aparțin (**set de date etichetate**)
- Fiecare înregistrare conține un set de **atribute**, iar unul dintre aceste atribute este **eticheta clasei**

Ce se caută?

- un **model** care „captează” relația dintre atributul asociat clasei și celelalte atribute (modelul este construit folosind un **set de antrenare** printr-un proces numit **antrenare/învățare supervizată**)

Care este scopul final?

- Să se poată utiliza modelul construit prin antrenare pentru a determina clasa căreia îi aparține o nouă dată

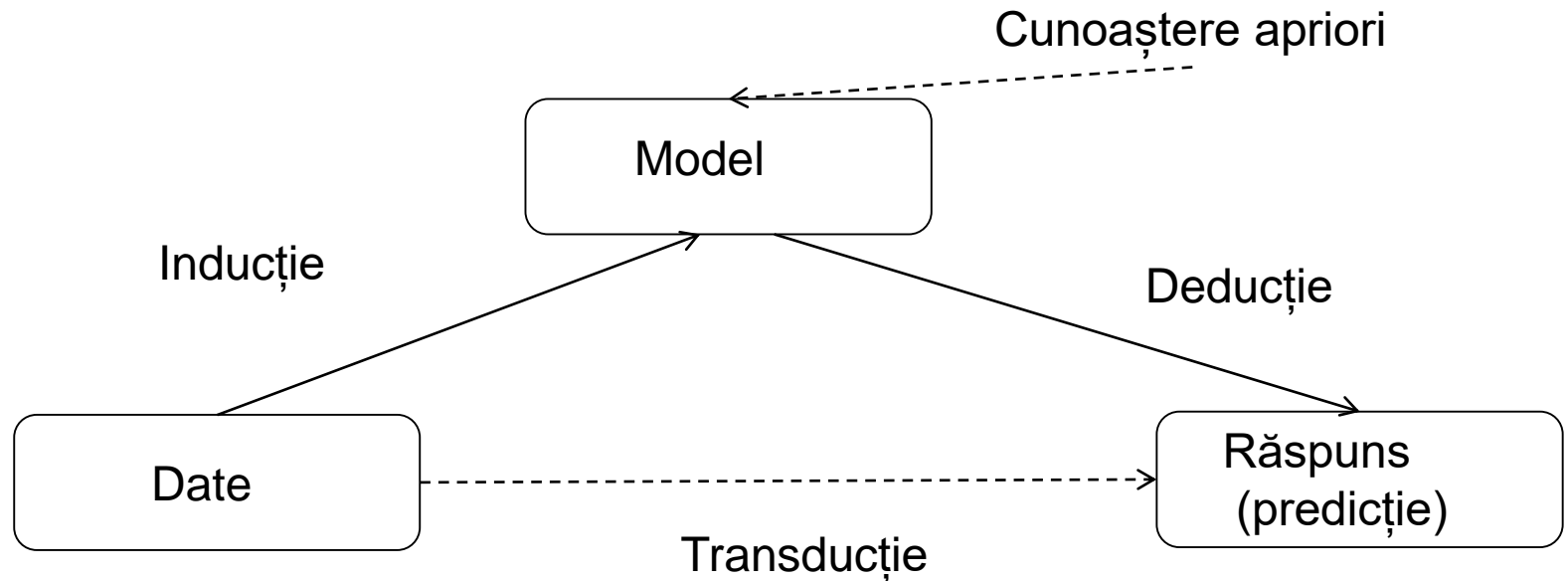
Obs:

- Pentru a fi util un model trebuie să aibă o bună **acuratețe**; acuratețea se măsoară analizând comportamentul modelului pentru date care nu au fost folosite în etapa de antrenare (**set de test**)

Concepte de bază

Invățare/ inducție/ inferență = construirea unui model pornind de la date (si eventual de la cunoștințe apriori privind domeniul)

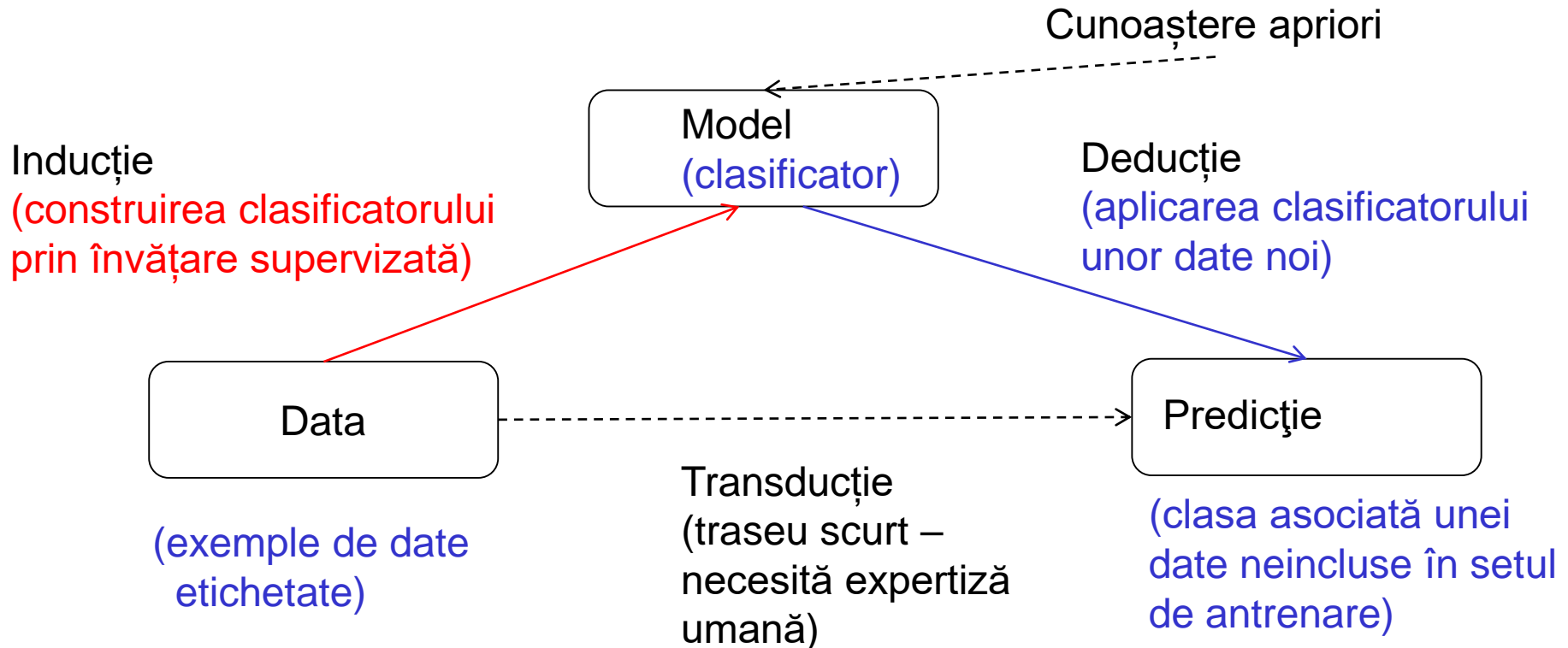
Între date, model, cunoștințe si răspunsuri există diferite legături:
inducție vs deducție vs transducție



Concepte de bază

Învățare/ inducție/ inferență = construirea unui model pornind de la date (și eventual de la cunoștințe apriori privind domeniul)

Între date, model, cunoștințe și răspunsuri există diferite legături:
inducție vs deducție vs transducție



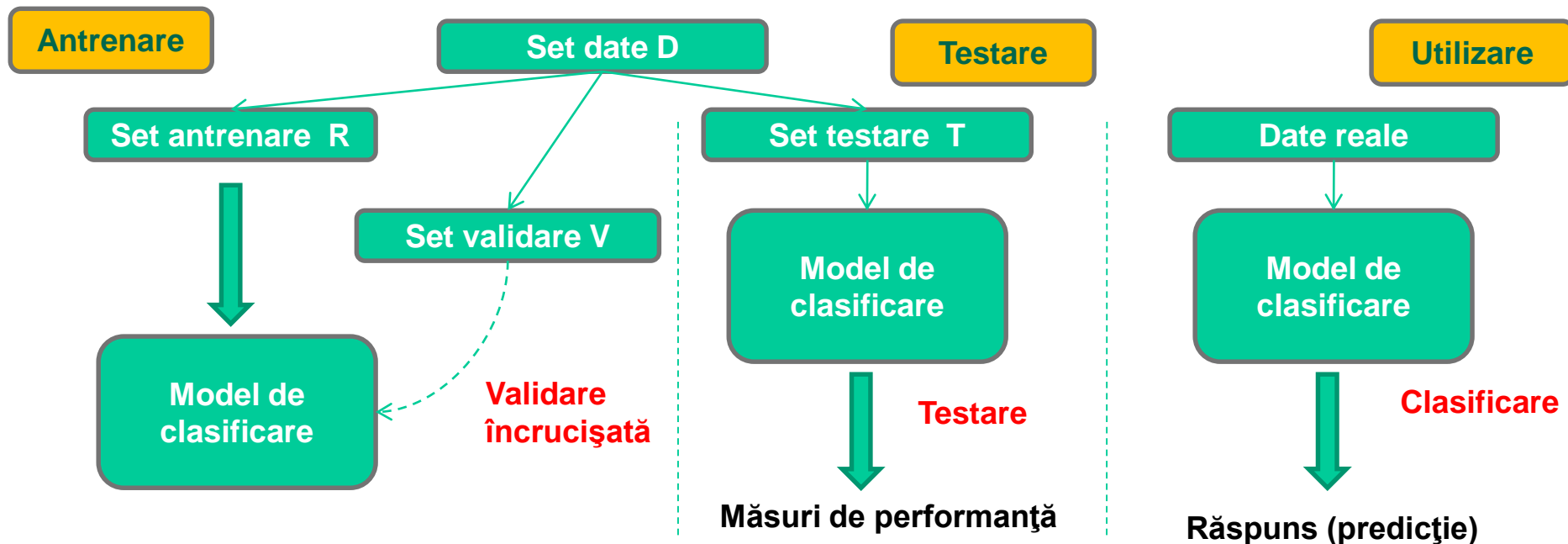
Concepte de bază

Informație disponibilă:

- Set de date etichetate:
 - $D = \{(x_1, c_1), (x_2, c_2), \dots, (x_N, c_N)\}$
 - fiecare x_i are n attribute
 - Eticheta clasei c_i aparține mulțimii $\{1, 2, \dots, K\}$

Scop:

- Construirea unui clasificator C folosind setul de date D a.î.
 - C poate prezice cărei clase îi aparține o nouă dată x



Modele de clasificare

Un model de clasificare este o “mapare” între valori ale atributelor și etichete ale claselor

Exemple de modele de clasificare:

- Arbori de decizie
- Reguli de clasificare
- Modele bazate pe prototipuri
- Modele probabiliste
- Rețele neuronale etc.

Un model de clasificare trebuie să fie:

- **Acurat:**
 - Identifică clasa corectă
- **Compact / comprehensibil**
 - Ușor de interpretat de către utilizator (preferabil să nu fie de tip “cutie neagră”)
- **Eficient în**
 - Etapa de antrenare
 - Etapa de clasificare

Modele de clasificare

Exemplu

```
@relation wisconsin-breast-cancer
@attribute Clump_Thickness integer [1,10]
@attribute Cell_Size_Uniformity integer [1,10]
@attribute Cell_Shape_Uniformity integer [1,10]
@attribute Marginal_Adhesion integer [1,10]
@attribute Single_Epi_Cell_Size integer [1,10]
@attribute Bare_Nuclei integer [1,10]
@attribute Bland_Chromatin integer [1,10]
@attribute Normal_Nucleoli integer [1,10]
@attribute Mitoses integer [1,10]
@attribute Class { benign, malignant}
@data
5,1,1,1,2,1,3,1,1,benign
5,4,4,5,7,10,3,2,1,benign
3,1,1,1,2,2,3,1,1,benign
8,10,10,8,7,10,9,7,1,malignant
1,1,1,1,2,10,3,1,1,benign
....
```

Regulă simplă de clasificare:
IF (Cell_Size_Uniformity < 3.5)
THEN benign
ELSE malignant

Modele de clasificare

Exemplu

```
@relation wisconsin-breast-cancer
@attribute Clump_Thickness integer [1,10]
@attribute Cell_Size_Uniformity integer [1,10]
@attribute Cell_Shape_Uniformity integer [1,10]
@attribute Marginal_Adhesion integer [1,10]
@attribute Single_Epi_Cell_Size integer [1,10]
@attribute Bare_Nuclei integer [1,10]
@attribute Bland_Chromatin integer [1,10]
@attribute Normal_Nucleoli integer [1,10]
@attribute Mitoses integer [1,10]
@attribute Class { benign, malignant}
@data
5,1,1,1,2,1,3,1,1,benign
5,4,4,5,7,10,3,2,1,benign
3,1,1,1,2,2,3,1,1,benign
8,10,10,8,7,10,9,7,1,malignant
1,1,1,1,2,10,3,1,1,benign
....
```

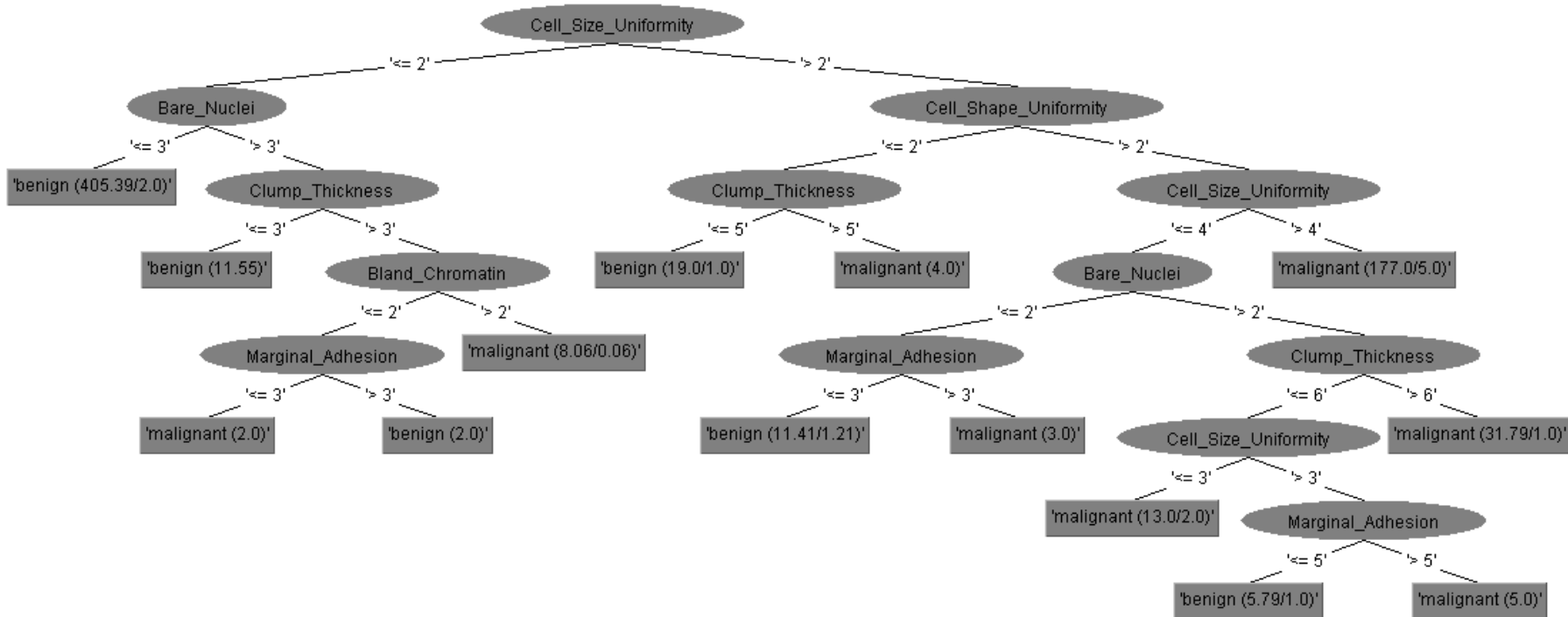
Regulă simplă de clasificare:
IF (Cell_Size_Uniformity < 3.5)
THEN benign
ELSE malignant

Întrebare: Cât de bună este
această regulă?

In **92.7%** din cazuri regula indică clasa corectă
Cum a fost calculată această valoare?
Cum ar trebui interpretată?

Modele de clasificare

Exemplu: un model mai complex (arbore de decizie)



Performanța: în 94.56% din cazuri clasificatorul indică clasa corectă

Ce se poate spune despre lizibilitatea clasificatorului?

Măsuri de performanță

Context: considerăm o problemă de clasificare în 2 clase

- Clasa 1 – pozitivă (ex: malign)
- Clasa 2 – negativă (ex: benign)

Cel mai simplu mod de a măsura performanța este de a analiza în câte cazuri clasificatorul indică răspunsul corect – această informație poate fi furnizată prin intermediul matricii de confuzie

Matrice de confuzie:

	C1	C2	← răspunsul clasificatorului
C1	TP	FN	
C2	FP	TN	



Clasa adevărată

TP = True Positive = nr de cazuri din C1 care sunt clasificate (corect) în C1

TN = True Negative = nr de cazuri din C2 care sunt clasificate (corect) în C2

FP = False Positive = nr de cazuri din C2 dar care sunt clasificate (incorect) în C1

FN = False Negative = nr de cazuri din C1 dar care sunt clasificate (incorect) în C2

Măsuri de performanță

Cazul a K clase:

- Se poate construi câte o matrice de confuzie 2x2 pt fiecare dintre clase (clasa curentă este considerată clasa pozitivă și toate celelalte clase sunt agregate în clasa negativă)
- Se extinde matricea la cazul a K clase: K linii și K coloane

Matrice de confuzie:

	C_1	C_2	...	C_j	...	C_K
C_1	T_1	F_{12}	...	F_{1j}	...	F_{1K}
C_2	F_{21}	T_2	...	F_{2j}	...	F_{2K}
...		
C_i	F_{i1}	F_{i2}		F_{ij}		F_{iK}
...		
C_K	F_{K1}	F_{K2}		F_{1Kj}		T_K

↑
Clasa adevărată

← răspunsul clasificatorului

F_{ij} = nr de cazuri care ar trebui clasificate în C_i dar sunt clasificate în C_j

Clasa adevărată

Măsuri de performanță

TP = True Positive = nr de cazuri din C1 care sunt clasificate (corect) în C1

TN = True Negative = nr de cazuri din C2 care sunt clasificate (corect) în C2

FP = False Positive = nr de cazuri din C2 dar care sunt clasificate (incorect) în C1

FN = False Negative = nr de cazuri din C1 dar care sunt clasificate (incorect) în C2

Acuratețe = $(TP+TN)/(TP+TN+FP+FN)$ = nr date clasificate corect/ nr total de date

Sensitivitate = $TP/(TP+FN)$ (TP rate sau **recall = rata de regăsire**)

Specificitate = $TN/(TN+FP)$ (TN rate), $1\text{-specificitate} = FP/(TN+FP) = \text{FP rate}$

Precizie = $TP/(TP+FP)$ (nr cazuri real pozitive/ nr cazuri clasificate ca fiind pozitive)

Obs:

- Toate valorile sunt în $[0,1]$; valori mai mari sugerează performanță mai bună
- **Sensitivitatea** și **specificitatea** sunt utilizare frecvent în analiza datelor medicale
- **Precizia** și **rata de regăsire** se folosesc în domeniul regăsirii informației (information retrieval)

Măsuri de performanță

TP = True Positive = nr de cazuri din C1 care sunt clasificate (corect) în C1

TN = True Negative = nr de cazuri din C2 care sunt clasificate (corect) în C2

FP = False Positive = nr de cazuri din C2 dar care sunt clasificate (incorect) în C1

FN = False Negative = nr de cazuri din C1 dar care sunt clasificate (incorect) în C2

In contextul regăsirii informației:

Precision = $TP / (TP + FP)$ = card(relevant și regăsit) / card(regăsit)

Recall = $TP / (TP + FN)$ = card(relevant și regăsit) / card(relevant)

O variantă agregată frecvent utilizată este **media armonică** a acestora:

F-measure = $2 * precision * recall / (precision + recall)$

Măsuri de performanță

TP = True Positive = nr de cazuri din C1 care sunt clasificate (corect) în C1

TN = True Negative = nr de cazuri din C2 care sunt clasificate (corect) în C2

FP = False Positive = nr de cazuri din C2 dar care sunt clasificate (incorect) în C1

FN = False Negative = nr de cazuri din C1 dar care sunt clasificate (incorect) în C2

Acuratețe ponderată de costuri (Cost sensitive accuracy)

- In anumite cazuri (ex: diagnoză medicală) clasificarea incorectă într-o clasă poate avea un impact mai mare decât clasificarea incorectă în altă clasă (e.g. nedetectarea unui caz de cancer poate fi mai periculoasă decât nedetectarea unui caz normal) - FN ar trebui să fie cât mai mic (senzitivitate mare)
- In alte cazuri (detectie atacuri informatice) ar trebui ca FP să fie cât mai mic

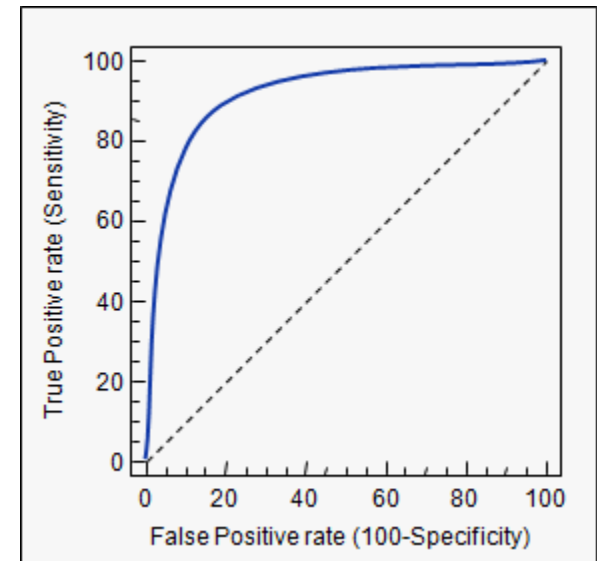
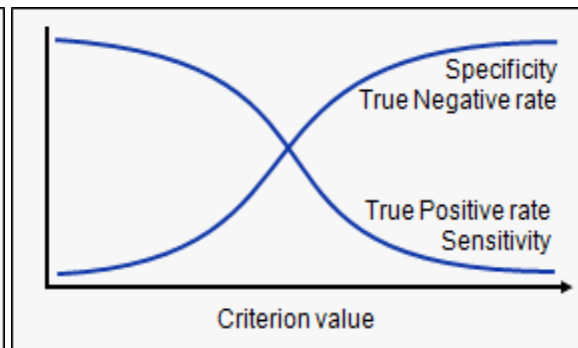
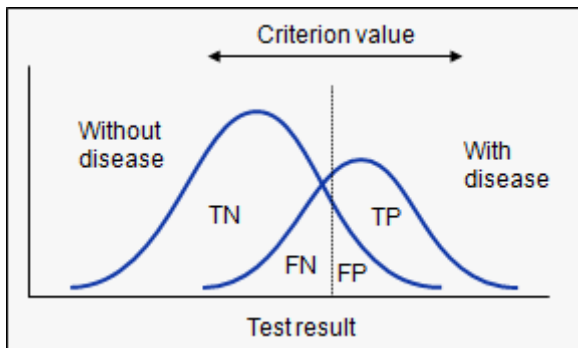
In astfel de situații se pot utiliza ponderi (interpretate ca și costuri ale erorii în clasificare) diferite pentru cele două tipuri de erori

- $CostAccuracy = (cost_1 * n_1 * sensitivity + cost_2 * n_2 * specificity) / (cost_1 * n_1 + cost_2 * n_2)$
 - $cost_i$ = costul clasificării incorecte a datelor din clasa C_i
 - n_i = numărul de date din C_i

Măsuri de performanță

Curba ROC (Receiver Operator Characteristics) si Aria de sub curbă (AUC)

- util pentru evaluarea performanței unui clasificator baza pe valoare prag
(ex: `IF (Cell_Size_Uniformity < 3.5) THEN benign ELSE malignant`)
- se reprezintă grafic punctele cu coordonatele
(FP rate, TP rate) = (senzitivitate, 1-specificitate)
pentru diferite valori ale pragului (sau pentru diferite subseturi de date
în cazul în care se folosește validare încrucișată)



Sursa: <https://www.medcalc.org/manual/roc-curves.php>

Cel mai simplu clasificator

Exemplu:

- Considerăm setul de date “sick” de la UCI Machine Learning
- Conține 3772 înregistrări (afereente unor pacienți), dintre care:
 - 231 sunt bolnavi (clasa C1 – pozitivă)
 - 3541 sunt sănătoși (clasa C2 – negativă)
- Ne interesează să construim un clasificator a cărui acuratețe pt acest set de date să fie cel puțin egală cu 0.9 (90%)
- Care este cel mai simplu clasificator care satisface această cerință?

Cel mai simplu clasificator

Exemplu:

- Considerăm setul de date “sick” de la UCI Machine Learning
- Conține 3772 înregistrări (afereente unor pacienți), dintre care:
 - 231 sunt bolnavi (clasa C1 – pozitivă) – 6% dintre pacienți
 - 3541 sunt sănătoși (clasa C2 – negativă) – 94% dintre pacienți
- Ne interesează să construim un clasificator a cărui acuratețe pt acest set de date să fie cel puțin egală cu 0.9 (90%)
- Care este cel mai simplu clasificator care satisface această cerință?
- Considerând regula: “indiferent de valorile atributelor clasa este C2 (negativă)” obținem $acurate\text{\u0219}ea = 3541/3772 = 0.94 > 0.9$
- Este un astfel de clasificator adecvat? Are vreo utilitate?

Cel mai simplu clasificator

- Este un astfel de clasificator adecvat? Are vreo utilitate?
- Acest clasificator, denumit ZeroR (întrucât se bazează pe o regulă de clasificare care nu conține nici un atribut în membrul stâng) utilizează doar distribuția datelor în cele două clase și va returna întotdeauna eticheta celei mai populare clase (se bazează pe un mecanism simplu de votare)
- **Nu este adecvat** întrucât produce răspuns incorect pt toate datele din clasa cu mai puține elemente
- Totuși ... poate fi utilizat pentru a determina o **margine inferioară** pt acuratețea unui clasificator: un clasificator cu acuratețe mai mică decât ZeroR ar trebui evitat

...din nou la evaluarea performanței

- Utilizarea întregului set de date disponibile pentru construirea clasificatorului nu este o abordare prea înțeleaptă întrucât poate conduce la **supra-antrenare**:
 - Clasificatorul se comportă bine pentru datele din setul de antrenare...
 - ... dar are performanțe slabe pentru alte date
 - O abordare mai bună este să se dividă setul de date în:
 - **Subset de antrenare** (utilizat pt construirea clasificatorului)
 - **Subset de testare** (utilizat pt estimarea performanței)
 - Există diferite strategii de divizare a setului de date în subseturi (antrenare și testare)
- Obs:** Pe lângă subsetul de testare se poate folosi și un subset de validare (utilizat pentru ajustarea parametrilor clasificatorului)

...din nou la evaluarea performanței

Strategii de divizare:

▪ Holdout

- Se reține 2/3 din set pt antrenare și 1/3 pt testare

▪ Holdout repetat

- Se repetă partiționarea (performanța este calculată ca medie a valorilor determinate la fiecare repetare a divizării)

▪ Validare încrucișată

- Se divide setul aleator în k subseturi disjuncte
- k -fold: se folosesc $k-1$ subseturi pt antrenare, iar al k -lea se folosește pt testare (evaluarea performanței)
- Leave-one-out: $k=n$

▪ Eșantionare repetată (util în cazul seturilor nebalansate)

- oversampling vs undersampling

▪ Bootstrap

- selecție cu revenire (se selectează un exemplu din set după care se pune la loc pentru a putea fi reselectat)

Dincolo de ZeroR

Set de date: [sick.arff](#), 29 attribute, 3772 instanțe (231 în clasa C1, 3541 în clasa C2), 2 clase

ZeroR (clasa e întotdeauna C2): **acuratețe=0.94**

OneR: permite construirea de reguli de clasificare care conțin un singur atribut în membrul stâng

Exemple de reguli (obținute folosind Weka OneR):

If $T3 < 0.25$ then C2 (negative)

If $T3$ in $[0.25, 0.35)$ then C1 (sick)

If $T3$ in $[0.35, 0.55)$ then C2 (negative)

If $T3$ in $[0.55, 1.15)$ then C1 (sick)

If $T3 \geq 1.15$ then C2 (negative)

If $T3$ value is missing then C2 (negative)

Acuaratețe: 0.96

OneR

Ideea principală: identifică atributul **cu cea mai mare putere de discriminare** și îl utilizează pentru a defini regulile de clasificare

Obs: este adecvat pentru attributele care au **valori discrete**

Algoritm:

FOR each attribute A_i do

FOR each value v_{ij} of A_i construct

R_{ij} : if $A_i = v_{ij}$ then class $C_{k(i,j)}$

(clasa majoritară pt instanțele care au $A_i = v_{ij}$)

se combină regulile într-un set R_i corespunzător lui A_i și se calculează

Err_i (nr date clasificate incorect)

ENDFOR

ENDFOR

Selectează setul de reguli cu eroarea cea mai mică

OneR

Exemplu: weather/play dataset

Outlook: err=4

sunny: 2 yes/ 3 no (→ no)

overcast: 4 yes/ 0 no (→ yes)

rainy: 3 yes/2 no (→yes)

Relation: weather.symbolic					
No.	outlook Nominal	temperature Nominal	humidity Nominal	windy Nominal	play Nominal
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	rainy	cool	normal	TRUE	no
7	overcast	cool	normal	TRUE	yes
8	sunny	mild	high	FALSE	no
9	sunny	cool	normal	FALSE	yes
10	rainy	mild	normal	FALSE	yes
11	sunny	mild	normal	TRUE	yes
12	overcast	mild	high	TRUE	yes
13	overcast	hot	normal	FALSE	yes
14	rainy	mild	high	TRUE	no

OneR

Example: weather/play dataset

Outlook: err=4

sunny: 2 yes/ 3 no (→ no)

overcast: 4 yes/ 0 no (→ yes)

rainy: 3 yes/2 no (→yes)

Temperature: err=5

hot: 2 yes/2 no (→ yes)

mild: 4 yes/2 no (→ yes)

cool: 3 yes/ 1no (→ yes)

Relation: weather.symbolic					
No.	outlook Nominal	temperature Nominal	humidity Nominal	windy Nominal	play Nominal
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	rainy	cool	normal	TRUE	no
7	overcast	cool	normal	TRUE	yes
8	sunny	mild	high	FALSE	no
9	sunny	cool	normal	FALSE	yes
10	rainy	mild	normal	FALSE	yes
11	sunny	mild	normal	TRUE	yes
12	overcast	mild	high	TRUE	yes
13	overcast	hot	normal	FALSE	yes
14	rainy	mild	high	TRUE	no

OneR

Example: weather/play dataset

Outlook: err=4

sunny: 2 yes/ 3 no (→ no)

overcast: 4 yes/ 0 no (→ yes)

rainy: 3 yes/ 2 no (→ yes)

Temperature: err=5

hot: 2 yes/ 2 no (→ yes)

mild: 4 yes/ 2 no (→ yes)

cool: 3 yes/ 1 no (→ yes)

Humidity: err=5

high: 4 yes/ 4 no (→ yes)

normal: 6 yes/ 1 no (→ yes)

Relation: weather.symbolic					
No.	outlook Nominal	temperature Nominal	humidity Nominal	windy Nominal	play Nominal
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	rainy	cool	normal	TRUE	no
7	overcast	cool	normal	TRUE	yes
8	sunny	mild	high	FALSE	no
9	sunny	cool	normal	FALSE	yes
10	rainy	mild	normal	FALSE	yes
11	sunny	mild	normal	TRUE	yes
12	overcast	mild	high	TRUE	yes
13	overcast	hot	normal	FALSE	yes
14	rainy	mild	high	TRUE	no

OneR

Exemplu: weather/play dataset

Outlook: err=4

sunny: 2 yes/ 3 no (→ no)

overcast: 4 yes/ 0 no (→ yes)

rainy: 3 yes/2 no (→yes)

Temperature: err=5

hot: 2 yes/2 no (→ yes)

mild: 4 yes/2 no (→ yes)

cool: 3 yes/ 1no (→ yes)

Humidity: err=5

high: 4 yes/ 4 no (→ yes)

normal: 6 yes/ 1 no (→ yes)

Windy: err=5

true: 3 yes/3 no (→ yes)

false: 6 yes/ 2 no (→ yes)

Relation: weather.symbolic

No.	outlook Nominal	temperature Nominal	humidity Nominal	windy Nominal	play Nominal
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	rainy	cool	normal	TRUE	no
7	overcast	cool	normal	TRUE	yes
8	sunny	mild	high	FALSE	no
9	sunny	cool	normal	FALSE	yes
10	rainy	mild	normal	FALSE	yes
11	sunny	mild	normal	TRUE	yes
12	overcast	mild	high	TRUE	yes
13	overcast	hot	normal	FALSE	yes
14	rainy	mild	high	TRUE	no

Reguli: weather/play dataset

If outlook=sunny then “no”

If outlook=overcast then “yes”

If outlook=rainy then “yes”

Acuratețe (set antrenare): 0.71

Acuratețe (validare încrucișată): 0.43 (!!)

OneR

Exemplu: weather/play dataset

Reguli: weather/play dataset

If outlook=sunny then “no”

If outlook=overcast then “yes”

If outlook=rainy then “yes”

Relation: weather.symbolic

No.	outlook Nominal	temperature Nominal	humidity Nominal	windy Nominal	play Nominal
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	rainy	cool	normal	TRUE	no
7	overcast	cool	normal	TRUE	yes
8	sunny	mild	high	FALSE	no
9	sunny	cool	normal	FALSE	yes
10	rainy	mild	normal	FALSE	yes
11	sunny	mild	normal	TRUE	yes
12	overcast	mild	high	TRUE	yes
13	overcast	hot	normal	FALSE	yes
14	rainy	mild	high	TRUE	no

Etapa de clasificare:

- Altă zi: (outlook=rainy, temperature=cool, humidity=high, windy=false)
- Răspuns: Yes

OneR

Sumar implementare OneR

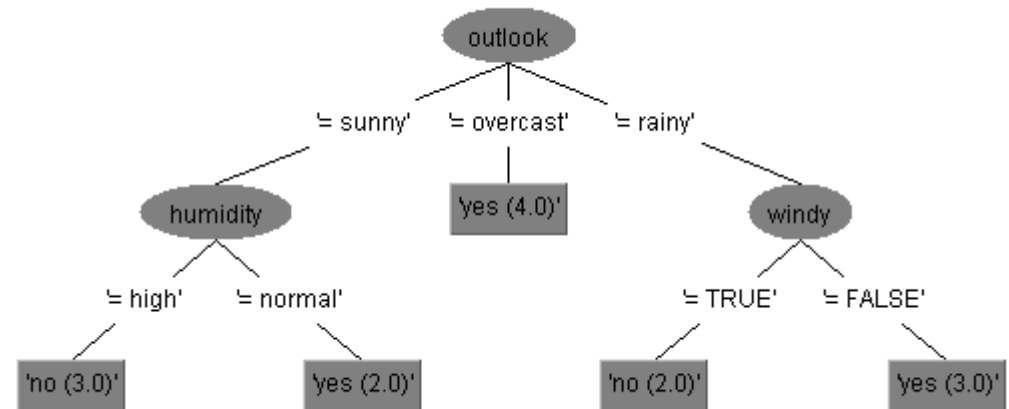
- **Construirea setului de reguli (etapa de antrenare)**
 - **Input:** set de antrenare (instanțe etichetate)
 - **Output:** set de reguli simple (toate regulile implică un singur atribut – același atribut în toate)
 - **Algoritm:** se analizează toate atributele și valorile corespunzătoare acestora și se selectează atributul pentru care eroarea de clasificare este minimă
- **Utilizarea regulilor (etapa de clasificare)**
 - **Input:** set de reguli, dată (instanță) nouă
 - **Output:** eticheta clasei
 - **Algoritm:**
 - Identifică regula care se potrivește cu data
 - Returnează clasa corespunzătoare regulii identificate

Arbori de decizie

Set de date: weather/play

Relation: weather.symbolic					
No.	outlook Nominal	temperature Nominal	humidity Nominal	windy Nominal	play Nominal
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	rainy	cool	normal	TRUE	no
7	overcast	cool	normal	TRUE	yes
8	sunny	mild	high	FALSE	no
9	sunny	cool	normal	FALSE	yes
10	rainy	mild	normal	FALSE	yes
11	sunny	mild	normal	TRUE	yes
12	overcast	mild	high	TRUE	yes
13	overcast	hot	normal	FALSE	yes
14	rainy	mild	high	TRUE	no

Arbore de decizie (construit folosind Weka)



Cum poate fi utilizat un arbore de decizie?

Ce clasă corespunde unei noi instanțe?

(outlook=sunny, temperature=mild, humidity=normal, windy=False)?

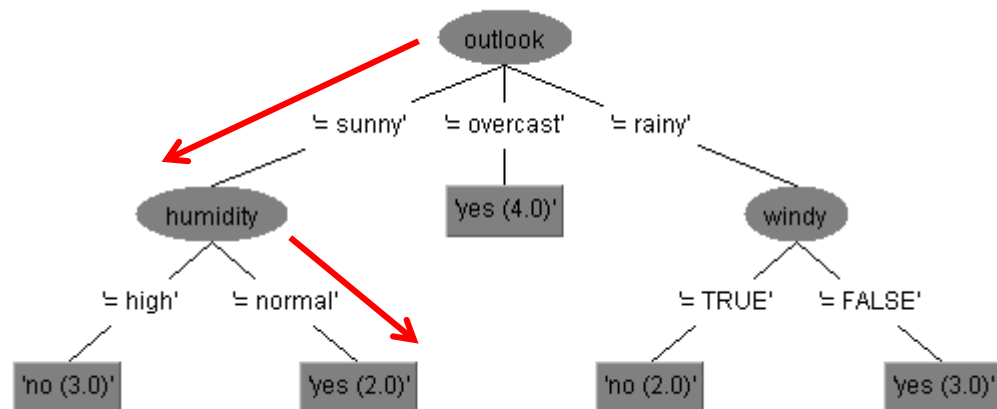
Arbori de decizie

Set de date: weather/play

Relation: weather.symbolic

No.	outlook Nominal	temperature Nominal	humidity Nominal	windy Nominal	play Nominal
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	rainy	cool	normal	TRUE	no
7	overcast	cool	normal	TRUE	yes
8	sunny	mild	high	FALSE	no
9	sunny	cool	normal	FALSE	yes
10	rainy	mild	normal	FALSE	yes
11	sunny	mild	normal	TRUE	yes
12	overcast	mild	high	TRUE	yes
13	overcast	hot	normal	FALSE	yes
14	rainy	mild	high	TRUE	no

Arbore de decizie (construit folosind Weka)



Cum poate fi utilizat un arbore de decizie?

Ce clasă corespunde unei noi instanțe?

(outlook=sunny, temperature=mild, humidity=normal, windy=False)?

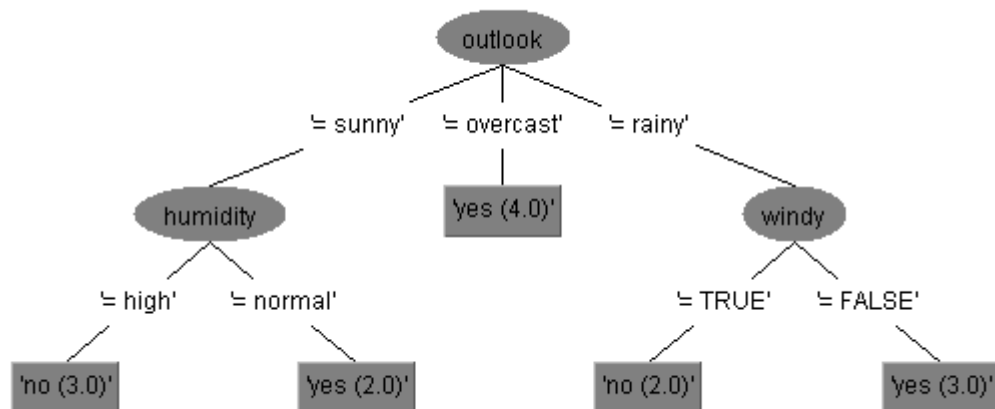
Clasa: Yes

Arbori de decizie

Set de date: weather/play

Relation: weather.symbolic					
No.	outlook Nominal	temperature Nominal	humidity Nominal	windy Nominal	play Nominal
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	rainy	cool	normal	TRUE	no
7	overcast	cool	normal	TRUE	yes
8	sunny	mild	high	FALSE	no
9	sunny	cool	normal	FALSE	yes
10	rainy	mild	normal	FALSE	yes
11	sunny	mild	normal	TRUE	yes
12	overcast	mild	high	TRUE	yes
13	overcast	hot	normal	FALSE	yes
14	rainy	mild	high	TRUE	no

Arbore de decizie (construit folosind Weka)



Cum poate fi tradus într-un set de reguli de clasificare? Fiecare ramură conduce la o regulă

- Regula 1:** IF outlook=sunny and humidity=high THEN play=no
- Regula 2:** IF outlook=sunny and humidity=normal THEN play=yes
- Regula 3:** IF outlook=overcast THEN play=yes
- Regula 4:** IF outlook=rainy and windy=True THEN play=no
- Regula 5:** IF outlook=rainy and windy=False THEN play=yes

Arbori de decizie

Cum poate fi construit un arbore de decizie pornind de la date?

- **Se alege un atribut** și se plasează în rădăcina arborelului
- Pt **fiecare valoare posibilă** a atributului (dintre cele prezente în setul de date) se **construiește o ramură**
- **Se partiționează setul de date** în subseturi corespunzătoare fiecărei ramuri
 - Dacă un subset conține date ce aparțin unei singure clase atunci el va corespunde unui nod frunză (nu se mai ramifică) – **nod pur**
 - Dacă subsetul conține date din mai multe clase atunci se continuă procesul de partiționare până când
 - se ajunge la un nod pur
 - pe ramura respectivă au fost deja analizate toate atributele
 - subsetul corespunzător ramurii este vid

Weather/play dataset

Relation: weather.symbolic					
No.	outlook Nominal	temperature Nominal	humidity Nominal	windy Nominal	play Nominal
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	rainy	cool	normal	TRUE	no
7	overcast	cool	normal	TRUE	yes
8	sunny	mild	high	FALSE	no
9	sunny	cool	normal	FALSE	yes
10	rainy	mild	normal	FALSE	yes
11	sunny	mild	normal	TRUE	yes
12	overcast	mild	high	TRUE	yes
13	overcast	hot	normal	FALSE	yes
14	rainy	mild	high	TRUE	no

Problemă: în ce ordine ar trebui analizate atributele? Ce condiție de testare ar trebui asociată cu fiecare nod?

Arbori de decizie

In ce ordine ar trebui analizate attributele?

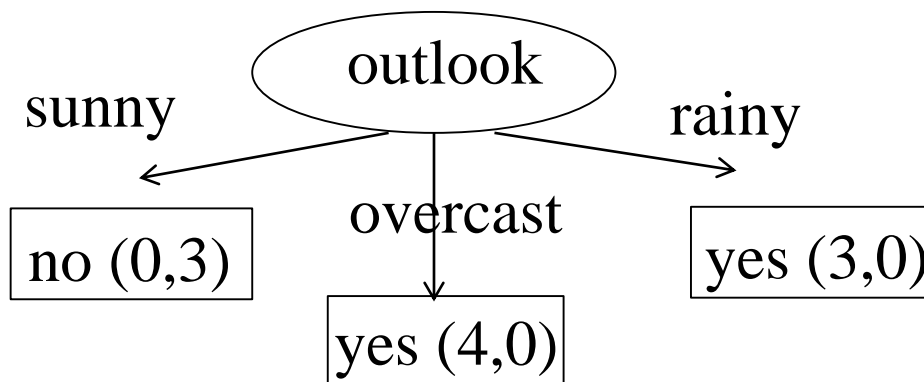
Set de date: weather/play (date selectate)

Ideea principală:

▪ Se selectează atributul care conduce la un arbore cât mai simplu adică un atribut cu grad de puritate cât mai mare (ideal ar fi ca pentru fiecare valoare a atributului datele corespunzătoare să aparțină aceleiași clase)

No.	outlook Nominal	temperature Nominal	humidity Nominal	windy Nominal	play Nominal
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	overcast	cool	normal	TRUE	yes
7	sunny	mild	high	FALSE	no
8	rainy	mild	normal	FALSE	yes
9	overcast	mild	high	TRUE	yes
10	overcast	hot	normal	FALSE	yes

Exemplu:



Obs:

- Toate nodurile frunză sunt “pure” (conțin date ce aparțin aceleiași clase)
- Conduce la reguli cu un singur atribut în membrul stâng
- O astfel de situație se întâmplă rar pentru date reale

Arbori de decizie

Principalele probleme ce trebuie soluționate la construirea unui arbore de decizie

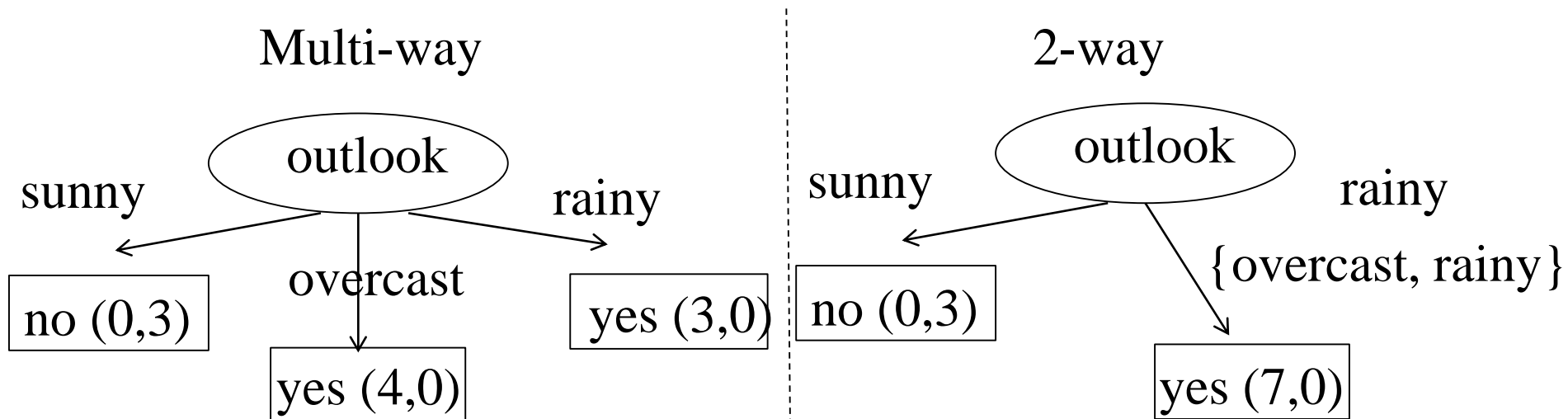
- **Ce condiții de test trebuie asignate ramurilor corespunzătoare unui nod ?**
 - Depinde de tipul atributului
 - Nominal, ordinal, continuu
 - Depinde de gradul de ramificare dorit:
 - Ramificare binară (setul curent de date este împărțit în două subseturi)
 - Ramificare multiplă (setul curent de date este împărțit în mai multe subseturi)
- **Ce atribut ar trebui selectat pentru partiționare?**
 - Cel cu puterea cea mai mare de discriminare – cel ce asigură partiționarea setului curent în subseturi cu **grad mare de puritate**
 - Criterii ce pot fi utilizate:
 - Bazate pe entropie (ex: câștig informațional)
 - Index Gini
 - Măsură a erorii de clasificare

Arbori de decizie

- Ce condiții de test trebuie asignate ramurilor corespunzătoare unui nod ?

Atribute nominale și ordinale:

- Ramificare multiplă (multi-way): atâtea ramuri câte valori posibile are atributul
- Ramificare binară (2-way): două ramuri



Arbori de decizie

Ce condiții de test trebuie asignate ramurilor corespunzătoare unui nod?

Atribute numerice:

- Trebuie discretizate în prealabil, după care se aplică strategia specifică atributelor nominale sau ordinale

Ce atribut se selectează pentru partiționare?

- Acel atribut care conduce la reducerea maximă în conținutul de informație necesar pentru a lua decizia corectă

Exemplu: information gain

Câștig informațional =
Entropia(distribuția datelor înainte de partiționare) –
EntropiaMedie (distribuția datelor după partiționare)

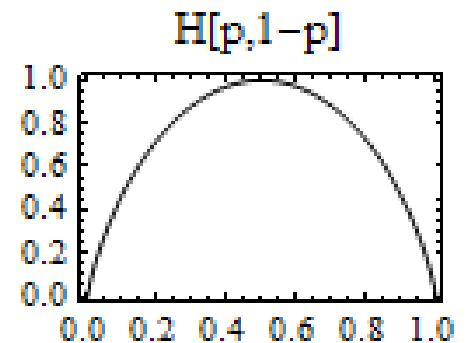
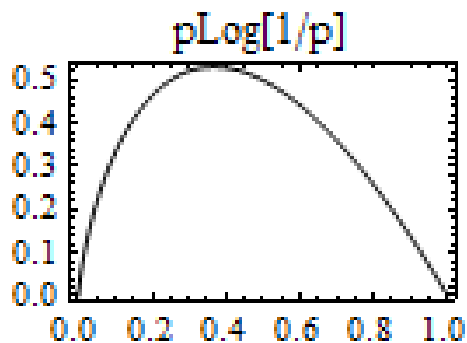
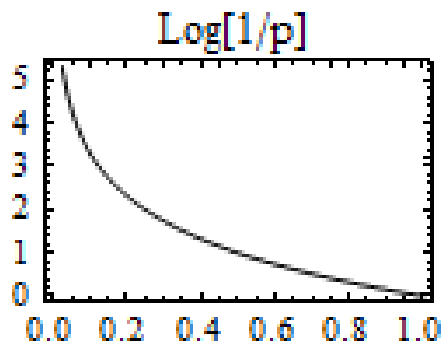
Reminder: entropie

Fie $D=(p_1, p_2, \dots, p_k)$ o distribuție de probabilitate. Entropia asociată acestei distribuții de probabilitate este caracterizată de:

$$H(D) = H(p_1, p_2, \dots, p_k) = -\sum_{i=1}^k p_i \log p_i$$

și poate fi interpretată ca o **măsură medie a incertitudinii** (sau surprizei) când se generează/selectează o valoare pe baza acestei distribuții

Caz particular: $k=2 \Rightarrow p_1=p, p_2=1-p$



Obs:

Interpretare $\text{Log}[1/p]$: surpriza de a observa un eveniment caracterizat de o probabilitate mică (eveniment neașteptat) este mai mare decât cea corespunzătoare unui eveniment de probabilitate mai mare (eveniment așteptat)

Reminder: entropie

In contextul rezolvării problemelor de clasificare:

- $D = \{C_1, C_2, \dots, C_k\}$ (set de date distribuit în k clase)
- Distribuția de probabilitate (p_1, p_2, \dots, p_k) , $p_i = \text{card}(C_i) / \text{card}(D)$
- Fie A un atribut și v_1, v_2, \dots, v_{m_A} valorile posibile ale acestui atribut
- Fie $D_j =$ setul de date din D pt care atributul A are valoarea v_j și P_j distribuția datelor din D_j în cele k clase ($C_{ji} =$ set de date din clasa C_i care au valoarea v_j pt atributul A)
- **Câștigul informațional** obținut prin partiționarea setului de date folosind atributul A este:

$$IG(D, A) = H(D) - \sum_{j=1}^{m_A} P(D_j | A = v_j) H(D_j | A = v_j), \quad H(D) = - \sum_{i=1}^k p_i \log p_i$$

$$H(D_j | A = v_j) = - \sum_{i=1}^k p_{ij} \log p_{ij}, \quad p_{ij} = \frac{\text{card}(C_{ji})}{\text{card}(C_i)}$$

$$P(D_j | A = v_j) = \frac{\text{card}(D_j)}{\text{card}(D)}$$

Alegerea atributului de ramificare

Exemplu

Relation: weather.symbolic					
No.	outlook Nominal	temperature Nominal	humidity Nominal	windy Nominal	play Nominal
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	rainy	cool	normal	TRUE	no
7	overcast	cool	normal	TRUE	yes
8	sunny	mild	high	FALSE	no
9	sunny	cool	normal	FALSE	yes
10	rainy	mild	normal	FALSE	yes
11	sunny	mild	normal	TRUE	yes
12	overcast	mild	high	TRUE	yes
13	overcast	hot	normal	FALSE	yes
14	rainy	mild	high	TRUE	no

- Distribuția claselor (C_1 ="yes", C_2 ="no"):
- $p_1=9/14$, $p_2=5/14$
- $H(p_1,p_2)=0.94$

Outlook

	C1 (yes)	C2(no)	Frequency
Sunny	2/5	3/5	5/14
Overcast	4/4	0/4	4/14
Rainy	3/5	2/5	5/14

$$H(\text{sunny}) = -2/5 \cdot \log(2/5) - 3/5 \cdot \log(3/5) = 0.97$$

$$H(\text{overcast}) = -1 \cdot \log(1) - 0 = 0$$

$$H(\text{rainy}) = -3/5 \cdot \log(3/5) - 2/5 \cdot \log(2/5) = 0.97$$

$$IG(\text{outlook}) = 0.94 - 5/14 \cdot 0.97 - 4/14 \cdot 0 - 5/14 \cdot 0.97 = 0.94 - 0.69 = 0.25$$

Alegerea atributului de ramificare

Exemplu

Relation: weather.symbolic					
No.	outlook Nominal	temperature Nominal	humidity Nominal	windy Nominal	play Nominal
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	rainy	cool	normal	TRUE	no
7	overcast	cool	normal	TRUE	yes
8	sunny	mild	high	FALSE	no
9	sunny	cool	normal	FALSE	yes
10	rainy	mild	normal	FALSE	yes
11	sunny	mild	normal	TRUE	yes
12	overcast	mild	high	TRUE	yes
13	overcast	hot	normal	FALSE	yes
14	rainy	mild	high	TRUE	no

- Distribuția claselor (C_1 ="yes", C_2 ="no"):
- $p_1=9/14$, $p_2=5/14$
- $H(p_1, p_2)=0.94$

Temperature

	C1 (yes)	C2(no)	Frequency
Hot	2/4	2/4	4/14
Mild	4/6	2/6	6/14
Cool	3/4	1/4	4/14

$$H(\text{hot}) = -2/4 \cdot \log(2/4) - 2/4 \cdot \log(2/4)$$

$$H(\text{mild}) = -4/6 \cdot \log(4/6) - 2/6 \cdot \log(2/6)$$

$$H(\text{cool}) = -3/4 \cdot \log(3/4) - 1/4 \cdot \log(1/4)$$

$$IG(\text{temperature}) = 0.03$$

Alegerea atributului de ramificare

Exemplu

Relation: weather.symbolic					
No.	outlook Nominal	temperature Nominal	humidity Nominal	windy Nominal	play Nominal
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	rainy	cool	normal	TRUE	no
7	overcast	cool	normal	TRUE	yes
8	sunny	mild	high	FALSE	no
9	sunny	cool	normal	FALSE	yes
10	rainy	mild	normal	FALSE	yes
11	sunny	mild	normal	TRUE	yes
12	overcast	mild	high	TRUE	yes
13	overcast	hot	normal	FALSE	yes
14	rainy	mild	high	TRUE	no

- Distribuția claselor (C_1 ="yes", C_2 ="no"):
- $p_1=9/14$, $p_2=5/14$
- $H(p_1,p_2)=0.94$

Humidity

	C1 (yes)	C2(no)	Frequency
High	3/7	4/7	7/14
Normal	6/7	1/7	7/14

$$H(\text{high})=-3/7*\log(3/7)-4/7*\log(4/7)$$

$$H(\text{normal})=-6/7*\log(6/7)-1/7*\log(1/7)$$

$$IG(\text{humidity})=0.15$$

Alegerea atributului de ramificare

Exemplu

Relation: weather.symbolic					
No.	outlook Nominal	temperature Nominal	humidity Nominal	windy Nominal	play Nominal
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	rainy	cool	normal	TRUE	no
7	overcast	cool	normal	TRUE	yes
8	sunny	mild	high	FALSE	no
9	sunny	cool	normal	FALSE	yes
10	rainy	mild	normal	FALSE	yes
11	sunny	mild	normal	TRUE	yes
12	overcast	mild	high	TRUE	yes
13	overcast	hot	normal	FALSE	yes
14	rainy	mild	high	TRUE	no

- Distribuția claselor (C_1 ="yes", C_2 ="no"):
- $p_1=9/14$, $p_2=5/14$
- $H(p_1, p_2)=0.94$

Windy

	C1 (yes)	C2(no)	Frequency
False	6/8	2/8	8/14
True	3/6	3/6	6/14

$$H(\text{false}) = -6/8 \cdot \log(6/8) - 2/8 \cdot \log(2/8)$$

$$H(\text{true}) = -3/6 \cdot \log(3/6) - 3/6 \cdot \log(3/6)$$

$$IG(\text{windy}) = 0.05$$

Alegerea atributului de ramificare

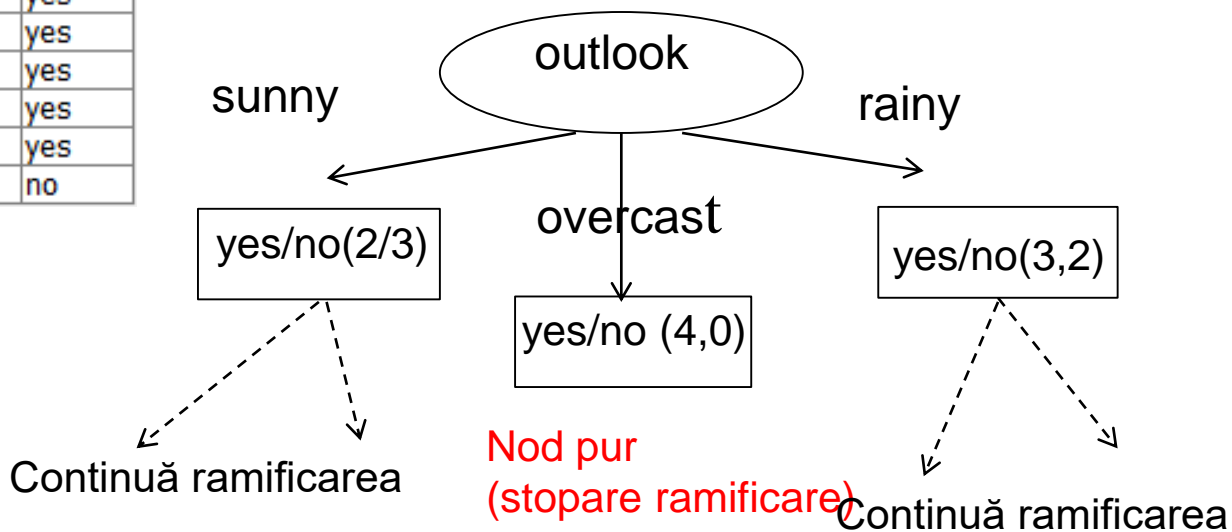
Exemplu

Relation: weather.symbolic					
No.	outlook Nominal	temperature Nominal	humidity Nominal	windy Nominal	play Nominal
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	rainy	cool	normal	TRUE	no
7	overcast	cool	normal	TRUE	yes
8	sunny	mild	high	FALSE	no
9	sunny	cool	normal	FALSE	yes
10	rainy	mild	normal	FALSE	yes
11	sunny	mild	normal	TRUE	yes
12	overcast	mild	high	TRUE	yes
13	overcast	hot	normal	FALSE	yes
14	rainy	mild	high	TRUE	no

Câștigul informațional al fiecărui atribut:

- $IG(\text{outlook})=0.25$
- $IG(\text{temperature})=0.03$
- $IG(\text{humidity})=0.15$
- $IG(\text{windy})=0.05$

Primul atribut selectat: outlook

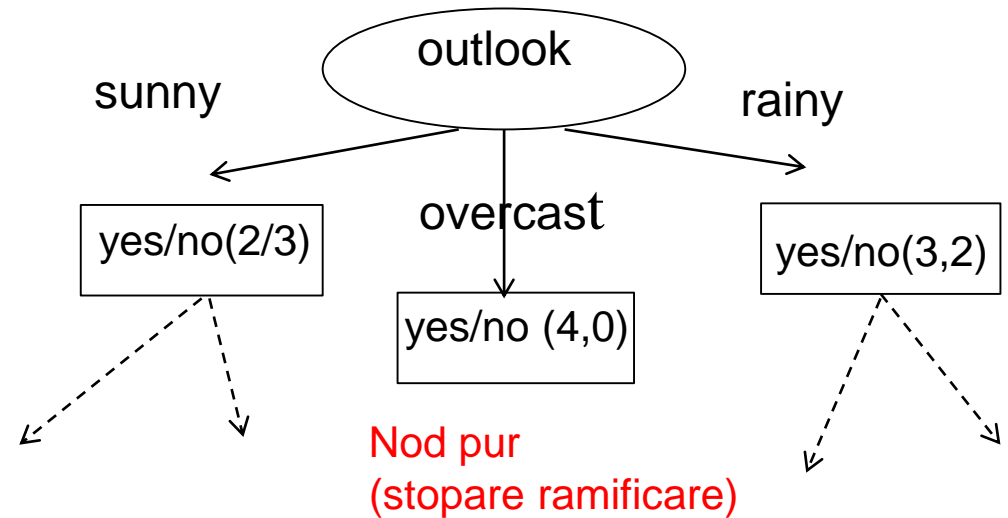


Alegerea atributului de ramificare

Exemplu

Relation: weather.symbolic

No.	outlook Nominal	temperature Nominal	humidity Nominal	windy Nominal	play Nominal
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	rainy	cool	normal	TRUE	no
7	overcast	cool	normal	TRUE	yes
8	sunny	mild	high	FALSE	no
9	sunny	cool	normal	FALSE	yes
10	rainy	mild	normal	FALSE	yes
11	sunny	mild	normal	TRUE	yes
12	overcast	mild	high	TRUE	yes
13	overcast	hot	normal	FALSE	yes
14	rainy	mild	high	TRUE	no



Nod pur
(stopare ramificare)

Temperature

	C1 (yes)	C2(no)	Freq.
Hot	0/2	2/2	2/5
Mild	1/2	1/2	2/5
Cool	1/1	0/1	1/5

Câștig informațional pt attributele rămase:

- Entropia pt subsetul "sunny" :
 $H(D(\text{sunny})) = -2/5 \cdot \log(2/5) - 3/5 \cdot \log(3/5) = 0.97$

- $H(\text{hot})=0$, $H(\text{mild})=1$, $H(\text{cool})=0$

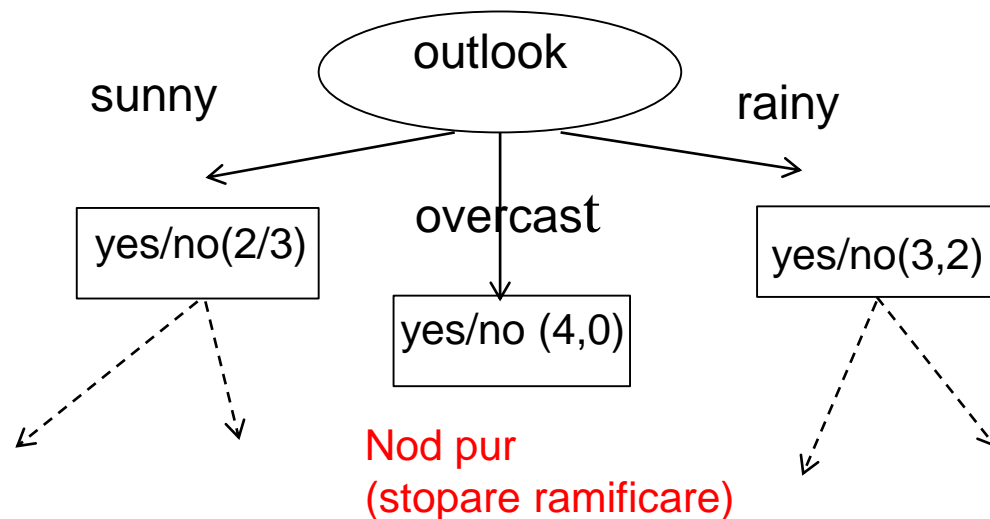
- $IG(\text{temperature}) = 0.97 - 2/5 = 0.57$

Alegerea atributului de ramificare

Exemplu

Relation: weather.symbolic

No.	outlook Nominal	temperature Nominal	humidity Nominal	windy Nominal	play Nominal
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	rainy	cool	normal	TRUE	no
7	overcast	cool	normal	TRUE	yes
8	sunny	mild	high	FALSE	no
9	sunny	cool	normal	FALSE	yes
10	rainy	mild	normal	FALSE	yes
11	sunny	mild	normal	TRUE	yes
12	overcast	mild	high	TRUE	yes
13	overcast	hot	normal	FALSE	yes
14	rainy	mild	high	TRUE	no



Humidity

	C1 (yes)	C2(no)	Freq.
High	0/3	3/3	3/5
Nor mal	2/2	0/2	2/5

Câștig informațional pt attributele rămase:

▪ Entropia pt subsetul "sunny" :

$$H(D(\text{sunny})) = -2/5 \cdot \log(2/5) - 3/5 \cdot \log(3/5) = 0.97$$

▪ $H(\text{high})=0$, $H(\text{normal})=0$

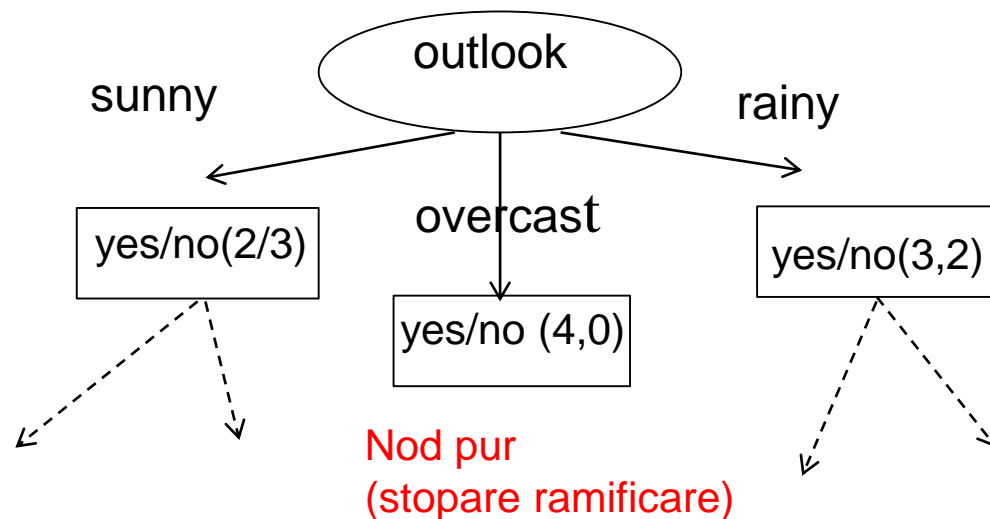
▪ $IG(\text{humidity})=0.97-0=0.97$

Alegerea atributului de ramificare

Exemplu

Relation: weather.symbolic

No.	outlook Nominal	temperature Nominal	humidity Nominal	windy Nominal	play Nominal
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	rainy	cool	normal	TRUE	no
7	overcast	cool	normal	TRUE	yes
8	sunny	mild	high	FALSE	no
9	sunny	cool	normal	FALSE	yes
10	rainy	mild	normal	FALSE	yes
11	sunny	mild	normal	TRUE	yes
12	overcast	mild	high	TRUE	yes
13	overcast	hot	normal	FALSE	yes
14	rainy	mild	high	TRUE	no



Windy

	C1 (yes)	C2(no)	Freq.
false	1/3	2/3	3/5
true	1/2	1/2	2/5

Câștig informațional pt attributele rămase:

▪ Entropia pt subsetul "sunny" :

$$H(D(\text{sunny})) = -2/5 \cdot \log(2/5) - 3/5 \cdot \log(3/5) = 0.97$$

▪ $H(\text{false})=0$, $H(\text{true})=1$

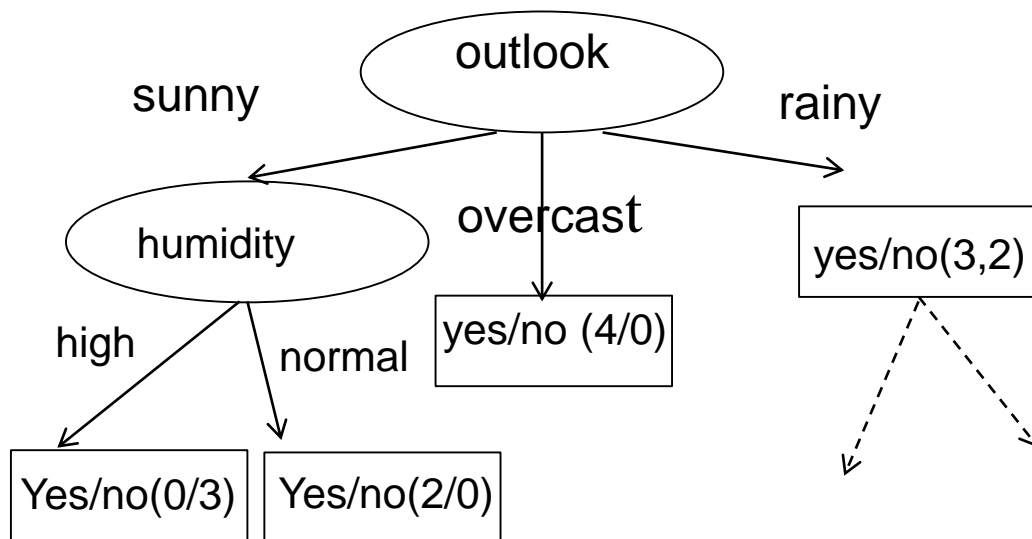
▪ $IG(\text{windy}) = 0.97 - 0.95 = 0.02$

Alegerea atributului de ramificare

Exemplu

Relation: weather.symbolic

No.	outlook Nominal	temperature Nominal	humidity Nominal	windy Nominal	play Nominal
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	rainy	cool	normal	TRUE	no
7	overcast	cool	normal	TRUE	yes
8	sunny	mild	high	FALSE	no
9	sunny	cool	normal	FALSE	yes
10	rainy	mild	normal	FALSE	yes
11	sunny	mild	normal	TRUE	yes
12	overcast	mild	high	TRUE	yes
13	overcast	hot	normal	FALSE	yes
14	rainy	mild	high	TRUE	no



Câștig informațional pt attributele rămase:

- $IG(\text{temperature}) = 0.97 - 2/5 = 0.57$
- $IG(\text{humidity}) = 0.97 - 0 = 0.97$
- $IG(\text{windy}) = 0.97 - 0.95 = 0.02$

Alegerea atributului de ramificare

Obs:

- Câștigul informațional favorizează attributele caracterizate printr-un număr mare de valori
- Pentru a limita această influență se poate utiliza raportul (Gain Ratio):

$$\text{GainRatio}(D, A) = \frac{IG(D, A)}{H(p_1^A, p_2^A, \dots, p_{m_A}^A)}$$

$$p_j^A = \frac{\text{card}(D, A = v_j)}{\text{card}(D)}$$

(proportia de date care au valoarea v_j pt atributul A)

- Atributul de ramificare poate fi determinat folosind indexul Gini = cât de frecvent un element ales aleator din set ar fi clasificat incorect dacă ar fi etichetat aleator pe baza distribuției corespunzătoare ramificării analizate (cu cât mai mică valoarea cu atât mai bună)

$$\text{Gini}(p_1, p_2, \dots, p_n) = 1 - \sum_{i=1}^n p_i^2$$

Algoritmi pentru construirea arborilor de decizie

ID3:

- Ințrare: set de date D
- Iesire: arbore de decizie (noduri interne etichetate cu atribute, noduri frunză etichetate cu clase, muchii etichetate cu valori ale atributelor)

```
DTinduction (D, DT, N) /* D=set date, DT=arbore de decizie, N=nod */
  find the best splitting attribute A
  label node N with A
  construct the splitting predicates (branches) for N
  FOR each branch i from N DO
    construct the corresponding data set  $D_i$ 
    create a new child node  $N_i$ 
    IF <stopping condition>
      THEN label  $N_i$  with the dominant class in  $D_i$  ( $N_i$  is a leaf node)
    ELSE DTinduction( $D_i$ ,DT,  $N_i$ )
```

Algoritmi pentru construirea arborilor de decizie

C4.5 = îmbunătățire a algoritmului ID3 pt a trata:

- **Atribute continue:**

- Incorporează procedură de discretizare

- **Valori absente:**

- Datele ce conțin valori absente sunt ignorate

sau

- Valorile absente sunt imputate

- **Atribut de ramificare:**

- Utilizează Gain Ratio pt selecția atributului

- **Simplificare sau fasonare (Pruning):**

- Anumiți subarbori sunt înlocuiți cu noduri frunză (dacă eroarea de clasificare nu crește semnificativ) – abordare bottom-up

Obs:

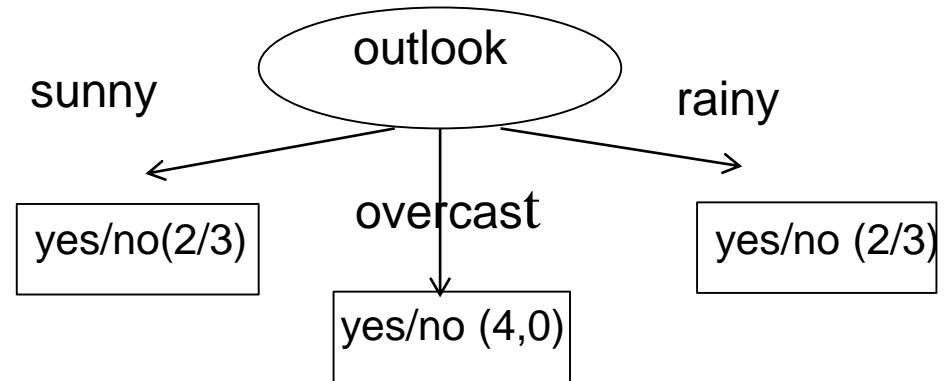
C5.0 – varianta comerciala a algoritmului C4.5 – accesibil pt analiza din 2011

J48 – implementarea din Weka a algoritmului C4.5

Algoritmi pentru construirea arborilor de decizie

Simplificare:

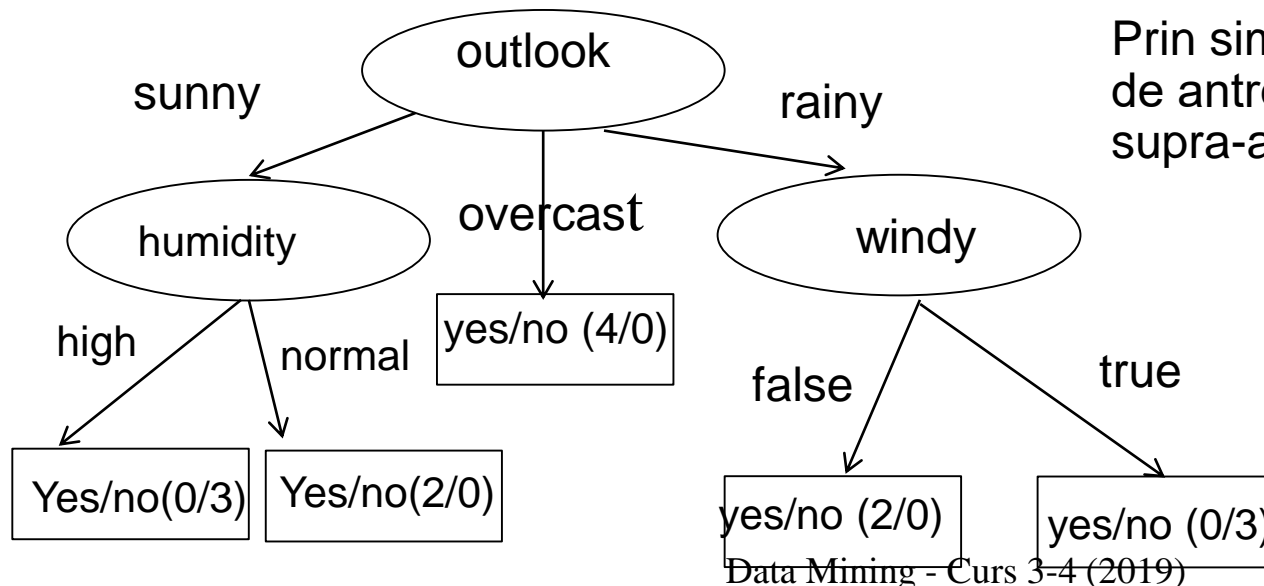
- Anumiți subarbori sunt înlocuiți cu noduri frunză (dacă eroarea de clasificare nu crește semnificativ) – abordare bottom-up



Arbore : error = 0

Arbore simplificat: error = 4/14

Prin simplificare eroarea pe setul de antrenare crește dar riscul de supra-antrenare poate să scadă



Extragerea regulilor de clasificare

Reminder: regulile de clasificare sunt structuri de tip IF ... THEN care conțin:

- In partea de **antecedent** (membrul stâng): condiții privind valorile atributelor (pot fi expresii logice care implică mai multe atribute)
- In partea de **consecință** (membrul drept): eticheta clasei

Exemple:

IF outlook=sunny THEN play=no

IF outlook=rainy THEN play=no

IF outlook=overcast THEN play=yes

Extragerea regulilor de clasificare

Obs:

- Dacă regulile sunt extrase dintr-un arbore de decizie atunci fiecare ramură conduce la o regulă
- Condițiile referitoare la noduri aflate pe aceeași ramură se combină prin AND:
IF (outlook=sunny) and (humidity=high) THEN play=no
- Regulile corespunzând unor ramuri diferite dar conducând la aceeași consecință (aceeași etichetă de clasă) pot fi reunite prin disjuncție (OR) între părțile de antecedent:

IF (outlook=sunny) OR (outlook=rainy) THEN play=no

Extragerea regulilor de clasificare

Alta variantă: Regulile de clasificare pot fi extrase direct din date **printr-un proces de învățare** utilizând algoritmi de acoperire (**covering algorithms**)

Noțiuni:

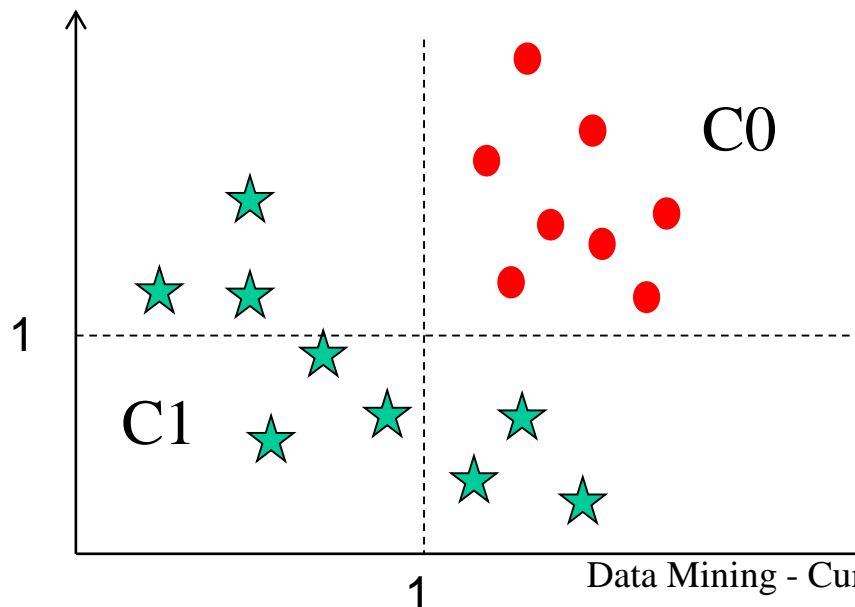
- O regulă **acoperă** o dată dacă valorile atributelor se potrivesc cu condițiile din antecedentul regulii
- Similar, despre o dată se spune că **activează** o regulă dacă valorile atributelor se potrivesc cu condițiile din antecedentul regulii
- **Suportul unei reguli (support)** = fracțiunea din setul total de date care este acoperită de către regulă și aparține aceleiași clase ca cea din regulă
$$= |\text{cover}(R) \cap \text{class}(R)| / |D|$$
- **Gradul de încredere în regulă (rule confidence)** = fracțiunea din datele acoperite de regulă care au aceeași clasă ca cea specificată de regulă =
$$|\text{cover}(R) \cap \text{class}(R)| / |\text{cover}(R)|$$
- **cover(R)** = subsetul de date acoperit de R
- **class(R)** = subsetul de date care au aceeași clasă cu cea specificată în R
- **D** = setul de date

Extragerea regulilor de clasificare

Noțiuni:

- **Reguli mutual exclusive** = regiunile acoperite de reguli sunt disjuncte (o instanță activează o singură regulă)
- **Set complet de reguli** = fiecare instanță activează cel puțin o regulă

Obs: dacă setul de reguli e complet și regulile sunt mutual exclusive atunci decizia privind apartenența unei date la o clasă este simplu de luat



Exemplu:

- R1:** IF $x > 1$ and $y > 1$ THEN C0
- R2:** IF $x \leq 1$ THEN C1
- R3:** IF $x > 1$ and $y \leq 1$ THEN C1

Ce se întâmplă însă dacă aceste proprietăți nu sunt satisfăcute?

Extragerea regulilor de clasificare

Obs: dacă regulile nu sunt mutual exclusive atunci pot să apară **conflicte** (o instanță poate activa reguli care au asociate clase diferite)

Conflictele pot fi rezolvate în unul dintre următoarele moduri:

- **Ordonarea regulilor** (pe baza unui criteriu) și decizia se ia conform primei reguli activate (prima regulă care se potrivește cu instanța).
- **Criteriul de ordonare** poate fi corelat cu:
 - **calitatea regulii** (e.g. Nivel încredere) – regulile cu nivel mare de încredere sunt mai bune
 - **specificitatea regulii** – o regulă este considerată mai bună dacă este mai specifică (e.g. Reguli care corespund claselor rare)
 - **complexitatea regulii** (e.g. Numărul de condiții din partea de antecedent a regulii) – regulile mai simple sunt mai bune

Extragerea regulilor de clasificare

Obs: aceste criterii pot fi conflictuale (o regulă cu coeficient mare de încredere nu este neapărat o regulă simplă)

- Rezultatul se obține considerând **clasa dominantă** pe baza tuturor regulilor activate de către instanță

Extragerea regulilor de clasificare

Algoritm secvențial de acoperire:

Intrare: set de date

Ieșire: set ordonat de reguli

Pas 1: se selectează una dintre clase și se identifică cea mai “bună” regulă care acoperă datele din D care au clasa selectată. Se adaugă regula la sfârșitul listei.

Pas 2: Se elimină datele din D care activează regula adăugată. Dacă încă există clase netratate și date în D go to Pas 1

Obs:

- Această este structura generală a algoritmilor secvențiali de acoperire
- Algoritmii pot să difere în funcție de strategia de selecție a claselor

Extragerea regulilor de clasificare

Exemplu: algoritmul RIPPER (Repeated Incremental Pruning to Produce Error Reduction)

Particularități:

- Setul de date e divizat la început în **growing set** (folosit pentru construirea unui set de reguli care acoperă setul de date) și **pruning set** (folosit pt simplificarea regulilor, de ex. prin eliminarea unor attribute din membrul stâng al regulii - se alege varianta de simplificare care reduce cel mai mult eroarea pe **pruning set**)
- Ordonare bazată pe clase: clasele sunt selectate crescător după dimensiune (clasele rare sunt selectate prima dată)
- Reguliile corespunzătoare unei clase sunt plasate consecutiv în lista de reguli

Extragerea regulilor de clasificare

Exemplu: algoritmul RIPPER (Repeated Incremental Pruning to Produce Error Reduction)

Particularități:

- Adăugarea unei noi reguli corespunzătoare unei clase este stopată când:
 - Când regula devine prea complexă
 - Când ‘următoarea’ regulă are o eroare de clasificare (pe setul de validare) mai mare decât un prag prestabilit
- Dacă la sfârșit rămân date “neacoperite” atunci se poate defini o regulă de tipul “**catch all**” careia i se asociază clasa dominantă din setul de date “neacoperite”

Clasificatori bazați pe instanțe

Ideea principală: datele similare aparțin aceleiași clase

- Modelul de clasificare constă tocmai din setul de antrenare
 - Procesul de antrenare constă doar în **stocarea datelor din set**
- Clasificarea unei noi date constă în:
 - Se calculează **similaritatea** (sau disimilaritatea) dintre noua dată și cele din setul de antrenare și se identifică **exemplarele cele mai apropiate**
 - Se **alege clasa cea mai frecvent întâlnită** în subsetul celor mai similare exemple

Clasificatori bazați pe instanțe

Ideea principală: datele similare aparțin aceleiași clase

Obs:

- Astfel de clasificatori sunt considerați leneși (“lazy”) deoarece faza de antrenare nu presupune nici un efort de calcul (întregul efort este amânat pentru faza de clasificare)
- Cei mai populari clasificatori din această categorie sunt cei bazați pe principiul celui/celor mai apropiat/apropiați vecin/vecini (k-Nearest Neighbour)

Clasificatori bazați pe instanțe

Ideea principală: datele similare aparțin aceleiași clase

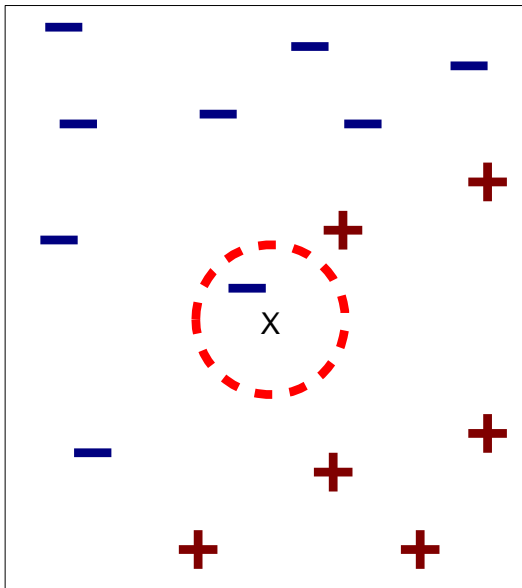
Obs:

- Astfel de clasificatori sunt considerați leneși (“lazy”) deoarece faza de antrenare nu presupune nici un efort de calcul (întregul efort este amânat pentru faza de clasificare)
- Cei mai populari clasificatori din această categorie sunt cei bazați pe principiul celui/celor mai apropiat/apropiați vecin/vecini (**k-Nearest Neighbour**)

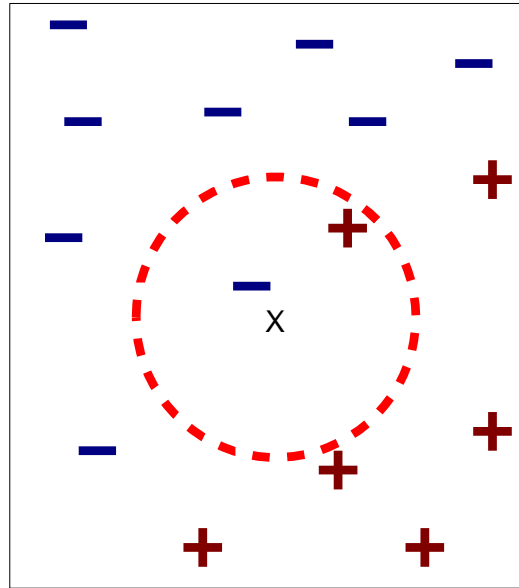
Clasificatori bazați pe instanțe

kNN – k Nearest Neighbour

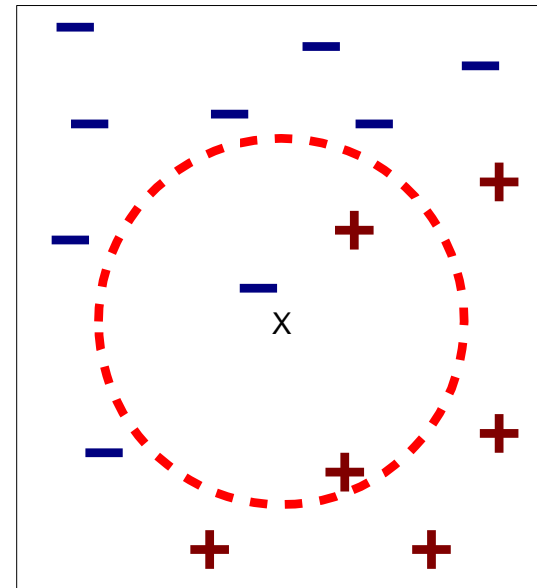
- Pt fiecare dată de clasificat:
 - Determină cele mai apropiate (mai similare) **k** exemple din setul de antrenare
 - Identifică cea mai frecventă clasă



(a) 1-nearest neighbor



(b) 2-nearest neighbor



(c) 3-nearest neighbor

Clasificatori bazați pe instanțe

kNN – k Nearest Neighbour

- Pt fiecare dată de clasificat:
 - Determină cele mai apropiate (mai similare) **k** exemple din setul de antrenare
 - Identifică cea mai frecventă clasă

Performanța clasificatorilor de tip kNN depinde de:

- Măsura de **similaritate/ disimilaritate**
 - Se alege în funcție de tipurile atributelor și de proprietățile problemei
- **Valoarea lui k (numărul de vecini)**
 - Cazul cel mai simplu: $k=1$ (nu e indicat în cazul datelor afectate de zgomot)

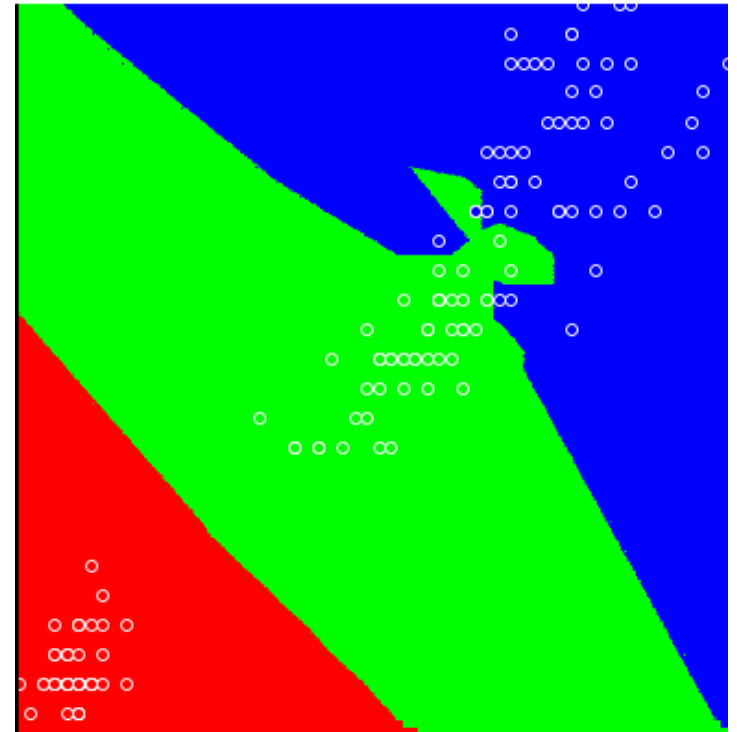
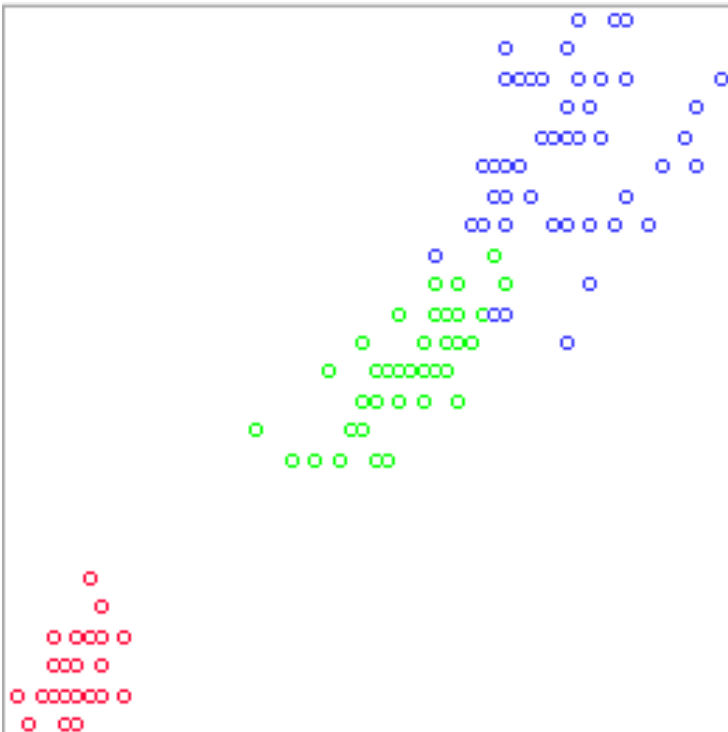
Obs: kNN induce o partiționare în regiuni a spațiului datelor; regiunile nu sunt calculate explicit ci sunt implicit determinate de măsura de similaritate (precum și de valoarea lui k)

Clasificatori bazați pe instanțe

1NN = Nearest Neighbor bazat pe cel mai apropiat vecin (și distanța euclidiană)

Ilustrarea regiunilor. Dataset: iris2D (“petal length” and “petal width”).

Plot: [Weka->Visualization->BoundaryVisualizer](#)



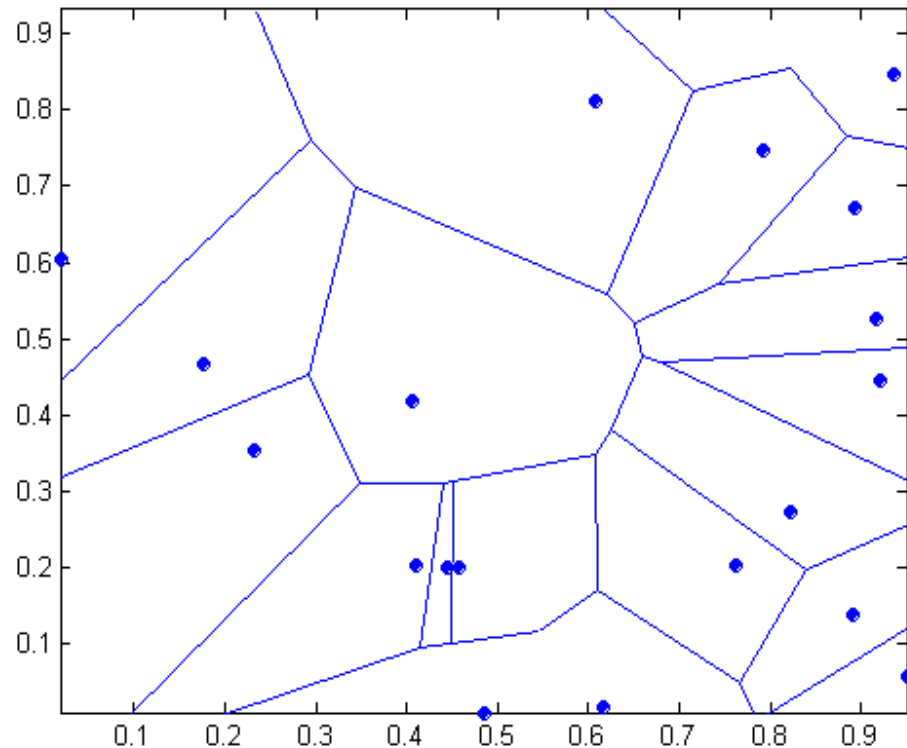
Clasificatori bazați pe instanțe

1NN = Nearest Neighbor bazat pe cel mai apropiat vecin (și distanța euclidiană)

1NN induce o partiționare a spațiului datelor (e.g. în 2D aceasta corespunde unei diagrame Voronoi)

Obs:

Fiecare instanță din setul de antrenare corespunde unei regiuni care cuprinde datele aflate în vecinătatea acelei instanțe



[Tan, Steinbach, Kumar; Introduction to Data Mining, slides, 2004]

Măsuri de similaritate/ disimilaritate

Considerăm două entități (e.g. vectori de date, serii de timp etc) A and B

- O măsură de **similaritate**, S , asociază perechii (A,B) un număr, $S(A,B)$, care este cu atât mai mare cu cât A și B sunt mai similare
- O măsură de **disimilaritate**, D , asociază perechii (A,B) un număr, $D(A,B)$, care este cu atât mai mare cu cât A și B sunt mai diferite

Alegerea măsurii depinde de:

- Tipul atributelor
- Numărul de attribute
- Distribuția datelor

Măsuri de similaritate/ disimilaritate

Atribute numerice

Cele mai populare măsuri de disimilaritate:

- Distanța euclidiană
- Distanța Manhattan

Obs:

- Distanța euclidiană este invariantă în raport cu rotații
- Dacă nu toate atributele au aceeași importanță atunci se folosește varianta ponderată ($w_i(a_i - b_i)^2$ în loc de $(a_i - b_i)^2$)

$$d_p(A, B) = \sqrt[p]{\sum_{i=1}^n (a_i - b_i)^p} \quad (\text{Minkowski, } L_p)$$

$$d_E(A, B) = \sqrt{\sum_{i=1}^n (a_i - b_i)^2} \quad (\text{Euclidean, } p = 2)$$

$$d_M(A, B) = \sum_{i=1}^n |a_i - b_i| \quad (\text{Manhattan, } p = 1)$$

$$d_\infty(A, B) = \max_{i=1, \dots, n} |a_i - b_i| \quad (p = \infty)$$

Ponderile se determină folosind tehnici de preprocesare

Măsuri de similaritate/ disimilaritate

Aspecte practice – problema dimensiunii (dimensionality curse):

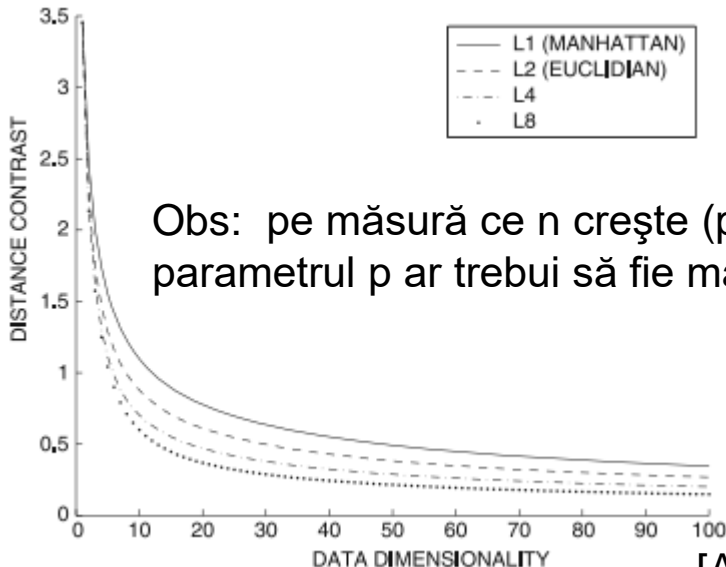
- Puterea de discriminare a acestor distanțe scade pe măsură ce nr de atribute (n) crește => pt date cu multe atribute clasificatorii bazați pe distanțe devin inefectivi

$$d_p(A, B) = \sqrt[p]{\sum_{i=1}^n (a_i - b_i)^p} \quad (\text{Minkowski, } L_p)$$

$$d_E(A, B) = \sqrt{\sum_{i=1}^n (a_i - b_i)^2} \quad (\text{Euclidean, } p = 2)$$

$$d_M(A, B) = \sum_{i=1}^n |a_i - b_i| \quad (\text{Manhattan, } p = 1)$$

$$d_\infty(A, B) = \max_{i=1, \dots, n} |a_i - b_i| \quad (p = \infty)$$



$$\text{Distance contrast: } \frac{d_{\max} - d_{\min}}{\sigma}$$

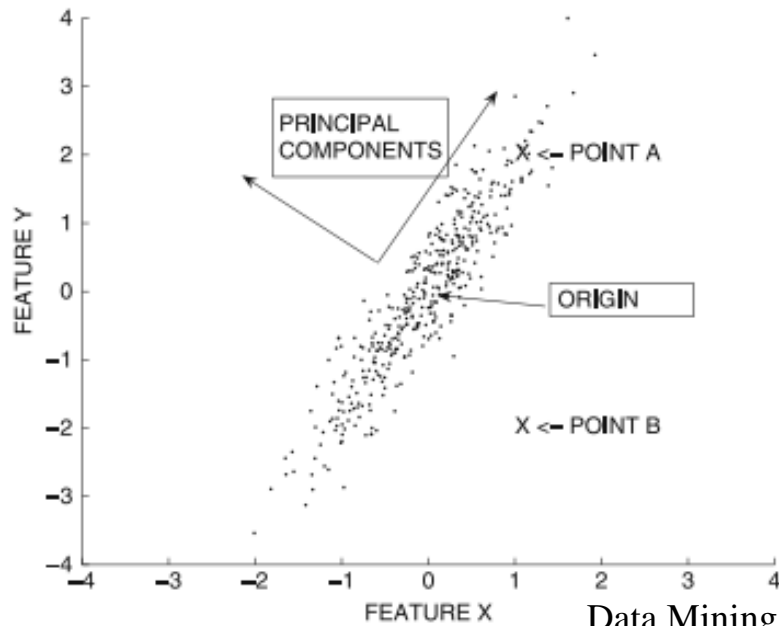
d_{\max}, d_{\min} = largest and smallest distance

σ = standard deviation of distances

Măsuri de similaritate/ disimilaritate

Aspecte practice – impactul distribuției datelor

Intrebare: Care punct e mai aproape de origine? A sau B?



Aggarwal, Data Mining Textbook, 2015

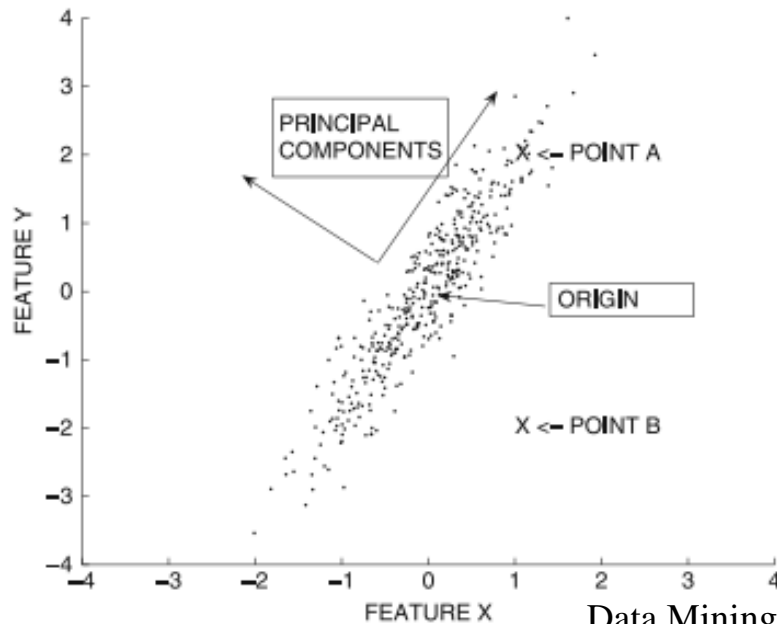
Măsuri de similaritate/ disimilaritate

Aspecte practice – impactul distribuției datelor

Intrebare: Care punct e mai aproape de origine? A sau B?

R: $d(O,A) = d(O,B)$ (distanțe euclidiene egale). Luând în considerare distribuția datelor: A este mai apropiat de O decât B

Altă întrebare: cum poate fi inclusă distribuția datelor în calculul distanței?



Distanța Mahalanobis

$$d_{Mah}(A, B) = \sqrt{(A - B)^T C^{-1} (A - B)}$$

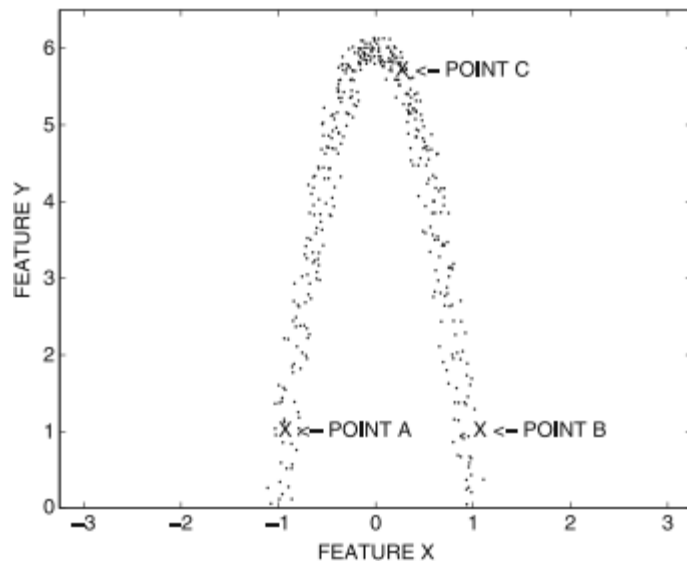
C^{-1} = inversa matricii de covarianța

Aggarwal, Data Mining Textbook, 2015

Măsuri de similaritate/ disimilaritate

Aspecte practice – impactul distribuției datelor

Intrebare: este distanța dintre A și B mai mică decât distanța dintre B și C?



Aggarwal, Data Mining Textbook, 2015

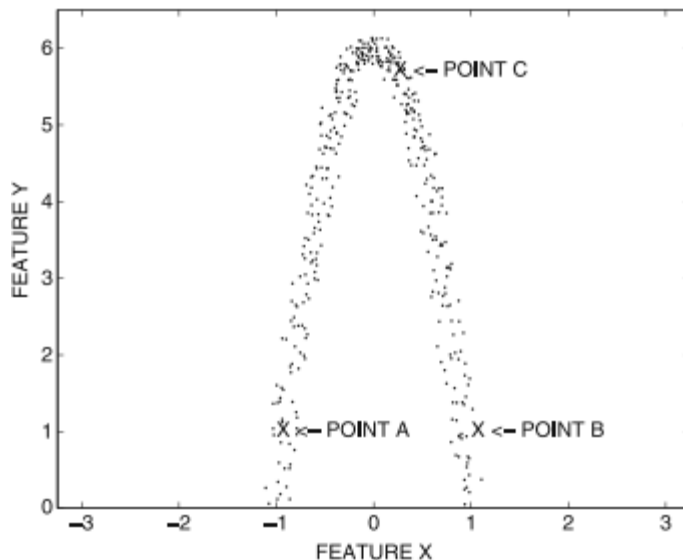
Măsuri de similaritate/ disimilaritate

Aspecte practice – impactul distribuției datelor

Intrebare: este distanța dintre A și B mai mică decât distanța dintre B și C?

R: da, dacă ignorăm distribuția datelor și folosim distanța euclidiană

Totuși, distribuția datelor nu poate fi ignorată întrucât este cea care furnizează contextul problemei, iar în acest context $d(A,B) > d(B,C)$



Distanța geodesică:

- Se construiește un graf ce are în noduri punctele iar muchiile unesc nodurile vecine (ex: cei mai apropiați k vecini)
- Calculează distanța dintre două puncte ca fiind cea mai scurtă cale în graf

Aggarwal, Data Mining Textbook, 2015

Măsuri de similaritate/ disimilaritate

Atribute numerice – măsură de similaritate

- Măsura cosinus: $\text{sim}(A,B)=A^T B / (\|A\| \|B\|)$ (produsul scalar dintre A și B împărțit la produsul normelor)

Remarcă:

- In cazul vectorilor normalizați ($\|A\|=\|B\|=1$) similaritatea e maximă când distanța euclidiană este minimă:

$$\begin{aligned}d_E^2(A, B) &= (A - B)^T (A - B) = A^T A - 2A^T B + B^T B = \\ &= 2(1 - A^T B) = 2(1 - \text{sim}(A, B))\end{aligned}$$

Măsuri de similaritate/ disimilaritate

Atribute nominale

Abordare 1: Transformarea atributelor nominale în atribute numerice (prin binarizare = one hot encoding) și utilizarea măsurilor de similaritate/disimilaritate pentru vectori binari:

- Disimilaritate: distanța **Hamming** = distanța Manhattan:
 $d_H(A,B)=d_M(A,B)$

Jaccard similarity:

$$J(A, B) = \frac{\sum_{i=1}^n a_i b_i}{\sum_{i=1}^n (a_i^2 + b_i^2 - a_i b_i)} = \frac{\text{card}(S_A \cap S_B)}{\text{card}(S_A \cup S_B)}$$

Obs: S_A și S_B sunt submulțimi ale mulțimii globale cu n atribute care corespund vectorilor de apartenență A și B .

Măsuri de similaritate/ disimilaritate

Atribute nominale

Abordare 2: Utilizează măsuri locale de similaritate (între valorile atributelor)

$$S(A, B) = \sum_{i=1}^n S(a_i, b_i)$$
$$S(a_i, b_i) = \begin{cases} 1 & \text{if } a_i = b_i \\ 0 & \text{if } a_i \neq b_i \end{cases}$$

Obs: similaritățile mai puțin frecvente pot fi considerate mai relevante decât cele frecvente

$$S(a_i, b_i) = \begin{cases} 1/f^2(a_i) & \text{if } a_i = b_i \\ 0 & \text{if } a_i \neq b_i \end{cases}$$

$f(a_i)$ = frecvența valorii a_i în setul de date
(pt atributul i)

Măsuri de similaritate/ disimilaritate

Atribute mixte: se combină măsurile corespunzătoare celor două tipuri de atribute (utilizând ponderi specifice)

$$S(A, B) = \lambda S_{numerical}(A, B) + (1 - \lambda) S_{nominal}(A, B)$$

Alte tipuri de date:

- **Siruri** (e.g. text sau secvențe biologice) – se utilizează distanța de editare
- **Concepte** (e.g. noduri într-o ontologie) – distanțe bazate pe cele mai scurte căi în grafuri sau arbori
- **Grafuri** (e.g. Rețele sociale sau biologice) – ponderea structurilor (tiparelor) similare în cele două structuri

kNN: alegerea lui k

Performanța clasificatorilor de tip kNN depinde de numărul de vecini

Cazuri extreme:

- $k=1$ - clasificatorul nu este robust (erorile din setul de date influențează răspunsul clasificatorului)
- $k=N$ - e echivalent cu ZeroR fiind bazat doar pe modul de distribuire a datelor în clase

Cum se alege k?

- Abordare de tip trial-and-error: se încearcă diferite valori și se alege valoarea care maximizează performanța

kNN: cost

Clasificarea unei noi instanțe necesită calculul a N distanțe (sau măsuri de similaritate pt un set de date cu N elemente precum și selecția celor mai mici k distanțe $\rightarrow O(N+kN)$)

Dacă N e mare această prelucrare poate fi costisitoare (întrucât trebuie efectuată pentru fiecare instanță care trebuie clasificată)

Abordări posibile:

- Crearea structuri de indexare a datelor din setul de antrenare care permite identificarea celor mai apropiați k vecini într-un mod eficient
- Reducerea numărului de date din setul de antrenare prin gruparea lor în clustere și înlocuirea fiecărui cluster cu un singur prototip
- Selecția unor prototipuri din set

Curs următor

- Modele probabiliste (Naive Bayes classifiers)
- Rețele neuronale (Multilayer Perceptrons)
- Clasificatori bazați pe vectori suport (Support Vector Machines)