# Curs 3-4:

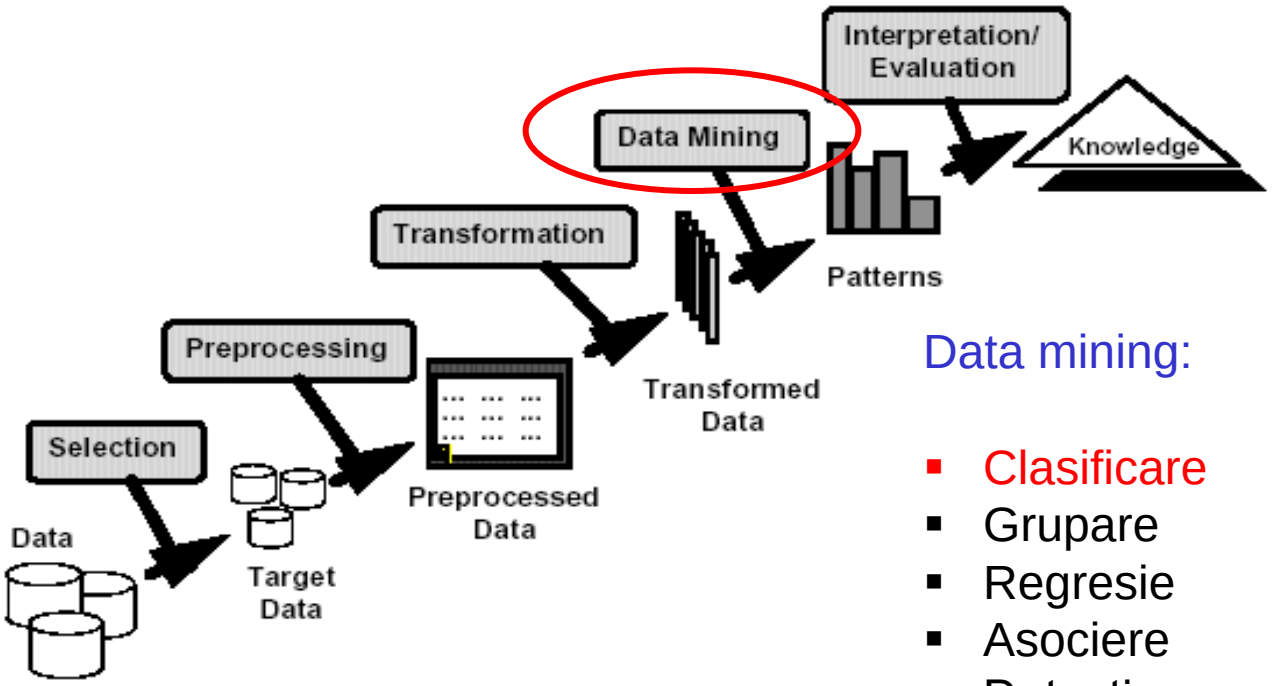
# Clasificarea datelor (I)

# Structura

- Motivare
- Concepte de bază în clasificare
- Măsuri de performanţă
- 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




Data mining:

- Clasificare
- Grupare
- Regresie
- Asociere
- Detecție anomalii

# Motivare

Reminder:  exemple de probleme de clasificare

- Clasificarea celulelor tumorale in benigne sau maligne (diagnoză medicală)
- Clasificarea tranzacţiilor 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 inregistrare 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

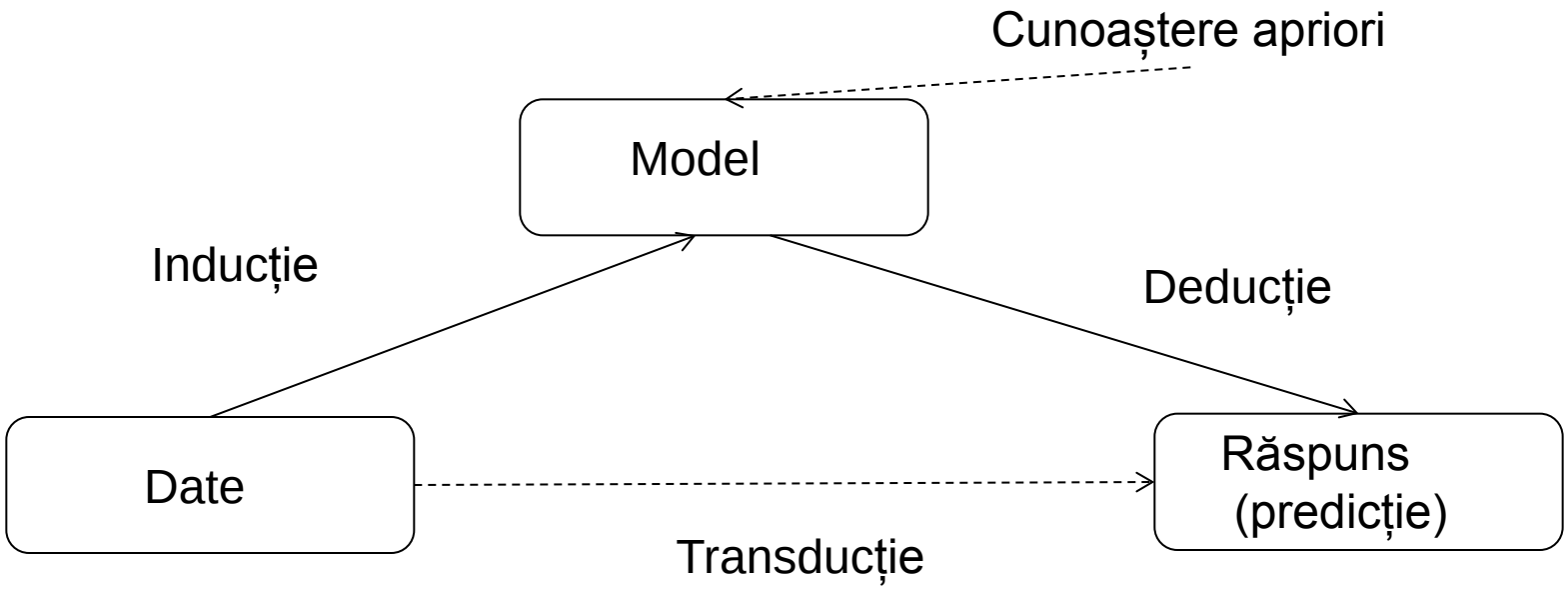
Obs:

- 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)

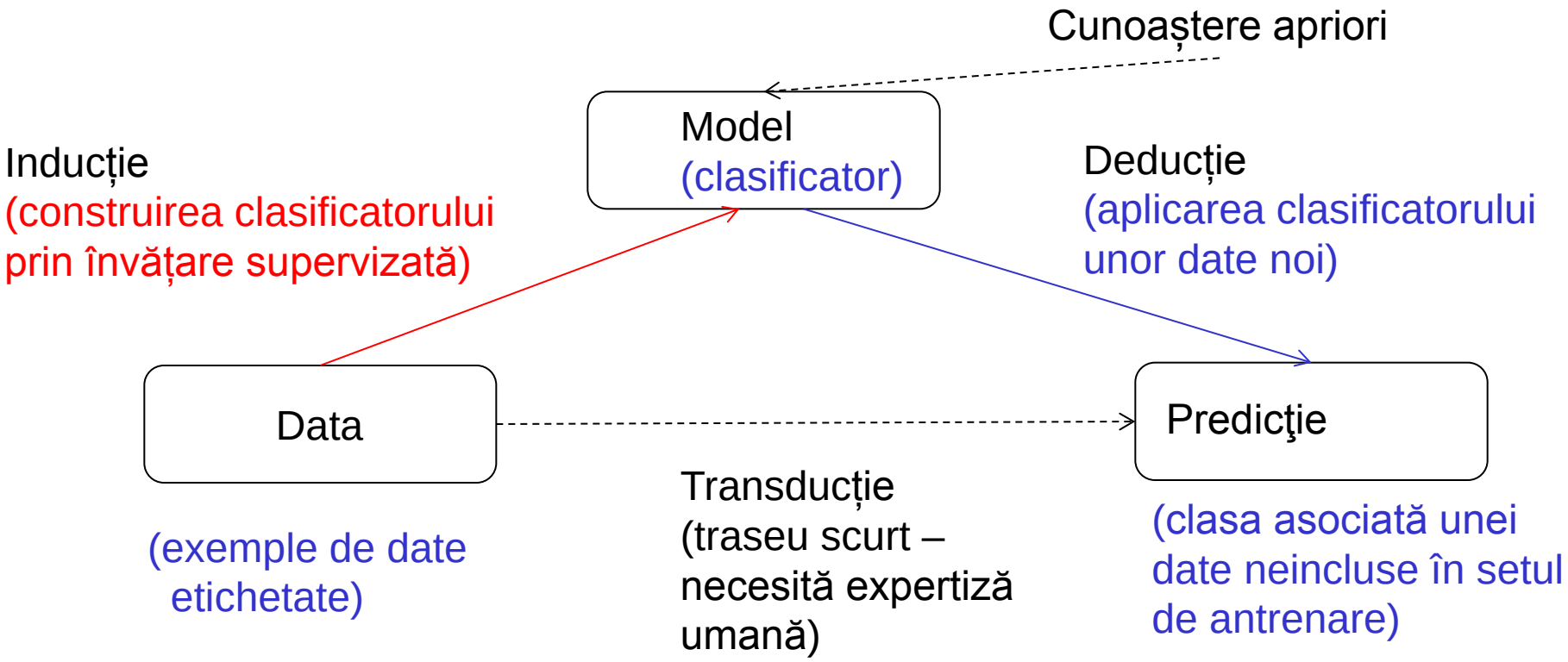
Intre date, model, cunostințe si răspunsuri există diferite legături:
inducție vs deducţie vs transducţie



# Concepte de bază

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

Intre date, model, cunostințe si răspunsuri există diferite legături:
inducție vs deductie vs transductie



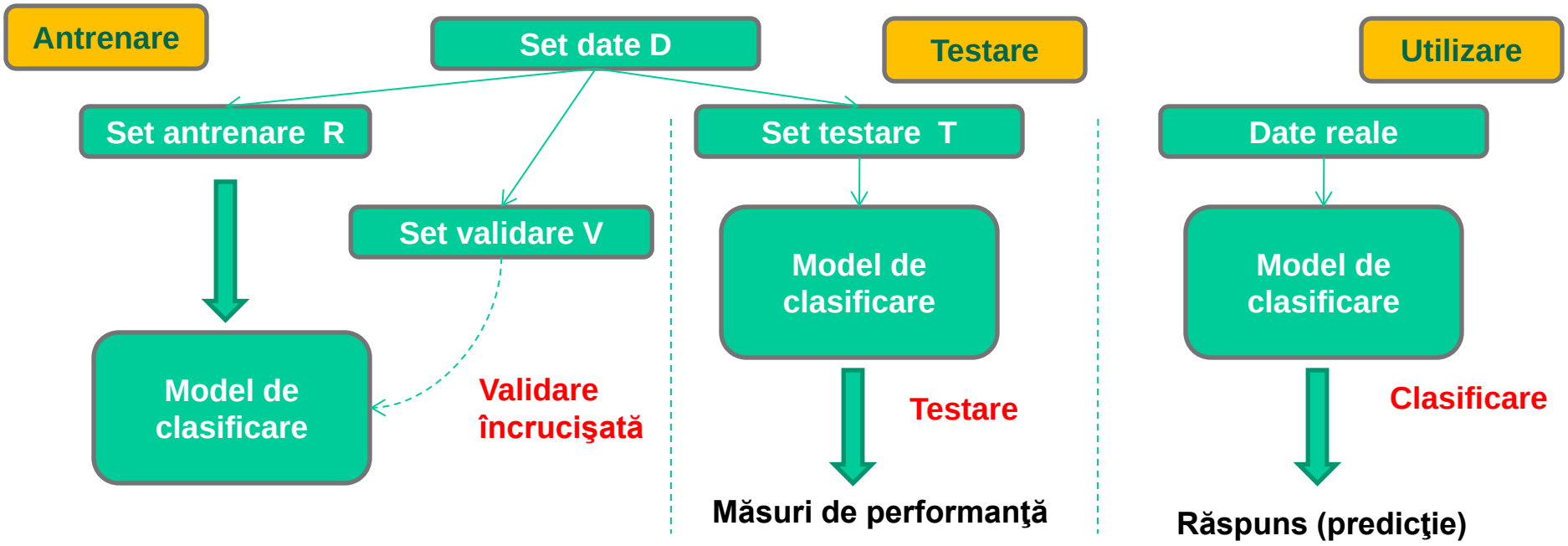
# Concepte de bază

Informaţie disponibilă:

- Set de date etichetate:
  - $D=\{(x_1,c_1),(x_2,c_2),\ldots,(x_N,c_N)\}$
  - fiecare $x_i$ are n atribute
  - Eticheta clasei $c_i$ aparţine mulţimii $\{1,2,\ldots,K\}$

Scop:

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



# Modele de clasificare

Un model de clasificare este o "mapare" între valori ale atributelor şi etichete ale claselor

Exemple de modele de clasificare:

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

Un model de clasificare trebuie să fie:

- Acurat:
  - Identifică clasa corectă
- Compact / comprehensibil
  - Uşor 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?

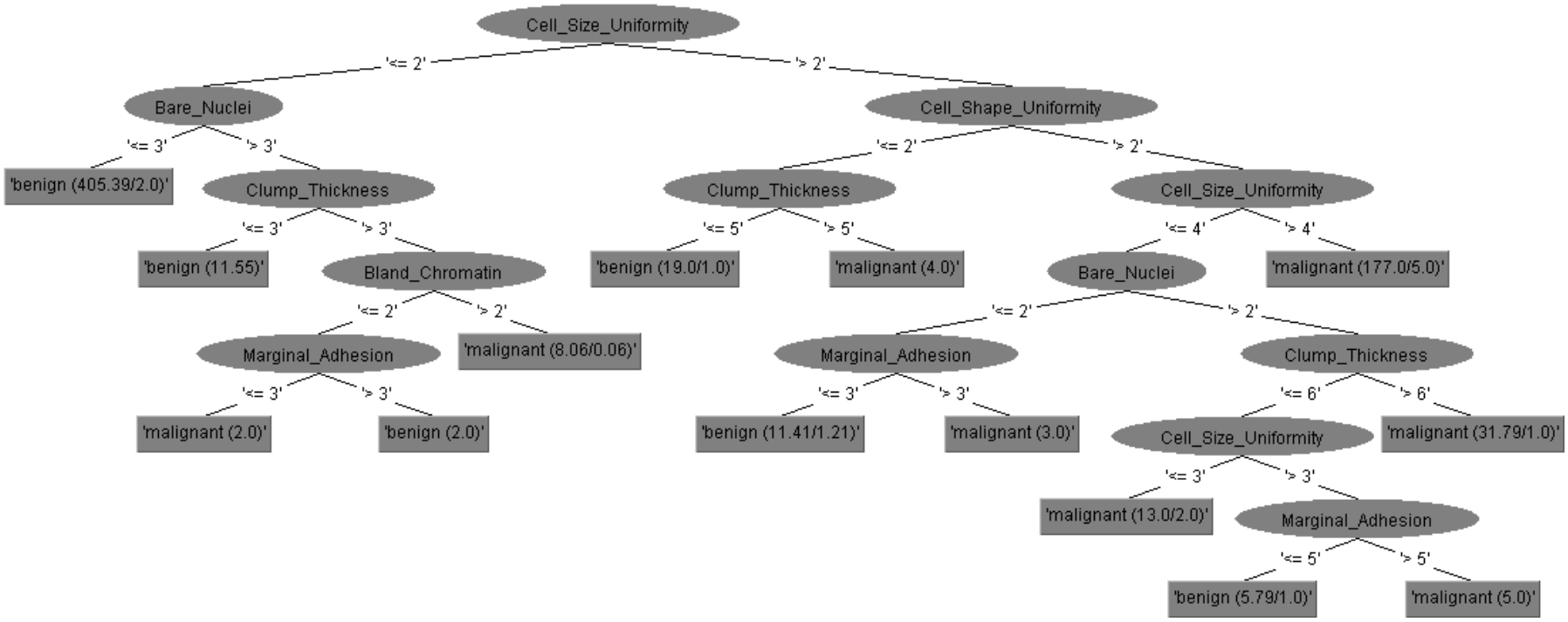
In 92.7% din cazuri regula indică clasa corectă

Cum a fost calculată această valoare?

Cum ar trebui interpretată?

# Modele de clasificare

Exemplu: un model mai complex (arbore de decizie)




Performanţa: în 94.56% din cazuri clasificatorul indică clasa corectă

Ce se poate spune despre lizibilitatea clasificatorului?

# Măsuri de performanţă

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 performanţa este de a analiza în câte cazuri clasificatorul indică răspunsul corect – această informaţie poate fi furnizată prin intermediul matricii de confuzie

Matrice de confuzie:

| | C1 | C2 |
|---|---|---|
| C1 | TP | FN |
| C2 | FP | TN |

← răspunsul clasificatorului

↑ 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 performanţă

Cazul a K clase:

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

# Măsuri de performanţă

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

Acurateţe = (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-specificitate=FP/(TN+FP) = 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 performanţă mai bună
- Sensitivitatea şi specificitatea sunt utilizare frecvent în analiza datelor medicale
- Precizia şi rata de regăsire se folosesc în domeniul regăsirii informaţiei (information retrieval)

# Măsuri de performanţă

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 informaţiei:

Precision = TP/(TP+FP) = card(relevant şi regăsit)/ card(regăsit)
Recall = TP/(TP+FN) = card(relevant şi regăsit)/ card(relevant)

O variantă agregată frecvent utilizată este media armonică a acestora:

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

# Măsuri de performanţă

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

Acurateţe 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 (detecţie atacuri informatice) ar trebui ca FP să fie cât mai mic

In astfel de situaţii se pot utiliza ponderi (interpretate ca şi 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 $C_i$

# Cel mai simplu clasificator

Exemplu:

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

# Cel mai simplu clasificator

Exemplu:

- Considerăm setul de date “sick” de la UCI Machine Learning
- Conţine 3772 înregistrări (aferente unor pacienţi), dintre care:
  - 231 sunt bolnavi (clasa C1 – pozitivă) – 6% dintre pacienţi
  - 3541 sunt sănătoşi (clasa C2 – negativă) – 94% dintre pacienţi
- Ne interesează să construim un clasificator a cărui acurateţe pt acest set de date să fie cel puţin egală cu 0.9 (90%)
- Care este cel mai simplu clasificator care satisface această cerinţă?
- Considerând regula: “indiferent de valorile atributelor clasa este C2 (negativă)” obţinem acurateţea=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 conţine nici un atribut în membrul stâng) utilizează doar distribuţia datelor în cele două clase şi 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 puţine elemente

- Totuşi ... poate fi utilizat pentru a determina o margine inferioară pt acurateţea unui clasificator: un clasificator cu acurateţe mai mică decât ZeroR ar trebui evitat

# …din nou la evaluarea performanţei

- Utilizarea întregului set de date disponibile pentru construirea clasificatorului nu este o abordare prea înţeleaptă întrucât poate conduce la supra-antrenare:
  - Clasificatorul se comportă bine pentru datele din setul de antrenare...
  - ... dar are performanţe 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 performanţei)

- Există diferite strategii de divizare a setului de date în subseturi (antrenare şi testare)

Obs: Pe lângă subsetul de testare se poate folosi şi un subset de validare (utilizat pentru ajustarea parametrilor clasificatorului)

# …din nou la evaluarea performanţei

Strategii de divizare:

- Holdout
  - Se reţin 2/3 din set pt antrenare şi 1/3 pt testare
- Holdout repetat
  - Se repetă partiţionarea (performanţa este calculată ca medie a valorilor determinate la fiecare repetare a divizării)
- Validare încrucişată
  - Se divide setul aleator în k subseturi disjuncte
  - k-fold: se folosesc k-1 subseturi pt antrenare, iar al k-lea se foloseşte pt testare (evaluarea performanţei)
  - Leave-one-out: k=n
- Eşantionare repetată (util în cazul seturilor nebalansate)
  - oversampling vs undersampling
- Bootstrap
  - Selecţie cu revenire

# Dincolo de ZeroR

Set de date: sick.arff, 29 atribute, 3772 instanţe (231 în clasa C1, 3541 în clasa C2), 2 clase

ZeroR (clasa e întotdeauna C2): acurateţe=0.94

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

Exemple de reguli (obţinute 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 >= 1.15 then C2 (negative)

If T3 value is missing then C2 (negative) Acurateţe: 0.96

# OneR

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

Obs: este adecvat pentru atributele 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 instanţele care au $A_i = v_{ij}$)

    se combină regulile într-un set $R_i$ corespunzător lui $A_i$ şi 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

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 |

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)

# OneR

Example: 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 |

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)

# OneR

Exemplu: 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 |

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)

Reguli: weather/play dataset

If outlook=sunny then "no"

If outlook=overcast then "yes"

If outlook=rainy then "yes"

Acurateţe (set antrenare): 0.71

Acurateţe (validare încrucişată): 0.43 (!!)

# OneR

Exemplu: 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 |

Reguli: weather/play dataset

If outlook=sunny then “no”

If outlook=overcast then “yes”

If outlook=rainy then “yes”

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 (instanţe etichetate)
  - Output: set de reguli simple (toate regulile implică un singur atribut – acelaşi atribut în toate)
  - Algoritm: se analizează toate atributele şi valorile corespunzătoare acestora şi se selectează atributul pentru care eroarea de clasificare este minimă
- Utilizarea regulilor (etapa de clasificare)
  - Input: set de reguli, dată (instanţă) nouă
  - Output: eticheta clasei
  - Algoritm:
    - Identifică regula care se potriveşte cu data
    - Returnează clasa corespunzătoare regulii identificate

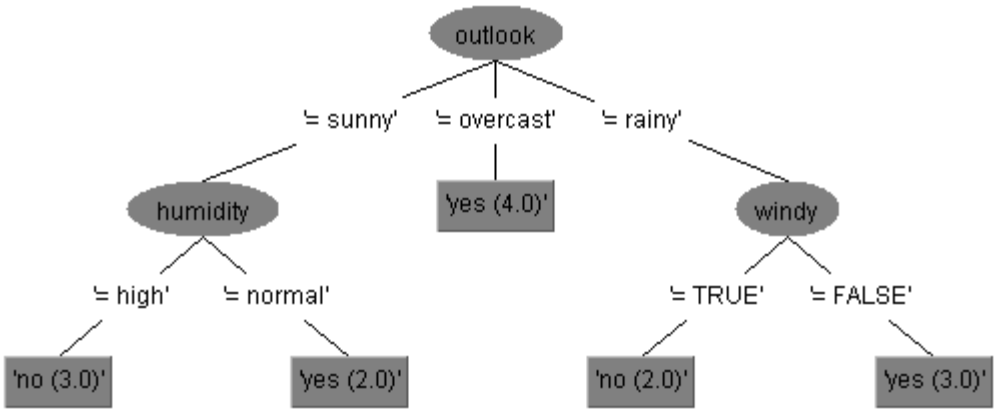
# Arbori de decizie

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

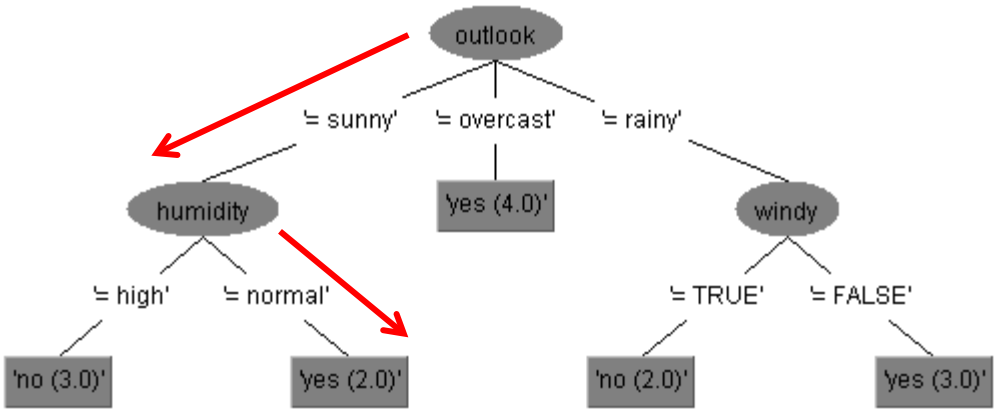
# Arbori de decizie

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 instanţe?
(outlook=sunny, temperature=mild, humidity=normal, windy=False)?
Clasa: Yes

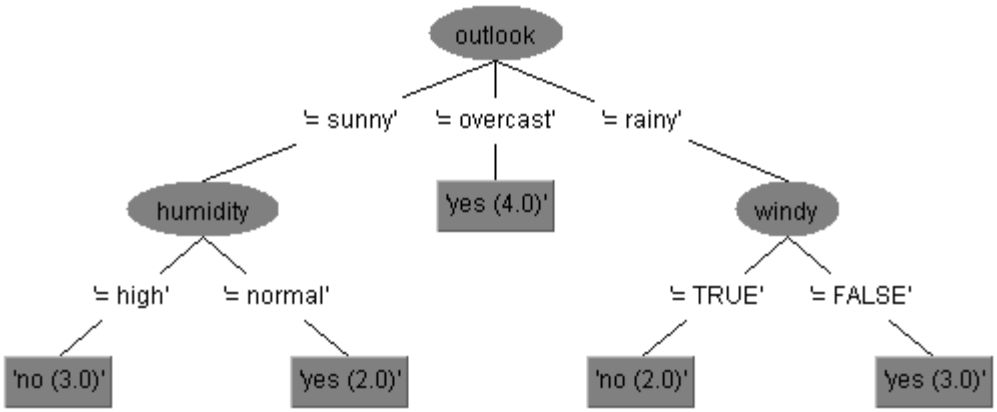
# Arbori de decizie

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 şi se plasează în rădăcina arborelului
- Pt fiecare valoare posibilă a atributului of (cele prezente în setul de date) se construieşte o ramură
- Se partiţionează setul de date în subseturi corespunzătoare fiecărei ramuri
  - Dacă un subset conţine date ce aparţin unei singure clase atunci el va corespunde unui nod frunză (nu se mai ramifică) –nod pur
  - Dacă subsetul conţine date din mai multe clase atunci se continuă procesul de partiţionare 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 condiţie de testare ar trebui asociată cu fiecare nod?

# Arbori de decizie

In ce ordine ar trebui analizate atributele?

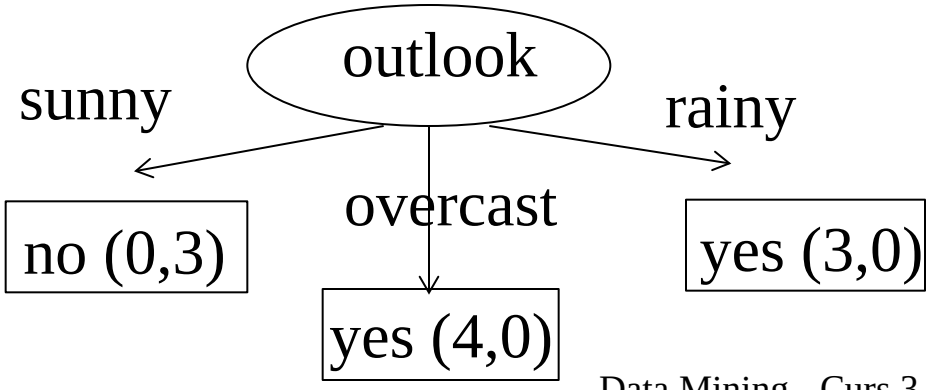
Ideea principală:

- 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 aparţină aceleiaşi clase)

Set de date: weather/play (date selectate)

| 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” (conţin date ce aparţin aceleiaşi clase)
- Conduce la reguli cu un singur atribut în membrul stâng
- O astfel de situaţie se întâmplă rar pentru date reale

# Arbori de decizie

Principalele probleme ce trebuie soluţionate la construirea unui arbore de decizie

- Ce condiţii 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 împărţit în două subseturi)
    - Ramificare multiplă (setul curent de date este împărţit în mai multe subseturi)

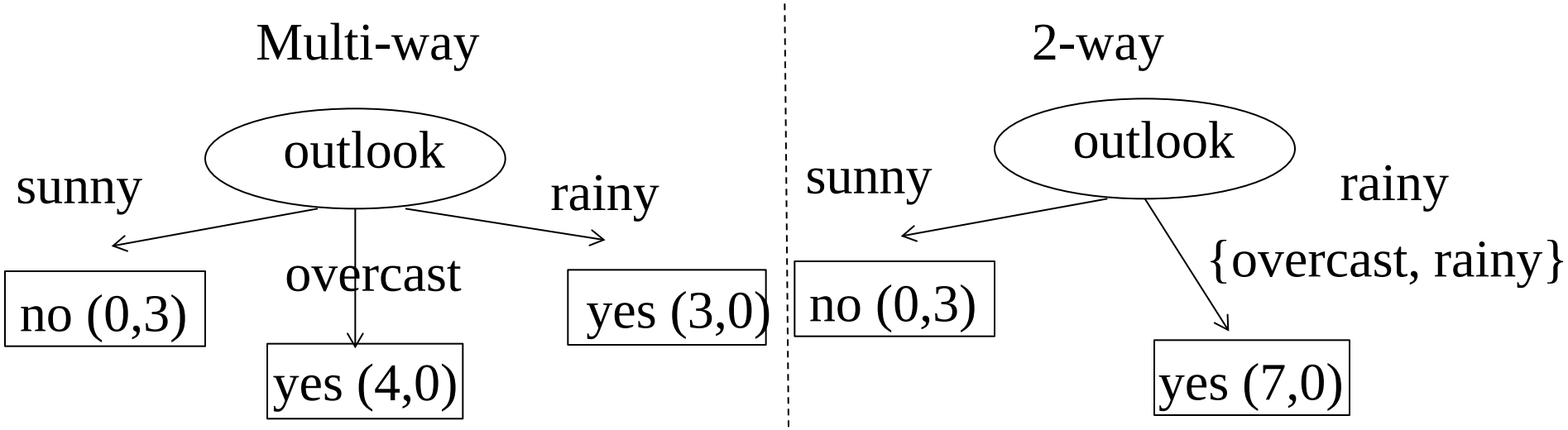
- Ce atribut ar trebui selectat pentru partiţionare?
  - Cel cu puterea cea mai mare de discriminare – cel ce asigură partiţionarea setului curent în subseturi cu grad mare de puritate
  - Criterii ce pot fi utilizate:
    - Bazate pe entropie (ex: câştig informaţional)
    - Index Gini
    - Măsură a erorii de clasificare

# Arbori de decizie

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

Atribute nominale şi ordinale:

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



# Arbori de decizie

Ce condiţii 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 partiţionare?

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

Exemplu: information gain

Câştig informaţional =
Entropia(distribuţia datelor înainte de partiţionare) –
EntropiaMedie (distribuţia datelor după partiţionare)

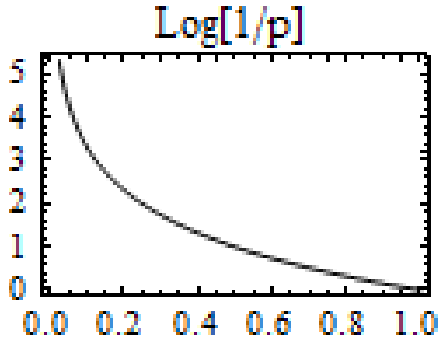
# Reminder: entropie

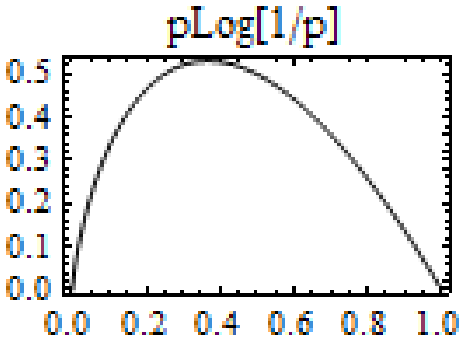
Fie D=($p_1$,$p_2$, …, $p_k$) o distribuţie de probabilitate. Entropia asociată acestei distribuţii de probabilitate este caracterizată de:

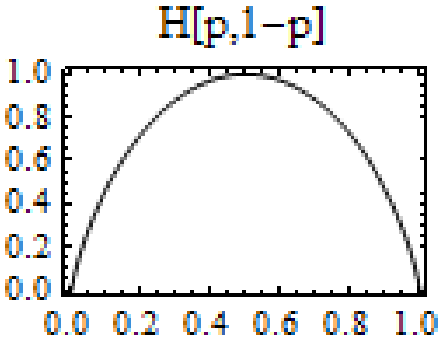
$$H(D) = H(p_1, p_2, ..., p_k) = -\sum_{i=1}^{k} p_i \log p_i$$

şi poate fi interpretată ca o măsură a incertitudinii (sau surprizei) când se generează/selectează o valoare în pe baza acestei distribuţii

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




Obs:

Interpretare Log[1/p] : surpriza de a observa un eveniment caracterizat de o probabilitate mică (eveniment neaşteptat) este mai mare decât cea corespunzătoare unui eveniment de probabilitate mai mare (eveniment aşteptat)

# Reminder: entropie

In contextul rezolvării problemelor de clasificare:

- D={$C_1$,$C_2$,…,$C_k$} (set de date distribuit in k clase)
- Distribuţia de probabilitate ($p_1$,$p_2$, …, $p_k$), $p_i$=card($C_i$)/card(D)
- Fie A un atribut şi $v_1$,$v_2$,…,$v_{mA}$ valorile posibile ale acestui atribut
- Fie $D_j$=setul de date din D pt care atributul A are valoarea $v_j$ şi $P_j$ distribuţia 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)
- Câştigul informaţional obţinut prin partiţionarea setului de date folosind atributul A este:

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

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

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

# 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 |

- Distribuţia 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(sunny)=-2/5*log(2/5)-3/5*log(3/5)=0.97
H(overcast)=-1*log(1)-0=0
H(rainy)=-3/5*log(3/5)-2/5*log(2/5)=0.97

IG(outlook)=0.94-5/14*0.97-4/14*0-5/14*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 |

H(hot)=-2/4*log(2/4)-2/4*log(2/4)
H(mild)=-4/6*log(4/6)-2/6*log(2/6)
H(cool)=-3/4*log(3/4)-1/4*log(1/4)

IG(temperature)=0.03

- Distribuţia 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 |

# 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 |

- Distribuţia 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(high)=-3/7*log(3/7)-4/7*log(4/7)
H(normal)=-6/7*log(6/7)-1/7*log(1/7)

IG(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 |

H(false)=-6/8*log(6/8)-2/8*log(2/8)
H(true)=-3/6*log(3/6)-3/6*log(3/6)

IG(windy)=0.05

- Distribuţia 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 |

# Alegerea atributului de ramificare

Exemplu

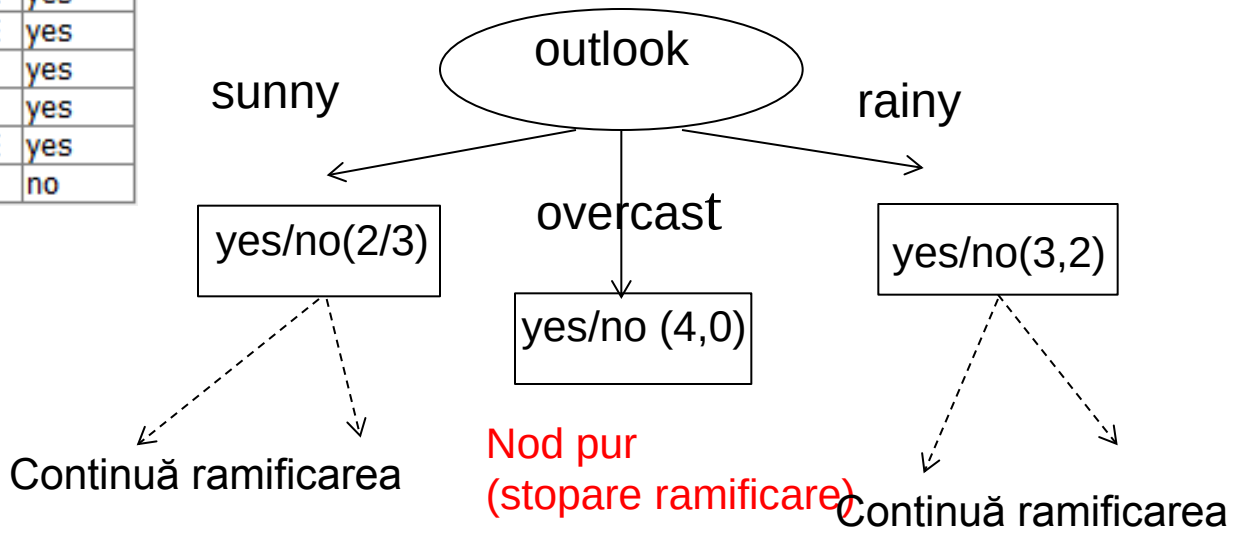
Relation: weather.symbolic

| 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 |

Câştigul informaţional al fiecărui atribut:

- IG(outlook)=0.25
- IG(temperature)=0.03
- IG(humidity)=0.15
- IG(windy)=0.05

Primul atribut selectat: outlook

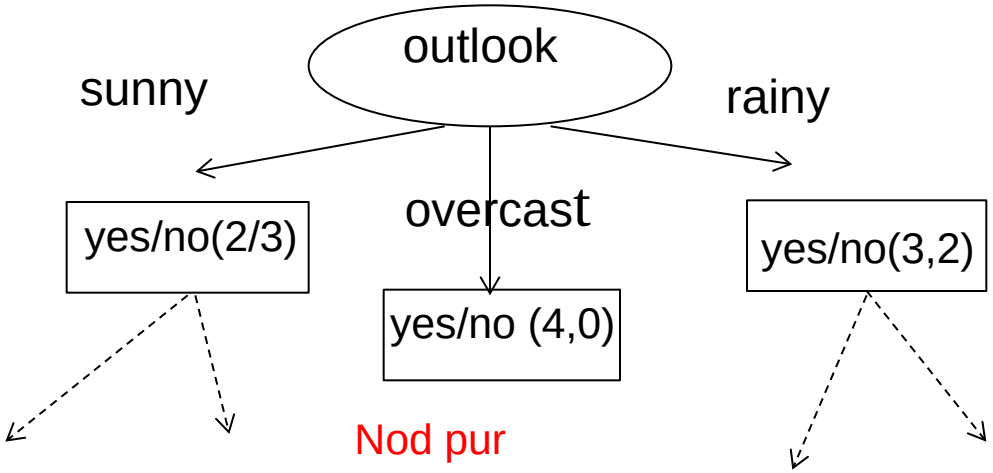


# Alegerea atributului de ramificare

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)

Câştig informaţional pt atributele rămase:

- Entropia pt subsetul “sunny” :
  H(D(sunny))=-2/5*log(2/5)-3/5*log(3/5)=0.97
- H(hot)=0, H(mild)=1, H(cool)=0
- IG(temperature)=0.97-2/5=0.57

Temperature

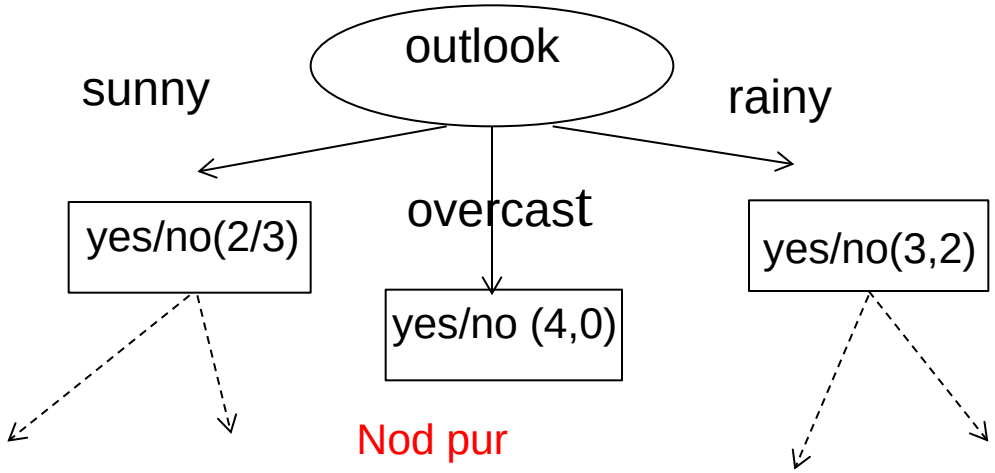
| | C1 (yes) | C2(no) | Freq. |
|---|---|---|---|
| Hot | 0/2 | 2/2 | 2/5 |
| Mild | 1/2 | 1/2 | 2/5 |
| Cool | 1/1 | 0/1 | 1/5 |

# 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)

Câştig informaţional pt atributele rămase:

- Entropia pt subsetul “sunny” :

H(D(sunny))=-2/5*log(2/5)-3/5*log(3/5)=0.97

- H(high)=0, H(normal)=0
- IG(humidity)=0.97-0=0.97

Humidity

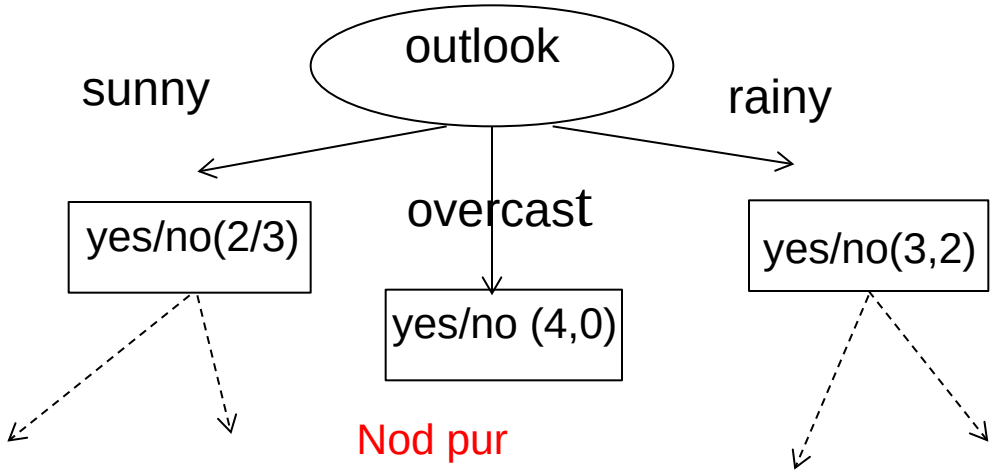
| | C1 (yes) | C2(no) | Freq. |
|---|---|---|---|
| High | 0/3 | 3/3 | 3/5 |
| Normal | 2/2 | 0/2 | 2/5 |

# 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)

Câştig informaţional pt atributele rămase:

- Entropia pt subsetul “sunny” :

H(D(sunny))=-2/5*log(2/5)-3/5*log(3/5)=0.97

- H(false)=0, H(true)=1
- IG(windy)=0.97-0.95=0.02

Windy

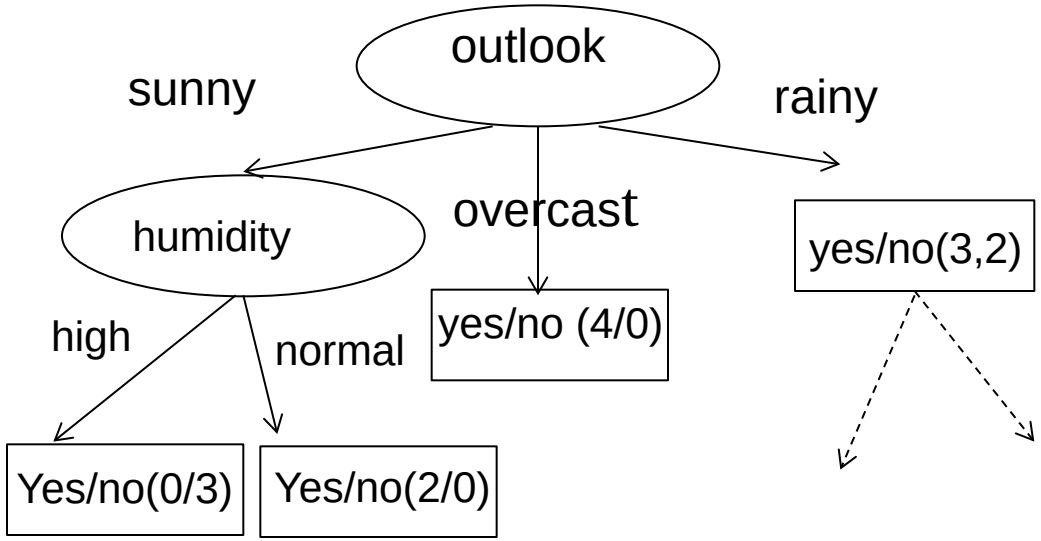
| | C1 (yes) | C2(no) | Freq. |
|---|---|---|---|
| false | 1/3 | 2/3 | 3/5 |
| true | 1/2 | 1/2 | 2/5 |

# 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 |




Câştig informaţional pt atributele rămase:

- IG(temperature)=0.97-2/5=0.57
- IG(humidity)=0.97-0=0.97
- IG(windy)=0.97-0.95=0.02

# Alegerea atributului de ramificare

Obs:

- Câştigul informaţional favorizează atributele caracterizate printr-un număr mare de valori
- Pentru a limita această influenţă se poate utiliza raportul (Gain Ratio):

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

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

(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 distribuţiei corespunzătoare ramificării analizate (cu cât mai mică valoarea cu atât mai bună)

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

# Algoritmi pentru construirea arborilor de decizie

ID3:

- Intrare: 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 = îmbunătăţire a algoritmului ID3 pt a trata:

- Atribute continue:
  - Incorporează procedură de discretizare
- Valori absente:
  - Datele ce conţin valori absente sunt ignorate

sau

  - Valorile absente sunt imputate

- Atribut de ramificare:
  - Utilizează Gain Ratio pt selecţia atributului

- Simplificare sau fasonare (Pruning):
  - Anumiţi subarbori sunt înlocuiţi cu noduri frunză (dacă eroarea de clasificare nu creşte semnificativ) – abordare bottom-up

Obs:

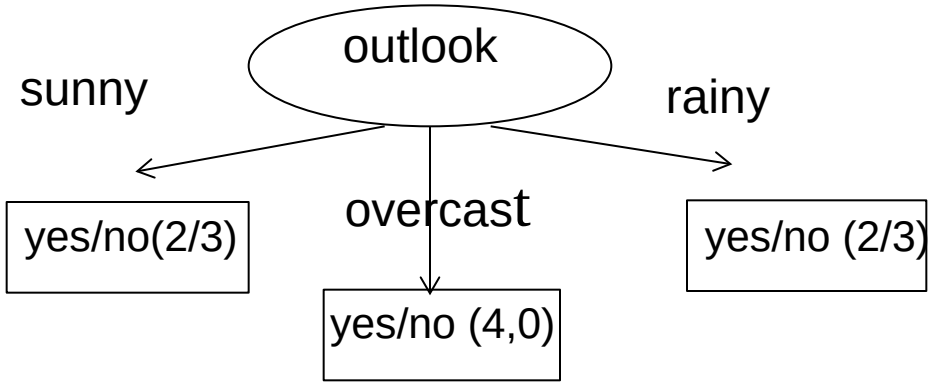
C5.0 – varianta comerciala a algoritmului C4.5

J48 – implementarea din Weka a algoritmului C4.5

# Algoritmi pentru construirea arborilor de decizie

Simplificare:

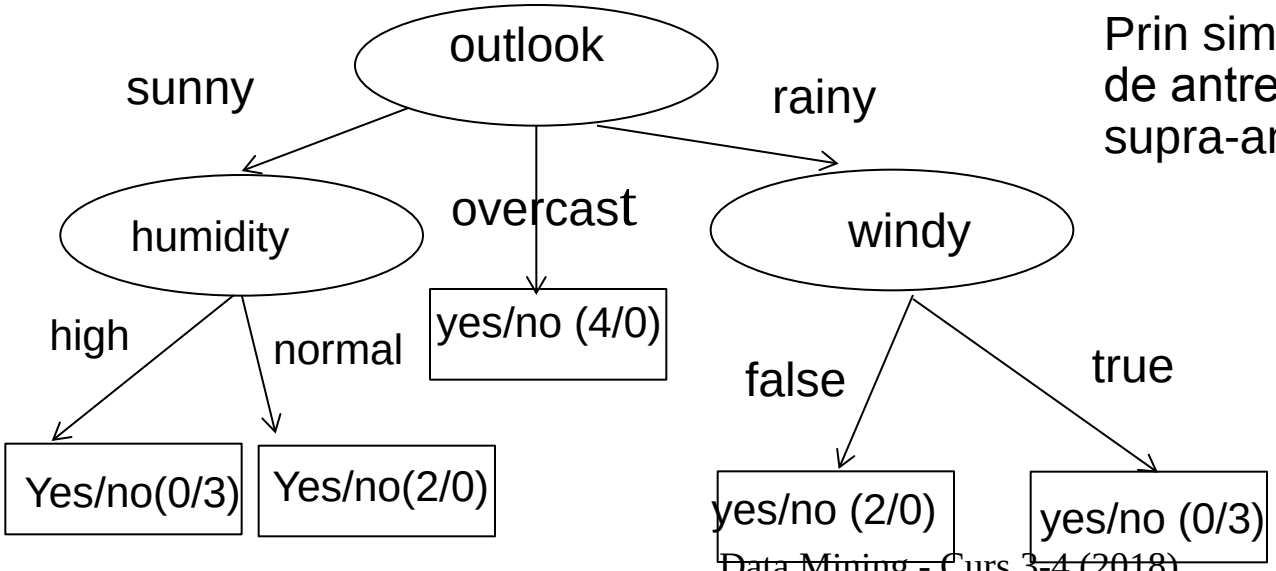
- Anumiţi subarbori sunt înlocuiţi cu noduri frunză (dacă eroarea de clasificare nu creşte semnificativ) – abordare bottom-up




Arbore : error = 0

Arbore simplificat: error = 4/14

Prin simplificare eroarea pe setul de antrenare creşte dar riscul de supra-antrenare poate sa scada



# Extragerea regulilor de clasificare

Reminder: regulile de clasificare sunt structuri de tip IF … THEN care conţin:

- In partea de antecedent (membrul stâng): condiţii privind valorile atributelor (pot fi expresii logice care implică mai multe atribute)
- In partea de consecinţă (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:

- Aceste reguli sunt extrase dintr-un arbore de decizie – fiecare ramură conduce la o regulă

- Condiţiile referitoare la noduri aflate pe aceaşi ramură se combină prin AND:

  IF (outlook=sunny) and (humidity=high) THEN play=no

- Regulile corespunzând unor ramuri diferite dar conducând la aceeaşi consecinţă (aceeaşi etichetă de clasă) pot fi reunite prin disjuncţie (OR) între părţile de antecedent:

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

# Extragerea regulilor de clasificare

Regulile de clasificare pot fi extrase direct din date printr-un proces de învăţare utilizând algoritmi de acoperire (covering algorithms)

Noţiuni:

- O regulă acoperă o dată dacă valorile atributelor se potrivesc cu condiţiile din antecedentul regulii
- Similar, despre o dată se spune că activează o regulă dacă valorile atributelor se potrivesc cu condiţiile din antecedentul regulii

- Suportul unei reguli (support) = fracţiunea din setul de date care este acoperită de către regulă şi aparţin aceleiaşi clase ca şi regula
  = |cover(R) ∩class(R ) |/|D|
- Gradul de încredere în regulă (rule confidence) = fracţiunea din datele acoperite de regulă care au aceeaşi clasă ca cea specificată de regulă = |cover(R) ∩class(R )|/|cover(R )|

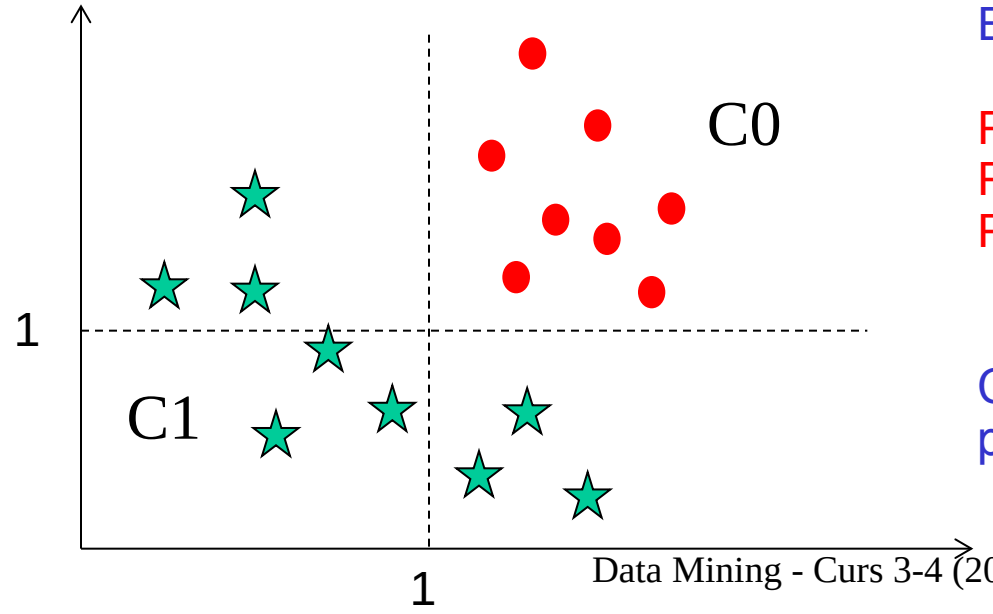
cover( R) = subsetul de date acoperit de R
class(R) = subsetul de date care au aceeaşi clasă cu R
D= setul de date

# Extragerea regulilor de clasificare

Noţiuni:

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

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




Exemplu:

R1: IF x>1 and y>1 THEN C0
R2: IF x<=1 THEN C1
R3: IF x>1 and y<=1 THEN C1

Ce se întâmplă însă dacă aceste proprietăţi nu sunt satisfăcute?

# Extragerea regulilor de clasificare

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

Conflictele pot fi rezolvate în unul dintre următoarele moduri:

- Ordonarea regulilor (pe baza unui criteriu) şi decizia se ia cf primei reguli activate (prima regulă care se potriveşte cu instanţa).

- 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 condiţii 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 obţine considerând clasa dominantă pe baza tuturor regulilor activate de către instanţă

# Extragerea regulilor de clasificare

Algoritm secvenţial de acoperire:

Intrare: set de date
Ieşire: set ordonat de reguli

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

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

Obs:

- Aceasta este structura generală a algoritmilor secvenţiali de acoperire
- Algoritmii pot să difere în funcţie de strategia de selecţie a claselor

# Extragerea regulilor de clasificare

Exemplu: algoritmul RIPPER

Particularităţi:

- Ordonare bazată pe clase: clasele sunt selectate crescător după dimensiune (clasele rare sunt selectate prima dată)
- Regulile corespunzătoare unei clase sunt plasate consecutiv în lista de reguli
- 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 sfârşit rămân date "neacoperite" atunci se poate defini o regulă de tipul "catch all" căreia i se asociază clasa dominantă din setul de date "neacoperite"

# Clasificatori bazaţi pe instanţe

Ideea principală: datele similare aparţin aceleiaşi 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 şi cele din setul de antrenare şi se identifică exemplarele cele mai apropiate
  - Se alege clasa cea mai frecventă întâlnită în subsetul celor mai similare exemple

Obs:

- Astfel de clasificatori sunt consideraţi leneşi (“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 bazaţi pe principiul celui/celor mai apropiat/apropiati vecin/vecini (k-Nearest

# Clasificatori bazaţi pe instanţe

Ideea principală: datele similare aparţin aceleiaşi clase

Obs:

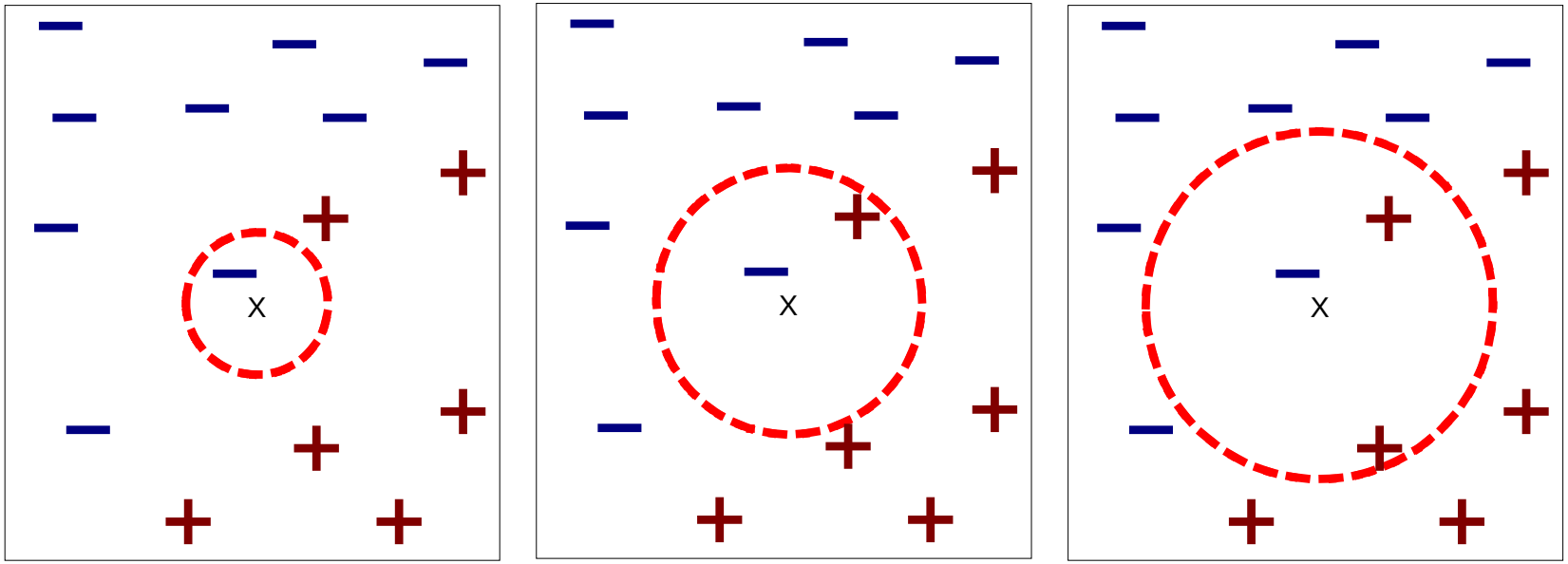
- Astfel de clasificatori sunt consideraţi leneşi (“lazy”) deoarece faza de antrenare nu presupune nici un efort de calcul (întregul efort este amânat pentru faza de clasificare)

- Cei mai populari clasificatori din această categorie sunt cei bazaţi pe principiul celui/celor mai apropiat/apropiaţi vecin/vecini (k-Nearest Neighbour)

# Clasificatori bazaţi pe instanţe

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

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

# Clasificatori bazaţi pe instanţe

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

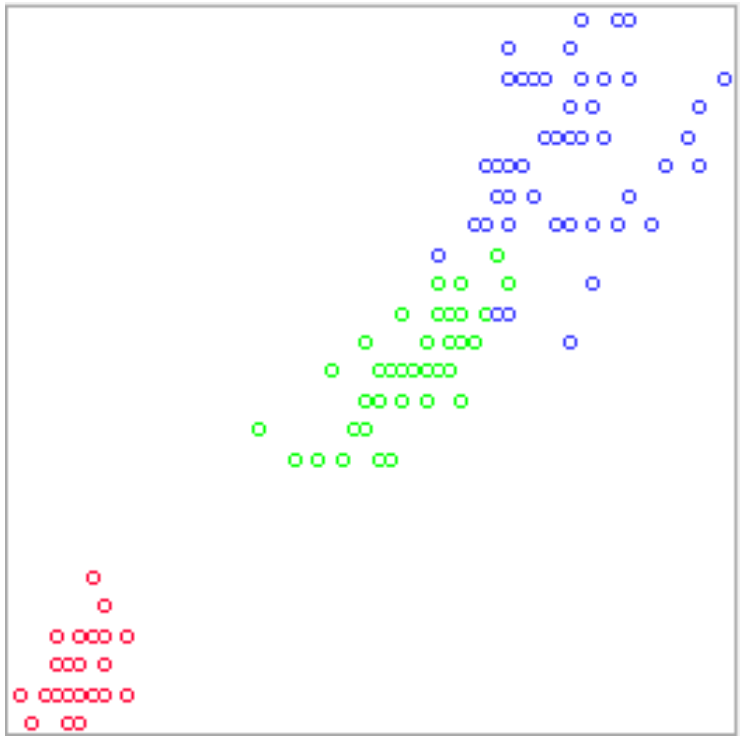
Performanţa clasificatorilor de tip kNN depinde de:

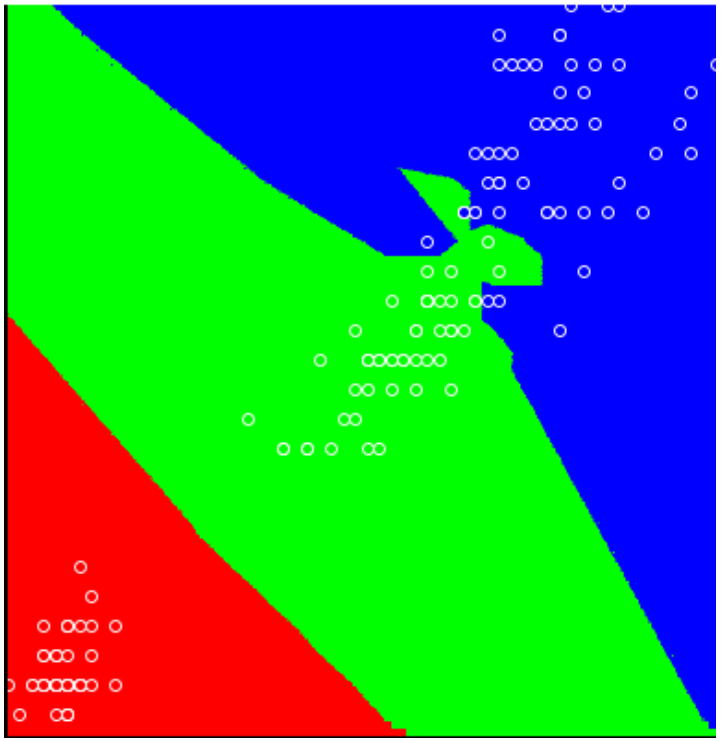
- Măsura de similaritate/ disimilaritate
  - Se alege în funcţie de tipurile atributelor şi de proprietăţile problemei
- Valoarea lui k (numărul de vecini)
  - Cazul cel mai simplu: k=1 (nu e indicat în cazul datelor afectate de zgomot)

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

# Clasificatori bazaţi pe instanţe

1NN = Nearest Neighbor bazat pe cel mai apropiat vecin (şi distanţa euclidiană)
Ilustrarea regiunilor. Dataset: iris2D ("petal length" and "petal width").
Plot: Weka->Visualization->BoundaryVisualizer





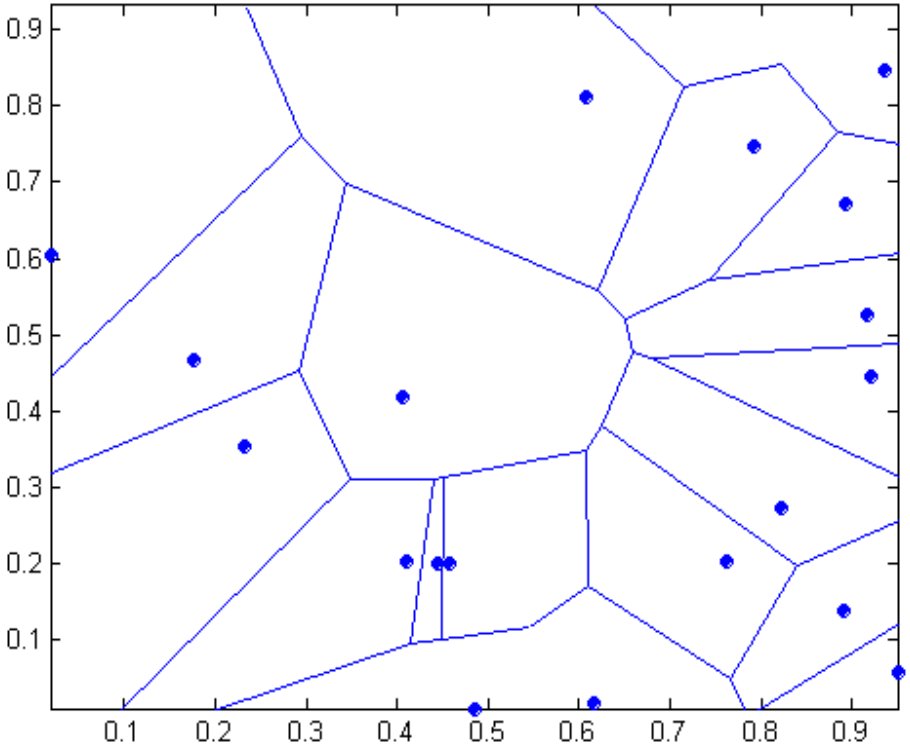
# Clasificatori bazaţi pe instanţe

1NN = Nearest Neighbor bazat pe cel mai apropiat vecin (şi distanţa euclidiană)

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

Obs:

Fiecare instanţă din setul de antrenare corespunde unei regiuni care cuprinde datele aflate în vecinătatea acelei instanţe




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

# Măsuri de similaritate/ disimilaritate

Considerăm două entităţi (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 şi 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 şi B sunt mai diferite

Alegerea măsurii depinde de:

- Tipul atributelor
- Numărul de atribute
- Distribuţia datelor

# Măsuri de similaritate/ disimilaritate

Atribute numerice

Cele mai populare măsuri de disimilaritate:

- Distanţa euclidiană
- Distanţa Manhattan

Obs:

- Distanţa euclidiană este invariantă în raport cu rotaţii
- Dacă nu toate atributele au aceeaşi importanţă atunci se foloseşte varianta ponderată (e.g. $w_i(a_i-b_i)^2$ în loc de $(a_i-b_i)^2$ )

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

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

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

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

Ponderile se determină folosind tehnici de preprocesare

# Măsuri de similaritate/ disimilaritate

Aspecte practice – problema dimensiunii (dimensionality curse):

- Puterea de discriminara a acestor distanţe scade pe măsură ce nr de atribute (n) creşte =>

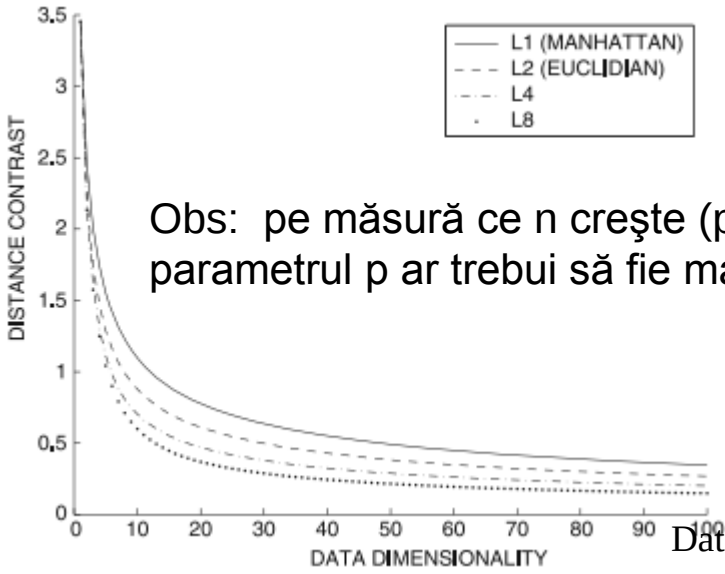
Pt date cu multe atribute clasificatorii bazaţi pe distanţe devin inefectivi

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

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

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

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




Obs: pe măsură ce n creşte (p) parametrul p ar trebui să fie mai mic

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

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

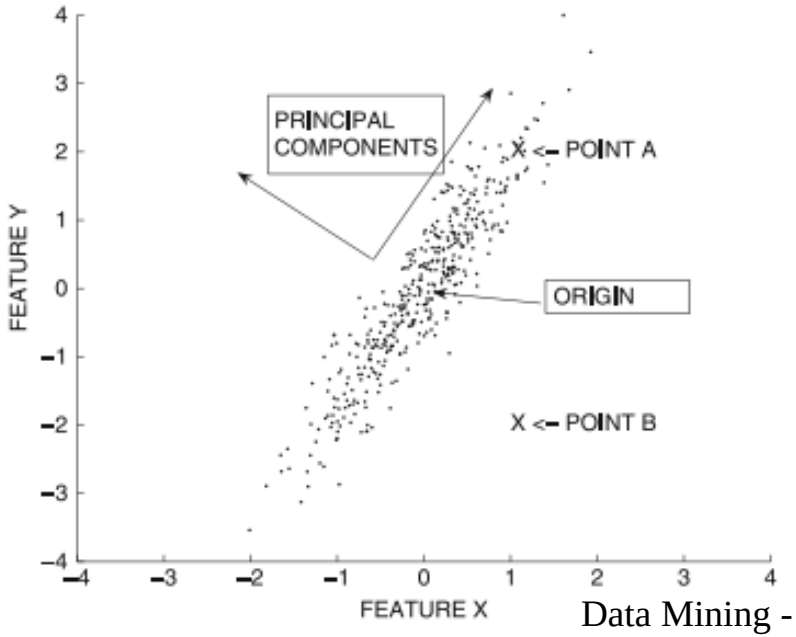
$\sigma$ = standard deviation of distances

Aggarwal, Data Mining Textbook, 2015

# Măsuri de similaritate/ disimilaritate

## Aspecte practice – impactul distribuţiei datelor

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




Aggarwal, Data Mining Textbook, 2015

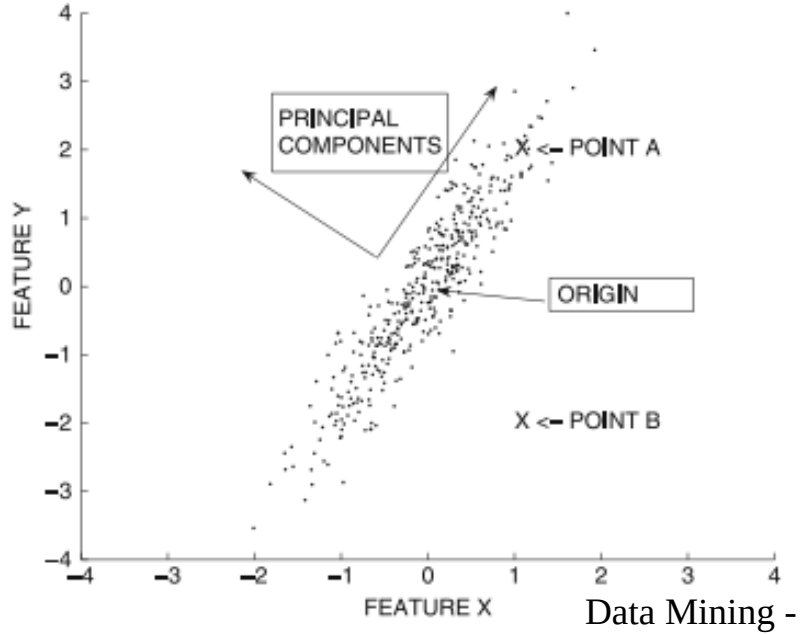
# Măsuri de similaritate/ disimilaritate

Aspecte practice – impactul distribuţiei datelor

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

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

Altă întrebare: cum poate fi inclusă distribuţia datelor în calculul distanţei?




Distanta Mahalanobis

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

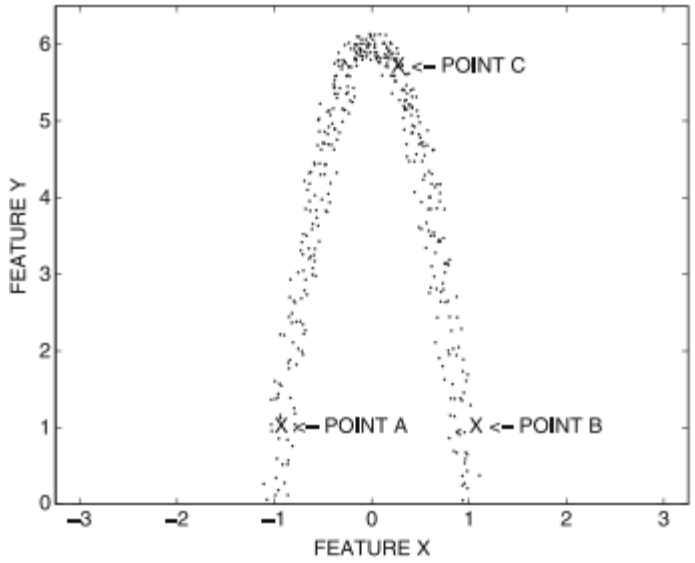
$$C^{-1} = \text{inversa matricii de covarianta}$$

Aggarwal, Data MiningTextbook, 2015

# Măsuri de similaritate/ disimilaritate

## Aspecte practice – impactul distribuţiei datelor

Intrebare: este distanţa dintre A şi B mai mică decât distanţa dintre B şi C?




Aggarwal, Data Mining Textbook, 2015

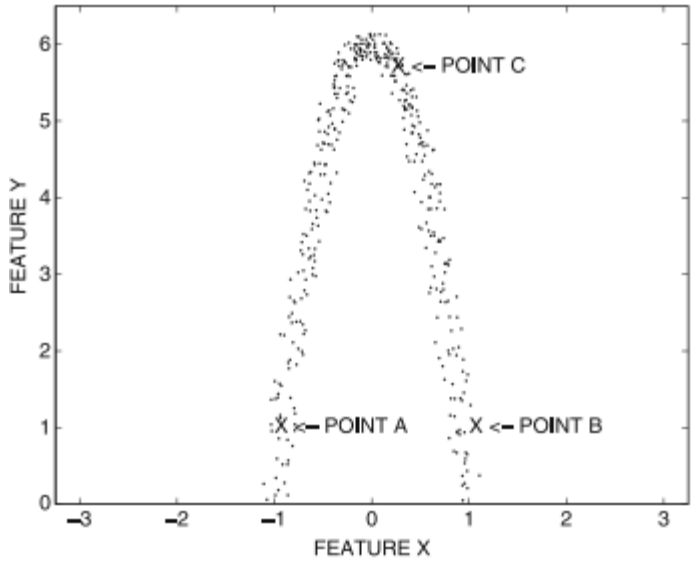
# Măsuri de similaritate/ disimilaritate

Aspecte practice – impactul distribuţiei datelor

Intrebare: este distanţa dintre A şi B mai mică decât distanţa dintre B şi C?

R: da, dacă ignorăm distribuţia datelor şi folosim distanţa euclidiană

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




Distanţa geodesică:

- Se construieşte un graf ce are în noduri punctele iar muchiile unesc nodurile vecine (ex: cei mai apropiaţi k vecini)
- Calculează distanţa 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: $sim(A,B)=A^TB/(||A||\ ||B||)$ (produsul scalar dintre A şi B împărţit la produsul normelor)

Remarcă:

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

$$d_E^2(A,B) = (A-B)^T(A-B) = A^TA - 2A^TB + B^TB =$$

$$= 2(1\text{-}A^TB) = 1(1\text{-}sim(A,B))$$

# Măsuri de similaritate/ disimilaritate

Atribute nominale

Abordare 1: Transformarea atributelor nominale în atribute numerice (prin binarizare) şi utilizarea măsurilor de similaritate/disimilaritate pentru vectori binari:

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

- Similaritate: coeficientul Jaccard

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

Obs: $S_A$ şi $S_B$ sunt submulţimi ale mulţimii globale cu n atribute care corespund vectorilor de apartenenţă A şi 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: similarităţile mai puţin frecvente pot fi considerate mai relevante decât cele frecvente

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

$f(a_i) = \text{frecventa valorii } a_i \text{ in setul de date}$

(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 secvenţe biologice) – se utilizează distanţa de editare
- Concepte (e.g. noduri într-o ontologie) – distanţe bazate pe cele mai scurte căi în grafuri sau arbori
- Grafuri (e.g. Reţele sociale sau biologice) – ponderea structurilor (tiparelor) similare în cele două structuri

# kNN: alegerea lui k

Performanţa clasificatorilor de tip kNN depinde de numărul de vecini

Cazuri extreme:

- k=1 - clasificatorul nu este robust (erorile din setul de date influenţează 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 şi se alege valoarea care maximizează performanţa

# kNN: cost

Clasificarea unei noi instanţe necesită calculul a N distanţe (sau măsuri de similaritate pt un set de date cu N elemente precum şi selecţia celor mai mici k distanţe ➔ O(N+kN)

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

Abordări posibile:

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

# Curs următor

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