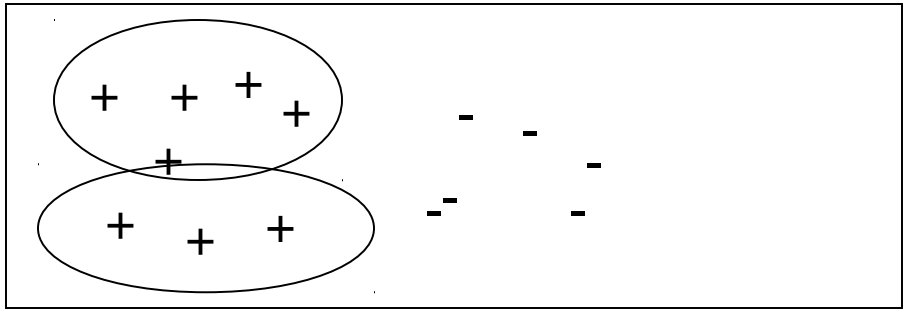
Apprendimento di insiemi disgiuntivi di regole

Apprendimento di insiemi disgiuntivi di regole

- Consideriamo prima l'apprendimento di un singolo concetto target (attributo target booleano).
- Regole della forma:
  - If Antecedente then Class=pos
- Classificazione: se nessuna regola fa match, allora classe negativa
- Metodo 1: Utilizzare un algoritmo di copertura sequenziale:
  - Ripeti finche' l'insieme di esempi positivi non e' vuoto
    - Apprendi una regola
    - Rimuovi gli esempi positivi coperti dalla regola
- Metodo 2: apprendere un albero di decisione e poi convertirlo in regole

Copertura Sequenziale

- Ad ogni esecuzione, si trova una regola congiuntiva che e' coerente con alcuni esempi positivi e con tutti gli esempi negativi
- Gli esempi positivi che sono stati considerati sono eliminati dal training set e l'algoritmo e' ripetuto finche' sono coperti tutti gli esempi positivi



Algoritmo di apprendimento sequenziale

APPRENDIMENTO_SEQUENZIALE(Attributo-target,Attributi,Esempi)

Regole_apprese$\leftarrow\varnothing$

Ripeti

Regola$\leftarrow$
APPRENDI_UNA_REGOLA(Attributo-target, Attributi,Esempi)

Regole_apprese$\leftarrow$ Regole_apprese$\cup${Regola}

Esempi $\leftarrow$Esempi-{esempi positivi coperti da Regola}

finche' l'insieme degli esempi positivi non e' vuoto

restituisci le Regole_apprese

APPRENDI_UNA_REGOLA

- APPRENDI_UNA_REGOLA accetta un insieme di esempi positivi e negativi come input, e produce una regola singola che copre molti esempi positivi e pochi o nessun esempio negativo
- Alta accuratezza, ma non necessariamente alta copertura
- Come implementare questa procedura?
- Una possibilita': per mezzo dell'algoritmo Candidate-Elimination

APPRENDI_UNA_REGOLA

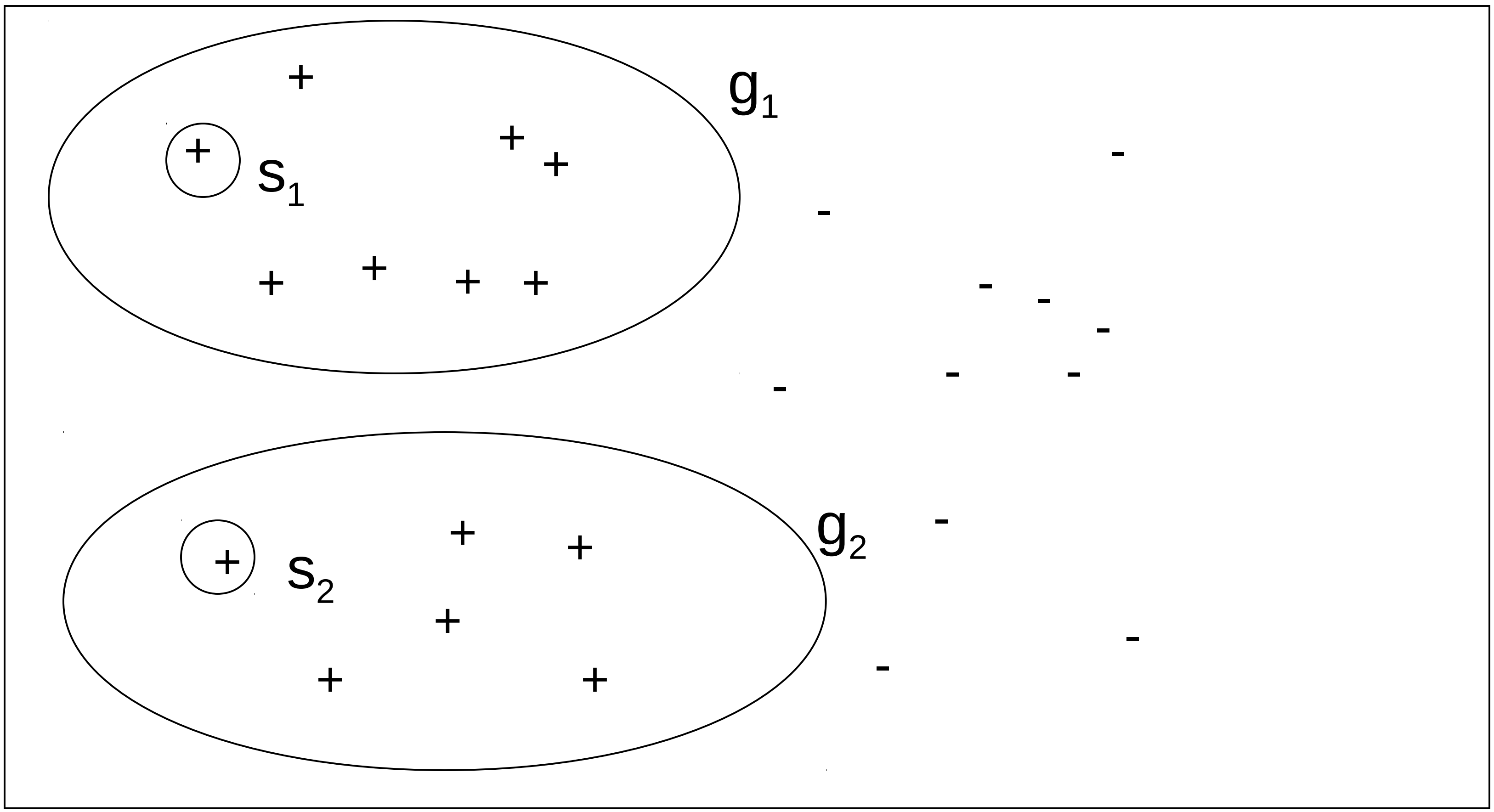
1-Inizializza S a contenere un esempio positivo (esempio seme)

2-Per ciascun esempio negativo applica la routine Update-G a G

3-Restituisci come regola una descrizione g da G

Dato che Update-G e' stato applicato usando tutti gli esempi negativi, g non copre nessun esempio negativo, ma puo' coprire diversi esempi positivi

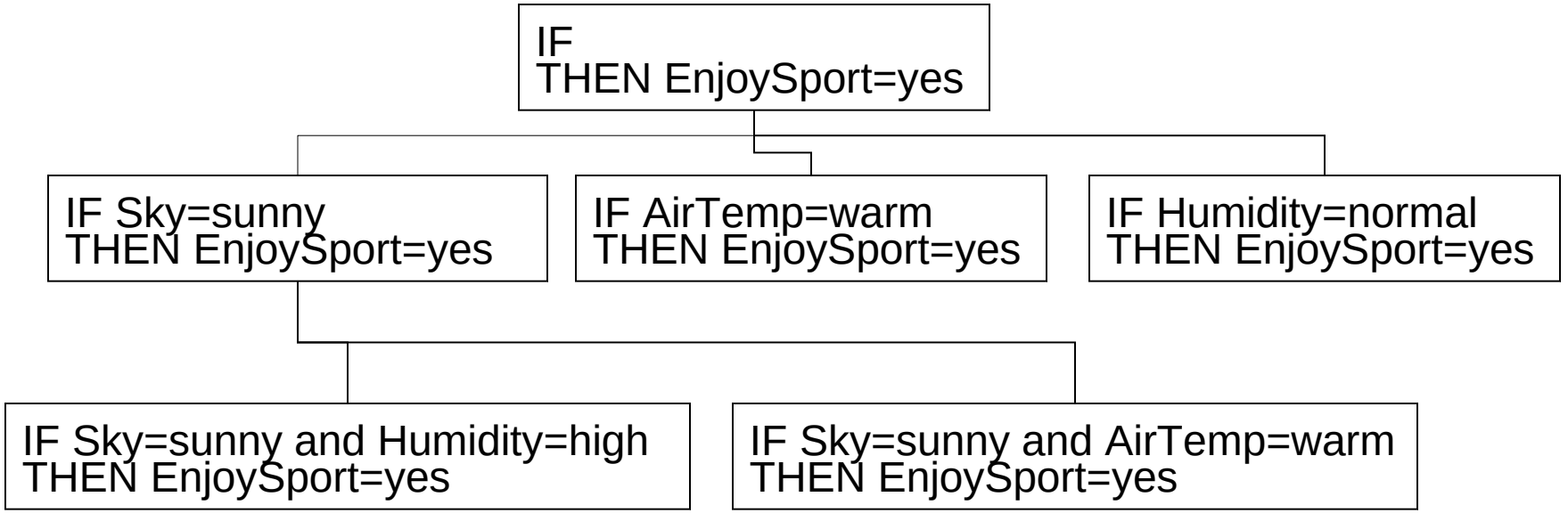
Copertura sequenziale + Candidate Elimination



Ricerca da generale a specifico

- Un approccio differente per realizzare APPRENDI_UNA_REGOLA consiste nel compiere una ricerca nello spazio delle regole partendo dalla regola più generale (con antecedente vuoto) e aggiungendo via via letterali all'antecedente
- Ad ogni passo si aggiunge il letterale che produce la regola più promettente
- Ricerca greedy top-down o da generale a specifico attraverso lo spazio delle regole

Spazio di ricerca



- Ad ogni passo viene scelta il letterale che migliora maggiormente la performance della regola.

Misure di performance

- In APPRENDI_UNA_REGOLA si puo' usare
  - Accuratezza sul training set: sia n il numero di esempi coperti dalla regola c e n_c il numero degli esempi classificati correttamente (ovvero il numero di e+ coperti dalla regola). Più correttamente precision della regola

$$P(+|c)=n_c/n$$

Questa stima prende il nome di stima attraverso la frequenza relativa

Misure di performance

- Problemi nel caso di pochi esempi:
  - se n_c=0 P(+|c)=0 anche se n e' molto piccolo
  - se n=0 P(+|c) diventa 0/0 e non si puo' calcolare
- Cestnik ha mostrato che in questi casi la stima attraverso la frequenza relativa non e' affidabile ed e' meglio usare la stima di Laplace: nel caso di due classi

$$P(+\mid c) = \frac{n_c + 1}{n + 2}$$

  - nel caso di N classi

$$P(+\mid c) = \frac{n_c + 1}{n + N}$$

Misure di performance

- La stima di Laplace assume che le 2 (o N) classi siano uniformemente distribuite nel training set.
- Se questo non e' vero bisogna usare la stima-m: sia p la probabilita' che un esempio estratto a caso dal training set abbia la classificazione assegnata dalla regole e m un peso

$$P(+ \mid c) = \frac{n_c + mp}{n + m}$$

  - Se m=0 si ottiene l'accuratezza
  - Se m=N, p=1/N si ottiene la stima di Laplace

Misure di performance

- Per grandi valori di n_c ed n la stima di Laplace e la stima-m si avvicinano alla frequenza relativa

Ricerca beam

- La ricerca da generale a specifico e' generalmente una ricerca hill-climbing senza backtracking
- Pericolo di una scelta subottimale ad ogni passo
- Per ridurre il rischio viene a volte effettuata una ricerca beam
- Ad ogni passo, viene mantenuta una lista dei migliori k candidati, invece del singolo miglior candidato
- Sia CN2 [Clark&Niblett, 1989] che AQ [Michalski, 1969, 1986] compiono una ricerca beam top-down. AQ e' basato su un esempio seme per guidare la ricerca di regole.

Apprendimento di piu' concetti

- Caso in cui l'attributo target non e' booleano
- Non si apprendono piu' solamente regole che hanno nel conseguente Vero, ma si apprende un insieme di regole con diversi conseguenti
- Due approcci:
  - Lista ordinata di regole: la prima regola che fa match con l'esempio da classificare fornisce la classe. L'ultima regola e' una regola di default (CN2)
  - Lista non ordinata di regole: la classificazione e' data dalla classe piu' comune tra gli esempi di training coperti dalle regole che fanno match con l'esempio. Se non fa match con nessuna, la classificazione e' data dalla classe piu' frequente tra gli esempi di training (AQ)

Algoritmo di apprendimento sequenziale di CN2

- Si differenzia dall'algoritmo per l'apprendimento sequenziale visto prima perche':
  - E' in grado di imparare regole per diversi concetti
  - La procedura APPRENDI_UNA_REGOLA non restituisce una regola, ma solo un antecedente, il conseguente e' aggiunto successivamente

Algoritmo di apprendimento sequenziale di CN2

CN2(Attributo_target,Attributi,Esempi)

Regole_apprese←∅

Ripeti

Best_ant ←APPRENDI_UN_ANTECEDENTE (Attributo_target, Attributi, Esempi,k)

Sia E' l'insieme di esempi coperti da Best_ant

Esempi ←Esempi-E'

Sia C la classe piu' comune tra gli esempi di E'

Aggiungi la regola 'if Best_ant then class=C' alla fine di Regole_apprese

Finche' Best_ant e' nullo o Esempi e' vuoto

Aggiungi alla fine di Regole_apprese la regola di default 'class=C' dove C e' la classe piu' frequente in Esempi

restituisci le Regole_apprese

APPRENDI_UN_ANTECEDENTE (1)

APPRENDI_UN_ANTECEDENTE(Attributo_target,Attributi, Esempi,k)

Beam←{∅}

Best_ant ← ∅

Finche' Beam non e' vuoto fai

Specializza tutti gli antecedenti di Beam:

All_constraints ←insieme di tutti i vincoli della forma a=v

Nuovo_beam ←per ciascun h in Beam, per ciascun c in All_constraints: crea una specializzazione di h aggiungendo c

Rimuovi da Nuovo_beam tutte le ipotesi che sono duplicate, inconsistenti o che erano gia' in Beam

APPRENDI_UN_ANTECEDENTE (2)

Aggiorna Best_ant:

Per ciasun antecedente h in Nuovo_beam

se Performance(h,Esempi,Attributo_target)> Performance(Best_ant,Esempi,Attributo_target) then Best_ant ←h

Aggiorna Beam:

Beam ← i migliori k membri di Nuovo_beam secondo la misura di Performance.

Fine ciclo

Ritorna Best_ant

CN2

- Dato un concetto c, gli esempi positivi per gli altri concetti sono esempi negativi per c
- In questo caso, togliendo gli esempi di un concetto c nel ciclo di covering si rimuovono anche degli esempi negativi per gli altri concetti
- Rischio di generare regole troppo generali per gli altri concetti
- Questo problema e' risolto dal fatto che le regole sono valutate in ordine. Quindi, anche se le ultime regole sono molto generali questo non e' un problema perche' le regole precedenti intercettano gli esempi.

Misure di performance

- In APPRENDI_UN_ANTECEDENTE si usa
  - Entropia

$$info(S) = -\sum_{j=1}^{k} \frac{freq(c_j, S)}{|S|} \times \log_2 \left(\frac{freq(c_j, S)}{|S|} \right)$$

- Dove S e' l'insieme degli esempi coperti dall'antecedente e c_j sono le k classi
- Di solito si considera –info(S) in modo che le clausole migliori abbiamo i valori piu' alti

Esempio

No	Outlook	Temp	Humid	Windy	Class
D1	sunny	mild	normal	T	P
D2	sunny	hot	high	T	N
D3	sunny	hot	high	F	N
D4	sunny	mild	high	F	N
D5	sunny	cool	normal	F	P
D6	overcast	mild	high	T	P
D7	overcast	hot	high	F	P
D8	overcast	cool	normal	T	P
D9	overcast	hot	normal	F	P
D10	rain	mild	high	T	N
D11	rain	cool	normal	T	N
D12	rain	mild	normal	F	P
D13	rain	cool	normal	F	P
D14	rain	mild	high	F	P

Esempio

- Apprendi un antecedente:

Beam←{∅}

Best_ant ← ∅

Specializza gli antecedenti di Beam:

Nuovo_beam ←{outlook=sunny, outlook=overcast, outlook=rain, temp=hot, temp=mild, temp=cool, humid=high, humidity=normal, windy=t, windy=f}

Esempio

Calcola la performance di ciascun antecedente

outlook=sunny	-info(S)=$2/5\log_2 2/5+3/5\log_2 3/5$=-0.97
outlook=overcast	-info(S)=0
outlook=rain	-info(S)=-0.97
temp=hot	-info(S)=-1
temp=mild	-info(S)=-0.92
temp=cool	-info(S)=-0.81
humid=high	-info(S)=-0.99
humid=normal	-info(S)=-0.59
wind=t,	-info(S)=-1
wind=f	-info(S)=-0.81

Esempio

- Best_ant ← outlook=overcast
- Aggiorna Beam (dimensione=2):
- Beam ← {outlook=overcast, humid=normal }
- Nuova iterazione:
- Nuovo_beam ←{

outlook=overcast and temp=hot,

outlook=overcast and temp=mild,

outlook=overcast and temp=cool,

outlook=overcast and humid=high,

outlook=overcast and humid=normal,

outlook=overcast and windy=t,

outlook=overcast and windy=f,

Esempio

humid=normal and outlook=sunny,
humid=normal and outlook=overcast,
humid=normal and outlook=rain,
humid=normal and temp=hot,
humid=normal and temp=mild,
humid=normal and temp=cool,
humid=normal and windy=t,
humid=normal and windy=f}

Esempio

- Calcola la performance di ciascun antecedente

outlook=overcast and temp=hot, -info(S)=0

outlook=overcast and temp=mild, -info(S)=0

outlook=overcast and temp=cool, -info(S)=0

outlook=overcast and humid=high, -info(S)=0

outlook=overcast and humid=normal, -info(S)=0

outlook=overcast and windy=t, -info(S)=0

outlook=overcast and windy=f, -info(S)=0

Esempio

humid=normal and outlook=sunny,	-info(S)=0
humid=normal and outlook=overcast,	-info(S)=0
humid=normal and outlook=rain,	-info(S)=-0.92
humid=normal and temp=hot,	-info(S)=0
humid=normal and temp=mild,	-info(S)=0
humid=normal and temp=cool,	-info(S)=-0.81
humid=normal and windy=t,	-info(S)=-0.92
humid=normal and windy=f	-info(S)=0

Esempio

- Best_ant rimane outlook=overcast
- Aggiorna Beam (dimensione=2):
- Beam ← {outlook=overcast and temp=hot,

outlook=overcast and temp=mild }

Esempio

- Nuova iterazione:
- Nuovo_beam ←{

outlook=overcast and temp=hot and humid=normal,
outlook=overcast and temp=hot and humid=high,
outlook=overcast and temp=hot and windy=t,
outlook=overcast and temp=hot and windy=f,
outlook=overcast and temp=mild and humid=normal,
outlook=overcast and temp=mild and humid=high,
outlook=overcast and temp=mild and windy=t,
outlook=overcast and temp=mild and windy=f}

Esempio

- Calcola la performance di ciascun antecedente

outlook=overcast and temp=hot and humid=normal	0
outlook=overcast and temp=hot and humid=high	0
outlook=overcast and temp=hot and windy=t	0
outlook=overcast and temp=hot and windy=f	0
outlook=overcast and temp=mild and humid=normal	0
outlook=overcast and temp=mild and humid=high	0
outlook=overcast and temp=mild and windy=t	0
outlook=overcast and temp=mild and windy=f	0

Esempio

- Best_ant rimane outlook=overcast
- Aggiorna Beam (dimensione=2):
- Beam ← {

outlook=overcast and temp=hot and humid=normal,

outlook=overcast and temp=hot and humid=high}

- Nuova iterazione:
- Nuovo_beam ←{

outlook=overcast and temp=hot and humid=normal and windy=t,

outlook=overcast and temp=hot and humid=normal and windy=f,

outlook=overcast and temp=hot and humid=high and windy=t,

outlook=overcast and temp=hot and humid=high and windy=t}

Esempio

- Calcola la performance di ciascun antecedente

outlook=overcast and temp=hot and humid=normal and windy=t 0
outlook=overcast and temp=hot and humid=normal and windy=f 0
outlook=overcast and temp=hot and humid=high and windy=t 0
outlook=overcast and temp=hot and humid=high and windy=t 0

- Best_ant rimane outlook=overcast
- Aggiorna Beam (dimensione=2):
- Beam ← {

outlook=overcast and temp=hot and humid=normal and windy=t,
outlook=overcast and temp=hot and humid=normal and windy=f

}

Esempio

- Nuova iterazione:
- Nuovo_beam $\leftarrow \varnothing$
- Fine del ciclo
- Restituisci Best_ant=outlook=overcast
- Nel ciclo esterno:
  - Gli es coperti da outlook=overcast sono D6, D7, D8 e D9
  - Rimuovi da E questi esempi
  - Aggiungi la regola ‘if outlook=overcast then class=P’ alla teoria
  - Esegui un’altra iterazione del ciclo esterno
  - .....

Apprendimento di piu' concetti da parte di AQ

- Si apprende un concetto alla volta usando come esempi negativi gli esempi positivi per gli altri concetti usando l'algoritmo APPRENDIMENTO_SEQUENZIALE
  - Dopo aver appreso le regole per un concetto, l'insieme di esempi per il concetto viene riportato allo stato iniziale (tutti gli esempi)
- L'unione degli insiemi di regole appresi per ciascun concetto forma l'output del sistema di apprendimento

Determinazione delle regole di produzione da un albero decisionale

- Da un albero decisionale puo' essere generato un insieme di regole, una per ogni foglia, componendo tutti i test dalla radice dell'albero alle foglie
- Possibili successivi passi di semplificazione

Esempio

- Dall'albero

Outlook=sunny
    | humidity ≤ 75: P
    | humidity > 75: N
Outlook=overcast: P
Outlook=rain
    | windy=True: N
    | windy=False: P

- Si estraggono le regole

If Outlook=sunny and humidity ≤ 75 then Class=P

If Outlook=sunny and humidity > 75 then Class=N

if Outlook=overcast then Class=P

if Outlook=rain and windy=True then Class=N

if Outlook=rain and windy=False then Class=P

Copertura sequenziale simultanea vs sequenziale

Gli algoritmi di apprendimento di alberi di decisione possono essere considerati algoritmi di copertura simultanea, in contrasto con gli algoritmi di copertura sequenziale.

- Gli algoritmi di copertura sequenziale sono più lenti.
- Per imparare un insieme di n regole, ciascuna contenente k test attributo-valore nelle loro precondizioni, gli algoritmi di copertura sequenziale devono compiere n*k passi di ricerca primitivi (scelte indipendenti)

Copertura sequenziale simultanea vs sequenziale

- Gli algoritmi di copertura simultanea sono generalmente piu' veloci, dato che ogni test con m possibili risultati contribuisce alla scelta delle precondizioni per almeno m regole
- Se ogni test ha m risultati e ho n regole, la profondita' dell'albero sara' $\log_m(n)$ e le regole hanno ciascuna $\log_m(n)$ test. Il numero di test da scegliere sara' quindi:

$$\sum_{i=0}^{(\log_m n)-1} m^i = \frac{m^{\log_m(n)} - 1}{m-1} = \frac{n-1}{m-1} \rightarrow_{n \rightarrow \infty} \frac{n}{m-1}$$

- Con la copertura sequenziale: n* $\log_m(n)$ scelte

Copertura sequenziale simultanea vs sequenziale

- Gli algoritmi di copertura sequenziale compiono un grande numero di scelte indipendenti, mentre gli algoritmi di copertura simultanea compiono un numero ristretto di scelte dipendenti.

Apprendere regole direttamente o convertire un albero in regole?

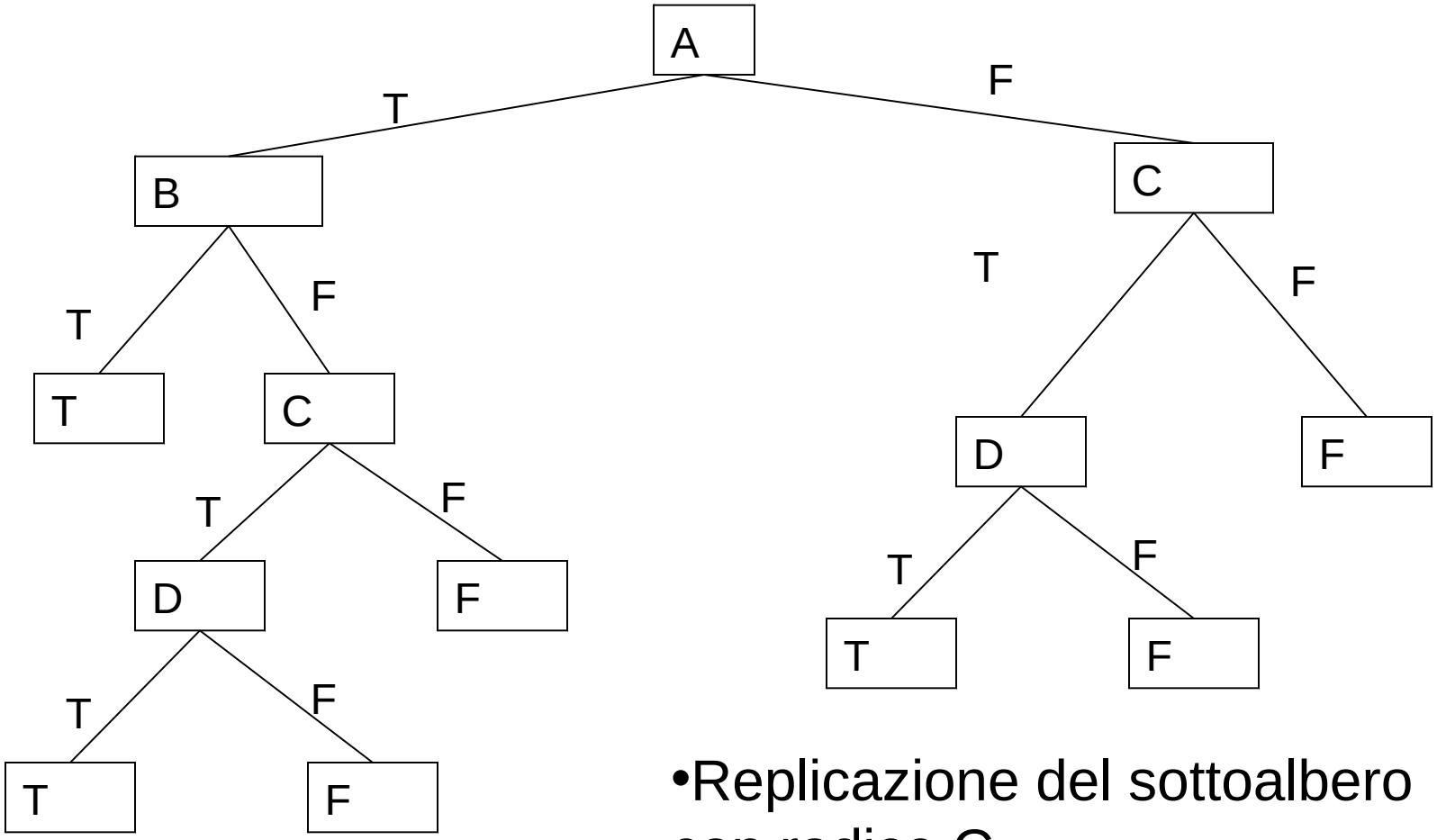
- A seconda della quantità di dati a disposizione:
  - Se si hanno molti dati, si possono usare gli algoritmi di copertura sequenziale perche' i dati possono supportare le molteplici scelte indipendenti
  - Se i dati sono scarsi, sono meglio gli algoritmi di copertura simultanea in quanto vengono usati tutti i dati nelle scelte

Apprendere regole direttamente o convertire un albero in regole?

- A seconda del tipo di concetto da apprendere
  - Se il concetto è altamente disgiuntivo con molte condizioni indipendenti, l'apprendimento di alberi ha basse prestazioni nel caso di dati scarsi

Concetti disgiuntivi rappresentati con alberi di decisione

- Albero che rappresenta il concetto AB $\vee$ CD



- Replicazione del sottoalbero con radice C

Concetti digiuntivi rappresentati mediante regole

- Concetto AB $\vee$ CD:
  - If A=T and B=T then class=T
  - If C=T and D=T then class=T
  - class=F

Apprendimento di regole con c4.5

- c4.5 e' in grado di apprendere anche regole (con il programma c4.5rules)
- Non converte semplicemente un albero in regole ma effettua una serie di elaborazioni
  - Generalizzazione delle regole singole
  - Selezione delle regole
  - Ordinamento delle regole
  - Scelta della regola di default
  - Seconda selezione di regole
- L'obiettivo di queste elaborazioni e' quello di ottenere un classificatore che sia piu' comprensibile per gli esseri umani dell'albero e che sia altrettanto accurato sui casi non visti

Generalizzazione delle regole

- Ad esempio la piu' profonda foglia T dell'albero precedente corrisponde alla regola
  - if A=T and B=F and C=T and D=T then class=T
- L'antecedente delle regole puo' contenere condizioni irrilevanti
  - Ad esempio, rimuovendo le condizioni A=T, B=F si ottiene una regola con la stessa accuratezza (1)

Uso di c4.5

- Comando per estrarre le regole dai dati:

c4.5rules –f labor-neg –u

dove l'opzione –f serve ad indicare il nome dell'insieme di file da esaminare e –u significa che le regole prodotte devono essere testate sui casi del file labor-neg.test

Output: regole di produzione

```
C4.5 [release 8] rule generator Sat May 18 17:05:42 1996
-------------------------------

    Options:
        File stem <labor-neg>
        Rulesets evaluated on unseen cases

Read 40 cases (16 attributes) from labor-neg

------------------
Processing tree 0

Final rules from tree 0:

Rule 6:
        wage increase first year > 2.5
        statutory holidays > 10
        ->  class good  [93.0%]

Rule 5:
        wage increase first year > 4
        ->  class good  [90.6%]
```

Output: regole di produzione

```
Rule 4:
        wage increase first year <= 4
        statutory holidays <= 10
        ->  class bad  [87.1%]

Rule 3:
        wage increase first year <= 2.5
        working hours > 38
        ->  class bad  [79.4%]

Default class: good
```

Valutazione delle regole

```
Evaluation on training data (40 items):

Rule  Size  Error  Used  Wrong            Advantage
----  ----  -----  ----  -----            ---------
   6     2   7.0%    19      0 (0.0%)       0 (0|0)    good
   5     1   9.4%     3      0 (0.0%)       0 (0|0)    good
   4     2  12.9%    10      0 (0.0%)       7 (7|0)    bad
   3     2  20.6%     3      0 (0.0%)       3 (3|0)    bad

Tested 40, errors 1 (2.5%)   <<


          (a)   (b)      <-classified as
         ----  ----
           26           (a): class good
            1    13     (b): class bad
```

Valutazione delle regole (2)

```
Evaluation on test data (17 items):

Rule  Size  Error  Used  Wrong            Advantage
----  ----  -----  ----  -----            ---------
   6     2   7.0%     9      1 (11.1%)       0 (0|0)    good
   5     1   9.4%     3      1 (33.3%)       0 (0|0)    good
   4     2  12.9%     1      0 (0.0%)        1 (1|0)    bad
   3     2  20.6%     2      0 (0.0%)        2 (2|0)    bad

Tested 17, errors 3 (17.6%)   <<


          (a)   (b)      <-classified as
         ----  ----
           11           (a): class good
            3     3     (b): class bad
```

Interpretazione dell'output

- Errror:stima pessimistica dell'errore
- Advantage **A (B|C)**
  - **B**: casi classificati bene dalla regola che verrebbero classificati scorrettamente se la regola non fosse presente
  - **C**: casi classificati male dalla regola che verrebbero classificati correttamente se la regola non fosse presente
  - **A = B-C**: vantaggio fornito dall'includere la regola nell'insieme di regole.

Confronto tra algoritmi per apprendere regole

- Ricerca da specifico a generale o da generale a specifico?
  - Vantaggio di da generale a specifico: c'è un sola ipotesi massimamente generale, molte ipotesi massimamente specifiche
- Ricerca generate and test (CN2, c4.5) oppure example-driven (Find-S, Candidate-Elimination, AQ)?
  - Vantaggio di generate and test: ogni scelta nella ricerca è basata su molti esempi, quindi l'impatto di un esempio rumoroso è basso

Confronto tra algoritmi per apprendere regole

- Quando e come fare post-pruning?
  - Ad esempio: potare ciascuna regola dopo l'apprendimento rimuovendo precondizioni quando questo porti ad un miglioramento della performance su un insieme di esempi distinti da quelli di training
- Quali misure di performance utilizzare?

Software

- CN2 è scaricabile da

http://www.cs.utexas.edu/users/pclark/software.html#cn2

- C4.5 è scaricabile da

http://www.rulequest.com/Personal/

Bibliografia

Clark, P. e Niblett, T. *The CN2 induction algorithm*, Machine Learning 3(4), 1989.

Cervone G., Panait, L.A. and Michalski R.S., *The Development of the AQ20 Learning System and Initial Experiments*, Tenth International Symposium on Intelligent Information Systems, Zakopane, Poland, giugno, 2001. http://www.mli.gmu.edu/papers/IIS-01.ps

Rissanen, J. *A universal prior for integers and estimation by minimum description length*, Annals of Statistics 11(2), 416-431, 1983.