Classificatori Rule-Based

Prof. Matteo Golfarelli
Alma Mater Studiorum - Università di Bologna

Classificatori basati su regole

- Classificano i record utilizzando insiemi di regole del tipo "if…then…"
- Una regola ha la forma: (*Condizione*) $\rightarrow$ *y*
  - ✓ dove
    - *Condizione* è una congiunzione di predicati su attributi
    - *y* è l'etichetta di classe
  - ✓ *LHS*: antecedente o condizione
  - ✓ *RHS*: conseguente o etichetta della classe
  - ✓ Esempi di regole:
    - (Blood Type=Warm) $\wedge$ (Lay Eggs=Yes) $\rightarrow$ Birds
    - (Taxable Income < 50K) $\wedge$ (Refund=Yes) $\rightarrow$ Evade=No

Costruire un modello significa identificare un insieme di regole

Vantaggi e svantaggi

- Espressivi quanto un Decision Tree
- Facili da interpretare
- Facili da generare
- Rapidi nella classificazione di nuove istanze

- Il costo di costruzione non scala al crescere del training set
- Risentono fortemente del rumore sui dati

Classificatori basati su regole

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

r1: (Give Birth = no) $\wedge$ (Can Fly = yes) $\rightarrow$ Birds

r2: (Give Birth = no) $\wedge$ (Live in Water = yes) $\rightarrow$ Fishes

r3: (Give Birth = yes) $\wedge$ (Blood Type = warm) $\rightarrow$ Mammals

r4: (Give Birth = no) $\wedge$ (Can Fly = no) $\rightarrow$ Reptiles

r5: (Live in Water = sometimes) $\rightarrow$ Amphibians

Classificatori basati su regole

- Una regola *r* copre un'istanza **x** se i valori di **x** soddisfano l'antecedente di *r*

r1: (Give Birth = no) $\wedge$ (Can Fly = yes) $\rightarrow$ Birds

r2: (Give Birth = no) $\wedge$ (Live in Water = yes) $\rightarrow$ Fishes

r3: (Give Birth = yes) $\wedge$ (Blood Type = warm) $\rightarrow$ Mammals

r4: (Give Birth = no) $\wedge$ (Can Fly = no) $\rightarrow$ Reptiles

r5: (Live in Water = sometimes) $\rightarrow$ Amphibians

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

La regola *r1* copre hawk => Bird

La regola *r3* copre grizzly bear => Mammal

Copertura e accuratezza

- Dato un dataset D e una regola di classificazione A→y definiamo
- Copertura:
  - ✓ Frazione dei record che soddisfano l'antecedente della regola

$$Coverage(r) = \frac{|A|}{|D|}$$

- Accuratezza:
  - ✓ Frazione dei record che, soddisfando l'antecedente, soddisfano anche il conseguente della regola

$$Accuracy(r) = \frac{|A \cap y|}{|A|}$$

RTid	Refund	Marital Status	Taxable Income	Class
1	Yes	**Single**	125K	**No**
2	No	Married	100K	**No**
3	No	**Single**	70K	**No**
4	Yes	Married	120K	**No**
5	No	Divorced	95K	**Yes**
6	No	Married	60K	**No**
7	Yes	Divorced	220K	**No**
8	No	**Single**	85K	**Yes**
9	No	Married	75K	**No**
10	No	**Single**	90K	**Yes**

r = (*Status*=Single) → No

Coverage(r) = 40%, *Accuracy*(r) = 50%

Come funziona un classificatore basato su regole?

r1: (Give Birth = no) $\wedge$ (Can Fly = yes) $\rightarrow$ Birds

r2: (Give Birth = no) $\wedge$ (Live in Water = yes) $\rightarrow$ Fishes

r3: (Give Birth = yes) $\wedge$ (Blood Type = warm) $\rightarrow$ Mammals

r4: (Give Birth = no) $\wedge$ (Can Fly = no) $\rightarrow$ Reptiles

r5: (Live in Water = sometimes) $\rightarrow$ Amphibians

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

Un lemure attiva la regola *r3*, quindi è classificato come mammifero

Una tartaruga attiva sia *r4* sia *r5* che determinano classi diverse!

Uno squalo non attiva nessuna regola!!

Proprietà dei classificatori basati su regole

- Regole mutuamente esclusive: un insieme di regole R è detto mutuamente esclusivo se nessuna coppia di regole può essere attivata dallo stesso record
  - ✓ Ogni record è coperto da al più una regola

- Regole esaustive: un insieme di regole R ha una copertura esaustiva se esiste una regola per ogni combinazione di valori degli attributi
  - ✓ Ogni record è coperto da almeno una regola

Proprietà dei classificatori basati su regole

- Non sempre è possibile determinare un insieme di regole esaustive e mutualmente esclusive
- Mancanza di mutua esclusività
  - ✓ Un record può attivare più regole dando vita a classificazioni discordanti
  - ✓ Soluzione
    - Definire un ordine di attivazione delle regole. Si parla in questo caso di liste di decisione
    - Assegnare il record alla classe per la quale vengono attivate più regole (voto)
- Mancanza di esaustività
  - ✓ Un record può non attivare nessuna regola
  - ✓ Soluzione
    - Utilizzare una classe di default (“altro”) a cui il record viene associato in caso di non attivazione delle regole

Modalità di ordinamento

- Ordinamento Rule-based
  - ✓ Le singole regole sono ordinate in base alla loro qualità
- Ordinamento Class-based
  - ✓ Gruppi di regole che determinano la stessa classe compaiono di seguito nella lista. L'ordinamento rilevante diventa quello tra le classi che può dipendere dall'importanza della classe o dalla gravità di commettere un errore di classificazione per quella classe
  - ✓ Il rischio con questa soluzione è che una regola di buona qualità sia superata da una regola di qualità inferiore ma appartenente a una classe importante
  - ✓ Soluzione utilizzata dai principali algoritmi di costruzione delle regole(RIPPER, C4.5rules)

Rule-based Ordering

(Refund=Yes) ==> No

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

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

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

Class-based Ordering

(Refund=Yes) ==> No

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

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

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

Esercizio

- Attributi e valori
  - ✓ Air Conditioner = {*Working, Broken*}
  - ✓ Engine = {*Good, Bad*}
  - ✓ Mileage = {*High, Medium, Low*}
  - ✓ Rust = {*Yes, No*}
- Insieme di regole
  - ✓ Mileage = High → *Value = Low*
  - ✓ Mileage = Low → *Value = High*
  - ✓ Air Conditioner = Working, Engine = Good → *Value = High*
  - ✓ Air Conditioner = Working, Engine = Bad → *Value = Low*
  - ✓ Air Conditioner = Broken → *Value = Low*
- *Quesiti*
  - ✓ *Le regole sono esaustive?*
  - ✓ *Le regole sono mutualmente esclusive?*
  - ✓ *E' necessario un ordinamento?*
  - ✓ *E' necessaria una classe di default?*

Costruzione delle regole

- Metodi diretti
  - ✓ Estraggono le regole direttamente dai dati (es. RIPPER, CN2, Holte's 1R)
- Metodi indiretti
  - ✓ Estraggono le regole dal risultato di altri metodi di classificazione come per esempio i decision tree (C4.5rules)

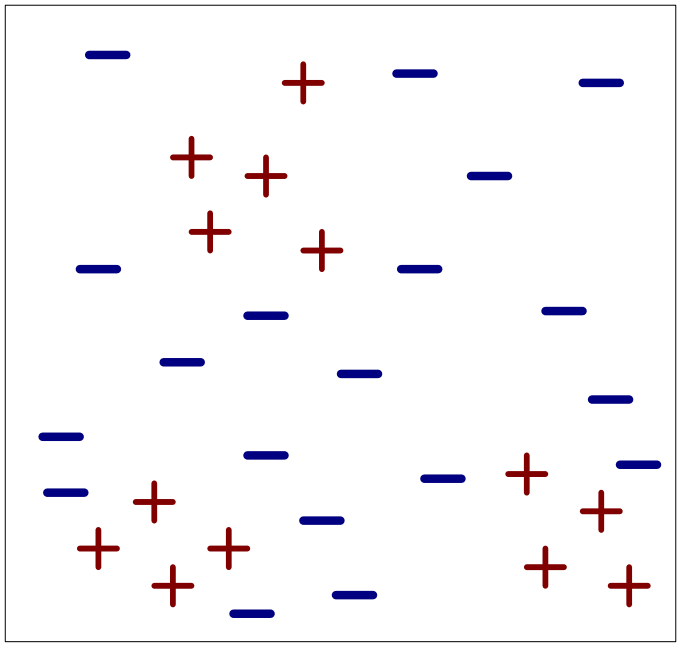
Metodi diretti: sequential covering

Let E be the training set, $A=\{(A_j, v_j)\}$ the set of attribute-value pairs and $Y_0=\{y_1 \ldots y_k\}$ the set of classes where y_k is the default class

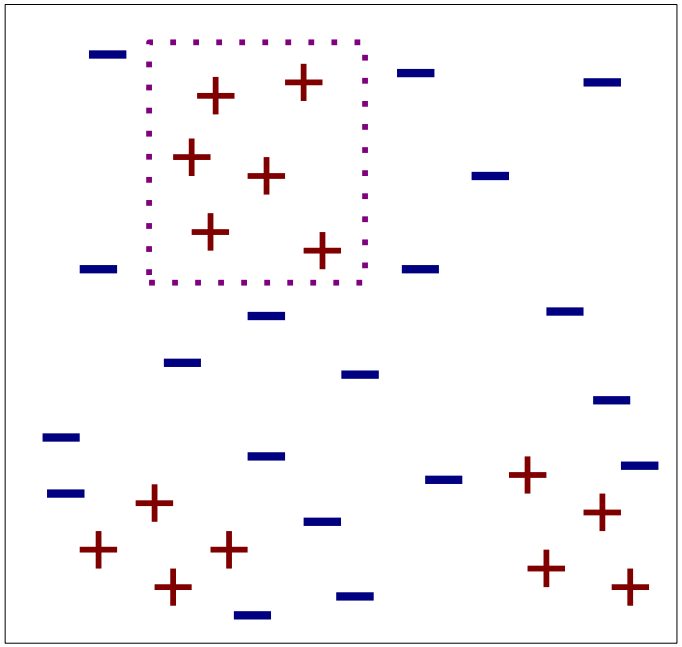
```
set R = ∅
for each class y ∈ Y0 – {yk} do
 stop=FALSE;
  while !stop do
    r =  Learn-One-Rule(E,A,y)
    remove from E training records that are covered by r
    If Quality(r,E) < Threshold then
      stop=TRUE;
    else
      R = R ∪ r // Add r at the bottom of the rule list
   end while
end for
R = R ∪ {{}->yk}// Add the default rule at the bottom of the rule list
PostPruning(R);
```

- L'ordinamento delle regole è determinato da quello delle classi
- Punti di attenzione
  - ✓ Modalità di creazione delle regole (Learn-One-Rule)
  - ✓ Eliminazione delle istanze
  - ✓ Criterio di stop

Sequential covering: esempio



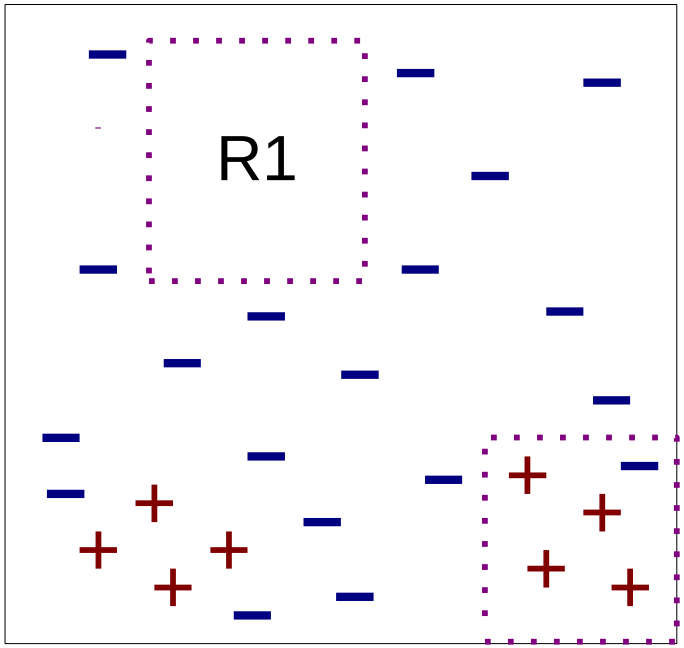
(i) Original Data



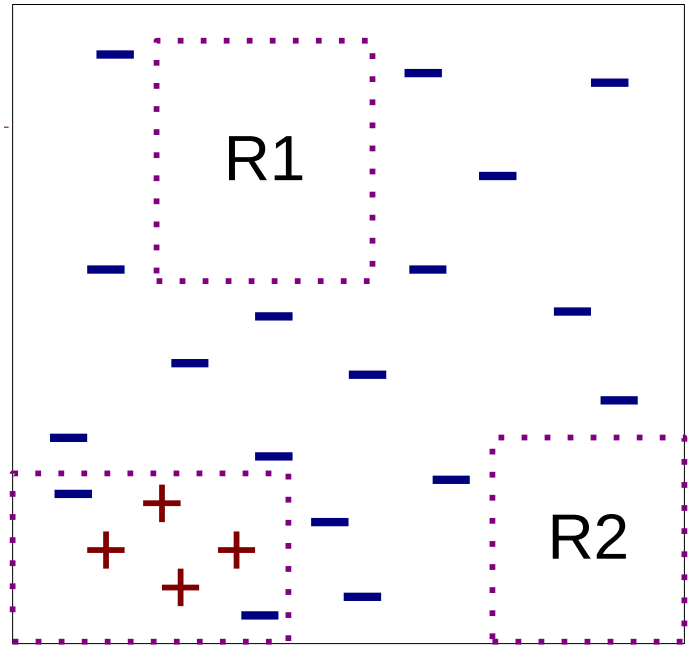
(ii) Step 1

- Chiameremo esempi positivi (+) tutte le tuple del training set che appartengono alla classe y. Sono esempi negativi (-) tutti gli altri.

Sequential covering: esempio



(iii) Step 2



(iv) Step 3

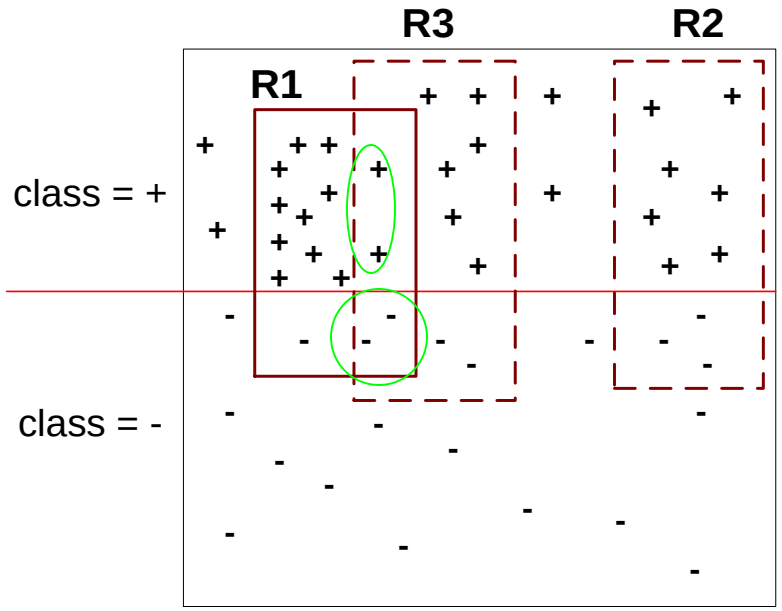
Eliminazione delle istanze dal training set

- L'eliminazione delle istanze dal training set serve a:
  - ✓ Istanze classificate correttamente: evitare di generare sempre la stessa regola, evitare di sovrastimare l'accuracy della prossima regola
  - ✓ Istanze classificate non correttamente: a evitare di sottostimare l'accuracy della prossima regola
    - Le istanze negative conteggiate per la regola 1 non devono essere conteggiate anche per la regola 3

Accuracy(R1)=12/15 (80%)
Accuracy(R2)=7/10 (70%)
Accuracy(R3)=8/12 (66.7%)

Dopo aver scelto R1

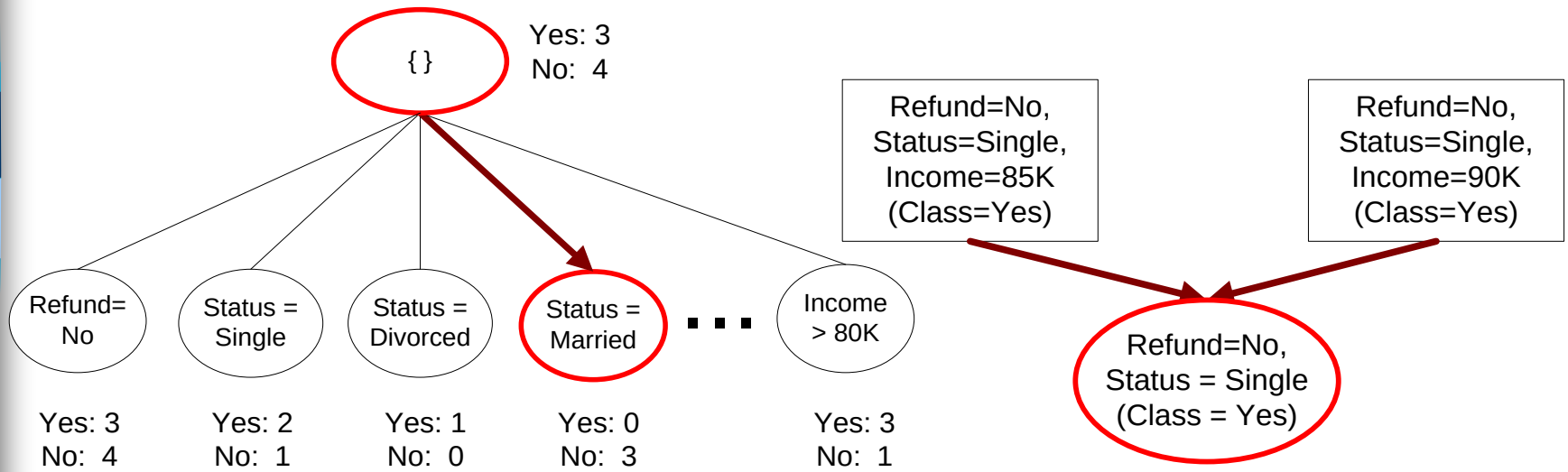
Accuracy(R2)=7/10 (70%)
Accuracy(R3)=6/8 (75%)



Learn-One-Rule

- L'obiettivo dell'algoritmo è trovare una regola che copra quanti più possibili esempi positivi e quanto meno possibili (o nessun) esempio negativo
  - ✓ Il problema è complesso dato che lo spazio di ricerca è esponenziale nel numero di possibili combinazioni dei predicati

- Le regole sono costruite considerando progressivamente un nuovo possibile predicato ossia una possibile coppia (Attributo,Valore)



(a) General-to-specific

(b) Specific-to-general

Learn-One-Rule

- E' necessario un criterio per scegliere quale predicato aggiungere
  - *n :* numero di istanze coperte dalla regola
  - n_r *:* numero di istanze correttamente classificate dalla regola
  - *k* : numero di classi

✓ Accuratezza: limitata applicabilità poiché non considera la coverage della regola

$$Accuracy(r) = \frac{n_r}{n}$$

- *r1*: copre 50 esempi positivi e 5 esempi negativi → Accuracy(*r1*) = 90.9%
- *r2*: copre 2 esempi positivi e 0 esempi negativi → Accuracy(*r2*) = 100%

✓ Metriche che considerano la coverage: pesano l'accuracy in base alla coverage

$$Laplace(r) = \frac{n_r + 1}{n + k}$$

- Se la coverage è 0 (n_r=n=0) e assumendo distribuzione uniforme dei dati , l'indice si riduce alla probabilità a priori della classe (1/*k*)
- Se la coverage è alta l'indice tende asintoticamente al valore dell'accuracy $\frac{n_r}{n}$
- Per valori bassi di coverage *k* tende a ridurre il valore dell'indice

Learn-One-Rule

- ✓ Metriche che considerano il supporto della regola ossia il numero di esempi positivi coperti dalla regola

$$FoilGain(r, r') = t\left(\log_2 \frac{p'}{p'+n'} - \log_2 \frac{p}{p+n}\right)$$

- *r* / *r'*: regola prima / dopo l'inserimento del nuovo predicato
- *t*: numero di esempi positivi coperti sia da *r* sia da *r'*
- *p* / *p'* : numero di esempi positivi coperti da *r* / *r'*
- *n* / *n'* : numero di esempi negativi coperti da *r* / *r'*
- L'indice è proporzionale a *t* e a (*p'*)/(*p'*+*n'*) quindi tende a favorire regole che hanno elevato coverage e accuracy

Training set composto da 100 esempi positivi e 400 negativi
R1: A → + (copertura: 4 esempi positivi e 1 negativo)
R2: A → + (copertura: 30 esempi positivi e 10 negativi)
R3: A → + (copertura: 100 esempi positivi e 90 negativi)

Determinare quale è la regola migliore sulla base dell'accuracy e del FoilGain

Learn-One-Rule

- Criterio di stop nell'estensione di una regola
  - ✓ Calcola il gain
  - ✓ Se il gain non è significativo stop!

- Rule Pruning: ha l'obiettivo di semplificare le regole per migliorare l'errore di generalizzazione delle regole, può essere utile visto che l'approccio di costruzione è greedy
  - ✓ Possono essere utilizzate le tecniche viste per gli alberi decisionali
    - Minimum Description Length
    - Approccio pessimistico
    - Utilizzo del validation set
  - ✓ Esempio (Reduced error pruning):
    - Rimuovi a turno uno dei predicati dalla regola
    - Elimina il predicato la cui rimozione comporta il massimo miglioramento dell'error rate sul validation set
    - Ripeti i passi precedenti sino a che continuano a determinare un miglioramento

Metodi diretti: RIPPER

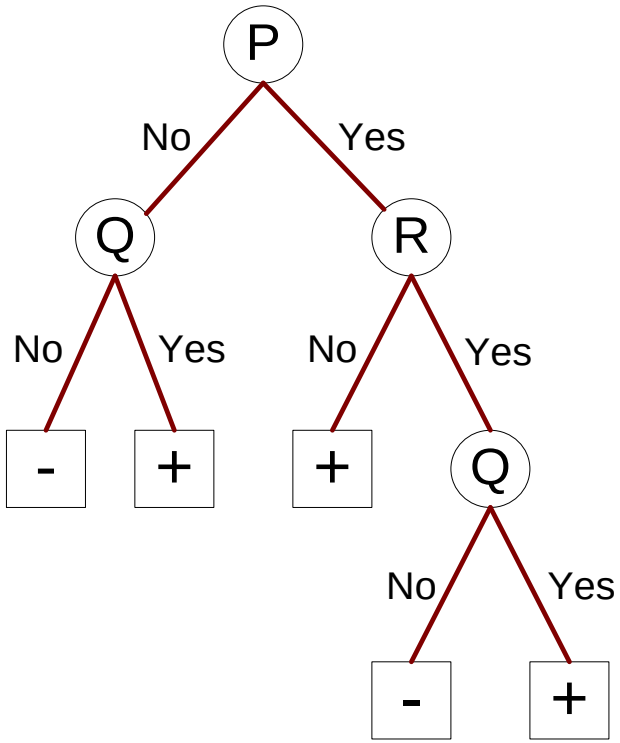
- Approccio basato sul sequential covering
  - ✓ Una sua versione denominata **JRip** è implementata in WEKA
- Per problemi a 2 classi, scegli una delle classi come classe positiva e l'altra come classe negativa
  - ✓ Impara le regole per la classe positiva
  - ✓ La classe negativa sarà la classe di default
- Per problemi multi-classe
  - ✓ Ordina le classi in base a un criterio di rilevanza della classe (frazione delle istanze che appartengono alla classe)
  - ✓ Impara le regole a partire dalla classe più piccola e considerando il resto delle istanze come classe negativa
  - ✓ Ripeti l'operazione con le classi successive
- Dopo l'aggiunta di una regola calcola il description length
  - ✓ Interrompi la costruzione di nuove regole quando il nuovo valore di description length è superiore di d bits rispetto a quello precedente

Metodi diretti: RIPPER

- Learn-one-rule:
  - ✓ Inizia con una regola vuota
  - ✓ Aggiungi predicati finchè determinano un miglioramento del FoilGain
  - ✓ Esegui il pruning utilizzando la tecnica dell'*incremental reduced error pruning*
    - Esegue il pruning dopo ogni passo di growing
  - ✓ Metodo di pruning: elimina dalla regola i predicati la cui rimozione massimizza: $v = (p-n)/(p+n)$
    - p: numero di esempi positivi coperti dalla regola nel validation set
    - n: numero di esempi negativi coperti dalla regola nel validation set

Metodi indiretti

- E' possibile trasformare un decision tree in un insieme di regole (vedi C4.5rules)
  - ✓ Ad ogni percorso radice-foglia corrisponderà una regola
  - ✓ Saranno poi applicati opportune tecniche di pruninig

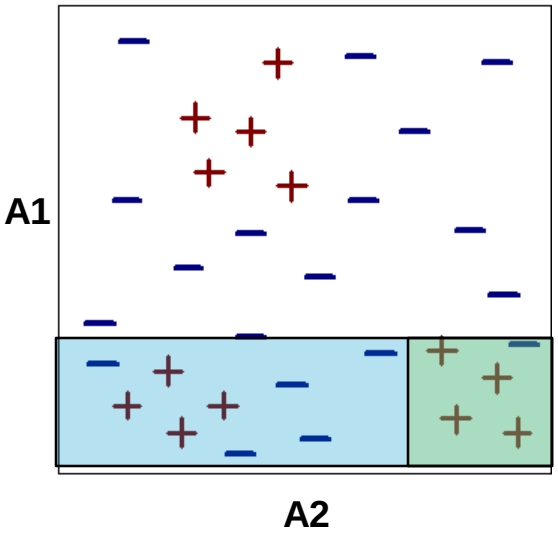


Rule Set

r1: (P=No,Q=No) ==> -
r2: (P=No,Q=Yes) ==> +
r3: (P=Yes,R=No) ==> +
r4: (P=Yes,R=Yes,Q=No) ==> -
r5: (P=Yes,R=Yes,Q=Yes) ==> +

Alberi decisionali vs regole

- Sebbene l'espressivatà delle due tecniche sia similare l'insieme delle regole prodotte è generalmente diverso
- La differenza fondamentale tra le due tecniche consiste nel fatto che nel procedimento di costruzione:
  - ✓ la bontà del criterio di split scelto in ogni nodo dell'albero considera la qualità di tutti i figli generati
  - ✓ il criterio con cui viene aggiunto un predicato alla regola valuta solo la bontà della classe che si viene a determinare



R1: A1<20

R1: A1<20 and A2>60