



Università degli Studi di Bari

DIPARTIMENTO DI INFORMATICA
Corso di Laurea Magistrale in Informatica

PROGETTO DI INTELLIGENZA ARTIFICIALE

CONFRONTO TRA ALGORITMI DI APPRENDIMENTO

Esaminando:

Giuseppe Rizzi

Matricola 591275

Docenti:

Prof. Floriana Esposito

Prof. Nicola Di Mauro

Indice

1	Introduzione	2
2	Descrizione dei dati	3
2.1	German Credit dataset	3
2.2	Hepatitis dataset	4
2.3	Vehicle Silhouettes dataset	5
2.4	Wisconsin Breast Cancer dataset	6
3	Naive Bayes	8
3.1	Teorema di Bayes	8
3.2	Naive Bayes Classifier	9
4	REPTree	12
4.1	Information Gain	12
4.2	Reduced Error Pruning	13
5	RIPPER	15
5.1	Incremental Reduced Error Pruning	16
5.2	Miglioramenti ad IREP	18
6	Esecuzione	21
6.1	Risultati su German Credit	21
6.2	Risultati su Hepatitis	30
6.3	Risultati su Vehicle Silhouettes	37
6.4	Risultati su Wisconsin Breast Cancer	45
7	Analisi	50
7.1	Test	50
7.2	Interpretazione dei risultati	52
8	Conclusioni	53

Capitolo 1: Introduzione

Il seguente lavoro si propone di confrontare tre algoritmi di apprendimento supervisionato, *Naive Bayes*, *REPTree* e *RIPPER*: il primo sfrutta un modello bayesiano, il secondo la metodologia di costruzione di alberi di decisione, il terzo quello di costruzione di regole.

Verranno testati su quattro dataset messi a disposizione dall'*UCI Machine Learning Repository*¹, procedendo con la presentazione dei risultati e dei modelli di predizione ottenuti.

Il software utilizzato è *Weka*², una suite di algoritmi di *machine learning*, fortemente utilizzato sia in ambito accademico che industriale.

¹<https://archive.ics.uci.edu/ml/datasets.html>

²<http://www.cs.waikato.ac.nz/ml/weka/>

Capitolo 2: Descrizione dei dati

Di seguito vengono descritti i 4 dataset utilizzati nella .

2.1 German Credit dataset

Il dataset contiene informazioni in ambito finanziario su clienti ritenuti a rischio o meno.

- Numero di istanze: 1000
- Numero di attributi: 21
- Attributo target: **class**
- Valori target: {good, bad}

Attributo	Tipo
checking_status	nominal
duration	numeric
credit_history	nominal
purpose	nominal
credit_amount	numeric
savings_status	nominal
employment	nominal
installment_commitment	numeric
personal_status	nominal
other_parties	nominal
residence_since	numeric
property_magnitude	nominal
age	numeric
other_payment_plans	nominal
housing	nominal
existing_credits	numeric
job	nominal
num_dependents	numeric
own_telephone	nominal
foreign_worker	nominal
class	nominal

2.2 Hepatitis dataset

Il dataset contiene informazioni su un vari casi di epatite.

- Numero di istanze: 135
- Numero di attributi: 20
- Attributo target: **Class**
- Valori target: {DIE, LIVE}

Attributo	Tipo
AGE	numeric
SEX	nominal
STEROID	nominal
ANTIVIRALS	nominal
FATIGUE	nominal
MALAISE	nominal
ANOREXIA	nominal
LIVER_BIG	nominal
LIVER_FIRM	nominal
SPLEEN_PALPABLE	nominal
SPIDERS	nominal
ASCITES	nominal
VARICES	nominal
BILIRUBIN	numeric
ALK_PHOSPHATE	numeric
SGOT	numeric
ALBUMIN	numeric
PROTIME	numeric
HISTOLOGY	nominal
Class	nominal

2.3 Vehicle Silhouettes dataset

Il dataset contiene informazioni per discriminare le silhouette di diversi veicoli tra automobili, van e bus.

- Numero di istanze: 846
- Numero di attributi: 19
- Attributo target: **Class**
- Valori target: {opel, saab, bus, van}

Attributo	Tipo.
COMPACTNESS	numeric
CIRCULARITY	numeric
DISTANCE CIRCULARITY	numeric
RADIUS RATIO	numeric
PR.AXIS ASPECT RATIO	numeric
MAX.LENGTH ASPECT RATIO	numeric
SCATTER RATIO	numeric
ELONGATEDNESS	numeric
PR.AXIS RECTANGULARITY	numeric
MAX.LENGTH RECTANGULARITY	numeric
SCALED VARIANCE_MAJOR	numeric
SCALED VARIANCE_MINOR	numeric
SCALED RADIUS OF GYRATION	numeric
SKEWNESS ABOUT_MAJOR	numeric
SKEWNESS ABOUT_MINOR	numeric
KURTOSIS ABOUT_MAJOR	numeric
KURTOSIS ABOUT_MINOR	numeric
HOLLOWS RATIO	numeric
Class	nominal

2.4 Wisconsin Breast Cancer dataset

Il dataset contiene informazioni riguardo a vari casi di tumore al seno, che permettono di stabilire se esso è benigno o maligno.

- Numero di istanze: 699
- Numero di attributi: 10
- Attributo target: **Class**
- Valori target: {benign, malignant}

Attributo	Tipo
Clump_Thickness	numeric
Cell_Size_Uniformity	numeric
Cell_Shape_Uniformity	numeric
Marginal_Adhesion	numeric
Single_Epi_Cell_Size	numeric
Bare_Nuclei	numeric
Bland_Chromatin	numeric
Normal_Nucleoli	numeric
Mitoses	numeric
Class	nominal

Capitolo 3: Naive Bayes

Il primo algoritmo sfrutta il ragionamento bayesiano, un approccio probabilistico all'inferenza. Si basa sull'assunzione che i dati sono governati da distribuzioni di probabilità e che possono essere prese decisioni ottimali relative a queste probabilità insieme agli esempi a disposizione. Un modello bayesiano non è complicato da costruire, soprattutto su grandi dataset. Nonostante la sua semplicità, tale classificatore spesso si comporta meglio di altri classificatori più sofisticati[5].

3.1 Teorema di Bayes

Nel contesto di classificazione, quello che interessa è determinare la migliore ipotesi h appartenente ad uno spazio delle ipotesi H e i dati osservati D . Un modo per determinare la *migliore* ipotesi è ricercare la *più probabile*, grazie ai dati a disposizione più una conoscenza iniziale sulle probabilità a priori delle varie ipotesi in H [8]. Il teorema di Bayes fornisce un modo diretto per calcolare queste probabilità, in particolare:

- $P(h)$, la *probabilità a priori* che l'ipotesi h sia valida, prima di aver osservato i dati di training. Riflette una qualche conoscenza pregressa che abbiamo circa la possibilità che h sia corretta.
- $P(D)$ denota la probabilità a priori che i dati di training D saranno osservati, senza fare alcuna considerazione sulle ipotesi.
- $P(D|h)$ denota la probabilità di osservare i dati D in un mondo in cui l'ipotesi h regga.
- $P(h|D)$ è la probabilità che h sia valida dopo aver osservato i dati di training D . Si tratta della *probabilità a posteriori* di h perché riflette la confidenza che h sia corretta dopo aver visto D .

Quello che ci interessa è $P(h|D)$, che è possibile calcolare combinando le probabilità succitate:

$$P(h|D) = \frac{P(D|h)P(h)}{P(D)}$$

Come è facile intuire, $P(h|D)$ aumenta con $P(h)$ e con $P(D|h)$. Analogamente, $P(h|D)$ diminuisce all'aumentare di $P(D)$, perché più è probabile che D venga osservato non considerando h , meno evidenza D fornisce in supporto ad h .

In molti scenari di apprendimento vengono considerate un insieme H di ipotesi candidate e, tra di esse, ci interessa trovare quella più probabile dopo aver osservato i dati D (o almeno quella massimamente più probabile, se ce n'è più di una). Questa ipotesi è chiamata ipotesi *maximum a posteriori* (MAP). Si può determinare l'ipotesi MAP usando il teorema di Bayes per calcolare la probabilità a posteriori di ogni ipotesi candidata, e poi trovare quella che massimizza tale probabilità:

$$\begin{aligned} h_{MAP} &= \operatorname{argmax}_{h \in H} P(h|D) \\ &= \operatorname{argmax}_{h \in H} \frac{P(D|h)P(h)}{P(D)} \\ &= \operatorname{argmax}_{h \in H} P(D|h)P(h) \end{aligned}$$

Il termine $P(D)$ può essere tolto perché è una costante indipendente da h .

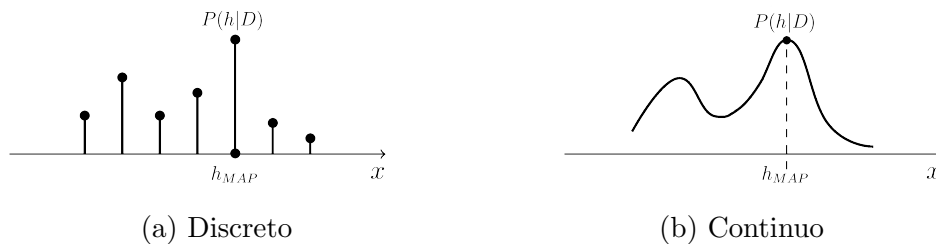


Figura 3.1: Rappresentazione di MAP

3.2 Naive Bayes Classifier

Un metodo di apprendimento molto efficace è il classificatore **Naive Bayes**. Esso si applica a task di apprendimento in cui ogni istanza è descritta come

una congiunzione di valori di attributo $\langle a_1, \dots, a_n \rangle$ e dove l'attributo target può assumere un qualsiasi valore da un insieme finito V .

Ad una nuova istanza viene assegnato il più probabile valore target v_{MAP} , considerati i valori di attributo $\langle a_1, \dots, a_n \rangle$:

$$v_{MAP} = \operatorname{argmax}_{v_j \in V} P(v_j | a_1, \dots, a_n)$$

Riapplicando le trasformazioni relative alla MAP definite sopra, possiamo riscrivere l'espressione come:

$$\begin{aligned} v_{MAP} &= \operatorname{argmax}_{h \in H} \frac{P(a_1, \dots, a_n | v_j) P(v_j)}{P(a_1, \dots, a_n)} \\ &= \operatorname{argmax}_{h \in H} P(a_1, \dots, a_n | v_j) P(v_j) \end{aligned}$$

Ora bisogna stimare le due probabilità sui dati di training. I vari $P(v_j)$ possono essere facilmente calcolati contando la frequenza con cui ogni valore target v_j occorre nei dati. Non è altrettanto semplice calcolare $P(a_1, \dots, a_n | v_j)$ allo stesso modo. Il problema è che ci sono molte probabilità da calcolare e pochi dati a disposizione per ottenere delle stime affidabili: servirebbero dataset molto grandi.

Qui entra in gioco il punto cardine del NBC, ossia presupporre che esista l'*indipendenza condizionale* tra gli attributi. In altre parole, l'assunzione è che, dato il valore target, la probabilità di osservare la congiunzione a_1, \dots, a_n è semplicemente il prodotto delle probabilità dei singoli attributi:

$$P(a_1, \dots, a_n | v_j) = \prod_i P(a_i | v_j)$$

Facendo le opportune sostituzioni otteniamo:

$$v_{NB} = v_{MAP} = \operatorname{argmax}_{v_j \in V} P(v_j) \prod_i P(a_i | v_j)$$

dove v_{NB} è il valore target restituito da NBC, che è uguale a v_{MAP} quando si assume l'indipendenza condizionale. Si noti che $P(a_i | v_j)$ è semplicemente il numero dei valori di attributo distinti moltiplicato il numero dei valori target distinti, chiaramente un numero molto più piccolo e gestibile rispetto a $P(a_1, \dots, a_n | v_j)$ senza l'ipotesi di indipendenza.

L'algoritmo di un NBC, pertanto, si può riassumere come:

- Calcola i vari $P(v_j)$ e $P(a_i | v_j)$, basandoti sulle loro frequenze sui dati di training.

- Usa queste stime per formare l'ipotesi.
- Ogniqualevolta l'assunzione di indipendenza condizionale è soddisfatta, la classificazione v_{NB} è uguale alla classificazione MAP.

Un aspetto interessante è che NBC non effettua alcuna ricerca esplicita nello spazio delle ipotesi. Viene costruita l'ipotesi semplicemente contando le frequenze delle varie combinazioni dei dati all'interno del training set.

Attributi continui

Contare le frequenze delle diverse combinazioni di valori è possibile per gli attributi discreti, quindi bisogna trovare un modo per calcolare le probabilità degli eventuali attributi continui. Ci sono due strade: la prima è appunto discretizzare i valori numerici in vari *bin*, l'altra è assumere l'esistenza di una distribuzione normale (Gaussiana) per i valori $\langle a_1, \dots, a_n \rangle$ dell'attributo A_k . Questa funzione ha bisogno di due parametri, media e deviazione standard degli attributi:

$$\mu_{kj} = E[A_k|v_j] = \frac{1}{n} \sum_{i=1}^n a_i$$

$$\sigma_{kj} = \sqrt{E[(A_k - \mu_{kj})^2|v_j]} = \sqrt{\frac{1}{n-1} \sum_{i=1}^n (a_i - \mu_{kj})^2}$$

Infine, serve la funzione di densità di probabilità della Gaussiana, che applicata al nostro contesto diventa:

$$P(a_i|v_j) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(a_i-\mu)^2}{2\sigma^2}}$$

Capitolo 4: REPTree

Come altro algoritmo si è scelto di usare **REPTree**, che costruisce alberi di decisione usando l'*information gain* per i valori nominali e la varianza per i valori numerici[19].

Visto che sono stati presi in considerazione dataset con attributi di classe nominali per un task di classificazione e non di regressione, verrà discusso il criterio dell'*information gain*, analogo all'algoritmo *C4.5*[12].

4.1 Information Gain

Per selezionare l'attributo che meglio classifica i dati D ed in particolare, su quale dei suoi valori occorre fare uno *split*, può convenire usare l'*entropia*[16], ossia l'incertezza contenuta nei dati, che è calcolata come:

$$E(D) = - \sum_i p_i \log_2 p_i$$

cioè la media dei logaritmi delle probabilità di ciascun oggetto i pesato per la probabilità stessa. Nel contesto di classificazione, gli oggetti sono gli esempi nel dataset di cui viene calcolata la probabilità che essi appartengano o meno ad una delle classi presenti nell'attributo target. Più è probabile che un esempio appartenga ad una certa classe, più la sua influenza nel calcolo della media sarà mitigata dal logaritmo della sua stessa probabilità.

È possibile calcolare l'entropia anche per sottoinsiemi del dataset, in particolare quegli esempi D_v che presentano lo stesso valore v di un certo attributo a , poi sommare tutte le entropie relative a tutti i valori V dell'attributo per ottenere l'entropia dei dati dopo aver preso in considerazione l'attributo a . Ogni entropia viene pesata per il numero di esempi che presentano quel valore diviso per il totale di esempi esistenti nel dataset:

$$E(D|a) = \sum_{v \in V} \frac{|D_v|}{|D|} \cdot E(D_v)$$

Da queste formule si ricava l'information gain, cioè la riduzione di incertezza totale prendendo in considerazione un attributo a :

$$IG = E(D) - E(D|a)$$

Come radice verrà utilizzato l'attributo che massimizza l'information gain, come archi i valori dell'attributo e si ripete la procedura per i nodi figli fino a generare le foglie.

4.2 Reduced Error Pruning

Per evitare l'*overfitting*, ossia un sovra-adattamento del modello ai dati di training che compromette la bontà delle sue predizioni su nuovi esempi, può essere ragionevole semplificare il modello, rischiando di commettere qualche errore ma garantendoci una migliore copertura per dati non visti.

Questa semplificazione viene chiamata *pruning* (potatura), in cui, una volta costruito il modello utilizzando i dati del *growing set*, esso viene testato su una parte dei dati, accantonati e non adoperati per la predizione, che fanno parte del *pruning set*.

Una tecnica di potatura è REP (**R**educed **E**rror **P**runing)[11] che utilizza un pruning set per stimare l'accuratezza dei nodi intermedi e confrontarla con quella dei suoi sottoalberi.

Viene calcolato il guadagno dall'eventuale potatura sottraendo il numero di errori (esempi classificati scorrettamente) al sottoalbero T al numero di errori al nodo radice v del sottoalbero:

$$Gain_{REP} = \varepsilon_T - \varepsilon_v$$

L'albero è potato se il guadagno è positivo quando vengono commessi più errori nell'intero sottoalbero, e non al nodo. C'è un'altra condizione da rispettare per procedere alla potatura: può avvenire solo se il sottoalbero T non ha un sottoalbero che ha un tasso d'errore minore di T stesso (*bottom-up restriction*).

L'algoritmo di REP è il seguente:

- Si parte dall'albero completo e lo si visita in post-ordine.
- Per ogni nodo intermedio v
 - Calcolo l'accuratezza sul pruning set dell'albero completo.
 - Calcolo l'accuratezza sul pruning set rispetto a v e al suo sottoalbero T .

- Se l'accuratezza aumenta, pota. In caso di uguaglianza pota per semplificare (*rasoio di Occam*).

Inoltre è dimostrato che, tra tutti i possibili sottoalberi potati che è possibile generare, REP trova il sottoalbero più piccolo e più accurato rispetto al pruning set[6].

Capitolo 5: RIPPER

Come ultimo algoritmo si è scelto di usare **RIPPER**[3], in particolare nella versione implementata da Weka, **JRip**[20].

RIPPER (**R**epeated **I**ncremental **P**runing to **P**roduce **E**rror **R**eduction) è un algoritmo di induzione di regole proposto da William W. Cohen nel 1995. Esso si è dimostrato competitivo con C4.5Rules rispetto ai tassi di errore, scala in maniera lineare con il numero di esempi di training e può elaborare in maniera efficiente dataset rumorosi che contengono centinaia di migliaia di esempi. RIPPER si basa su IREP (**I**ncremental **R**educed **E**rror **P**runing))[7], di cui si discuterà nei prossimi paragrafi.

Molte delle tecniche usate nei moderni sistemi di apprendimento di regole sono state adattate dall'apprendimento degli alberi di decisione. La maggior parte dei sistemi di apprendimento di alberi di decisione usa una strategia di apprendimento *overfit-and-simplify* (sovradata-e-semplifica) per gestire dati rumorosi: viene generata un'ipotesi prima facendo crescere un albero complesso che "overfitta" i dati, e poi si semplifica o pota tale albero (un'operazione di *pruning*). Una tecnica di pruning efficace è *reduced error pruning* (*REP*), discussa in 4.2. Essa può essere facilmente adattata ai sistemi di apprendimento di regole[9][1].

In REP per le regole, il training set viene diviso in *growing set* e *pruning set*. All'inizio, viene creato un *rule set* di partenza che overfitta il growing set, usando qualche metodo euristico. Questo rule set spropositato viene poi semplificato ripetutamente applicando qualche operatore di pruning. Ad ogni fase di semplificazione, l'operatore di pruning scelto è quello che produce la più grande riduzione di errore sul pruning set. La semplificazione finisce quando il tasso di errore non si riduce ulteriormente applicando gli operatori di pruning.

REP per le regole di solito migliora davvero la performance di generalizzazione sui dati rumorosi[9][1][17][7]; tuttavia è computazionalmente costoso per grandi dataset[2].

In risposta all'inefficienza di REP, Fürnkranz e Widmer [1994] proposero un algoritmo di apprendimento chiamato *incremental reduced error pruning*

(IREP)[7].

5.1 Incremental Reduced Error Pruning

L'idea di usare un pruning set separato per la potatura è REP. La variante che pota una regola subito dopo averla "fatta crescere" si chiama *incremental reduced error pruning* (IREP)[18]. Quest'ultima integra saldamente REP con un algoritmo di apprendimento di regole *separate-and-conquer*. L'algoritmo 1 ne presenta una versione a due classi. Come ogni algoritmo *separate-and-conquer* standard, IREP costruisce un ruleset in maniera *greedy*, una regola alla volta. Dopo averne trovata una, tutti gli esempi coperti da quella regola (sia positivi che negativi) sono cancellati. Questo processo si ripete finché non ci sono più esempi positivi, o finché la regola trovata da IREP non presenta un grande tasso di errore, cosa inaccettabile.

Per costruire una regola, IREP usa la seguente strategia. Prima, gli esempi non coperti sono partizionati a caso in due sottoinsiemi, un growing e un pruning set. Nell'implementazione di Cohen, il growing set contiene 2/3 degli esempi.

Poi, una regola viene "fatta crescere". L'implementazione di Cohen di *GrowRule* è una versione proposizionale di FOIL (First Order Inductive Learner), dove i letterali non si servono di predicati ma di uguaglianze (per valori discreti) e confronti numerici (per valori continui)[15]. Esso inizia con una congiunzione vuota di condizioni (la regola vuota) e considera di aggiungere a questa qualsiasi condizione nella forma $A_d = v$, $A_c \leq \theta$ oppure $A_c \geq \theta$ dove A_d è un attributo discreto e v è un valore che può assumere, mentre A_c è un attributo continuo e θ è un valore soglia. *GrowRule* aggiunge ripetutamente la condizione che massimizza un'euristica di *information gain*, nello specifico quella di FOIL, finché la regola non copre più esempi negativi nel growing set.

Siano R_0 e R_1 due regole, la seconda ottenuta dall'aggiunta di una condizione nel corpo della prima. L'information gain viene così calcolato:

$$Gain_{IREP}(R_0, R_1) = t \cdot \left(\log \frac{p_1}{p_1 + n_1} - \log \frac{p_0}{p_0 + n_0} \right)$$

dove t riguarda gli esempi positivi coperti da R_0 che soddisfano anche R_1 dopo aver aggiunto una condizione, p_0 (rispettivamente p_1) sono gli esempi positivi coperti da R_0 (rispettivamente R_1) e n_0 (rispettivamente n_1) sono gli esempi negativi coperti da R_0 (rispettivamente R_1).

L'idea alla base è che l'informazione totale che si guadagna è dato dal numero di tuple che soddisfano la nuova condizione moltiplicato l'informazione guadagnata in merito a ciascuna[10].

Dopo aver espanso una regola, essa viene immediatamente potata. Per prunarla, l'implementazione di Cohen cancella qualsiasi sequenza finale di condizioni dalla regola e sceglie l'eliminazione che massimizza la funzione

$$v(Rule, PrunePos, PruneNeg) \equiv \frac{p + (N - n)}{P + N} \quad (5.1)$$

dove P (rispettivamente N) è il numero totale di esempi in $PrunePos$ ($PruneNeg$) e p (n) è il numero di esempi in $PrunePos$ ($PruneNeg$) coperti da $Rule$. Questo processo è ripetuto finché nessun'altra cancellazione migliora il valore di v .

L'algoritmo IREP descritto sopra è per i problemi di apprendimento a due classi. L'implementazione di Cohen gestisce classi multiple, come spiegato di seguito:

1. Le classi vengono ordinate secondo la prevalenza, cioè l'ordine è C_1, \dots, C_k dove C_1 è la classe di minoranza e C_k è la classe di maggioranza.
2. Viene trovata una regola che separi C_1 dal resto delle classi; questo viene fatto con una singola chiamata ad IREP dove $PosData$ contiene gli esempi di classe C_1 e $NegData$ contiene gli esempi di classi C_2, C_3, \dots, C_k .
3. Tutte le istanze coperte dal ruleset appena addestrato sono rimosse dal dataset e IREP si appresta a separare C_2 dalle classi C_3, \dots, C_k .
4. Si ripete finché rimane la sola classe C_k . Quest'ultima verrà usata come classe di default.

L'implementazione di Cohen differisce da quella di Fürnkranz e Widmer sotto molti aspetti. Quando le regole vengono potate, la nuova implementazione permette di cancellare qualsiasi sequenza finale di condizioni, mentre l'implementazione di Fürnkranz e Widmer permette solo la cancellazione di una singola condizione finale. L'algoritmo rivisitato permette anche di fermare l'aggiunta di regole al ruleset quando la regola appresa ha un tasso di errore superiore al 50%, mentre quello di Fürnkranz e Widmer la ferma quando l'accuratezza della regola è minore dell'accuratezza della regola vuota.

Algoritmo 1 IREP(Pos, Neg)

```
1:  $Ruleset \leftarrow \emptyset$ 
2: while  $Pos \neq \emptyset$  do
3:   dividi  $(Pos, Neg)$  in  $(GrowPos, GrowNeg)$  e  $(PrunePos, PruneNeg)$ 
4:    $Rule \leftarrow GrowRule(GrowPos, GrowNeg)$ 
5:    $Rule \leftarrow PruneRule(Rule, PrunePos, PruneNeg)$ 
6:   if il tasso di errore su  $(PrunePos, PruneNeg) > 50\%$  then
7:     return  $Ruleset$ 
8:   else
9:     aggiungi  $Rule$  a  $Ruleset$ 
10:    rimuovi gli esempi coperti da  $Rule$  da  $(Pos, Neg)$ 
11: return  $Ruleset$ 
```

5.2 Miglioramenti ad IREP

Sono state implementate tre modifiche ad IREP: una metrica alternativa per determinare il valore delle regole nella fase di potatura; una nuova euristica per decidere quando fermare l'aggiunta di regole al ruleset; un successivo passaggio di "ottimizzazione" del ruleset per tentare di avvicinarsi di più al REP convenzionale (cioè, non incrementale).

Metrica per il valore delle regole

Il fallimento occasionale di IREP a convergere al crescere del numero degli esempi può essere facilmente fatto risalire alla metrica usata per guidare la potatura (ossia la (5.1)). Le scelte intraprese nella definizione di tale metrica non sono intuitive; per esempio (assumendo che P e N siano fissati) la metrica preferisce una regola R_1 che copre $p_1 = 2000$ esempi positivi e $n_1 = 1000$ esempi negativi rispetto ad una regola R_2 che copre $p_2 = 1000$ esempi positivi e $n_2 = 1$ esempio negativo; si noti comunque che R_2 è altamente predittiva, al contrario di R_1 . Quindi si è deciso di sostituire la metrica di IREP con

$$v^*(Rule, PrunePos, PruneNeg) \equiv \frac{p - n}{p + n}$$

che sembra avere un comportamento più intuitivo e soddisfacente.

Condizione di stop

L'implementazione di IREP di Cohen si ferma in maniera greedy aggiungendo regole al ruleset quando l'ultima regola costruita ha un tasso d'errore

maggiore del 50% sui dati di pruning. Questa euristica, spesso, si ferma troppo presto con campioni di dimensioni moderate; questo è vero soprattutto quando si apprende un concetto con regole a bassa copertura (pochi esempi coperti).

La soluzione a questo problema è la seguente. Dopo l'aggiunta di ogni regola, viene calcolata la *description-length* totale del ruleset e degli esempi. La nuova versione di IREP ferma l'aggiunta di regole quando questa *description-length* è maggiore di d bit rispetto alla più piccola *description-length* ottenuta sinora, o quando non ci sono più esempi positivi. Nell'implementazione si è usato $d = 64$. Il ruleset viene poi semplificato esaminando ogni regola a turno (cominciando dall'ultima) e cancellando regole così da ridurre la *description-length* totale.

Il principio *MDL* (Minimum Description Length) può essere meglio espresso immaginando un modello di comunicazione in cui un mittente trasmette ad un ricevente una descrizione che consiste in una teoria T e i dati D da cui essa è derivata[14].

Il metodo usato per la codifica è lo stesso usato in *C4.5rules*[13]. Esso parte da un *bias* in cui il numero di falsi positivi e falsi negativi sia lo stesso e si procede come segue: i messaggi da inviare si presentano con probabilità p_j , e servono $-\log(p_j)$ bit (in base 2) per costruirli: più un messaggio è frequente, meno bit saranno necessari per rappresentarlo. Si inviano i dati codificati, poi anziché inviare i messaggi di errore per tutti i dati, il mittente prima trasmette gli errori e nei C casi coperti dalla teoria e poi negli U casi non coperti. Sotto l'assunzione che i falsi positivi fp e i falsi negativi fn siano bilanciati, la probabilità di errore nei casi coperti è $e/2C$ e questa probabilità è usata per codificare i messaggi di errore per i casi coperti. Una volta che i falsi positivi sono stati identificati, il destinatario può calcolare il vero numero dei falsi negativi come $e - fp$, quindi la probabilità di errore per i casi non coperti è fn/U . Il costo totale quindi diventa:

$$\begin{aligned} & \log(|D| + 1) \\ & + fp \times (-\log(\frac{e}{2C})) \\ & + (C - fp) \times (-\log(1 - \frac{e}{2C})) \\ & + fn \times (-\log(\frac{fn}{U})) \\ & + (U - fn) \times (-\log(1 - \frac{fn}{U})) \end{aligned}$$

Ottimizzazione delle regole

L'approccio ripetuto *grow-and-simplify* usato in IREP può produrre risultati abbastanza differenti dal REP convenzionale (non incrementale). Un modo per migliorarlo è elaborare a posteriori le regole prodotte da IREP così da avvicinarsi di più all'effetto del REP convenzionale. Per esempio, si potrebbe ri-potare ogni regola al fine di minimizzare l'errore del ruleset completo.

Il metodo sviluppato per ottimizzare un ruleset R_1, R_2, \dots, R_k consiste nel costruire due regole alternative per ogni R_i . La *sostituta* di R_i viene generata espandendo e poi potando R_i . La *revisione* di R_i viene generata in maniera analoga, tranne per il fatto che la revisione è espansa in modo greedy aggiungendo condizioni a R_i , piuttosto che alla regola vuota. Infine si sceglie tra le tre regole quale includere nella teoria. Questa decisione viene presa in base all'euristica MDL. L'implementazione di questo metodo in IREP avviene in questo modo:

1. Viene usato IREP per ottenere un ruleset iniziale.
2. Esso viene ottimizzato, come descritto sopra.
3. Vengono aggiunte le regole in modo tale da coprire gli esempi positivi rimanenti.

L'ottimizzazione può essere ripetuta più volte elaborando il ruleset ottenuto dalla passata precedente dell'algoritmo.

IREP, con l'aggiunta del passo di post-ottimizzazione, forma un nuovo algoritmo che è stato chiamato **RIPPER** (**R**epeated **I**ncremental **P**runing to **P**roduce **E**rror **R**eduction).

L'implementazione in Weka di RIPPER si chiama JRip.

Capitolo 6: Esecuzione

Qui vengono confrontati i due algoritmi REPTree e RIPPER/JRip. Entrambi hanno sfruttato una *10-fold cross validation*.

6.1 Risultati su German Credit

L'esecuzione ha coinvolto 900 istanze di training e 100 istanze di testing ad ogni iterazione del CV.

Esecuzione NaiveBayesSimple

```
=== Run information ===

Scheme:weka.classifiers.bayes.NaiveBayesSimple
Relation:    german_credit
Instances:   1000
Attributes:  21
             checking_status
             duration
             credit_history
             purpose
             credit_amount
             savings_status
             employment
             installment_commitment
             personal_status
             other_parties
             residence_since
             property_magnitude
             age
             other_payment_plans
             housing
             existing_credits
             job
             num_dependents
             own_telephone
```

```

foreign_worker
class
Test mode:10-fold cross-validation

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances      756           75.6   %
Incorrectly Classified Instances    244           24.4   %
Kappa statistic                    0.3876
Mean absolute error                 0.294
Root mean squared error             0.4209
Relative absolute error             69.9613 %
Root relative squared error         91.8429 %
Total Number of Instances          1000

=== Detailed Accuracy By Class ===

      TP Rate  FP Rate  Precision  Recall  F-Measure  ROC Area  Class
      0.864    0.497    0.802    0.864    0.832     0.785   good
      0.503    0.136    0.614    0.503    0.553     0.785   bad
Weighted Avg.   0.756    0.388    0.746    0.756    0.748     0.785

=== Confusion Matrix ===

  a   b  <-- classified as
605  95 |  a = good
149 151 |  b = bad

```

Modello Naive Bayes										
Class good : P(C) = 0.6996008										
Attribute checking_status										
“<0” 0.19886364		“0<=X<200” 0.234375			“>=200” 0.07102273			no checking 0.49573864		
Attribute duration										
Mean: 19.20714286					Standard Deviation: 11.07956427					
Attribute credit_history										
no credits/all paid 0.02269504		all paid 0.03120567		existing paid 0.51347518		delayed previously 0.08652482		critical/other existing credit 0.34609929		
Attribute purpose										
new car 0.20534459	used car 0.12236287	furniture/equipment 0.17440225	radio/tv 0.30801688	domestic appliance 0.01265823	repairs 0.02109705	education 0.04078762	vacation 0.00140647	retraining 0.01265823	business 0.09001406	other 0.01125176
Attribute credit_amount										
Mean: 2985.45714286					Standard Deviation: 2401.47227783					
Attribute savings_status										
“<100” 0.54893617		“100<=X<500” 0.09929078			“500<=X<1000” 0.0751773		“>=1000” 0.06099291		no known savings 0.21560284	
Attribute employment										
unemployed 0.05673759		“<1” 0.14609929			“1<=X<4” 0.33475177		“4<=X<7” 0.1929078		“>=7” 0.26950355	
Attribute installment_commitment										
Mean: 2.92					Standard Deviation: 1.12807837					
Attribute personal_status										
male div/sep 0.04397163		female div/dep/mar 0.28652482			male single 0.57163121		male mar/wid 0.0964539		female single 0.00141844	
Attribute other_parties										
none 0.90469417			co applicant 0.0341394				guarantor 0.06116643			
Attribute residence_since										
Mean: 2.84285714					Standard Deviation: 1.10837254					
Attribute property_magnitude										
real estate 0.31676136		life insurance 0.23011364			car 0.328125			no known property 0.125		
Attribute age										
Mean: 36.22428571					Standard Deviation: 11.38114466					
Attribute other_payment_plans										
bank 0.11806543			stores 0.04125178				none 0.84068279			
Attribute housing										
rent 0.15647226			own 0.75106686				for free 0.09246088			
Attribute existing_credits										
Mean: 1.42428571					Standard Deviation: 0.58472099					
Attribute job										
unemp/unskilled non res 0.02272727		unskilled resident 0.20596591			skilled 0.63210227			high qualif/self emp/mgmt 0.13920455		
Attribute num_dependents										
Mean: 1.15571429					Standard Deviation: 0.36284353					
Attribute own_telephone										
none 0.58404558					yes 0.41595442					
Attribute foreign_worker										
yes 0.95156695					no 0.04843305					
Class bad : P(C) = 0.3003992										
Continua...										

Modello Naive Bayes										
Attribute checking_status										
“<0” 0.44736842		“0<=X<200” 0.34868421			“>=200” 0.04934211			no checking 0.15460526		
Attribute duration										
Mean: 24.86					Standard Deviation: 13.28263886					
Attribute credit_history										
no credits/all paid 0.0852459		all paid 0.09508197		existing paid 0.55737705		delayed previously 0.09508197		critical/other existing credit 0.16721311		
Attribute purpose										
new car 0.28938907	used car 0.05787781	furniture/ equipment 0.18971061	radio/tv 0.20257235	domestic appliance 0.01607717	repairs 0.02893891	education 0.07395498	vacation 0.00321543	retraining 0.00643087	business 0.11254019	other 0.0192926
Attribute credit_amount										
Mean: 3938.12666667					Standard Deviation: 3535.81895505					
Attribute savings_status										
“<100” 0.7147541		“100<=X<500” 0.1147541			“500<=X<1000” 0.03934426		“>=1000” 0.02295082		no known savings 0.10819672	
Attribute employment										
unemployed 0.07868852		“<1” 0.23278689			“1<=X<4” 0.3442623		“4<=X<7” 0.13114754		“>=7” 0.21311475	
Attribute installment_commitment										
Mean: 3.09666667					Standard Deviation: 1.08839526					
Attribute personal_status										
male div/sep 0.06885246		female div/dep/mar 0.36065574			male single 0.48196721		male mar/wid 0.0852459		female single 0.00327869	
Attribute other_parties										
none 0.9009901				co applicant 0.06270627			guarantor 0.03630363			
Attribute residence_since										
Mean: 2.85					Standard Deviation: 1.0946052					
Attribute property_magnitude										
real estate 0.20065789			life insurance 0.23684211			car 0.33881579		no known property 0.22368421		
Attribute age										
Mean: 33.96333333					Standard Deviation: 11.2223792					
Attribute other_payment_plans										
bank 0.19141914				stores 0.0660066			none 0.74257426			
Attribute housing										
rent 0.23432343				own 0.61716172			for free 0.14851485			
Attribute existing_credits										
Mean: 1.36666667					Standard Deviation: 0.5597021					
Attribute job										
unemp/unskilled non res 0.02631579			unskilled resident 0.1875			skilled 0.61513158		high qualif/self emp/mgmt 0.17105263		
Attribute num_dependents										
Mean: 1.15333333					Standard Deviation: 0.36091053					
Attribute own_telephone										
none 0.62251656					yes 0.37748344					
Attribute foreign_worker										
yes 0.98344371					no 0.01655629					

Esecuzione REPTree

=== Run information ===

Scheme:weka.classifiers.trees.REPTree -M 2 -V 0.001 -N 3 -S 1 -L -1

Relation: german_credit

Instances: 1000

Attributes: 21

checking_status
duration
credit_history
purpose
credit_amount
savings_status
employment
installment_commitment
personal_status
other_parties
residence_since
property_magnitude
age
other_payment_plans
housing
existing_credits
job
num_dependents
own_telephone
foreign_worker
class

Test mode:10-fold cross-validation

=== Stratified cross-validation ===

=== Summary ===

Correctly Classified Instances	718	71.8	%
Incorrectly Classified Instances	282	28.2	%
Kappa statistic	0.2702		
Mean absolute error	0.3417		
Root mean squared error	0.4424		
Relative absolute error	81.3157	%	
Root relative squared error	96.532	%	
Total Number of Instances	1000		

=== Detailed Accuracy By Class ===

TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area	Class
0.859	0.61	0.767	0.859	0.81	0.72	good
0.39	0.141	0.542	0.39	0.453	0.72	bad

```
Weighted Avg.    0.718    0.469    0.699    0.718    0.703    0.72
```

```
=== Confusion Matrix ===
```

```
   a   b  <-- classified as
601  99 |   a = good
183 117 |   b = bad
```

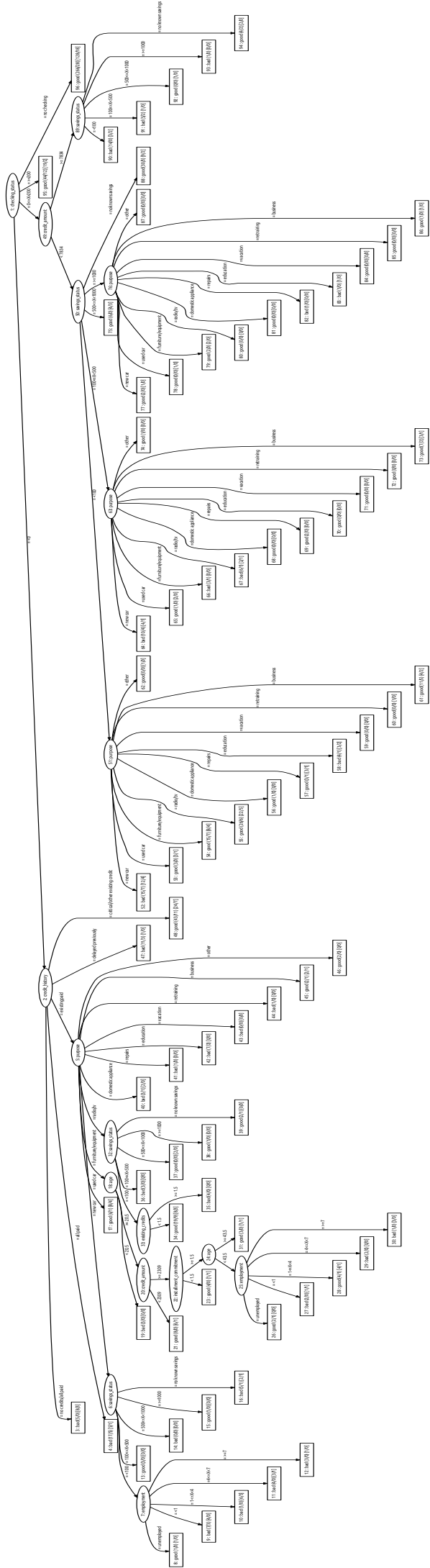


Figura 6.1: Modello di REPTree

Esecuzione JRip

=== Run information ===

Scheme:weka.classifiers.rules.JRip -F 3 -N 2.0 -O 2 -S 1

Relation: german_credit

Instances: 1000

Attributes: 21

checking_status
duration
credit_history
purpose
credit_amount
savings_status
employment
installment_commitment
personal_status
other_parties
residence_since
property_magnitude
age
other_payment_plans
housing
existing_credits
job
num_dependents
own_telephone
foreign_worker
class

Test mode:10-fold cross-validation

=== Stratified cross-validation ===

=== Summary ===

Correctly Classified Instances	717	71.7	%
Incorrectly Classified Instances	283	28.3	%
Kappa statistic	0.2513		
Mean absolute error	0.3781		
Root mean squared error	0.4472		
Relative absolute error	89.9974	%	
Root relative squared error	97.5906	%	
Total Number of Instances	1000		

=== Detailed Accuracy By Class ===

TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area	Class
0.873	0.647	0.759	0.873	0.812	0.593	good
0.353	0.127	0.544	0.353	0.428	0.593	bad

```
Weighted Avg.    0.717    0.491    0.694    0.717    0.697    0.593
```

```
=== Confusion Matrix ===
```

```
   a   b  <-- classified as
611  89 |   a = good
194 106 |   b = bad
```

Regole:

1. (checking_status = "<0") **AND** (job = skilled) \Rightarrow class=bad (172.0/76.0)
2. (checking_status = "0<=X<20") **AND** (duration >= 24) **AND** (savings_status = "<100") \Rightarrow class=bad (61.0/19.0)
3. [*Empty Rule*] \Rightarrow class=good (767.0/162.0)

6.2 Risultati su Hepatitis

L'esecuzione ha coinvolto 139 istanze di training e 16 istanze di testing ad ogni iterazione del CV.

Esecuzione NaiveBayesSimple

=== Run information ===

Scheme:weka.classifiers.bayes.NaiveBayesSimple

Relation: hepatitis

Instances: 155

Attributes: 20

AGE

SEX

STEROID

ANTIVIRALS

FATIGUE

MALAISE

ANOREXIA

LIVER_BIG

LIVER_FIRM

SPLEEN_PALPABLE

SPIDERS

ASCITES

VARICES

BILIRUBIN

ALK_PHOSPHATE

SGOT

ALBUMIN

PROTIME

HISTOLOGY

Class

Test mode:10-fold cross-validation

=== Stratified cross-validation ===

=== Summary ===

Correctly Classified Instances	131	84.5161 %
Incorrectly Classified Instances	24	15.4839 %
Kappa statistic	0.5483	
Mean absolute error	0.1688	
Root mean squared error	0.3665	
Relative absolute error	51.118 %	
Root relative squared error	90.5193 %	
Total Number of Instances	155	

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area	Class
	0.688	0.114	0.611	0.688	0.647	0.863	DIE
	0.886	0.313	0.916	0.886	0.901	0.863	LIVE
Weighted Avg.	0.845	0.271	0.853	0.845	0.848	0.863	

=== Confusion Matrix ===

```

a   b   <-- classified as
22  10 |   a = DIE
14 109 |   b = LIVE

```

Modello Naive Bayes	
Class DIE : $P(C) = 0.21019108$	
Attribute AGE	
Mean: 46.59375	Standard Deviation: 9.94455395
Attribute SEX	
male	female
0.02941176	0.97058824
Attribute STEROID	
no	yes
0.61764706	0.38235294
Attribute ANTIVIRALS	
no	yes
0.91176471	0.08823529
Attribute FATIGUE	
no	yes
0.08823529	0.91176471
Attribute MALAISE	
no	yes
0.29411765	0.70588235
Attribute ANOREXIA	
no	yes
0.67647059	0.32352941
Attribute LIVER_BIG	
no	yes
0.13793103	0.86206897
Attribute LIVER_FIRM	
no	yes
0.51724138	0.48275862
Attribute SPLEEN_PALPABLE	
no	yes
0.60606061	0.39393939
Attribute SPIDERS	
no	yes
0.3030303	0.6969697
Attribute ASCITES	
no	yes
0.54545455	0.45454545
Attribute VARICES	
no	yes
0.63636364	0.36363636
Attribute BILIRUBIN	
Continua...	

Modello Naive Bayes	
Mean: 2.54333333	Standard Deviation: 1.94008235
Attribute ALK_PHOSPHATE	
Mean: 122.375	Standard Deviation: 54.34896303
Attribute SGOT	
Mean: 99.83333333	Standard Deviation: 101.76818937
Attribute ALBUMIN	
Mean: 3.15185185	Standard Deviation: 0.60215662
Attribute PROTIME	
Mean: 43.5	Standard Deviation: 16.76217731
Attribute HISTOLOGY	
no	yes
0.23529412	0.76470588
Class LIVE : $P(C) = 0.78980892$	
Attribute AGE	
Mean: 39.79674797	Standard Deviation: 12.82860492
Attribute SEX	
male	female
0.136	0.864
Attribute STEROID	
no	yes
0.45967742	0.54032258
Attribute ANTIVIRALS	
no	yes
0.816	0.184
Attribute FATIGUE	
no	yes
0.42741935	0.57258065
Attribute MALAISE	
no	yes
0.68548387	0.31451613
Attribute ANOREXIA	
no	yes
0.81451613	0.18548387
Attribute LIVER_BIG	
no	yes
0.19166667	0.80833333
Attribute LIVER_FIRM	
no	yes
0.59663866	0.40336134
Attribute SPLEEN_PALPABLE	
no	yes
0.84297521	0.15702479
Attribute SPIDERS	
no	yes
0.75206612	0.24793388
Attribute ASCITES	
no	yes
0.94214876	0.05785124
Attribute VARICES	
no	yes
0.9338843	0.0661157
Attribute BILIRUBIN	
Mean: 1.14621849	Standard Deviation: 0.72214252
Attribute ALK_PHOSPHATE	
Mean: 101.31372549	Standard Deviation: 50.25479247
Attribute SGOT	
Mean: 82.43801653	Standard Deviation: 86.50490281
Continua...	

Modello Naive Bayes	
Attribute ALBUMIN	
Mean: 3.97767857	Standard Deviation: 0.55600081
Attribute PROTIME	
Mean: 66.57142857	Standard Deviation: 21.91126468
Attribute HISTOLOGY	
no	yes
0.632	0.368

Esecuzione REPTree		
=== Run information ===		
Scheme:weka.classifiers.trees.REPTree -M 2 -V 0.001 -N 3 -S 1 -L -1		
Relation:	hepatitis	
Instances:	155	
Attributes:	20	
	AGE	
	SEX	
	STEROID	
	ANTIVIRALS	
	FATIGUE	
	MALAISE	
	ANOREXIA	
	LIVER_BIG	
	LIVER_FIRM	
	SPLEEN_PALPABLE	
	SPIDERS	
	ASCITES	
	VARICES	
	BILIRUBIN	
	ALK_PHOSPHATE	
	SGOT	
	ALBUMIN	
	PROTIME	
	HISTOLOGY	
	Class	
Test mode:10-fold cross-validation		
=== Stratified cross-validation ===		
=== Summary ===		
Correctly Classified Instances	122	78.7097 %
Incorrectly Classified Instances	33	21.2903 %
Kappa statistic	0.0554	
Mean absolute error	0.304	
Root mean squared error	0.4067	
Relative absolute error	92.0602 %	
Root relative squared error	100.448 %	
Total Number of Instances	155	

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area	Class
	0.063	0.024	0.4	0.063	0.108	0.533	DIE
	0.976	0.938	0.8	0.976	0.879	0.533	LIVE
Weighted Avg.	0.787	0.749	0.717	0.787	0.72	0.533	

=== Confusion Matrix ===

```

a  b  <-- classified as
2  30 |   a = DIE
3 120 |   b = LIVE

```

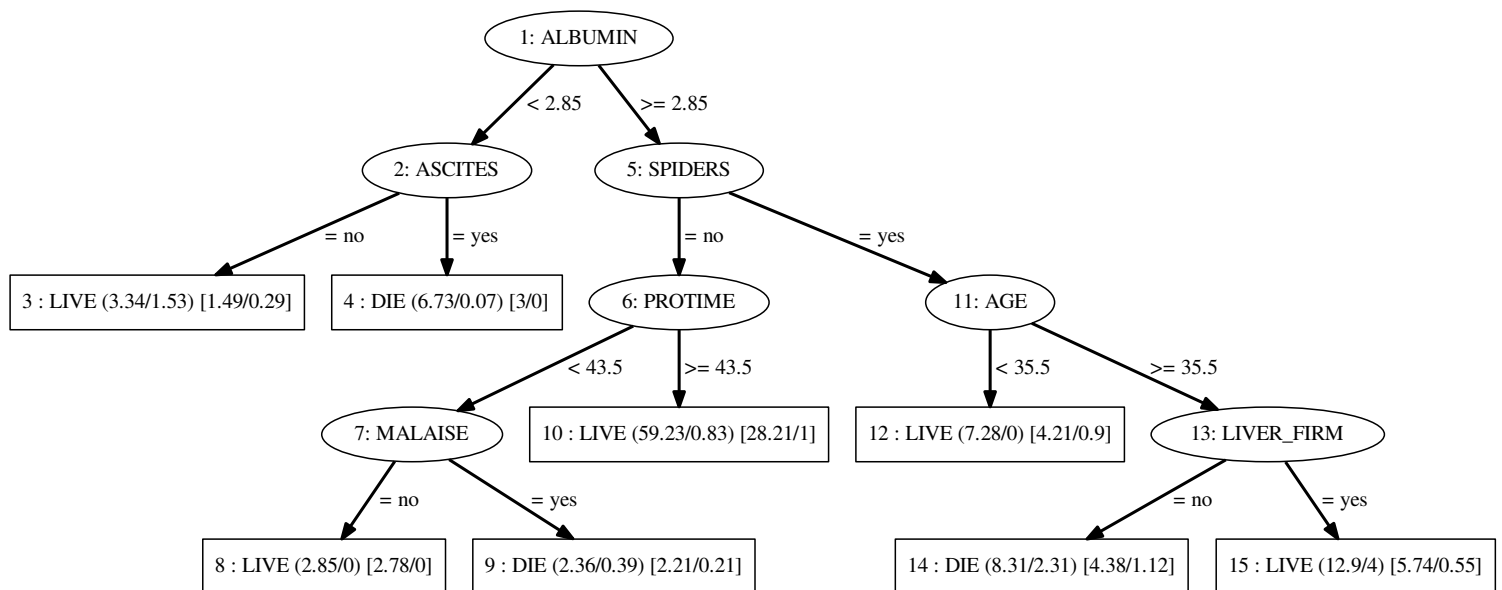


Figura 6.2: Modello di REPTree

Esecuzione JRip

=== Run information ===

```

Scheme:weka.classifiers.rules.JRip -F 3 -N 2.0 -O 2 -S 1
Relation:      hepatitis

```

```

Instances: 155
Attributes: 20
            AGE
            SEX
            STEROID
            ANTIVIRALS
            FATIGUE
            MALAISE
            ANOREXIA
            LIVER_BIG
            LIVER_FIRM
            SPLEEN_PALPABLE
            SPIDERS
            ASCITES
            VARICES
            BILIRUBIN
            ALK_PHOSPHATE
            SGOT
            ALBUMIN
            PROTIME
            HISTOLOGY
            Class

```

Test mode:10-fold cross-validation

=== Stratified cross-validation ===
 === Summary ===

Correctly Classified Instances	121	78.0645 %
Incorrectly Classified Instances	34	21.9355 %
Kappa statistic	0.2623	
Mean absolute error	0.2594	
Root mean squared error	0.4122	
Relative absolute error	78.5662 %	
Root relative squared error	101.7911 %	
Total Number of Instances	155	

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area	Class
	0.344	0.106	0.458	0.344	0.393	0.664	DIE
	0.894	0.656	0.84	0.894	0.866	0.664	LIVE
Weighted Avg.	0.781	0.543	0.761	0.781	0.768	0.664	

=== Confusion Matrix ===

```

a  b  <-- classified as
11 21 |  a = DIE
13 110 |  b = LIVE

```

Regole:

1. (ALBUMIN \leq 3.8) **AND** (ALBUMIN \leq 2.8) \Rightarrow Class=DIE (13.0/2.0)
2. (PROTIME \leq 42) \Rightarrow Class=DIE (15.0/7.0)
3. (SPIDERS = yes) **AND** (BILIRUBIN \geq 2) \Rightarrow Class=DIE (11.0/4.0)
4. [*Empty Rule*] \Rightarrow Class=LIVE (116.0/6.0)

6.3 Risultati su Vehicle Silhouettes

L'esecuzione ha coinvolto 761 istanze di training e 85 istanze di testing ad ogni iterazione del CV.

Esecuzione NaiveBayesSimple

=== Run information ===

Scheme:weka.classifiers.bayes.NaiveBayesSimple

Relation: vehicle

Instances: 846

Attributes: 19

COMPACTNESS

CIRCULARITY

DISTANCE CIRCULARITY

RADIUS RATIO

PR.AXIS ASPECT RATIO

MAX.LENGTH ASPECT RATIO

SCATTER RATIO

ELONGATEDNESS

PR.AXIS RECTANGULARITY

MAX.LENGTH RECTANGULARITY

SCALED VARIANCE_MAJOR

SCALED VARIANCE_MINOR

SCALED RADIUS OF GYRATION

SKEWNESS ABOUT_MAJOR

SKEWNESS ABOUT_MINOR

KURTOSIS ABOUT_MAJOR

KURTOSIS ABOUT_MINOR

HOLLOWS RATIO

Class

Test mode:10-fold cross-validation

=== Stratified cross-validation ===

=== Summary ===

Correctly Classified Instances	383	45.2719 %
Incorrectly Classified Instances	463	54.7281 %
Kappa statistic	0.2759	
Mean absolute error	0.2793	
Root mean squared error	0.4581	
Relative absolute error	74.4992 %	
Root relative squared error	105.8072 %	
Total Number of Instances	846	

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area	Class
	0.415	0.167	0.454	0.415	0.433	0.706	opel
	0.387	0.122	0.522	0.387	0.444	0.714	saab
	0.17	0.024	0.712	0.17	0.274	0.851	bus
	0.874	0.41	0.396	0.874	0.545	0.827	van
Weighted Avg.	0.453	0.176	0.524	0.453	0.422	0.774	

=== Confusion Matrix ===

```

a   b   c   d   <-- classified as
88  60   0  64 |   a = opel
63  84   3  67 |   b = saab
38   9  37 134 |   c = bus
5    8  12 174 |   d = van

```

Modello Naive Bayes	
Class opel : $P(C) = 0.25058824$	
Attribute COMPACTNESS	
Mean: 95.06132075	Standard Deviation: 8.23073734
Attribute CIRCULARITY	
Mean: 46.58018868	Standard Deviation: 7.23526016
Attribute DISTANCE CIRCULARITY	
Mean: 89.08962264	Standard Deviation: 15.58667817
Attribute RADIUS RATIO	
Mean: 180.30188679	Standard Deviation: 31.35492276
Attribute PR.AXIS ASPECT RATIO	
Mean: 60.87735849	Standard Deviation: 4.9561161
Attribute MAX.LENGTH ASPECT RATIO	
Mean: 8.85849057	Standard Deviation: 1.98065851
Attribute SCATTER RATIO	
Mean: 182.16509434	Standard Deviation: 32.82188191
Attribute ELONGATEDNESS	
Mean: 37.87735849	Standard Deviation: 7.72415775
Attribute PR.AXIS RECTANGULARITY	
Mean: 21.58962264	Standard Deviation: 2.55303664
Attribute MAX.LENGTH RECTANGULARITY	
Mean: 151.27358491	Standard Deviation: 18.15809092
Attribute SCALED VARIANCE MAJOR	
Mean: 198.61792453	Standard Deviation: 28.84096381
Attribute SCALED VARIANCE MINOR	
Mean: 508.53773585	Standard Deviation: 172.45924992
Attribute SCALED RADIUS OF GYRATION	
Mean: 179.77358491	Standard Deviation: 34.72144618
Attribute SKEWNESS ABOUT MAJOR	
Mean: 70.14150943	Standard Deviation: 5.08028189
Attribute SKEWNESS ABOUT MINOR	
Mean: 6.60377358	Standard Deviation: 5.19328631
Attribute KURTOSIS ABOUT MAJOR	
Mean: 15.01415094	Standard Deviation: 10.16661145
Attribute KURTOSIS ABOUT MINOR	
Mean: 189.27830189	Standard Deviation: 5.59306451
Attribute HOLLOWS RATIO	
Continua...	

Modello Naive Bayes	
Mean: 197.11320755	Standard Deviation: 5.84608388
Class saab : $P(C) = 0.25647059$	
Attribute COMPACTNESS	
Mean: 97.28110599	Standard Deviation: 9.08138423
Attribute CIRCULARITY	
Mean: 45.53456221	Standard Deviation: 6.81908772
Attribute DISTANCE CIRCULARITY	
Mean: 88.67281106	Standard Deviation: 17.00514214
Attribute RADIUS RATIO	
Mean: 180.80184332	Standard Deviation: 30.80482761
Attribute PR.AXIS ASPECT RATIO	
Mean: 61.14285714	Standard Deviation: 4.32187129
Attribute MAX.LENGTH ASPECT RATIO	
Mean: 8.79262673	Standard Deviation: 2.15130043
Attribute SCATTER RATIO	
Mean: 179.66820276	Standard Deviation: 31.52380865
Attribute ELONGATEDNESS	
Mean: 38.31797235	Standard Deviation: 7.48673515
Attribute PR.AXIS RECTANGULARITY	
Mean: 21.44700461	Standard Deviation: 2.45292772
Attribute MAX.LENGTH RECTANGULARITY	
Mean: 148.69124424	Standard Deviation: 16.13460644
Attribute SCALED VARIANCE MAJOR	
Mean: 197.15207373	Standard Deviation: 27.92002315
Attribute SCALED VARIANCE MINOR	
Mean: 493.79723502	Standard Deviation: 163.15624014
Attribute SCALED RADIUS OF GYRATION	
Mean: 179.4562212	Standard Deviation: 33.77607594
Attribute SKEWNESS ABOUT MAJOR	
Mean: 69.75576037	Standard Deviation: 5.29152681
Attribute SKEWNESS ABOUT MINOR	
Mean: 7.65898618	Standard Deviation: 5.81694721
Attribute KURTOSIS ABOUT MAJOR	
Mean: 15.29953917	Standard Deviation: 10.06679773
Attribute KURTOSIS ABOUT MINOR	
Mean: 189.71428571	Standard Deviation: 4.99179221
Attribute HOLLOWS RATIO	
Mean: 198.04147465	Standard Deviation: 6.61039681
Class bus : $P(C) = 0.25764706$	
Attribute COMPACTNESS	
Mean: 91.59174312	Standard Deviation: 8.6180245
Attribute CIRCULARITY	
Mean: 45.06880734	Standard Deviation: 5.03076612
Attribute DISTANCE CIRCULARITY	
Mean: 76.72018349	Standard Deviation: 12.09256373
Attribute RADIUS RATIO	
Mean: 166.00458716	Standard Deviation: 30.57415217
Attribute PR.AXIS ASPECT RATIO	
Mean: 63.43119266	Standard Deviation: 8.80287654
Attribute MAX.LENGTH ASPECT RATIO	
Mean: 7.01376147	Standard Deviation: 4.75722042
Attribute SCATTER RATIO	
Mean: 170.02293578	Standard Deviation: 33.35681176
Attribute ELONGATEDNESS	
Mean: 40.1146789	Standard Deviation: 6.50137633
Attribute PR.AXIS RECTANGULARITY	
Continua...	

Modello Naive Bayes	
Mean: 20.5733945	Standard Deviation: 2.72835063
Attribute MAX.LENGTH RECTANGULARITY	
Mean: 146.70183486	Standard Deviation: 10.49239779
Attribute SCALED VARIANCE MAJOR	
Mean: 192.88990826	Standard Deviation: 33.99656785
Attribute SCALED VARIANCE MINOR	
Mean: 448.89449541	Standard Deviation: 193.13831358
Attribute SCALED RADIUS OF GYRATION	
Mean: 180.94954128	Standard Deviation: 31.24737371
Attribute SKEWNESS ABOUT MAJOR	
Mean: 77.12385321	Standard Deviation: 7.68494128
Attribute SKEWNESS ABOUT MINOR	
Mean: 4.8440367	Standard Deviation: 3.22053982
Attribute KURTOSIS ABOUT MAJOR	
Mean: 10.21100917	Standard Deviation: 6.87287247
Attribute KURTOSIS ABOUT MINOR	
Mean: 187.81192661	Standard Deviation: 7.32123009
Attribute HOLLOWS RATIO	
Mean: 191.32568807	Standard Deviation: 7.91832622
Class van : $P(C) = 0.23529412$	
Attribute COMPACTNESS	
Mean: 90.56281407	Standard Deviation: 3.87230833
Attribute CIRCULARITY	
Mean: 42.07035176	Standard Deviation: 4.08311074
Attribute DISTANCE CIRCULARITY	
Mean: 73.33165829	Standard Deviation: 10.86884867
Attribute RADIUS RATIO	
Mean: 147.12060302	Standard Deviation: 29.89452042
Attribute PR.AXIS ASPECT RATIO	
Mean: 61.26130653	Standard Deviation: 11.38056381
Attribute MAX.LENGTH ASPECT RATIO	
Mean: 9.71356784	Standard Deviation: 7.22463287
Attribute SCATTER RATIO	
Mean: 141.53768844	Standard Deviation: 13.99557422
Attribute ELONGATEDNESS	
Mean: 47.93969849	Standard Deviation: 4.66843926
Attribute PR.AXIS RECTANGULARITY	
Mean: 18.57788945	Standard Deviation: 1.0311872
Attribute MAX.LENGTH RECTANGULARITY	
Mean: 145.1758794	Standard Deviation: 11.01533463
Attribute SCALED VARIANCE MAJOR	
Mean: 164.01005025	Standard Deviation: 19.68001343
Attribute SCALED VARIANCE MINOR	
Mean: 298.20100503	Standard Deviation: 55.89465017
Attribute SCALED RADIUS OF GYRATION	
Mean: 157.27638191	Standard Deviation: 22.82119773
Attribute SKEWNESS ABOUT MAJOR	
Mean: 72.77889447	Standard Deviation: 8.8672224
Attribute SKEWNESS ABOUT MINOR	
Mean: 6.41708543	Standard Deviation: 4.66579099
Attribute KURTOSIS ABOUT MAJOR	
Mean: 9.69849246	Standard Deviation: 6.24334375
Attribute KURTOSIS ABOUT MINOR	
Mean: 188.93969849	Standard Deviation: 6.37992289
Attribute HOLLOWS RATIO	
Mean: 196.14572864	Standard Deviation: 7.34529805

Esecuzione REPTree

=== Run information ===

Scheme: weka.classifiers.trees.REPTree -M 2 -V 0.001 -N 3 -S 1 -L -1 -I 0.0
 Relation: vehicle
 Instances: 846
 Attributes: 19

COMPACTNESS
 CIRCULARITY
 DISTANCE CIRCULARITY
 RADIUS RATIO
 PR.AXIS ASPECT RATIO
 MAX.LENGTH ASPECT RATIO
 SCATTER RATIO
 ELONGATEDNESS
 PR.AXIS RECTANGULARITY
 MAX.LENGTH RECTANGULARITY
 SCALED VARIANCE_MAJOR
 SCALED VARIANCE_MINOR
 SCALED RADIUS OF GYRATION
 SKEWNESS ABOUT_MAJOR
 SKEWNESS ABOUT_MINOR
 KURTOSIS ABOUT_MAJOR
 KURTOSIS ABOUT_MINOR
 HOLLOWS RATIO
 Class

Test mode: 10-fold cross-validation

=== Stratified cross-validation ===

=== Summary ===

Correctly Classified Instances	612	72.3404 %
Incorrectly Classified Instances	234	27.6596 %
Kappa statistic	0.6313	
Mean absolute error	0.1617	
Root mean squared error	0.3109	
Relative absolute error	43.1254 %	
Root relative squared error	71.8227 %	
Total Number of Instances	846	

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0,575	0,137	0,584	0,575	0,580	0,440	0,842	0,586	opel
	0,475	0,129	0,560	0,475	0,514	0,366	0,795	0,561	saab
	0,945	0,040	0,892	0,945	0,918	0,889	0,968	0,895	bus
	0,910	0,063	0,815	0,910	0,860	0,816	0,972	0,879	van
Weighted Avg.	0,723	0,093	0,711	0,723	0,716	0,625	0,893	0,728	

=== Confusion Matrix ===

a	b	c	d	<-- classified as
122	68	8	14	a = opel
81	103	13	20	b = saab
1	4	206	7	c = bus
5	9	4	181	d = van

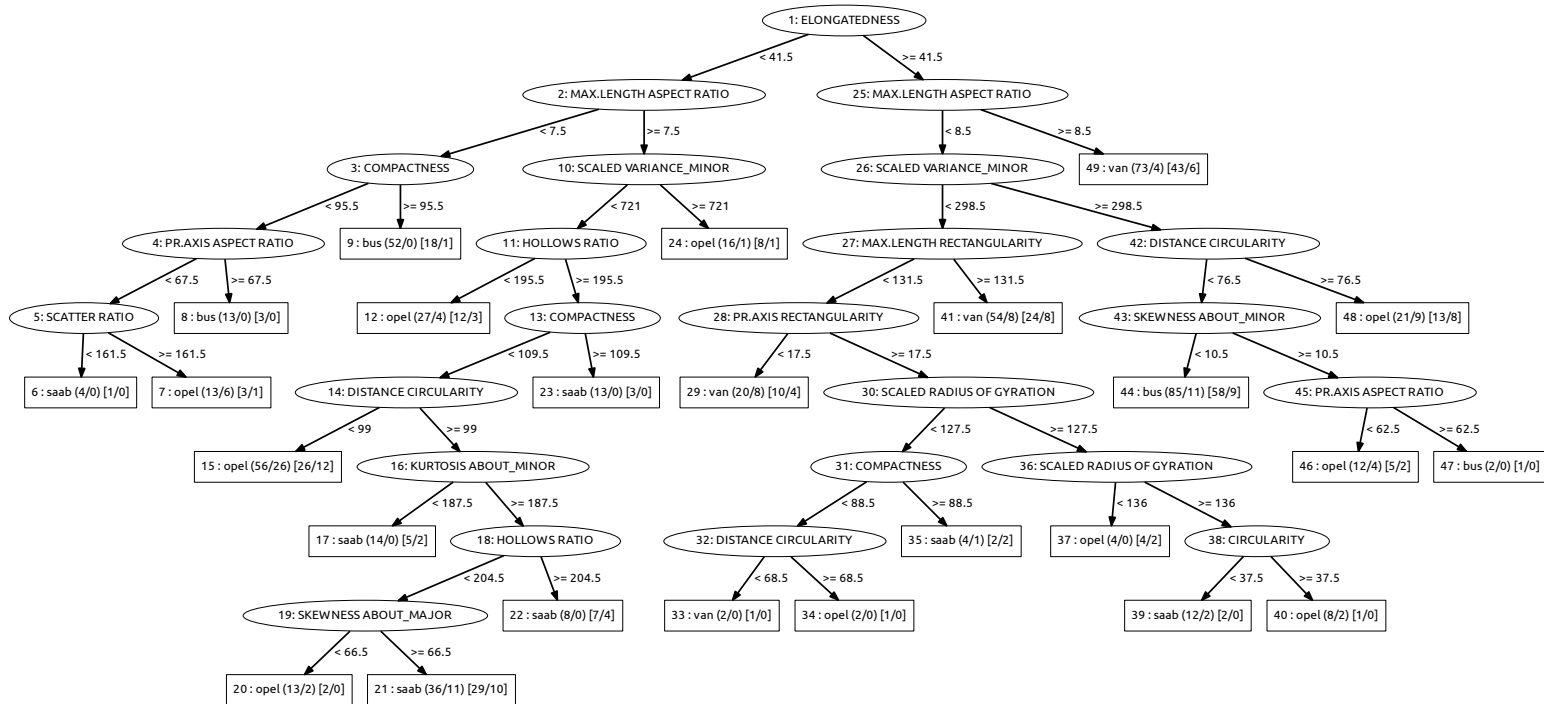


Figura 6.3: Modello di REPTree

Esecuzione JRip	
=== Run information ===	
Scheme:	weka.classifiers.rules.JRip -F 3 -N 2.0 -O 2 -S 1
Relation:	vehicle
Instances:	846
Attributes:	19
	COMPACTNESS
	CIRCULARITY
	DISTANCE CIRCULARITY
	RADIUS RATIO
	PR.AXIS ASPECT RATIO
	MAX.LENGTH ASPECT RATIO
	SCATTER RATIO
	ELONGATEDNESS
	PR.AXIS RECTANGULARITY
	MAX.LENGTH RECTANGULARITY
	SCALED VARIANCE_MAJOR
	SCALED VARIANCE_MINOR
	SCALED RADIUS OF GYRATION
	SKEWNESS ABOUT_MAJOR
	SKEWNESS ABOUT_MINOR
	KURTOSIS ABOUT_MAJOR
	KURTOSIS ABOUT_MINOR
	HOLLOWS RATIO

```

Class
Test mode: 10-fold cross-validation

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances      584          69.0307 %
Incorrectly Classified Instances    262          30.9693 %
Kappa statistic                    0.5868
Mean absolute error                 0.1914
Root mean squared error             0.3323
Relative absolute error             51.0598 %
Root relative squared error         76.7524 %
Total Number of Instances          846

=== Detailed Accuracy By Class ===

          TP Rate  FP Rate  Precision  Recall  F-Measure  MCC      ROC Area  PRC Area  Class
          0,481   0,126   0,560    0,481   0,518     0,374    0,785    0,545    opel
          0,484   0,146   0,533    0,484   0,507     0,349    0,793    0,501    saab
          0,940   0,099   0,768    0,940   0,845     0,792    0,940    0,812    bus
          0,864   0,043   0,860    0,864   0,862     0,820    0,927    0,861    van
Weighted Avg.  0,690   0,105   0,677    0,690   0,680     0,580    0,860    0,677

=== Confusion Matrix ===

  a   b   c   d  <-- classified as
102  71  27  12 |  a = opel
 74 105  27  11 |  b = saab
  2   6 205   5 |  c = bus
  4  15   8 172 |  d = van

```

Regole:

1. (ELONGATEDNESS ≥ 43) **AND** (MAX.LENGTH ASPECT RATIO ≥ 9) **AND** (DISTANCE CIRCULARITY ≥ 73) \Rightarrow Class=van (86.0/0.0)
2. (SCALED VARIANCE.MINOR ≤ 309) **AND** (MAX.LENGTH RECTANGULARITY ≥ 132) **AND** (DISTANCE CIRCULARITY ≤ 64) **AND** (SCALED RADIUS OF GYRATION ≤ 157) \Rightarrow Class=van (23.0/0.0)
3. (PR.AXIS RECTANGULARITY ≤ 18) **AND** (MAX.LENGTH RECTANGULARITY ≥ 128) **AND** (SCALED RADIUS OF GYRATION ≤ 140) \Rightarrow Class=van (42.0/6.0)
4. (SCALED VARIANCE.MINOR ≤ 309) **AND** (MAX.LENGTH RECTANGULARITY ≥ 142) \Rightarrow Class=van (33.0/5.0)
5. (ELONGATEDNESS ≥ 53) **AND** (SCALED RADIUS OF GYRATION ≥ 137) \Rightarrow Class=van (15.0/5.0)
6. (SCALED VARIANCE.MAJOR ≤ 177) **AND** (MAX.LENGTH ASPECT RATIO ≥ 10) \Rightarrow Class=van (8.0/1.0)
7. (MAX.LENGTH ASPECT RATIO ≥ 8) **AND** (MAX.LENGTH RECTANGULARITY ≥ 173) \Rightarrow Class=opel (45.0/8.0)
8. (MAX.LENGTH ASPECT RATIO ≥ 8) **AND** (COMPACTNESS ≤ 103) **AND** (ELONGATEDNESS ≤ 37) **AND** (HOLLOWS RATIO ≤ 195) \Rightarrow Class=opel (14.0/0.0)

9. (MAX.LENGTH ASPECT RATIO ≥ 8) **AND** (HOLLOWS RATIO ≤ 198) **AND** (KURTOSIS ABOUT_MINOR ≥ 189) \Rightarrow Class=opel (42.0/17.0)
10. (SKEWNESS ABOUT_MAJOR ≤ 67) **AND** (HOLLOWS RATIO ≤ 203) \Rightarrow Class=opel (66.0/30.0)
11. (SCALED RADIUS OF GYRATION ≤ 142) **AND** (HOLLOWS RATIO ≤ 194) **AND** (DISTANCE CIRCULARITY ≥ 57) \Rightarrow Class=opel (17.0/2.0)
12. (MAX.LENGTH ASPECT RATIO ≥ 9) **AND** (DISTANCE CIRCULARITY ≥ 100) **AND** (SCALED VARIANCE_MAJOR ≤ 231) \Rightarrow Class=saab (71.0/9.0)
13. (MAX.LENGTH ASPECT RATIO ≥ 9) **AND** (PR.AXIS ASPECT RATIO ≤ 61) \Rightarrow Class=saab (23.0/7.0)
14. (SCALED VARIANCE_MAJOR ≤ 165) **AND** (DISTANCE CIRCULARITY ≤ 66) \Rightarrow Class=saab (36.0/11.0)
15. (SKEWNESS ABOUT_MAJOR ≤ 72) **AND** (PR.AXIS ASPECT RATIO ≤ 65) **AND** (DISTANCE CIRCULARITY ≥ 81) **AND** (SKEWNESS ABOUT_MAJOR ≥ 66) \Rightarrow Class=saab (27.0/7.0)
16. (CIRCULARITY ≤ 40) **AND** (RADIUS RATIO ≤ 144) \Rightarrow Class=saab (16.0/6.0)
17. [*Empty Rule*] \Rightarrow Class=bus (282.0/69.0)

6.4 Risultati su Wisconsin Breast Cancer

L'esecuzione ha coinvolto 629 istanze di training e 70 istanze di testing per ogni fold.

Esecuzione NaiveBayesSimple

=== Run information ===

Scheme:weka.classifiers.bayes.NaiveBayesSimple

Relation: wisconsin-breast-cancer

Instances: 699

Attributes: 10

Clump_Thickness
Cell_Size_Uniformity
Cell_Shape_Uniformity
Marginal_Adhesion
Single_Epi_Cell_Size
Bare_Nuclei
Bland_Chromatin
Normal_Nucleoli
Mitoses
Class

Test mode:10-fold cross-validation

=== Stratified cross-validation ===

=== Summary ===

Correctly Classified Instances	671	95.9943 %
Incorrectly Classified Instances	28	4.0057 %
Kappa statistic	0.9127	
Mean absolute error	0.0405	
Root mean squared error	0.1987	
Relative absolute error	8.9562 %	
Root relative squared error	41.8102 %	
Total Number of Instances	699	

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area	Class
	0.952	0.025	0.986	0.952	0.969	0.99	benign
	0.975	0.048	0.914	0.975	0.944	0.984	malignant
Weighted Avg.	0.96	0.033	0.962	0.96	0.96	0.988	

=== Confusion Matrix ===

```
a  b  <-- classified as
436 22 |  a = benign
```

Modello Naive Bayes	
Class benign : $P(C) = 0.65477889$	
Attribute Clump_Thickness	
Mean: 2.95633188	Standard Deviation: 1.67431794
Attribute Cell_Size_Uniformity	
Mean: 1.32532751	Standard Deviation: 0.90769408
Attribute Cell_Shape_Uniformity	
Mean: 1.44323144	Standard Deviation: 0.99783575
Attribute Marginal_Adhesion	
Mean: 1.36462882	Standard Deviation: 0.99682975
Attribute Single_Epi_Cell_Size	
Mean: 2.12008734	Standard Deviation: 0.91712995
Attribute Bare_Nuclei	
Mean: 1.34684685	Standard Deviation: 1.1778482
Attribute Bland_Chromatin	
Mean: 2.10043668	Standard Deviation: 1.08033943
Attribute Normal_Nucleoli	
Mean: 1.29039301	Standard Deviation: 1.05885629
Attribute Mitoses	
Mean: 1.06331878	Standard Deviation: 0.50199548
Class malignant : $P(C) = 0.34522111$	
Attribute Clump_Thickness	
Mean: 7.19502075	Standard Deviation: 2.4288492
Attribute Cell_Size_Uniformity	
Mean: 6.57261411	Standard Deviation: 2.71951225
Attribute Cell_Shape_Uniformity	
Mean: 6.56016598	Standard Deviation: 2.56204475
Attribute Marginal_Adhesion	
Mean: 5.54771784	Standard Deviation: 3.21046547
Attribute Single_Epi_Cell_Size	
Mean: 5.29875519	Standard Deviation: 2.4516063
Attribute Bare_Nuclei	
Mean: 7.62761506	Standard Deviation: 3.116679
Attribute Bland_Chromatin	
Mean: 5.97925311	Standard Deviation: 2.2738516
Attribute Normal_Nucleoli	
Mean: 5.86307054	Standard Deviation: 3.3506724
Attribute Mitoses	
Mean: 2.58921162	Standard Deviation: 2.55793858

Esecuzione REPTree

=== Run information ===

Scheme:weka.classifiers.trees.REPTree -M 2 -V 0.001 -N 3 -S 1 -L -1

Relation: wisconsin-breast-cancer

Instances: 699

Attributes: 10
 Clump_Thickness
 Cell_Size_Uniformity
 Cell_Shape_Uniformity
 Marginal_Adhesion
 Single_Epi_Cell_Size
 Bare_Nuclei
 Bland_Chromatin
 Normal_Nucleoli
 Mitoses
 Class

Test mode:10-fold cross-validation

=== Stratified cross-validation ===

=== Summary ===

Correctly Classified Instances	656	93.8484 %
Incorrectly Classified Instances	43	6.1516 %
Kappa statistic	0.8653	
Mean absolute error	0.083	
Root mean squared error	0.2234	
Relative absolute error	18.3662 %	
Root relative squared error	47.0064 %	
Total Number of Instances	699	

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area	Class
	0.941	0.066	0.964	0.941	0.952	0.964	benign
	0.934	0.059	0.893	0.934	0.913	0.964	malignant
Weighted Avg.	0.938	0.064	0.94	0.938	0.939	0.964	

=== Confusion Matrix ===

a	b	<-- classified as
431	27	a = benign
16	225	b = malignant

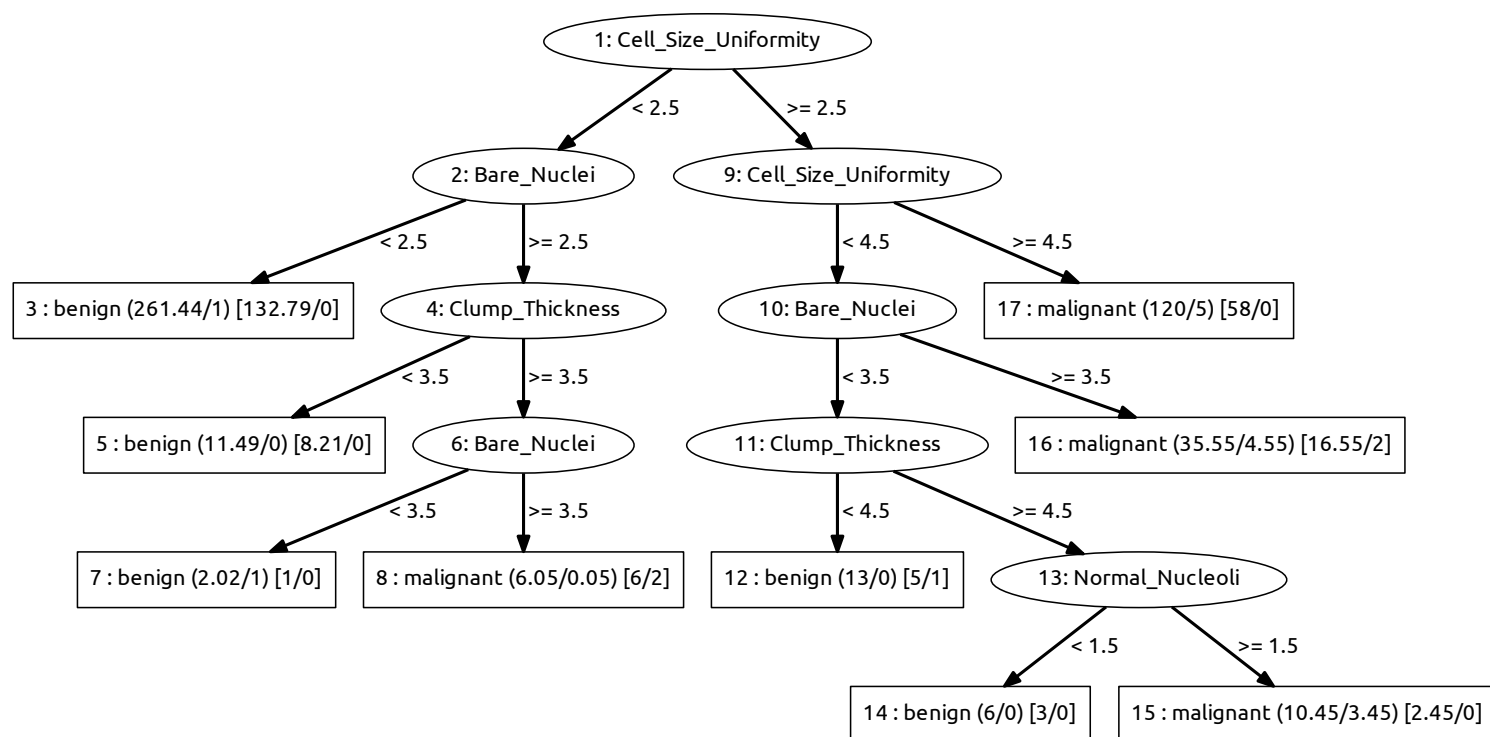


Figura 6.4: Modello di REPTree

Esecuzione JRip	
=== Run information ===	
Scheme:	weka.classifiers.rules.JRip -F 3 -N 2.0 -O 2 -S 1
Relation:	wisconsin-breast-cancer
Instances:	699
Attributes:	10
	Clump_Thickness
	Cell_Size_Uniformity
	Cell_Shape_Uniformity
	Marginal_Adhesion
	Single_Epi_Cell_Size
	Bare_Nuclei
	Bland_Chromatin
	Normal_Nucleoli
	Mitoses
	Class

Test mode:10-fold cross-validation

=== Stratified cross-validation ===

=== Summary ===

Correctly Classified Instances	667	95.422 %
Incorrectly Classified Instances	32	4.578 %
Kappa statistic	0.8999	
Mean absolute error	0.0618	
Root mean squared error	0.2022	
Relative absolute error	13.676 %	
Root relative squared error	42.5462 %	
Total Number of Instances	699	

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area	Class
	0.952	0.041	0.978	0.952	0.965	0.973	benign
	0.959	0.048	0.913	0.959	0.935	0.973	malignant
Weighted Avg.	0.954	0.044	0.955	0.954	0.954	0.973	

=== Confusion Matrix ===

a	b	<-- classified as
436	22	a = benign
10	231	b = malignant

Regole:

1. (Cell_Size_Uniformity ≥ 3) **AND** (Cell_Size_Uniformity ≥ 5) \Rightarrow Class=malignant (178.0/5.0)
2. (Bare_Nuclei ≥ 4) **AND** (Bare_Nuclei ≥ 7) \Rightarrow Class=malignant (48.0/4.0)
3. (Normal_Nucleoli ≥ 3) **AND** (Clump_Thickness ≥ 6) \Rightarrow Class=malignant (13.0/1.0)
4. (Bare_Nuclei ≥ 3) **AND** (Clump_Thickness ≥ 5) \Rightarrow Class=malignant (11.0/3.0)
5. (Marginal_Adhesion ≥ 8) \Rightarrow Class=malignant (2.0/0.0)
6. [Empty Rule] \Rightarrow Class=benign (447.0/2.0)

Capitolo 7: Analisi

In questo capitolo verranno analizzati i risultati dell'esperimento, ottenuti utilizzando un plugin di Microsoft Excel, **XLSTAT**. La configurazione ha riguardato i valori ottenuti sui quattro dataset dai tre algoritmi di interesse: Naive Bayes, REPTree e RIPPER.

7.1 Test

Per confrontare gli algoritmi è necessario definire due ipotesi, quella nulla H_0 e quella alternativa H_1 :

- H_0 : non ci sono differenze significative tra le variabili.
- H_1 : almeno due variabili sono significativamente diverse tra di loro.

Inoltre bisogna fare attenzione a scegliere lo strumento statistico adeguato, tenendo conto di alcuni aspetti:

- Non possono essere fatte considerazioni sul tipo di distribuzione che i dati assumono.
- È necessario uno test che vada bene per più di due algoritmi.
- I dati non sono indipendenti tra di loro.

Per questi motivi si è scelto di utilizzare il test di **Friedman**, che è un test non-parametrico utile per gestire dati accoppiati e per funzionare con più di due algoritmi[4].

La metrica utilizzata è la *F-Measure*, che è la media armonica di precisione e richiamo. Per fini di completezza riportiamo le relative formule:

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives}$$

$$Recall = \frac{True\ Positives}{True\ Positives + False\ Negatives}$$

$$F\text{-measure} = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$$

Esecuzione

Di seguito vengono mostrati i dati e i risultati di esecuzione del test:

	NB	REPTree	RIPPER
german credit	0,748	0,703	0,697
hepatitis	0,848	0,72	0,768
vehicle	0,422	0,716	0,68
wisconsin bc	0,96	0,939	0,954

Tabella 7.1: F-measure dei vari dataset

Valore osservato	1,500
Valore critico	5,991
Gradi di libertà	2
p-value	0,472
α	0,05

Tabella 7.2: Test di Friedman

7.2 Interpretazione dei risultati

Visto che il p-value è più grande del livello di significatività $\alpha = 0.05$, non si può rigettare l'ipotesi nulla H_0 , quindi gli algoritmi non si comportano diversamente gli uni dagli altri in modo statisticamente significativo.

Capitolo 8: Conclusioni

In questo documento sono stati confrontati tre algoritmi, Naive Bayes, REP-Tree e RIPPER, su quattro dataset provenienti dall'UCI Machine Learning Repository.

I risultati hanno evidenziato un pareggio tra le tre tecniche di classificazione, senza alcuna differenza significativa tra di esse.

In futuro si potrebbero eseguire ulteriori test su nuovi dataset e con nuove misure per vagliare altri aspetti.

Bibliografia

- [1] C. Brunk and M. J. Pazzani. An investigation of noise-tolerant relational concept learning algorithms, 1991.
- [2] W. W. Cohen. Efficient pruning methods for separate-and-conquer rule learning systems. In *In Proceedings of the 13th International Joint Conference on Artificial Intelligence*, pages 988–994. Morgan Kaufmann, 1993.
- [3] W. W. Cohen. Fast effective rule induction. In *In Proceedings of the Twelfth International Conference on Machine Learning*, pages 115–123. Morgan Kaufmann, 1995.
- [4] J. Demšar. Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.*, 7:1–30, Dec. 2006. ISSN 1532-4435.
- [5] P. Domingos and M. Pazzani. On the optimality of the simple bayesian classifier under zero-one loss. *Mach. Learn.*, 29(2-3):103–130, Nov. 1997. ISSN 0885-6125.
- [6] F. Esposito, D. Malerba, and G. Semeraro. A comparative analysis of methods for pruning decision trees. *IEEE Trans. Pattern Anal. Mach. Intell.*, 19(5):476–491, May 1997. ISSN 0162-8828.
- [7] J. Fürnkranz and G. Widmer. Incremental reduced error pruning, 1994.
- [8] T. M. Mitchell. *Machine Learning*, chapter 6. McGraw-Hill, Inc., New York, NY, USA, 1 edition, 1997. ISBN 0070428077, 9780070428072.
- [9] G. Pagallo and D. Haussler. Boolean feature discovery in empirical learning. *Mach. Learn.*, 5(1):71–99, May 1990. ISSN 0885-6125.
- [10] J. Quinlan. Learning logical definitions from relations. *Machine Learning*, 5(3):239–266, 1990. ISSN 1573-0565.

- [11] J. R. Quinlan. Simplifying decision trees. *Int. J. Man-Mach. Stud.*, 27 (3):221–234, Sept. 1987. ISSN 0020-7373.
- [12] J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993. ISBN 1-55860-238-0.
- [13] J. R. Quinlan. MDL and categorical theories (continued). In *In Machine Learning: Proceedings of the Twelfth International Conference, Lake Tahoe*, pages 464–470. Morgan Kaufmann, 1995.
- [14] J. R. Quinlan and R. L. Rivest. Inferring decision trees using the minimum description length principle. *Inf. Comput.*, 80(3):227–248, Mar. 1989. ISSN 0890-5401.
- [15] S. J. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Pearson Education, 2 edition, 2003. ISBN 0137903952.
- [16] C. E. Shannon. A Mathematical Theory of Communication. *The Bell System Technical Journal*, 27(3):379–423, 1948.
- [17] S. M. Weiss and N. Indurkha. Reduced complexity rule induction. In *In Proceedings of the 12th International Joint Conference on Artificial Intelligence (IJCAI-91)*, pages 678–684. Morgan Kaufmann, 1991.
- [18] I. H. Witten, E. Frank, and M. A. Hall. Decision trees. In *Data Mining: Practical Machine Learning Tools and Techniques*, chapter 6.2, page 206. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 3rd edition, 2011. ISBN 0123748569, 9780123748560.
- [19] I. H. Witten, E. Frank, and M. A. Hall. Decision trees. In *Data Mining: Practical Machine Learning Tools and Techniques*, chapter 11.4, page 456. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 3rd edition, 2011. ISBN 0123748569, 9780123748560.
- [20] I. H. Witten, E. Frank, and M. A. Hall. Decision trees. In *Data Mining: Practical Machine Learning Tools and Techniques*, chapter 6.11, page 303. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 3rd edition, 2011. ISBN 0123748569, 9780123748560.