



Decision trees

Sistemi informativi per le Decisioni

Slide a cura di Andrew W. Moore (Carnegie Mellon)



Classificazione e predizione

- estrarre modelli che descrivono classi di dati
 - predire valori categorici
 - es.: categorizzare le richieste di prestito a un istituto bancario in “sicura” e “rischiosa”
- effettuare previsioni riguardo ad andamenti futuri
 - predire valori continui
 - es.: prevedere le spese in materiale hi-tech a partire dal reddito e dall’occupazione dei clienti
- tecniche note da tempo, ma limitatamente a piccole dimensioni dei dati, manipolabili in memoria principale
 - estensioni e ottimizzazioni per la memoria secondaria



Classificazione

- processo in due passi:
 - costruzione del modello a partire da un insieme predeterminato di classi o concetti (**training set**)
 - predizione delle classi per i dati al di fuori del training set



Creazione del modello

- si analizzano le tuple e gli attributi di una parte del DB
- il training set deve essere selezionato a caso
 - supervisionato (si assume che esista un attributo di etichetta = **class label**)
 - non supervisionato (**clustering**, le classi vengono apprese)
- il modello appreso viene espresso in forma di regole di classificazione o formule matematiche
- il modello ha la class label come output, e non come input



Utilizzo del modello per predizione

- è necessario stimare l'accuratezza del classificatore
 - test set di dati etichettati correttamente
 - si applica il modello e si producono le etichette
 - si valuta la percentuale di concordanze tra le etichette corrette e quello prodotte
- è necessario utilizzare un test set diverso dal training set, per evitare il fenomeno di *overfitting*
 - il modello appreso potrebbe derivare da specificità del training set, riusare quest'ultimo come test set porterebbe a sovrastimare la sua qualità



Predizione

- costruzione ed uso di un modello per stabilire il valore o l'intervallo di valori che un attributo non noto dovrebbe avere
 - es.: nota la classe frequentata da uno studente si può predire l'anno di nascita, con una certa probabilità
 - regressione



Argomenti

- Information Gain per misurare le associazioni tra input e output
- Apprendimento dai dati di un albero di decisione per la classificazione

Ecco un dataset

- 48000 record, 16 attributi [Kohavi, 1995]

age	employe	education	edun	marital	...	job	relation	race	gender	hour	country	wealth
39	State_gov	Bachelors	13	Never_mar	...	Adm_cleric	Not_in_fan	White	Male	40	United_Stat	poor
51	Self_emp_	Bachelors	13	Married	...	Exec_man	Husband	White	Male	13	United_Stat	poor
39	Private	HS_grad	9	Divorced	...	Handlers_c	Not_in_fan	White	Male	40	United_Stat	poor
54	Private	11th	7	Married	...	Handlers_c	Husband	Black	Male	40	United_Stat	poor
28	Private	Bachelors	13	Married	...	Prof_speci	Wife	Black	Female	40	Cuba	poor
38	Private	Masters	14	Married	...	Exec_man	Wife	White	Female	40	United_Stat	poor
50	Private	9th	5	Married_sp	...	Other_serv	Not_in_fan	Black	Female	16	Jamaica	poor
52	Self_emp_	HS_grad	9	Married	...	Exec_man	Husband	White	Male	45	United_Stat	rich
31	Private	Masters	14	Never_mar	...	Prof_speci	Not_in_fan	White	Female	50	United_Stat	rich
42	Private	Bachelors	13	Married	...	Exec_man	Husband	White	Male	40	United_Stat	rich
37	Private	Some_coll	10	Married	...	Exec_man	Husband	Black	Male	80	United_Stat	rich
30	State_gov	Bachelors	13	Married	...	Prof_speci	Husband	Asian	Male	40	India	rich
24	Private	Bachelors	13	Never_mar	...	Adm_cleric	Own_child	White	Female	30	United_Stat	poor
33	Private	Assoc_acc	12	Never_mar	...	Sales	Not_in_fan	Black	Male	50	United_Stat	poor
41	Private	Assoc_voc	11	Married	...	Craft_repa	Husband	Asian	Male	40	*MissingV	rich
34	Private	7th_8th	4	Married	...	Transport_	Husband	Amer_Indi	Male	45	Mexico	poor
26	Self_emp_	HS_grad	9	Never_mar	...	Farming_fi	Own_child	White	Male	35	United_Stat	poor
33	Private	HS_grad	9	Never_mar	...	Machine_c	Unmarried	White	Male	40	United_Stat	poor
38	Private	11th	7	Married	...	Sales	Husband	White	Male	50	United_Stat	poor
44	Self_emp_	Masters	14	Divorced	...	Exec_man	Unmarried	White	Female	45	United_Stat	rich
41	Private	Doctorate	16	Married	...	Prof_speci	Husband	White	Male	60	United_Stat	rich
:	:	:	:	:	:	:	:	:	:	:	:	:



Classificazione

- Dato un attributo (es.: ricchezza), cercare di predire il valore di ricchezza di altre persone utilizzando gli altri attributi
- Si applica in genere ad attributi categorici (in output)
 - Attributo categorico: un attributo i cui valori possibili sono in numero finito (e limitato)
 - Confrontare con attributi reali

A proposito del dataset

- È un sottoinsieme dei dati del US Census 1990
- È disponibile online sul sito dell'UCI Machine Learning Datasets repository

Used Attributes

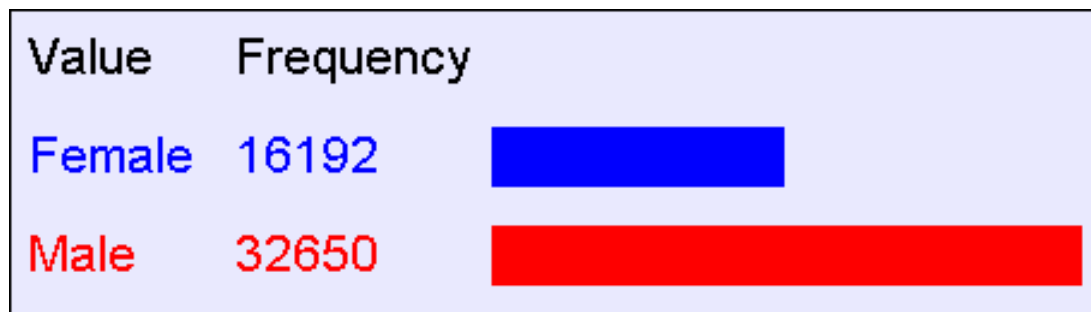
age	edunum	race	hours_worked
employment	marital	gender	country
taxweighting	job	capitalgain	wealth
education	relation	capitalloss	agegroup

This color = Real-valued This color = Symbol-valued

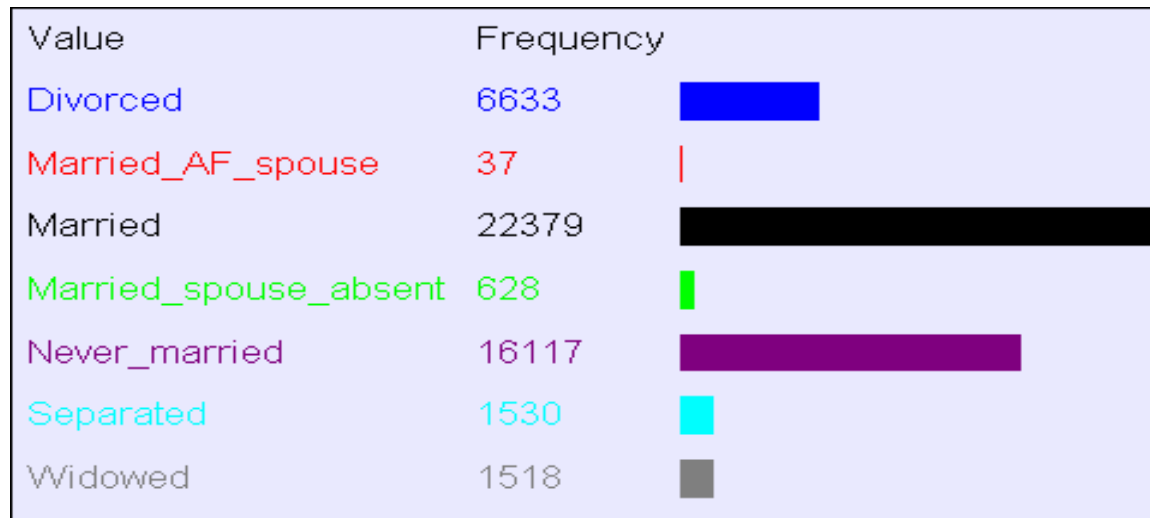
Successfully loaded a new dataset from the file \tadult.fds. It has 16 attributes and 48842 records.

Che ci facciamo con il dataset?

- Possiamo guardare gli istogrammi ...



Sesso



Stato
civile



Tabelle di contingenza

- Altro nome per un istogramma:

Una tabella di contingenza unidimensionale

- Algoritmo per creare tabelle di contingenza a k dimensioni:
 - Selezionare k attributi dal dataset:
 a_1, a_2, \dots, a_k
 - Per ogni possibile combinazione di valori,
 $a_1 = x_1, a_2 = x_2, \dots, a_k = x_k$,
memorizzare la frequenza dei record con tali valori
 - Chiamata anche “datacube k -dimensionale”








Una tabella di contingenza 2-d

wealth values:		poor	rich
agegroup	10s	2507	3
	20s	11262	743
	30s	9468	3461
	40s	6738	3986
	50s	4110	2509
	60s	2245	809
	70s	668	147
	80s	115	16
	90s	42	13

- Per ogni coppia di valori degli attributi (età, ricchezza) possiamo vedere il numero di record corrispondenti

Una tabella di contingenza 2-d

wealth values: poor rich

agegroup	10s	2507	3	
	20s	11262	743	
	30s	9468	3461	
	40s	6738	3986	
	50s	4110	2509	
	60s	2245	809	
	70s	668	147	
	80s	115	16	
	90s	42	13	

- Più semplice da valutare graficamente

Una tabella di contingenza 2-d

wealth values:		poor	rich	
agegroup	10s	2507	3	
	20s	11262	743	
	30s	9468	3461	
	40s	6738	3986	
	50s	4110	2509	
	60s	2245	809	
	70s	668	147	
	80s	115	16	
	90s	42	13	

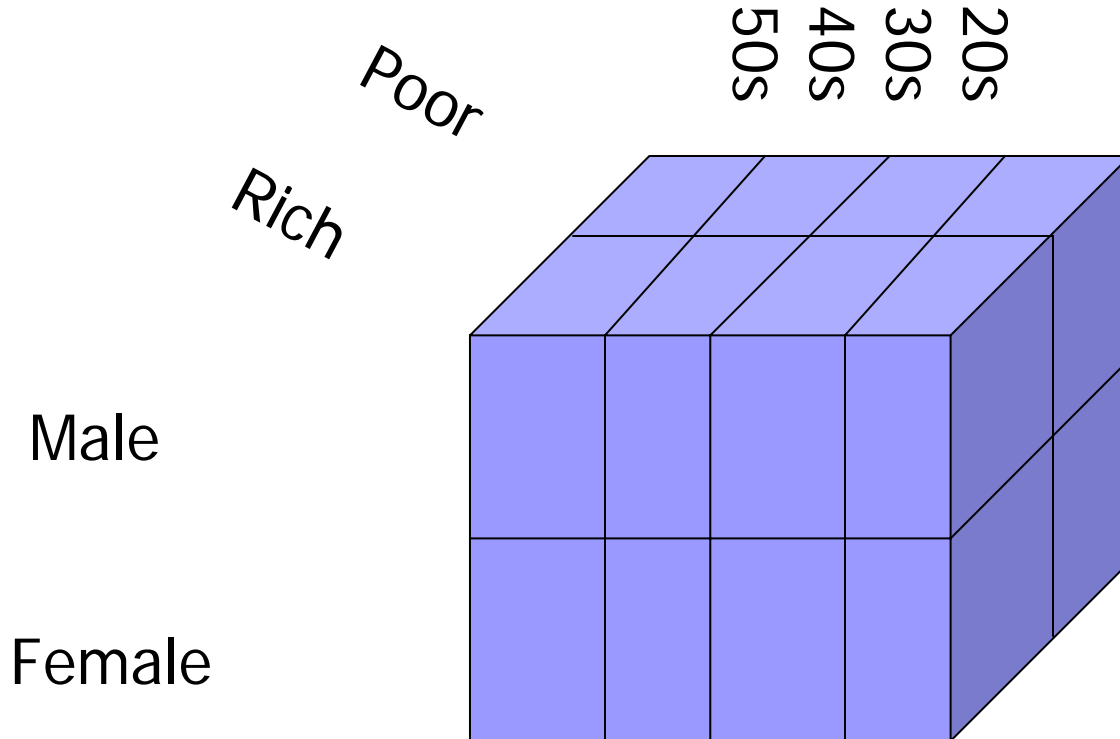
- Più semplice vedere casi “interessanti” se normalizziamo gli istogrammi

Una tabella di contingenza 2-d un po' più grande

job values:		Adm_clerical	Craft_repair	Farming_fishing	Machine_op_inspct	Priv_house_serv	Protective_serv	Tech_support									
MissingValue		Armed_Forces	Exec_managerial	Handlers_cleaners	Other_service	Prof_specialty	Sales	Transport_moving									
marital	Divorced	270	1192	0	679	890	90	197	434	762	46	795	121	664	239	254	
	Married_AF_spouse	5	6	0	4	3	1	1	1	5	0	4	1	5	0	1	
	Married	928	1495	7	3818	3600	869	724	1469	1088	27	3182	583	2491	609	1489	
	Married_spouse_absent	45	84	0	77	52	35	32	37	92	9	64	7	55	9	30	
	Never_married	1242	2360	8	1301	1260	434	1029	872	2442	99	1849	237	1992	506	486	
	Separated	97	224	0	160	126	23	63	123	275	21	145	23	146	48	56	
	Widowed	222	250	0	73	155	38	26	86	259	40	133	11	151	35	39	

Tabelle di contingenza 3-d

- Più difficili da comprendere!





Fermiamoci un po' a pensare ...

- Perché dovremmo volere studiare le tabelle di contingenza?
- Con 16 attributi, quante tabelle di contingenza 1-d esistono?
- Quante tabelle di contingenza 2-d esistono?
- Quante tabelle di contingenza 3-d?
- Con 100 attributi quante tabelle di contingenza 3-d esistono?

Fermiamoci un po' a pensare ...

- Perché dovremmo volere studiare le tabelle di contingenza?
- Con 16 attributi, quante tabelle di contingenza 1-d esistono? 16
- Quante tabelle di contingenza 2-d esistono? $16*15/2 = 120$
- Quante tabelle di contingenza 3-d? 560
- Con 100 attributi quante tabelle di contingenza 3-d esistono? 161700



Possiamo analizzare “manualmente” le tabelle di contingenza?

- Studiare 1 tabella di contingenza:
 - può essere interessante
- Studiare 10 tabelle:
 - già meno interessante
- Studiare 100 tabelle:
 - abbastanza noioso
- Studiare 100000 tabelle:
 - decisamente noioso

I bit...

- Supponiamo di osservare un insieme di valori casuali di un attributo X
- X ha solo quattro possibili valori

$P(X=A) = 1/4$	$P(X=B) = 1/4$	$P(X=C) = 1/4$	$P(X=D) = 1/4$
----------------	----------------	----------------	----------------

- Potremmo osservare: BAACBADCDADDDA...
- Trasmettiamo i dati su un collegamento seriale binario
- Codifichiamo ogni valore con 2 bit (es.: $A = 00$, $B = 01$, $C = 10$, $D = 11$)
- 0100001001001110110011111100...

Meno bit...

- Supponiamo che qualcuno ci dica che le probabilità non sono tutte uguali

$P(X=A) = 1/2$	$P(X=B) = 1/4$	$P(X=C) = 1/8$	$P(X=D) = 1/8$
----------------	----------------	----------------	----------------

- È possibile inventare una codifica che usi (in media) solamente 1.75 bit per simbolo. Come?
- Per esempio:
 - $A=0$, $B=10$, $C=110$, $D=111$

Ancora meno bit...

- Supponiamo di avere tre valori equiprobabili

$P(X=A) = 1/3$	$P(X=B) = 1/3$	$P(X=C) = 1/3$
----------------	----------------	----------------

- È possibile codificare i valori con 2 bit
 - A=00, B=01, C=10
- Possiamo ideare una codifica che usi solamente 1.6 bit per simbolo in media?
- In teoria, il risultato ottimo si ottiene con 1.58496 bit per simbolo

Caso generale

- Supponiamo che X assuma m valori: V_1, V_2, \dots, V_m

$$\boxed{P(X=V_1) = p_1 \mid P(X=V_2) = p_2 \mid \dots \mid P(X=V_m) = p_m}$$

- Qual è il minimo numero di bit per simbolo necessari, in media, per trasmettere un insieme di simboli ottenuti dalla distribuzione di X ?

$$H(X) = -p_1 \log_2 p_1 - p_2 \log_2 p_2 - \dots - p_m \log_2 p_m = -\sum_{j=1}^m p_j \log_2 p_j$$

- $H(X)$ = Entropia di X
 - “Entropia elevata” significa che X ha una distribuzione uniforme (noiosa)
 - “Entropia bassa” significa che X ha una distribuzione variegata (alti e bassi)

Caso generale

- Supponiamo che X assuma m valori: V_1, V_2, \dots, V_m

$P(X=V_1) = p_1$	$P(X=V_2) = p_2$	$P(X=V_3) = p_3$	\dots	$P(X=V_m) = p_m$
------------------	------------------	------------------	---------	------------------

- Qualche bit per sistema di valori di X sarebbe "piatto"

Un istogramma della distribuzione dei valori di X sarebbe "piatto"

Un istogramma della distribuzione dei valori di X avrebbe molti bassi e solo pochi "picchi"

$$H(X) = -p_1 \log_2 p_1 - p_2 \log_2 p_2 - \dots - p_m \log_2 p_m = -\sum_{j=1}^m p_j \log_2 p_j$$

- $H(X)$ = Entropia di X
 - "Entropia elevata" significa che X ha una distribuzione uniforme (noiosa)
 - "Entropia bassa" significa che X ha una distribuzione variegata (alti e bassi)

Caso generale

- Supponiamo che X assuma m valori: V_1, V_2, \dots, V_m

$P(X=V_1) = p_1$	$P(X=V_2) = p_2$	$P(X=V_3) = p_3$	\dots	$P(X=V_m) = p_m$
------------------	------------------	------------------	---------	------------------

- Qualche bit per sistema di dati
 media della distribuzione dei valori di X sarebbe "piatto"

Un istogramma della distribuzione dei valori di X sarebbe "piatto"

Un istogramma della distribuzione dei valori di X avrebbe molti bassi e solo pochi "picchi"

$$H(X) = -p_1 \log_2 p_1 - p_2 \log_2 p_2 - \dots - p_m \log_2 p_m = -\sum_{j=1}^m p_j \log_2 p_j$$

- $H(X)$ = Entropia
 - "Entropia alta" significa che X ha una distribuzione variegata (alti e bassi)
 - ... quindi i valori dei dati sarebbero presi un po' ovunque
 - "Entropia bassa" significa che X ha una distribuzione variegata (alti e bassi)
 - ... quindi i valori dei dati sarebbero più predicibili

Entropia condizionata specifica $H(Y|X=v)$

- Supponiamo di predire il valore di Y se conosciamo X
 X = College Major
 Y = Likes “Gladiator”
- Assumiamo che i valori riflettano le vere probabilità
- Dai dati stimiamo
 - $P(\text{LikeG} = \text{Yes}) = 0.5$
 - $P(\text{Major} = \text{Math} \ \& \ \text{LikeG} = \text{No}) = 0.25$
 - $P(\text{Major} = \text{Math}) = 0.5$
 - $P(\text{LikeG} = \text{Yes} \mid \text{Major} = \text{History}) = 0$
- Nota:
 - $H(X) = 1.5$
 - $H(Y) = 1$

X	Y
Math	Yes
History	No
CS	Yes
Math	No
Math	No
CS	Yes
History	No
Math	Yes

Entropia condizionata specifica $H(Y|X=v)$

- Definizione di Entropia condizionata specifica:

$H(Y|X=v)$ = L'entropia di Y solamente in quei record nei quali X assume il valore v

- **Esempio:**

- $H(Y|X=Math) = 1$
- $H(Y|X=History) = 0$
- $H(Y|X=CS) = 0$

X	Y
Math	Yes
History	No
CS	Yes
Math	No
Math	No
CS	Yes
History	No
Math	Yes

Entropia condizionata $H(Y|X)$

- Definizione di Entropia condizionata:

$H(Y|X)$ = La media delle entropie condizionate di Y

- = selezionando a caso un record quale sarebbe l'entropia di Y, condizionata al valore specifico di X
- = numero medio di bit necessari per trasmettere Y se sorgente e destinazione conoscono il valore di X
- = $\sum_j Prob(X=v_j) H(Y|X=v_j)$

X	Y
Math	Yes
History	No
CS	Yes
Math	No
Math	No
CS	Yes
History	No
Math	Yes

Entropia condizionata $H(Y|X)$

- Definizione di Entropia condizionata:

$$= \sum_j \text{Prob}(X=v_j) H(Y|X=v_j)$$

v_j	$\text{Prob}(X=v_j)$	$H(Y X=v_j)$
Math	0.5	1
History	0.25	0
CS	0.25	0

$$H(Y|X) = 0.5*1 + 0.25*0 + 0.25*0 = 0.5$$

X	Y
Math	Yes
History	No
CS	Yes
Math	No
Math	No
CS	Yes
History	No
Math	Yes

Information Gain

- Definizione di Information Gain:
= Devo trasmettere Y.
Quanti bit, in media, potrei risparmiare se entrambi conoscessimo X?
- $IG(Y|X) = H(Y) - H(Y|X)$
- Esempio:
 - $H(Y) = 1$
 - $H(Y|X) = 0.5$
 - Quindi $IG(Y|X) = 1 - 0.5 = 0.5$

X	Y
Math	Yes
History	No
CS	Yes
Math	No
Math	No
CS	Yes
History	No
Math	Yes

Esempio su Information Gain

wealth values: poor rich

gender Female 14423 1769  $H(\text{wealth} \mid \text{gender} = \text{Female}) = 0.497654$

Male 22732 9918  $H(\text{wealth} \mid \text{gender} = \text{Male}) = 0.885847$

$H(\text{wealth}) = 0.793844$ $H(\text{wealth} \mid \text{gender}) = 0.757154$

$IG(\text{wealth} \mid \text{gender}) = 0.0366896$

Altro esempio

wealth values: poor rich

agegroup	10s	2507	3		$H(\text{wealth} \mid \text{agegroup} = 10s) = 0.0133271$
	20s	11262	743		$H(\text{wealth} \mid \text{agegroup} = 20s) = 0.334906$
	30s	9468	3461		$H(\text{wealth} \mid \text{agegroup} = 30s) = 0.838134$
	40s	6738	3986		$H(\text{wealth} \mid \text{agegroup} = 40s) = 0.951961$
	50s	4110	2509		$H(\text{wealth} \mid \text{agegroup} = 50s) = 0.957376$
	60s	2245	809		$H(\text{wealth} \mid \text{agegroup} = 60s) = 0.834049$
	70s	668	147		$H(\text{wealth} \mid \text{agegroup} = 70s) = 0.680882$
	80s	115	16		$H(\text{wealth} \mid \text{agegroup} = 80s) = 0.535474$
	90s	42	13		$H(\text{wealth} \mid \text{agegroup} = 90s) = 0.788941$

$H(\text{wealth}) = 0.793844$ $H(\text{wealth} \mid \text{agegroup}) = 0.709463$

$IG(\text{wealth} \mid \text{agegroup}) = 0.0843813$

Information Gain relativo

- Definizione di Information Gain relativo:
= Devo trasmettere Y.
Che percentuale di bit, in media, potrei risparmiare se entrambi conoscessimo X?
- $RIG(Y|X) = (H(Y) - H(Y|X)) / H(Y)$
- Esempio:
 - $H(Y) = 1$
 - $H(Y|X) = 0.5$
 - Quindi $RIG(Y|X) = (1 - 0.5) / 1 = 0.5$

X	Y
Math	Yes
History	No
CS	Yes
Math	No
Math	No
CS	Yes
History	No
Math	Yes



A cosa ci serve l'Information Gain?

- Supponiamo di voler predire se l'aspettativa media di vita superi gli 80 anni
- Dai dati storici troviamo che:
 - $IG(\text{LongLife}|\text{HairColor}) = 0.01$
 - $IG(\text{LongLife}|\text{Smoker}) = 0.2$
 - $IG(\text{LongLife}|\text{Gender}) = 0.25$
 - $IG(\text{LongLife}|\text{LastDigitOfSSN}) = 0.00001$
- IG ci dice quanto può essere interessante una tabella di contingenza 2-d

Alla ricerca di IG elevati

- Dato un attributo (es.: wealth) che cerchiamo di predire, possiamo chiedere al computer di calcolare quale attributo abbia il maggior Information Gain.

wealth values: poor rich

relation	Husband	10870	8846		$H(\text{wealth} \mid \text{relation} = \text{Husband}) = 0.992385$
	Not_in_family	11307	1276		$H(\text{wealth} \mid \text{relation} = \text{Not_in_family}) = 0.473439$
	Other_relative	1454	52		$H(\text{wealth} \mid \text{relation} = \text{Other_relative}) = 0.216617$
	Own_child	7470	111		$H(\text{wealth} \mid \text{relation} = \text{Own_child}) = 0.110192$
	Unmarried	4816	309		$H(\text{wealth} \mid \text{relation} = \text{Unmarried}) = 0.328606$
	Wife	1238	1093		$H(\text{wealth} \mid \text{relation} = \text{Wife}) = 0.997207$

$H(\text{wealth}) = 0.793844$ $H(\text{wealth} \mid \text{relation}) = 0.628421$
 $IG(\text{wealth} \mid \text{relation}) = 0.165423$



Creazione di Decision Tree

- Un Decision Tree è una strutturazione gerarchica di un insieme di attributi da controllare per cercare di predire l'attributo di output
- Per decidere quale attributo controllare per primo, cerchiamo quello che fornisce l'information gain più elevato
- Quindi ricorsione ...

Un piccolo esempio: Miglia Per Gallone

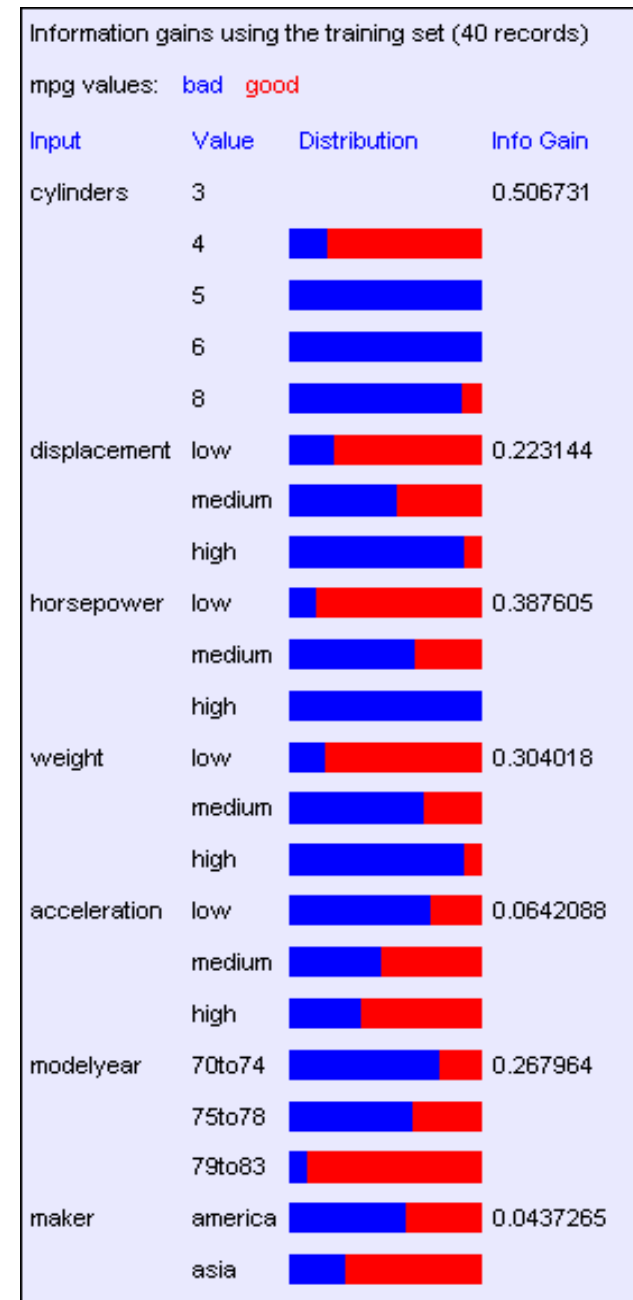
40 record

mpg	cylinders	displacement	horsepower	weight	acceleration	modelyear	maker
good	4	low	low	low	high	75to78	asia
bad	6	medium	medium	medium	medium	70to74	america
bad	4	medium	medium	medium	low	75to78	europa
bad	8	high	high	high	low	70to74	america
bad	6	medium	medium	medium	medium	70to74	america
bad	4	low	medium	low	medium	70to74	asia
bad	4	low	medium	low	low	70to74	asia
bad	8	high	high	high	low	75to78	america
:	:	:	:	:	:	:	:
:	:	:	:	:	:	:	:
:	:	:	:	:	:	:	:
bad	8	high	high	high	low	70to74	america
good	8	high	medium	high	high	79to83	america
bad	8	high	high	high	low	75to78	america
good	4	low	low	low	low	79to83	america
bad	6	medium	medium	medium	high	75to78	america
good	4	medium	low	low	low	79to83	america
good	4	low	low	medium	high	79to83	america
bad	8	high	high	high	low	70to74	america
good	4	low	medium	low	medium	75to78	europa
bad	5	medium	medium	medium	medium	75to78	europa

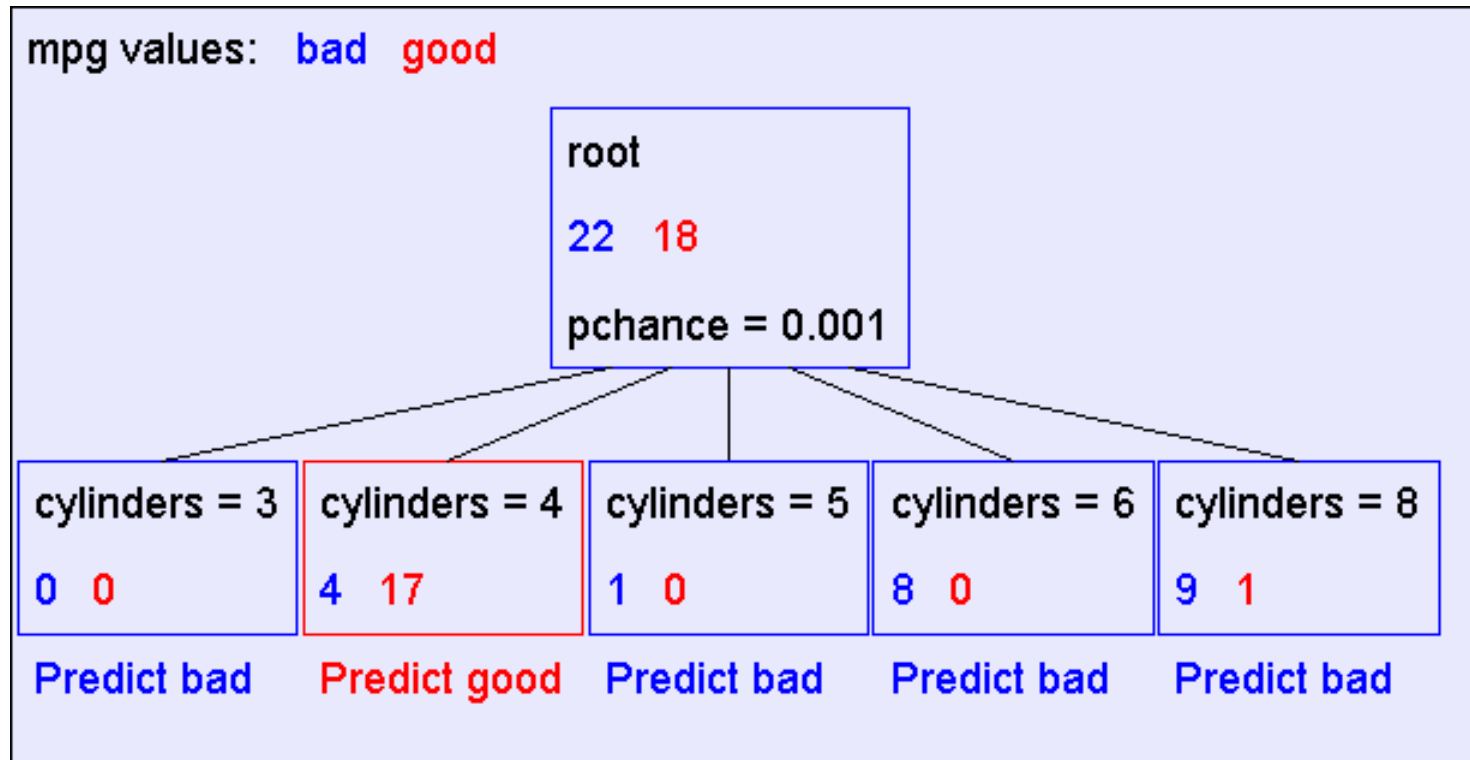
Dall'UCI repository

Supponiamo di voler predire MPG

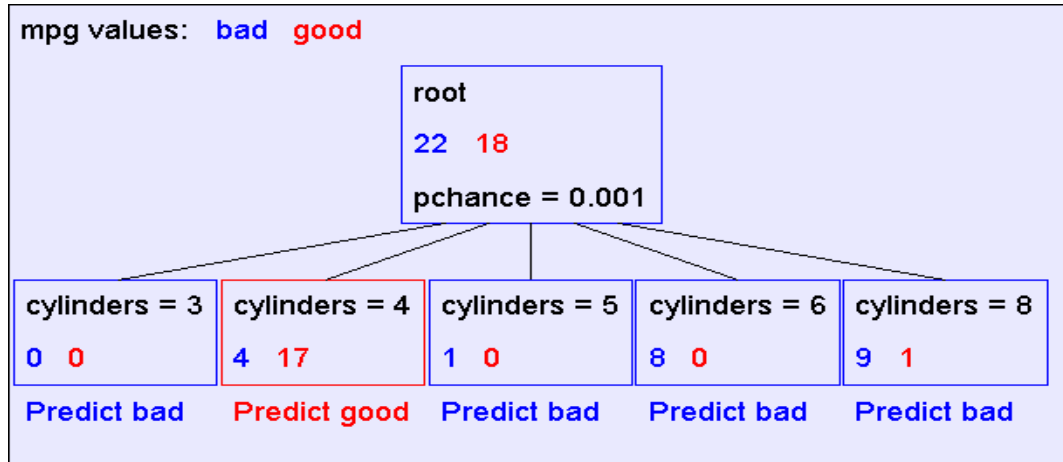
- Calcoliamo tutti gli information gain...



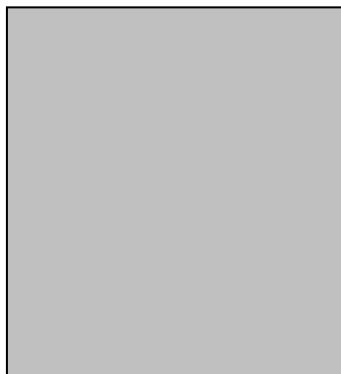
Un “Decision Stump”



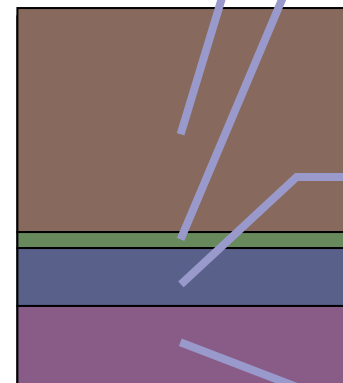
Ricorsione



Prendiamo
il dataset
originale



... e lo
partizioniamo
secondo i valori
dell'attributo
scelto



Auto con
4 cilindri

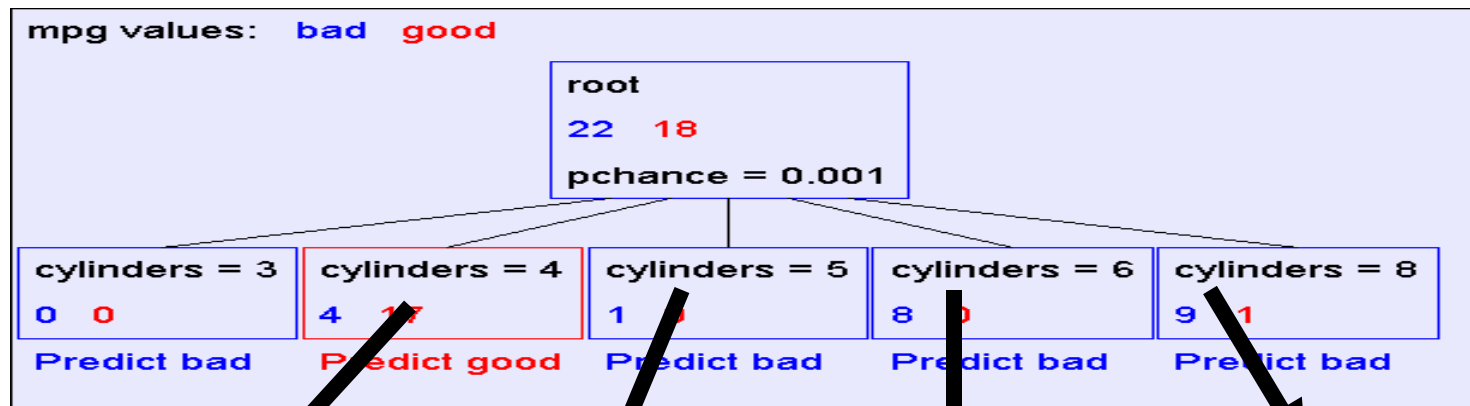
Auto con
5 cilindri

Auto con
6 cilindri

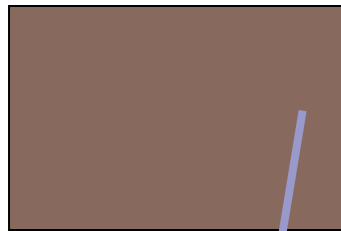
Auto con
8 cilindri

Decision trees

Ricorsione



Crea un albero con questi record...



Auto con 4 cilindri

Crea un albero con questi record...



Auto con 5 cilindri

Crea un albero con questi record...



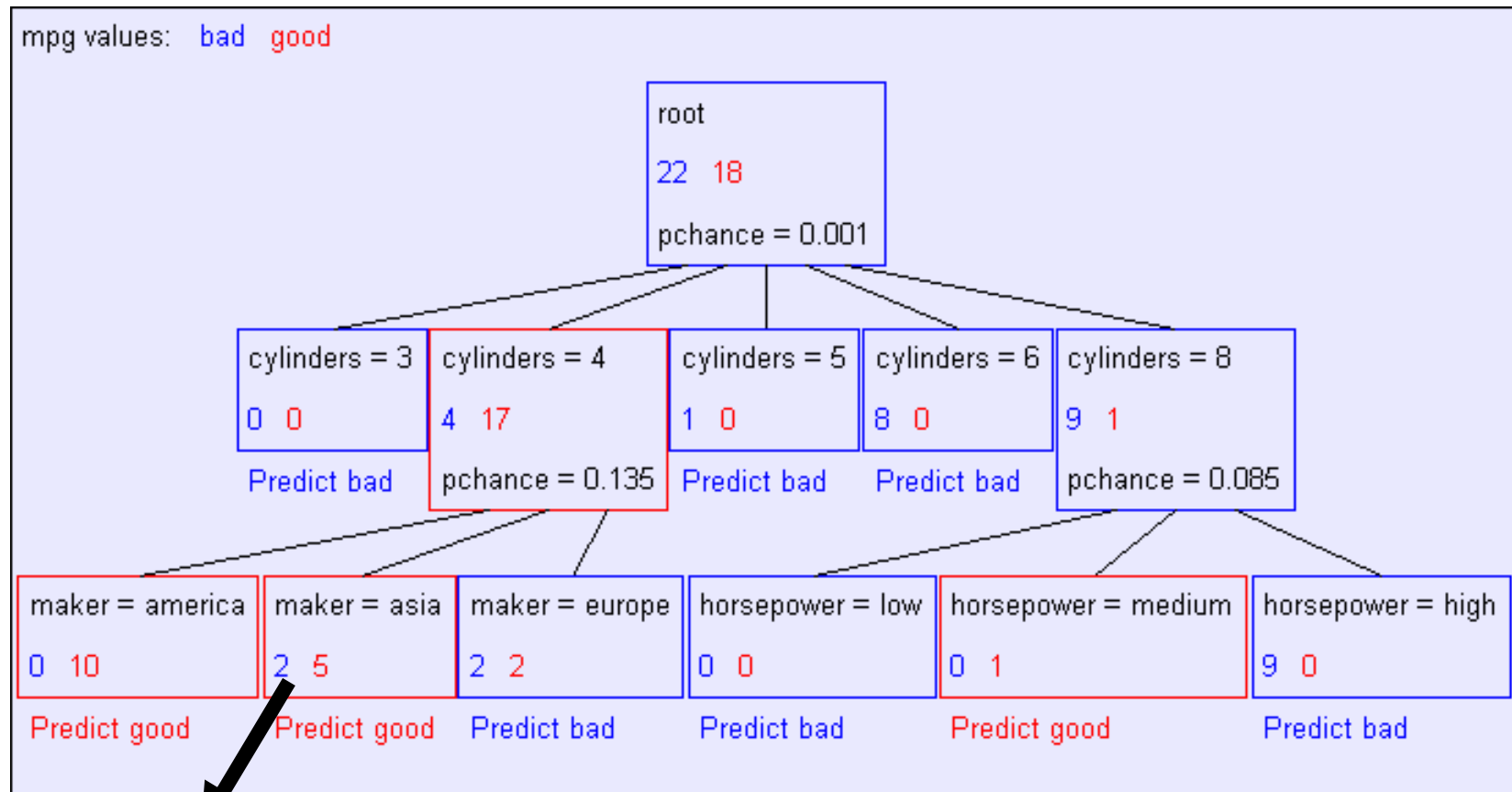
Auto con 6 cilindri

Crea un albero con questi record...



Auto con 7 cilindri

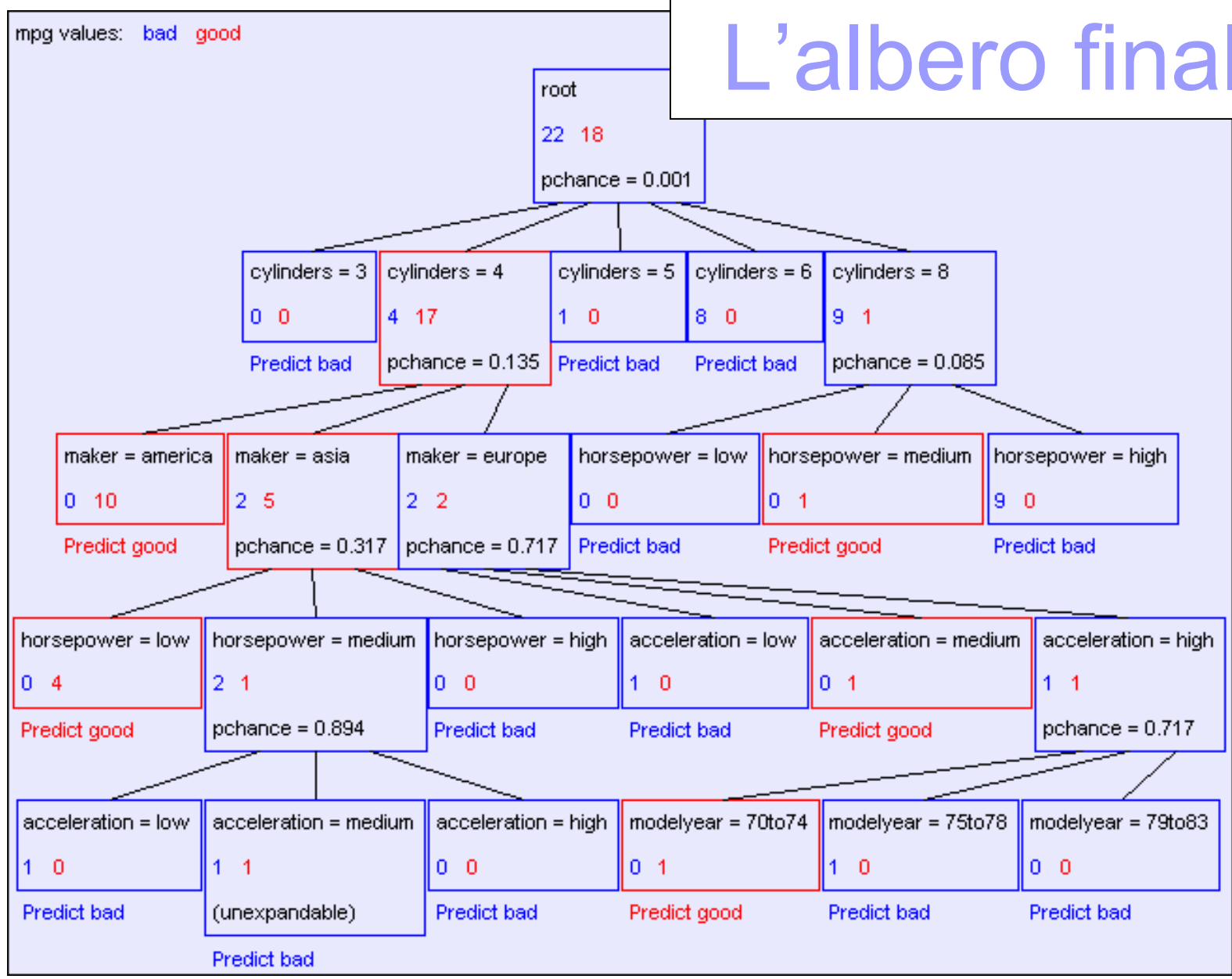
Secondo livello dell'albero



Costruisci ricorsivamente un albero sui 7 record con 4 cilindri ed il costruttore Asiatico

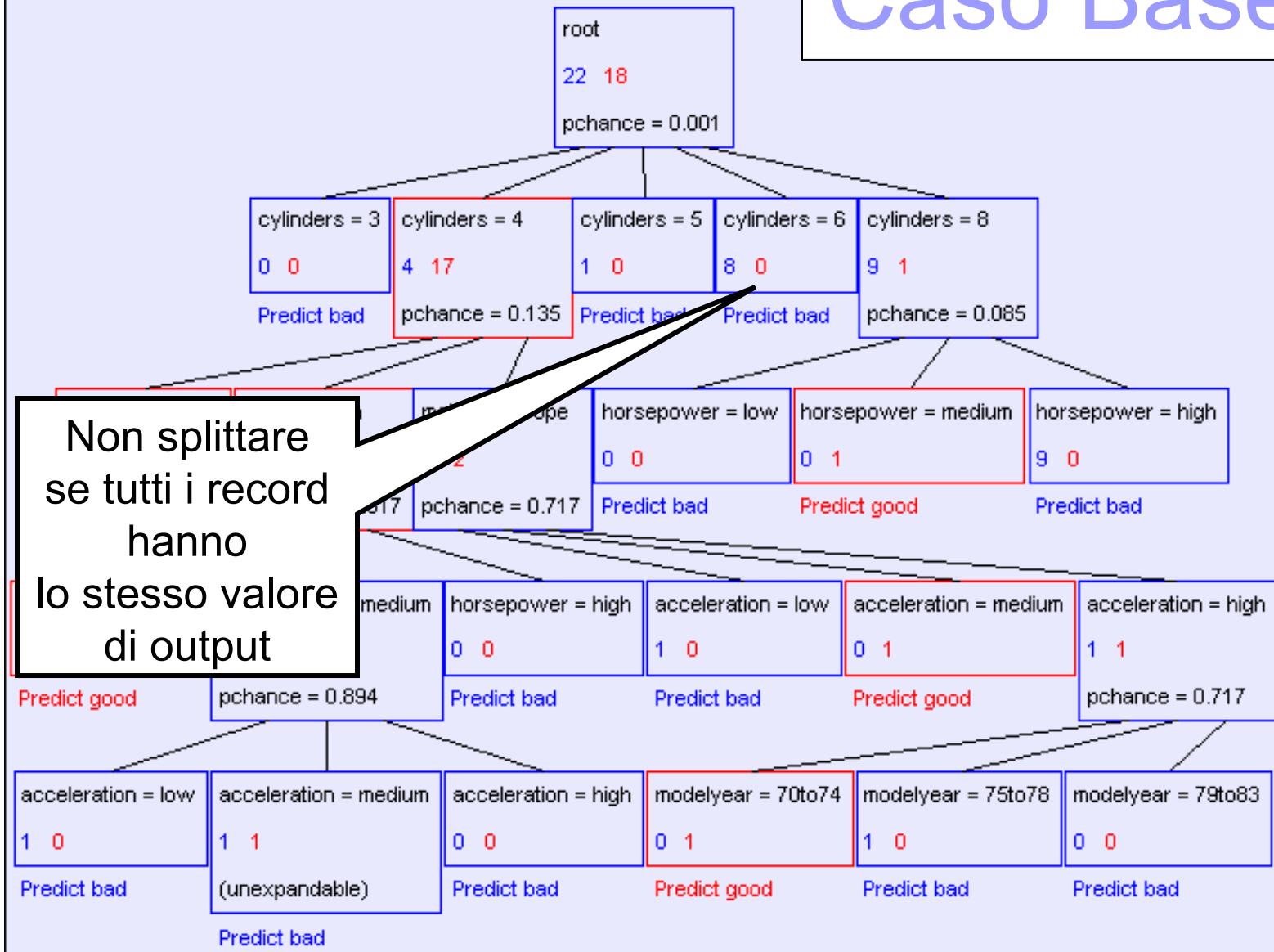
(Ricorsione simile negli altri casi)

L'albero finale



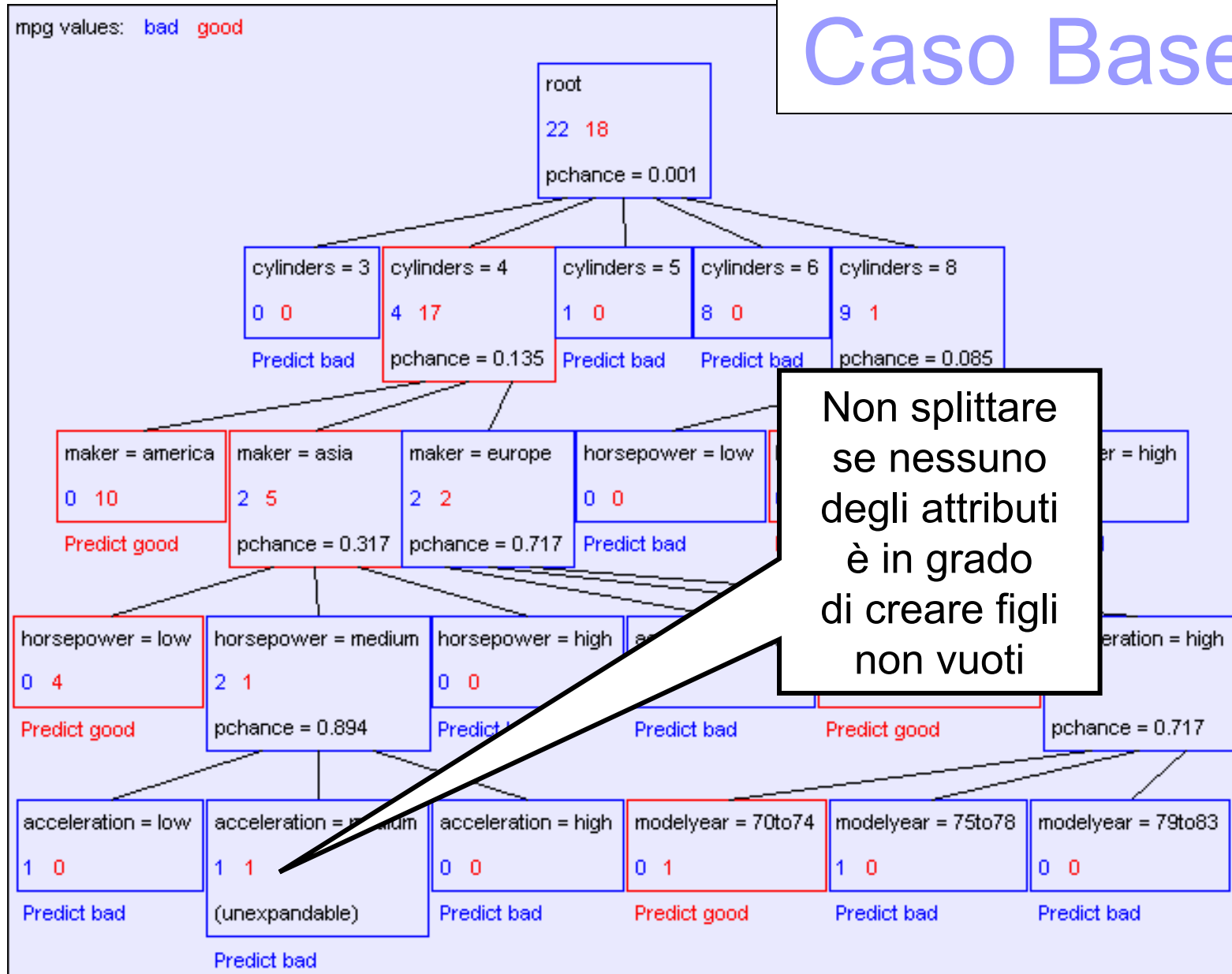
Caso Base 1

mpg values: bad good



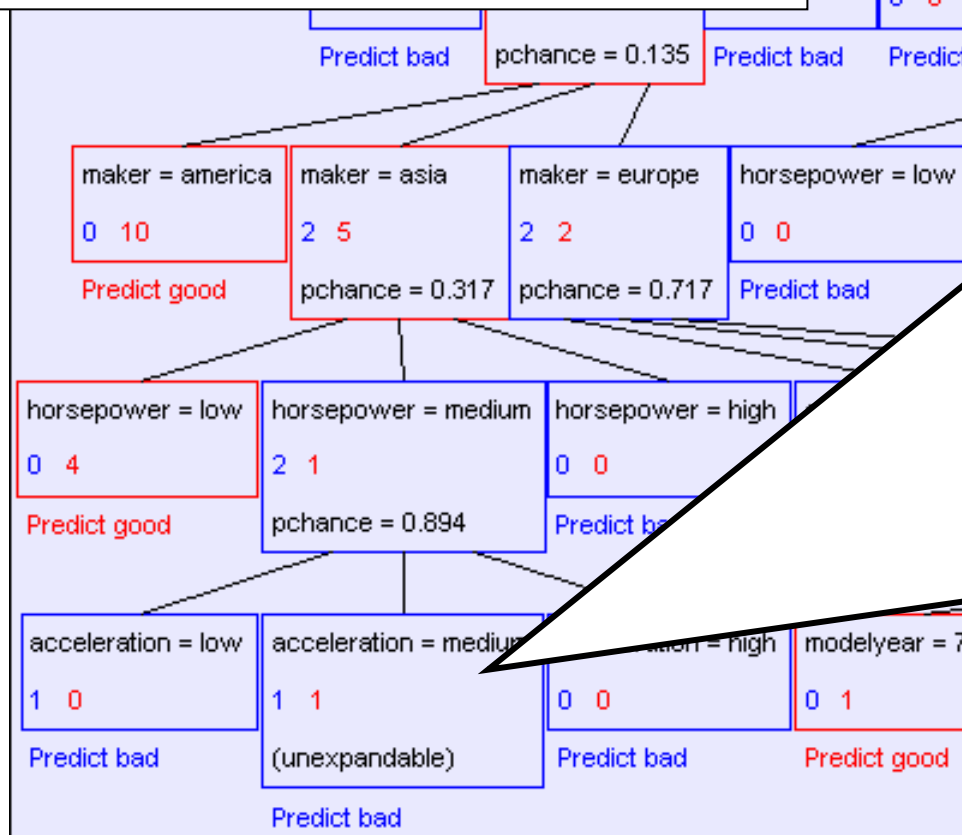
Non splittare se tutti i record hanno lo stesso valore di output

Caso Base 2



Non splittare
se nessuno
degli attributi
è in grado
di creare figli
non vuoti

Caso Base 2: nessun attributo è discriminante



Information gains using the training set (2 records)

mpg values: bad good

Input	Value	Distribution	Info Gain
cylinders	3		0
	4		
	5		
	6		
displacement	low		0
	medium		
	high		
horsepower	low		0
	medium		
	high		
weight	low		0
	medium		
	high		
acceleration	low		0
	medium		
	high		
modelyear	70to74		0
	75to78		
	79to83		
maker	america		0
	asia		
	europe		



Casi Base

- Caso Base 1: non effettuare ricorsione se tutti i record nell'insieme di dati attuale hanno lo stesso valore dell'attributo di output
- Caso Base 2: non effettuare ricorsione se tutti i record nell'insieme di dati attuale hanno gli stessi valori degli attributi di input

Creazione di Decision Tree

`BuildTree(DataSet, Output)`

- Se tutti i valori di output in *DataSet* sono identici, restituisci un nodo foglia “predici questo valore di output”
- Se non esistono altri attributi di input da considerare, restituisci un nodo foglia “predici il valore di output più frequente”
- Altrimenti trova l'attributo X con il più alto *RIG*
- Se X ha n_X valori distinti:
 - Crea e restituisci un nodo interno con n_X figli
 - L' i -esimo figlio va costruito chiamando
`BuildTree(DSi, Output)`
 - Dove DS_i consiste di quei record in *DataSet* per cui $X = i$ -esimo valore di X



Training Set Error

- Per ciascun record, navigare il decision tree per vedere il valore predetto
- Per quanti record la predizione del decision tree è sbagliata (ovvero, diversa dal valore memorizzato nel DB)?
- Questa quantità è detta

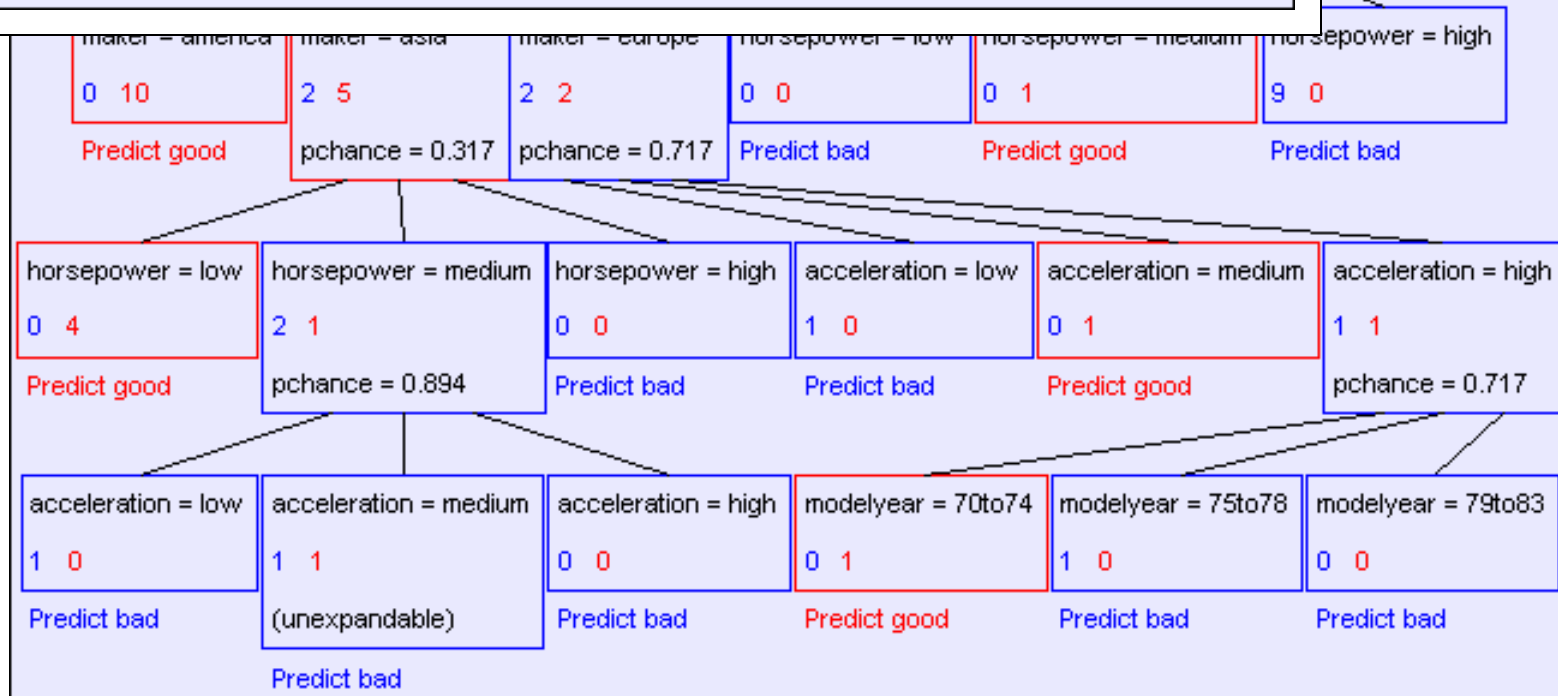
training set error


MPG Training error

mpg values: bad good

root
22 18
pchance = 0.001

	Num Errors	Set Size	Percent Wrong
Training Set 1	1	40	2.50





Un momento: perché stiamo facendo tutto questo?

- In genere, non per predire l'output su dati (il training set) che abbiamo già visto ...
- Molto più utile sarebbe predire i valori di output per dati che incontreremo nel futuro!

Attenzione: un pregiudizio comune sul Data Mining è che i due punti precedenti siano gli unici motivi che portano al learning. Ne esistono almeno un'altra decina ...



Test Set Error

- Supponiamo di guardare al futuro
- Quando abbiamo costruito il decision tree, abbiamo tenuto da parte alcuni dati
- Una volta effettuato il learning, possiamo vedere le predizioni dell'albero su tali dati
- Questa risulta essere una buona simulazione di ciò che accadrà quando proveremo a predire nel futuro
- L'errore ottenuto si chiama **Test Set Error**

Un esempio artificiale

■ Creiamo un training dataset

5 input, binari, sono generati in tutte le 32 possibili combinazioni

Output y = copia di e ,
tranne che per un 25%
(casuale) dei record
che hanno $y = !e$

32 record

a	b	c	d	e	y
0	0	0	0	0	0
0	0	0	0	1	0
0	0	0	1	0	0
0	0	0	1	1	1
0	0	1	0	0	1
:	:	:	:	:	:
1	1	1	1	1	1

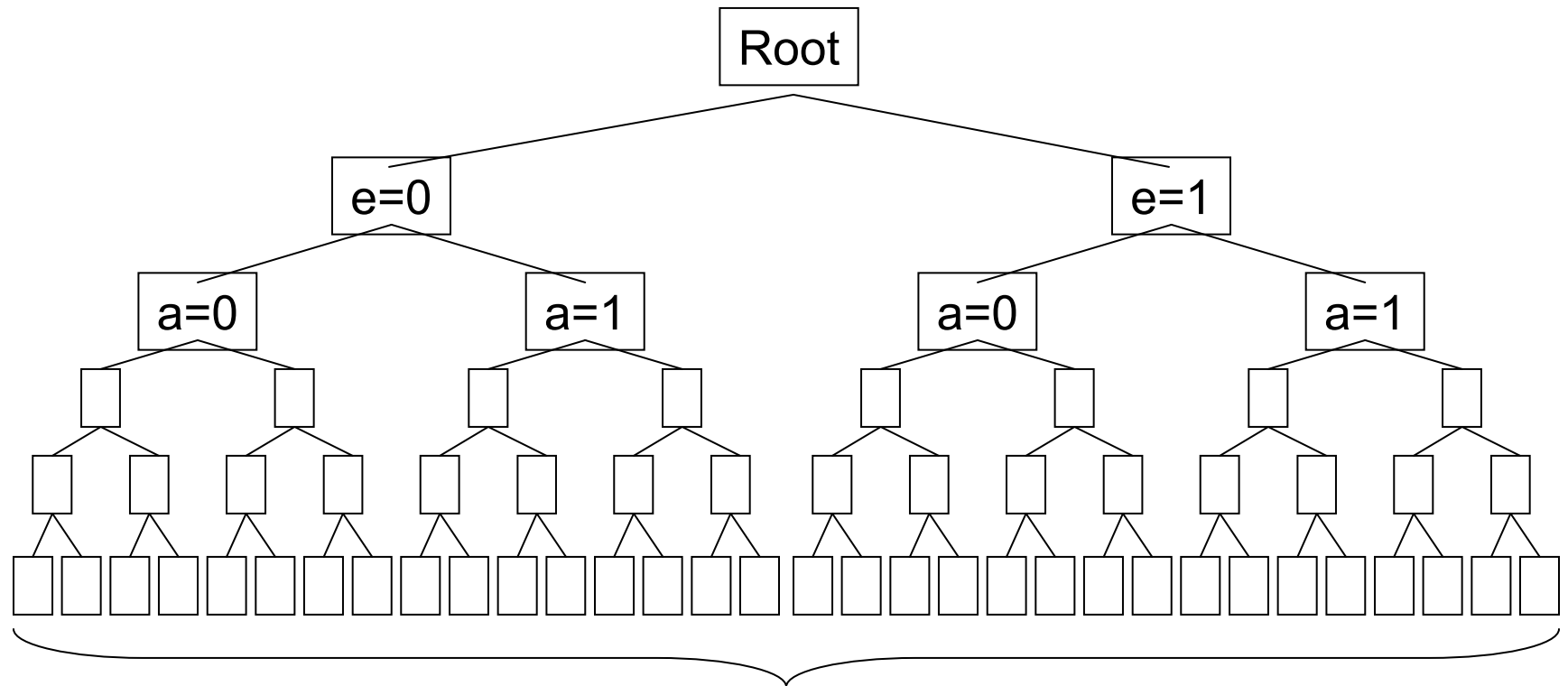


Nel nostro esempio artificiale

- Supponiamo che qualcuno generi un test set seguendo la stessa tecnica
- Il test set è uguale al training set, tranne che per alcune y che saranno diverse
- Alcune y che erano “errate” nel training set saranno “corrette” nel test set
- Alcune y che erano “corrette” nel training set saranno “errate” nel test set

Creare un albero con il training set artificiale

- Supponiamo di costruire un albero completo (splittiamo sempre fino al caso base 2)



25% di questi nodi foglia saranno “errati”



Training set error per il nostro albero

- Tutti i nodi foglia contengono esattamente un record, quindi ...
- Avremo un training set error = zero

Test dell'albero con il test set

	1/4 dei nodi sono errati	3/4 sono corretti
1/4 dei record del test set sono errati	1/16 del test set sarà predetto correttamente per il motivo sbagliato	3/16 del test set sarà predetto erroneamente perché il test record è errato
3/4 sono corretti	3/16 del test set sarà predetto erroneamente perché il nodo è errato	9/16 delle predizioni sul test set saranno corrette

In totale, ci attendiamo di sbagliare i 3/8 delle predizioni del test set



Cosa ci dimostra l'esempio?

- Per prima cosa, la differenza tra **training set error** e **test set error**
- ... ma, più importante, indica se c'è qualcosa da fare per predire correttamente dati futuri

Supponiamo di avere meno dati

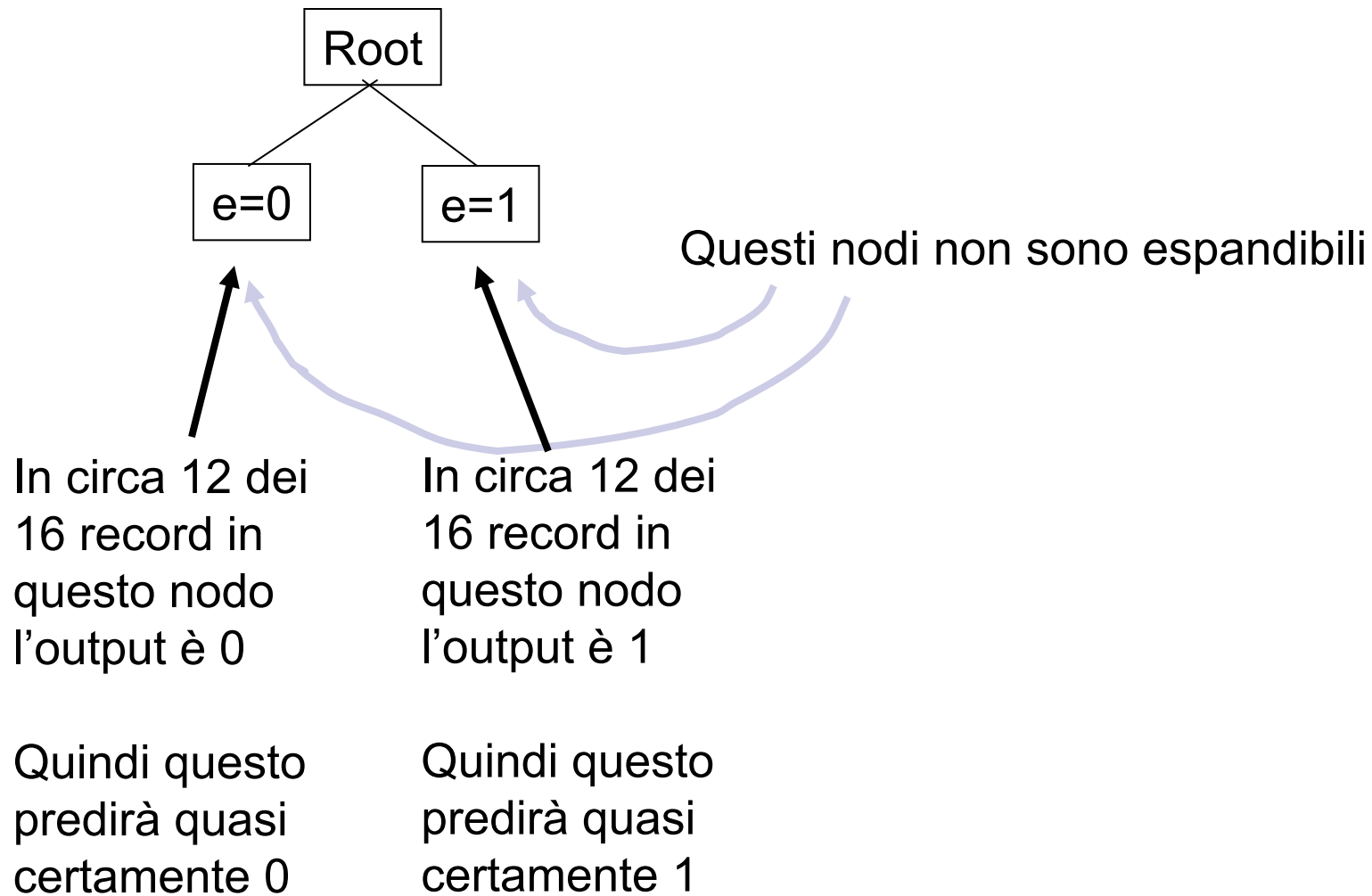
- Non consideriamo i bit irrilevanti



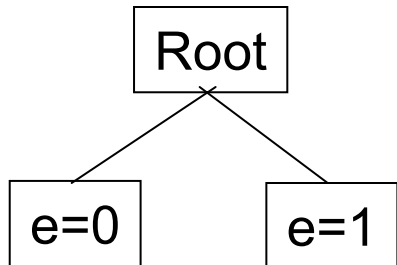
Senza accesso ai bit irrilevanti ...



Senza accesso ai bit irrilevanti ...



Senza accesso ai bit irrilevanti ...



	Quasi nessun nodo dell'albero è errato	Quasi tutti i nodi sono corretti
1/4 dei record del test set sono errati	n/a	1/4 del test set sarà predetto erroneamente perché il test record è errato
3/4 sono corretti	n/a	3/4 delle predizioni sul test set saranno corrette

In totale, ci aspettiamo di sbagliare solamente
1/4 delle predizioni sul test set



Overfitting

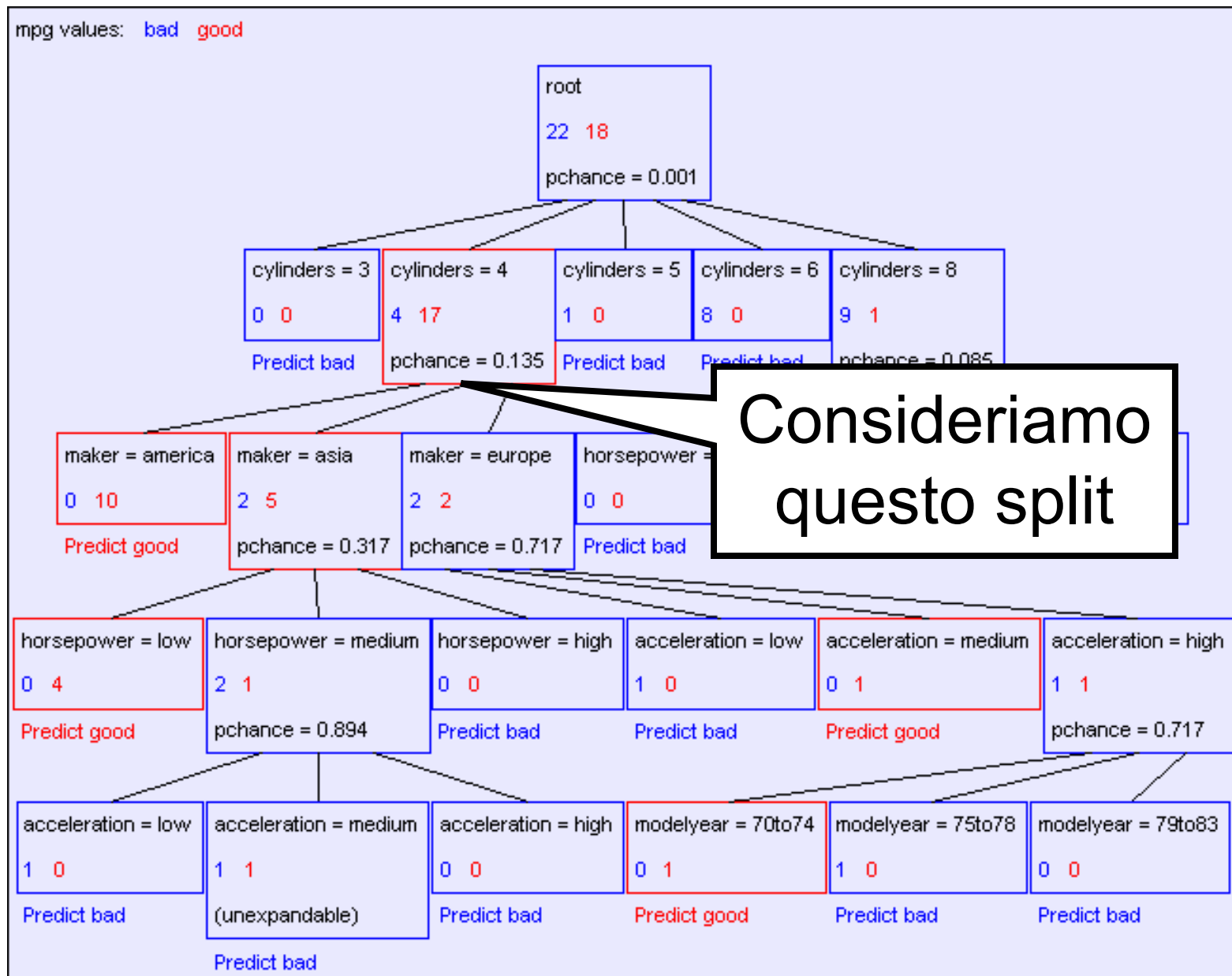
- Definizione: se l'algoritmo di machine learning predice il rumore (cioè presta attenzione alle parti irrilevanti dei dati) allora è in **overfitting**.
- Fatto (teorico ed empirico): se l'algoritmo di machine learning è in overfitting, le sue prestazioni sui dati del test set possono essere peggiori di un algoritmo "normale"



Evitare l'overfitting

- In genere, non sappiamo prima quali sono le variabili irrilevanti
- ... e queste potrebbero dipendere dal contesto
 - Per esempio, se $y = a \text{ AND } b$ allora b è una variabile irrilevante solamente nella porzione dell'albero in cui $a=0$

È possibile usare semplici statistiche per sapere quando si sta andando in overfitting



Un test chi-quadro

mpg values:		bad	good		
maker	america	0	10		$H(\text{mpg} \mid \text{maker} = \text{america}) = 0$
	asia	2	5		$H(\text{mpg} \mid \text{maker} = \text{asia}) = 0.863121$
	europa	2	2		$H(\text{mpg} \mid \text{maker} = \text{europa}) = 1$

$H(\text{mpg}) = 0.702467$ $H(\text{mpg} \mid \text{maker}) = 0.478183$
 $I_G(\text{mpg} \mid \text{maker}) = 0.224284$

- Supponiamo che MPG sia completamente scorrelato dal continente del produttore
- Qual'è la probabilità di vedere comunque dati che presentino questo livello apparente di associazione?

Usando un tipo particolare di test chi-quadro, la risposta è il 13.5%.



Il test chi quadro

- È usato per decidere se un campione di dati segue una certa distribuzione di probabilità
- Si considera l'attributo X diviso in k classi mutuamente esclusive
- Sia p_i la probabilità di ottenere un campione all'interno dell' i -esima classe
- Siano n_i il numero di campioni all'interno dell' i -esima classe e sia $n = \sum_i n_i$



Variabili casuali

- Per trovare una statistica di test appropriata, consideriamo $k=2$ e la variabile standardizzata Y :

$$Y = \frac{n_1 - n p_1}{\sqrt{n p_1 (1 - p_1)}}$$

- Se n è sufficientemente grande, Y ha una distribuzione normale
- Quindi Y^2 ha una distribuzione a chi quadro con 1 grado di libertà

Distribuzione chi quadro

- Segue che:

$$\begin{aligned}\frac{(n_1 - n p_1)^2}{n p_1 (1 - p_1)} &= \frac{(n_1 - n p_1)^2}{n p_1} + \frac{(n_1 - n p_1)^2}{n p_2} \\ &= \frac{(n_1 - n p_1)^2}{n p_1} + \frac{(n_2 - n p_2)^2}{n p_2}\end{aligned}$$

segue una distribuzione a chi quadro
con 1 grado di libertà

Generalizzando

- La statistica $\sum_{i=1}^k \frac{(n_i - n p_i)^2}{n p_i}$

ha distribuzione chi quadro con $k-1$ gradi di libertà

- Tale statistica (anche detta test di Pearson) ha valore 0 se c'è allineamento perfetto tra le frequenze osservate e quelle attese

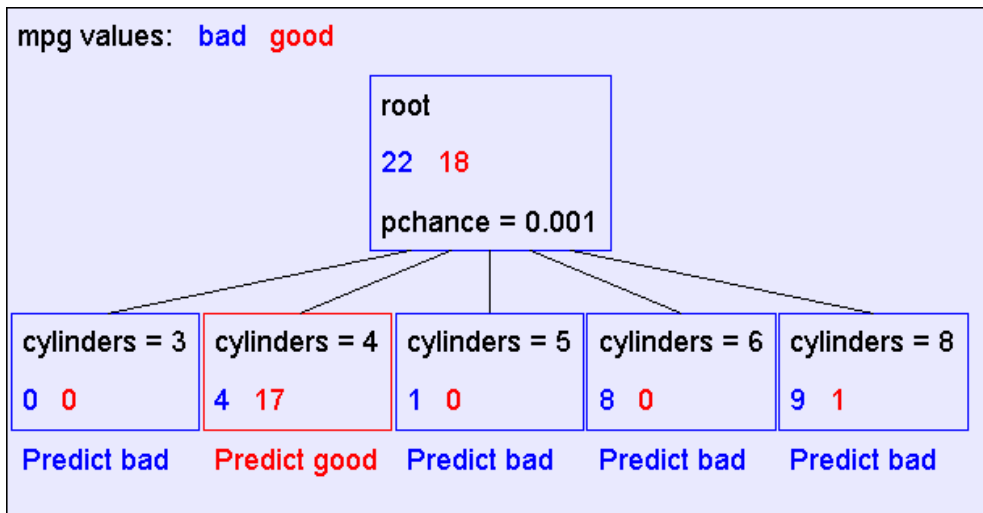


Usare chi-quadro per evitare l'overfitting

- Costruiamo il decision tree come al solito
- Quando non riusciamo a farlo crescere, cominciamo a “potarlo” :
 - Cominciando dal fondo dell'albero, cancellare gli split per cui $prob > MaxProb$
 - Continuare verso l'alto finché non ci sono più nodi da potare
- *MaxProb* è un parametro “magico” da specificare, che indica la volontà di rischiare l'overfitting

Esempio di “potatura”

- Con $MaxProb = 0.1$, otterremmo il seguente MPG decision tree:



Notare l'aumentata
precisione
sul test set rispetto
all'albero non potato

	Num Errors	Set Size	Percent Wrong
Training Set	5	40	12.50
Test Set	56	352	15.91

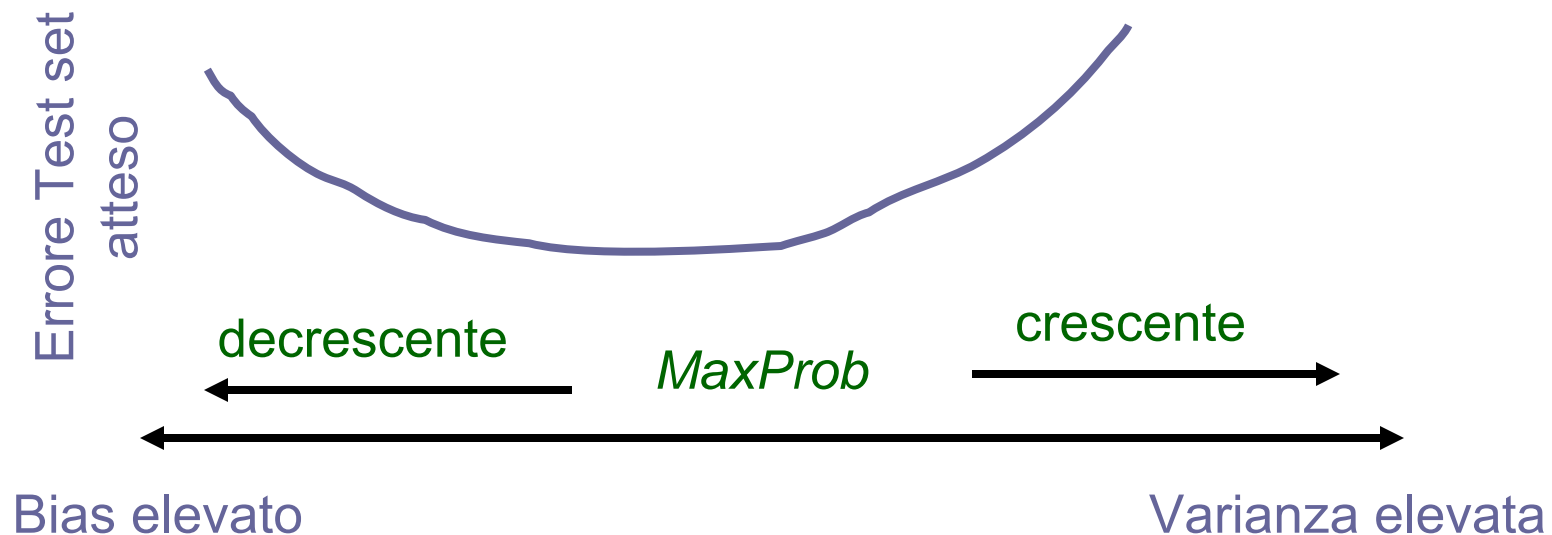


MaxProb

- **Buone notizie:** Il decision tree può automaticamente modificare le proprie decisioni sulla potatura a seconda della quantità apparente di rumore
- **Cattive notizie:** L'utente deve proporre un "buon" valore per *MaxProb*
- **Buone notizie:** Con un po' di lavoro in più, il valore ottimale di *MaxProb* può essere stimato con una tecnica detta di *cross-validazione*

MaxProb

- Nota tecnica: *MaxProb* è un parametro di regolarizzazione





L'albero più semplice

- Nota che questa potatura può essere difficile da trovare euristicamente

La struttura ad albero più semplice per cui tutti i contrasti all'interno di nodi foglia possono essere dovuti al caso

- Questo non equivale a chiedere “lo schema di classificazione più semplice per cui ...”
- I decision tree tendono a preferire i classificatori che possono essere espressi in forma d'albero



Espressività dei Decision Tree

- Assumiamo che tutti gli input e gli output siano Booleani
- Quali funzioni Booleane possono essere rappresentate da decision tree?
- Risposta: tutte le funzioni Booleane
- Dimostrazione:
 - Prendere una qualsiasi funzione Booleana
 - Convertirla in una tabella di verità
 - Costruire un decision tree nel quale ogni riga della tabella di verità corrisponda ad un percorso all'interno dell'albero

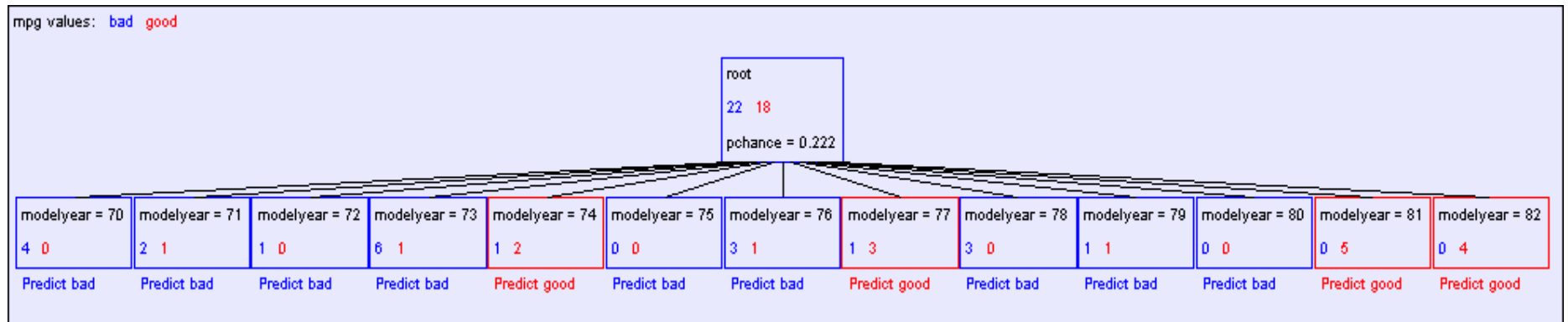
Input a Valori Reali

- Come facciamo se alcuni attributi di input sono a valori reali?

mpg	cylinders	displacemen	horsepower	weight	acceleration	modelyear	maker
good	4	97	75	2265	18.2	77	asia
bad	6	199	90	2648	15	70	america
bad	4	121	110	2600	12.8	77	europa
bad	8	350	175	4100	13	73	america
bad	6	198	95	3102	16.5	74	america
bad	4	108	94	2379	16.5	73	asia
bad	4	113	95	2228	14	71	asia
bad	8	302	139	3570	12.8	78	america
:	:	:	:	:	:	:	:
:	:	:	:	:	:	:	:
:	:	:	:	:	:	:	:
good	4	120	79	2625	18.6	82	america
bad	8	455	225	4425	10	70	america
good	4	107	86	2464	15.5	76	europa
bad	5	131	103	2830	15.9	78	europa

Idea 1: splittare su ogni possibile valore

Idea “un ramo per ogni valore”



- Senza speranza: con un fattore di diramazione così elevato, c'è per forza overfit
- Nota: *prob* è 0.222 nell'esempio ...
 - Fissando *MaxProb* a 0.05
avremmo potuto tutto l'albero ottenendo un singolo nodo radice

Un'idea migliore: split con soglia

- Supponiamo che X sia a valori reali
- Definiamo $IG(Y|X:t)$ come $H(Y) - H(Y|X:t)$
- Definiamo $H(Y|X:t) =$
$$H(Y|X < t) P(X < t) + H(Y|X \geq t) P(X \geq t)$$
 - $IG(Y|X:t)$ è l'information gain per predire Y se tutto ciò che sappiamo è se X è maggiore o minore di t
- Quindi definiamo $IG^*(Y|X) = \max_t IG(Y|X:t)$
- Per ogni attributo a valori reali, usiamo $IG^*(Y|X)$ per valutarne la capacità di split

Complessità Computazionale

- Possiamo calcolare $IG^*(Y|X)$ in tempo

$$R \log R + 2 R n_y$$

- dove

- R è il numero di record nel nodo attuale
- n_y è la cardinalità (numero di valori distinti) di Y

- Come?

- Ordiniamo i record secondo valori crescenti di X
- Quindi creiamo $2 * n_y$ tabelle di contingenza corrispondenti al calcolo di $IG(Y|X:x_{min})$
- Infine iteriamo sui record, controllando ogni soglia tra valori adiacenti di X , aggiornando incrementalmente le tabelle di contingenza
- Per un'ulteriore velocizzazione, si possono testare solamente i valori diversi di Y

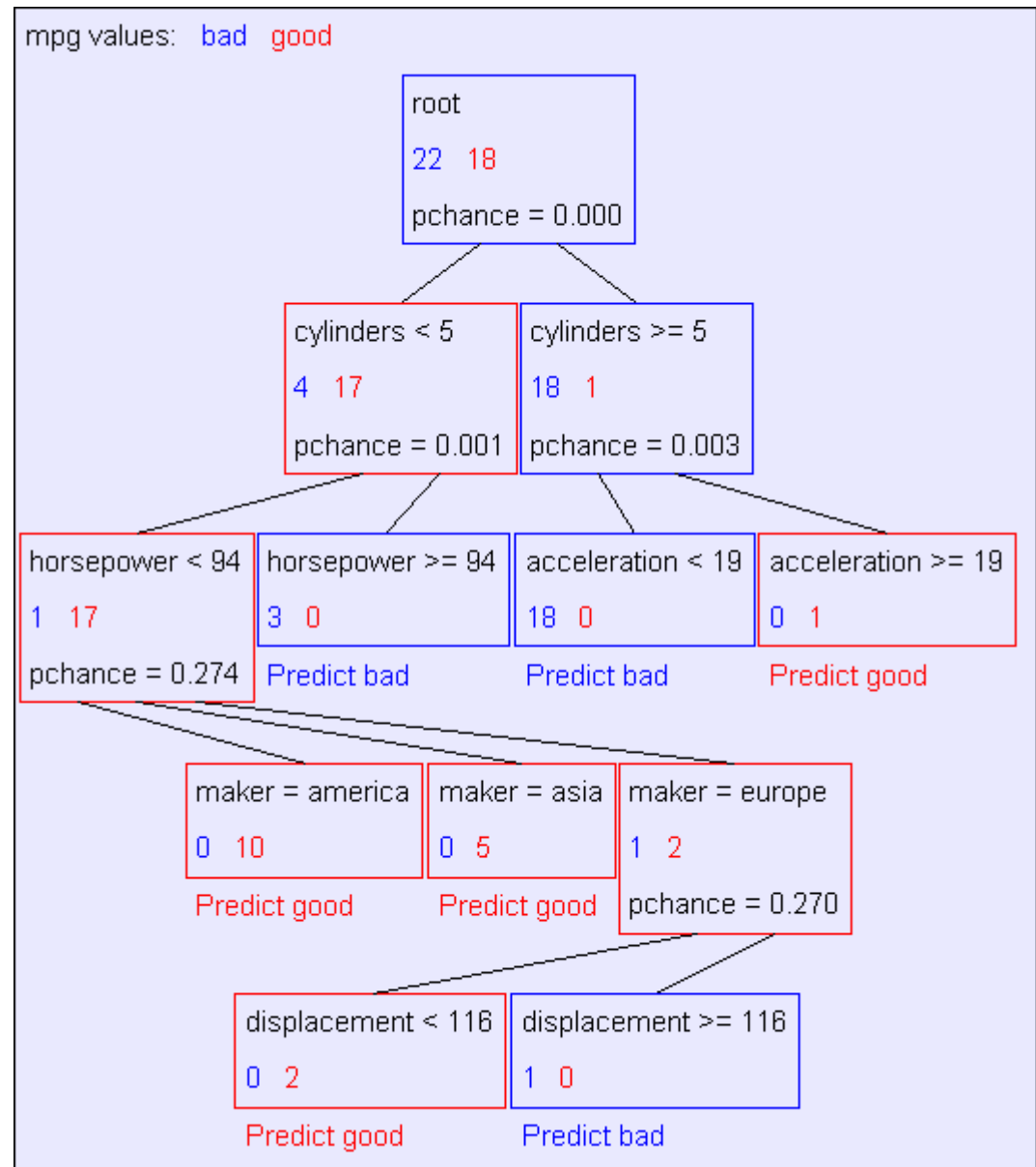
Esempio con MPG

Information gains using the training set (40 records)

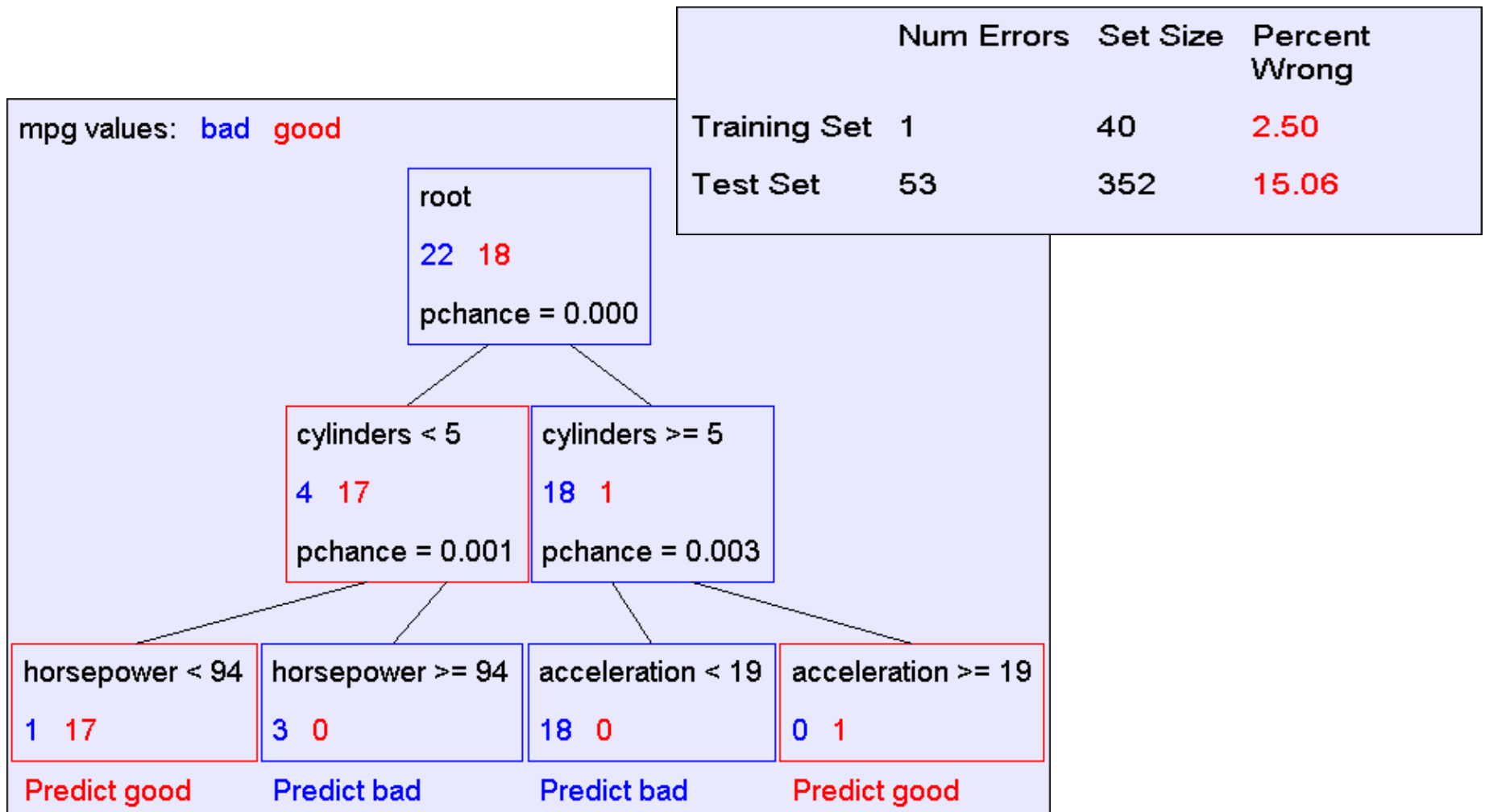
mpg values: bad good

Input	Value	Distribution	Info Gain
cylinders	< 5		0.48268
	>= 5		
displacement	< 198		0.428205
	>= 198		
horsepower	< 94		0.48268
	>= 94		
weight	< 2789		0.379471
	>= 2789		
acceleration	< 18.2		0.159982
	>= 18.2		
modelyear	< 81		0.319193
	>= 81		
maker	america		0.0437265
	asia		
	europa		

Albero non potato a valori reali

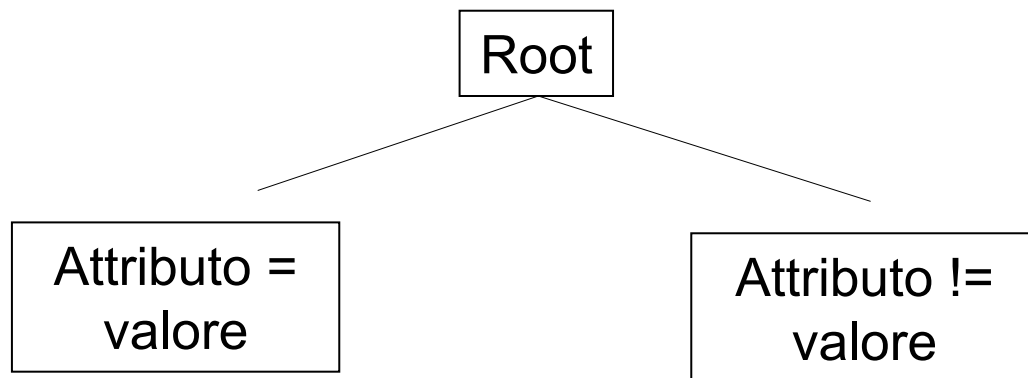


Albero potato a valori reali

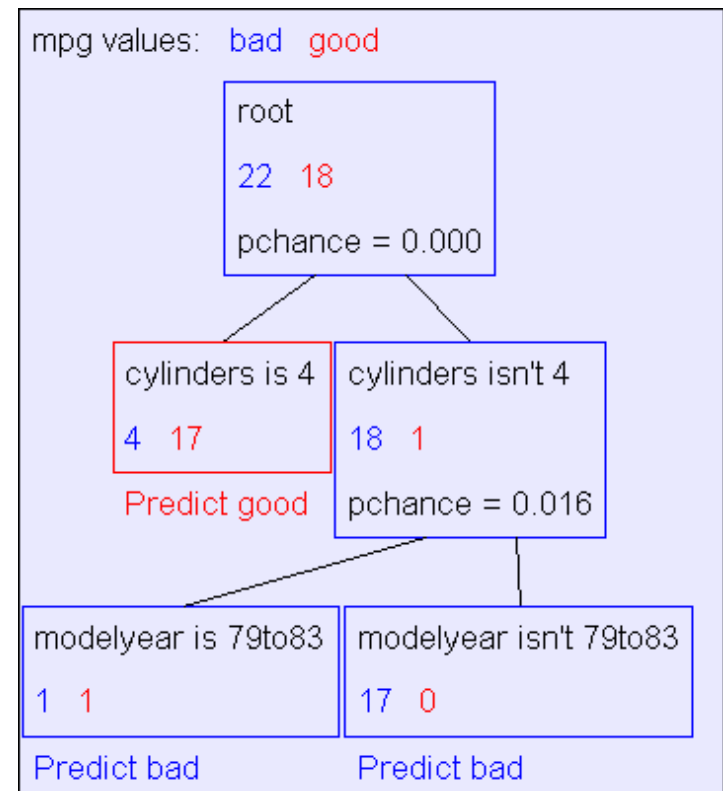


Split Categorici Binari

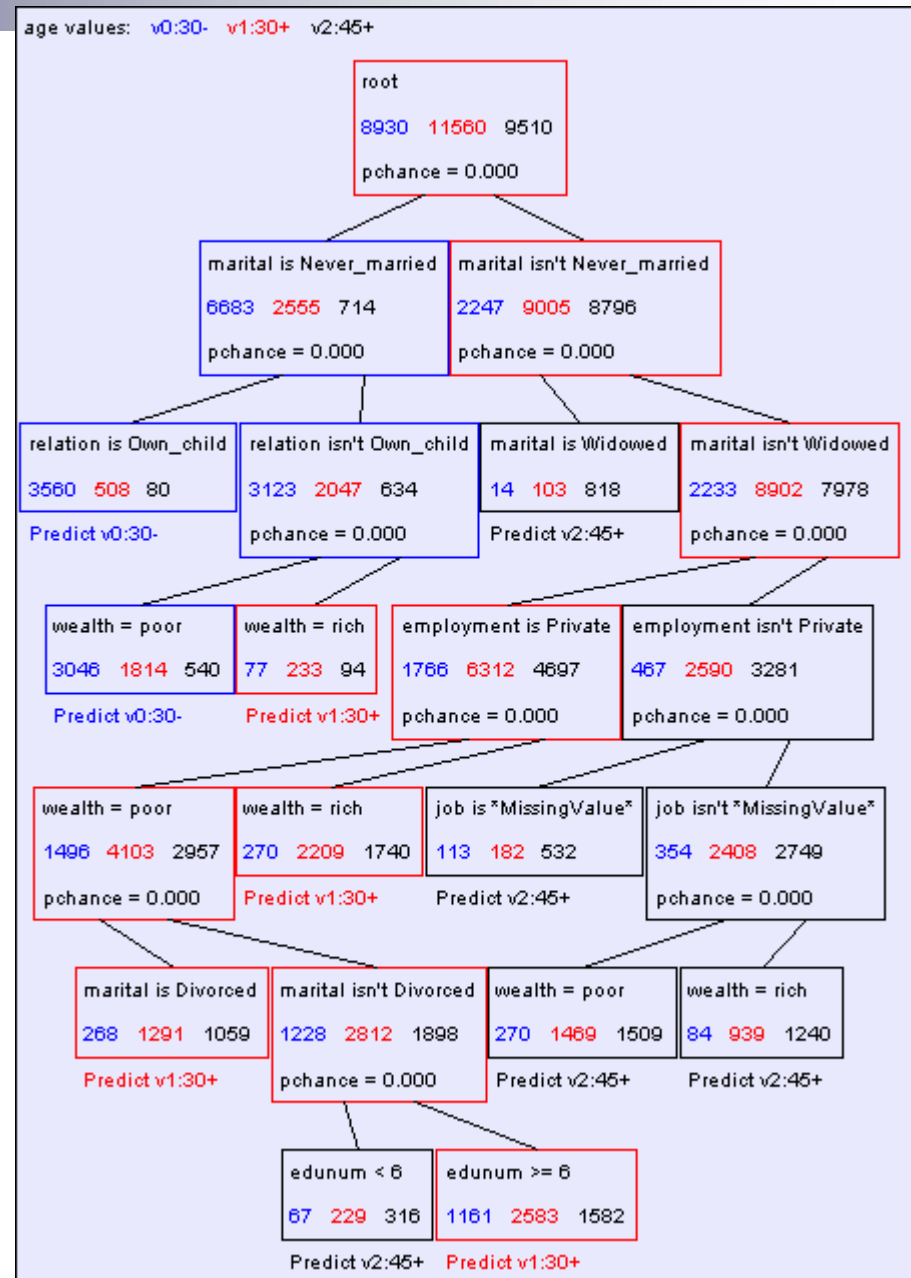
- Un classico “trucco”
- Permette split fatti così:



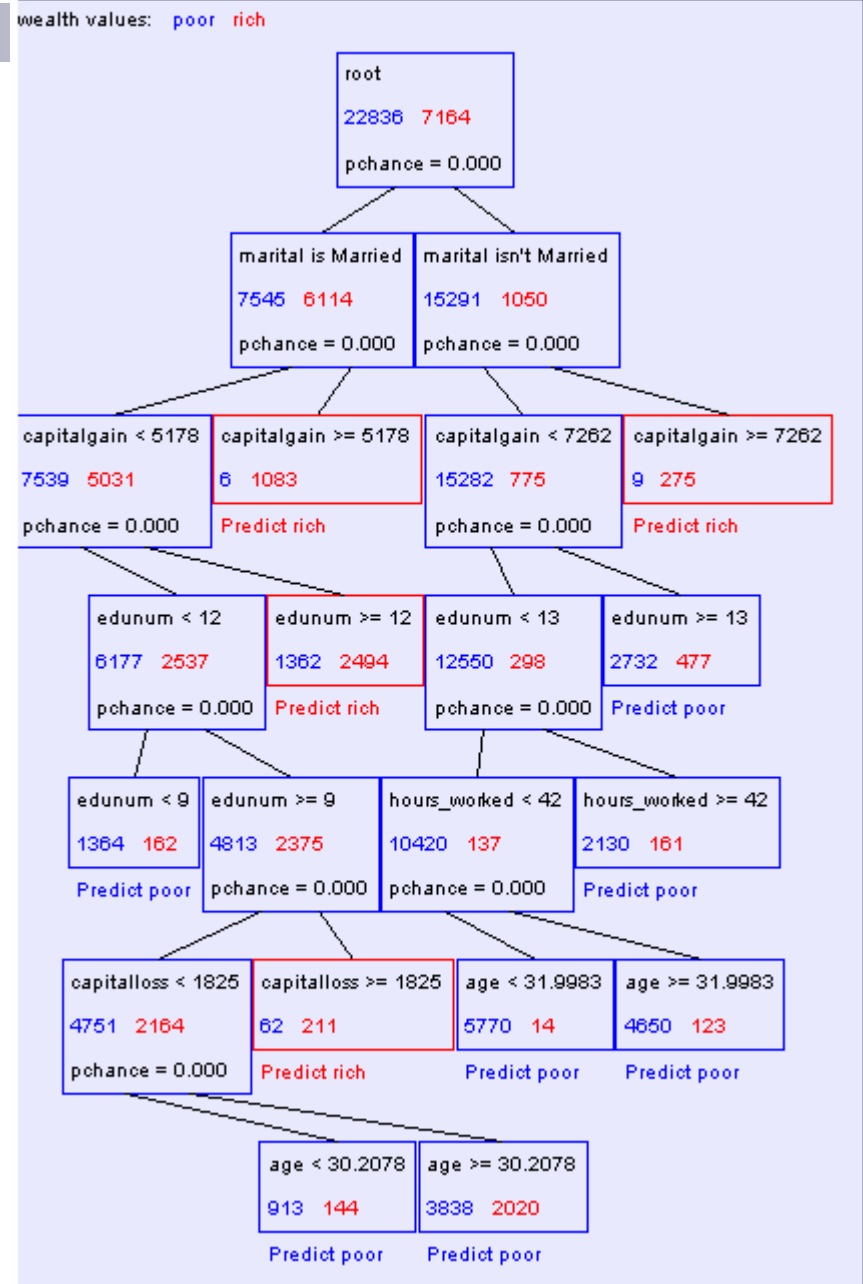
Esempio:



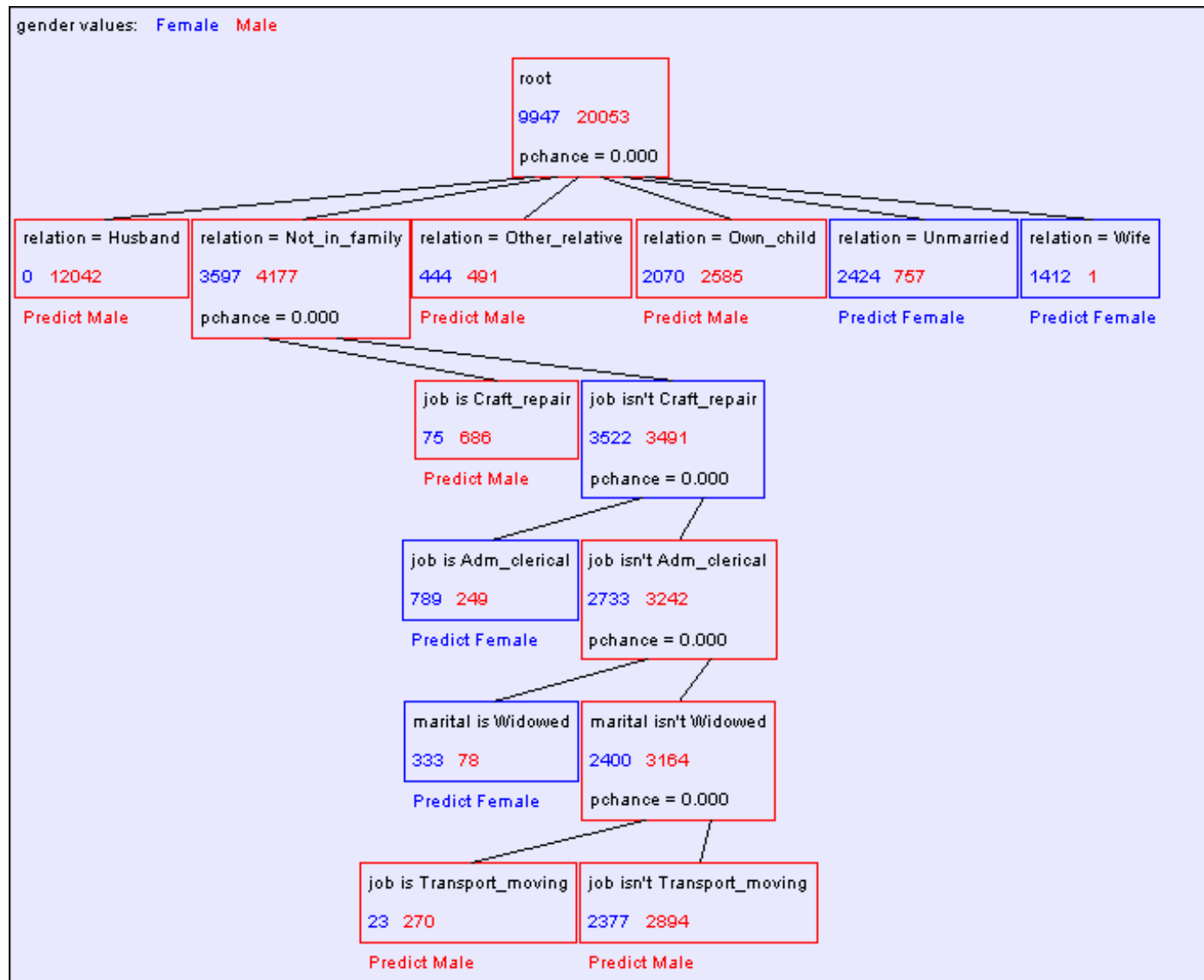
Predizione dell'età



Predizione della ricchezza



Predizione del sesso





Conclusioni

- I decision tree sono lo strumento più utilizzato di Data Mining
 - Semplici da comprendere
 - Semplici da implementare
 - Semplici da usare
 - Computazionalmente poco costosi
- È possibile avere problemi di overfitting
- Effettuano una classificazione:
predicono un attributo categorico di output
a partire da attributi categorici e/o reali in input



Cosa è necessario sapere

- Cos'è una tabella di contingenza?
- Cos'è l'information gain e come lo usiamo?
- L'algoritmo ricorsivo per costruire un decision tree non potato
- Cosa sono gli errori di training e test set
- Perché il test set error può essere più elevato del training set error
- Perché la potatura può ridurre l'errore di test set
- Come comportarsi in presenza di valori reali



Cosa non abbiamo trattato

- È possibile avere attributi a valori reali in output – si ottengono i Regression Tree
- Per prevenire l'overfitting si possono utilizzare i Decision Tree Bayesiani
- Complessità computazionale
- Alternative all'Information Gain per splittare i nodi
- Come selezionare automaticamente *MaxProb*
- I dettagli del test chi-quadro
- Boosting – un modo semplice per migliorare la precisione



Per approfondimenti

- Due libri:
 - L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. Classification and Regression Trees. Wadsworth, Belmont, CA, 1984.
 - C4.5 : Programs for Machine Learning (Morgan Kaufmann Series in Machine Learning) by J. Ross Quinlan
- Decine di articoli, tra cui:
 - Learning Classification Trees, Wray Buntine, Statistics and Computation (1992), Vol 2, pages 63-73
 - Kearns and Mansour, On the Boosting Ability of Top-Down Decision Tree Learning Algorithms, STOC: ACM Symposium on Theory of Computing, 1996
- Decine di implementazioni software disponibili sul Web gratuitamente e/o commercialmente a prezzi tra i \$50 ed i \$300,000



Discussione

- Invece che costruire l'albero in maniera euristica (e greedy) perché non effettuare una ricerca combinatoria dell'albero ottimale?
- Se costruiamo un decision tree per predire la ricchezza, e stato civile, età e sesso sono scelti come attributi nei livelli più alti dell'albero, è ragionevole concludere che questi tre attributi sono le maggiori cause di ricchezza (o povertà)?
- ... è ragionevole pensare che gli attributi non presenti dell'albero non sono cause di ricchezza?
- ... è ragionevole pensare che gli attributi non presenti dell'albero non sono correlati con la ricchezza?



Alberi di decisione e scalabilità

- La maggior parte degli algoritmi per l'apprendimento di DT richiede che il training set risieda tutto in memoria
- Nei casi reali sono frequenti training set di milioni di record
- Strategie:
 - discretizzazione, campionamento
 - ridurre i dati per adattarsi alla memoria
 - partizionamento
 - costruzione di DT su subset
 - combinazione dei DT su subset
 - peggiora la qualità del risultato finale
 - perché?
 - strutture dati ausiliarie
 - si mantengono in memoria soltanto strutture di sintesi, che permettono di indirizzare la costruzione dell'albero
 - numerose proposte



Rainforest

- Un metodo scalabile per la costruzione di DT
- Indipendente dall'algoritmo di costruzione dell'albero
 - separa gli aspetti della classificazione da quelli di gestione della memoria
- Non influenza la qualità del risultato
- Scala in modo praticamente lineare

- Gehrke, Ramakrishnan, Ganti, “Rainforest - A Framework for Fast Decision Tree Construction of Large Datasets”, Data Mining and Knowledge Discovery, 4, 127-162, 2000



La costruzione di un DT in sintesi

- Tecnica greedy
 - Si cerca un ottimo locale
- La radice contiene tutto il DB
- Si sceglie un criterio di split alla radice, generando k nodi figli, a ciascuno dei quali viene attribuita una partizione del DB del nodo padre
- Si sceglie ricorsivamente un criterio di split per tutti i nodi via via creati
 - La maggioranza degli algoritmi differiscono unicamente per la tecnica di scelta del criterio
- La fase di pruning può essere intercalata a quella di crescita o posteriore
- La fase di crescita è quella più costosa in termini di accesso ai dati

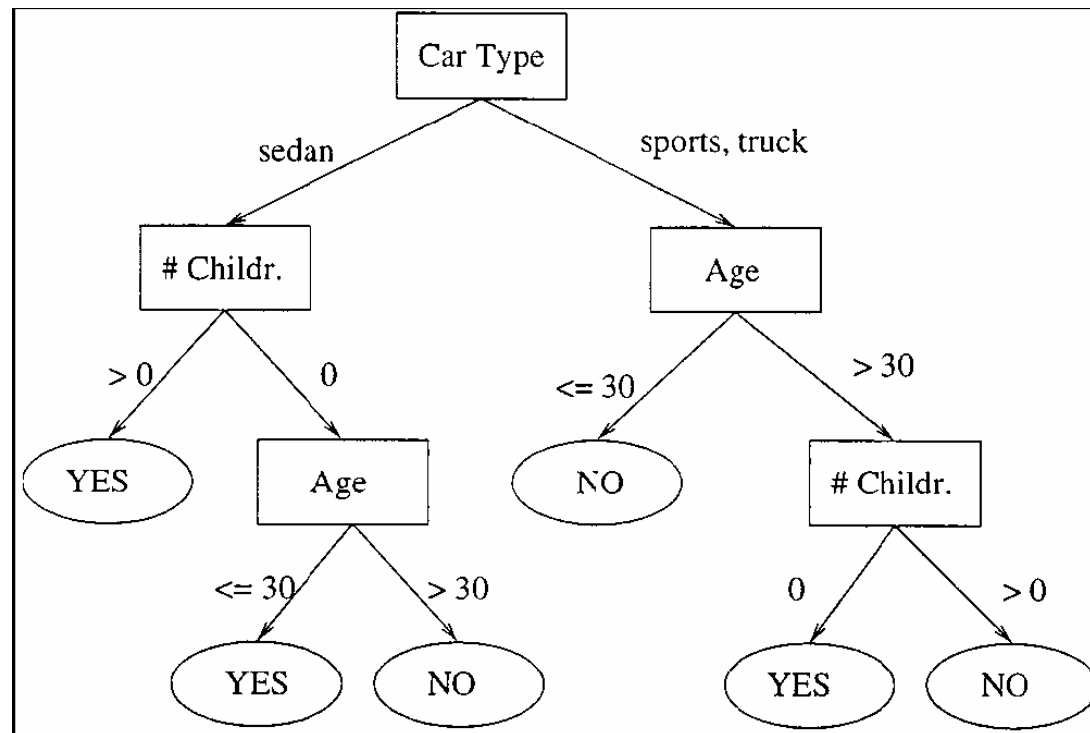


Esempio

Record Id	Car type	Age	Number of children	Subscription
1	sedan	23	0	yes
2	sports	31	1	no
3	sedan	36	1	no
4	truck	25	2	no
5	sports	30	0	no
6	sedan	36	0	no
7	sedan	25	0	yes
8	truck	36	1	no
9	sedan	30	2	yes
10	sedan	31	1	yes
11	sports	25	0	no
12	sedan	45	1	yes
13	sports	23	2	no
14	truck	45	0	yes

Esempio (ii)

- Tre attributi predittori
 - car type, age, number of children
- Si intende predire la classe *subscriber*



Struttura dati

- $AVC_n(X)$: per ogni attributo predittore X e per ogni nodo n si costruisce una struttura che contiene
 - i valori distinti dell'attributo
 - i conteggi del numero di tuple presenti nel nodo per ciascuna etichetta di classe

```
SELECT D.X, D.C
FROM D
WHERE  $f(n)$ 
GROUP BY D.X, D.C
```

- $AVC\text{-group}_n$ è l'insieme degli $AVC_n(X)$ per ogni attributo disponibile per lo split al nodo n
- La struttura per tutti gli attributi può essere costruita con una *union* che, con opportuni accorgimenti, può essere generata in un solo passo

Struttura dati – esempio

Car type	Subscription	
	Yes	No
sedan	5	2
sports	0	4
truck	1	2

Age	Subscription	
	Yes	No
23	1	1
25	1	2
30	1	1
31	1	1
36	0	3
45	2	0

Num. Children	Subscription	
	Yes	No
0	3	3
1	2	3
2	1	2



Osservazioni

■ AVC-group

- non è una rappresentazione compressa delle tuple selezionate da $f(n)$
- contiene informazioni aggregate sufficienti per la costruzione dell'albero di classificazione

■ Casistica

- AVC-group della radice sta in memoria
- i singoli AVC degli attributi stanno in memoria, ma non tutti insieme
- ci sono singoli AVC che non stanno in memoria



Attività svolta per ciascun nodo

- costruzione di AVC-group
 - leggere i dati $f(n)$ richiede una scansione del DB, o della partizione materializzata che compete al nodo, eventualmente ripetuta per ogni attributo se non si dispone di una implementazione efficiente della *union*
- scelta dell'attributo di split
 - si applicano i criteri dell'algoritmo scelto
 - tutti gli algoritmi considerano le informazioni contenute in AVC, un attributo alla volta
- partizionamento di $f(n)$ tra i nodi figli che vengono creati
 - una lettura completa e una scrittura, suddivisa fra i figli
 - se la memoria è sufficiente la scrittura può avvenire in un passo solo per tutti i figli



Valori mancanti

- L'assenza di un valore in un attributo predittore porterebbe a escludere la tupla dal training set
- Alternativa: stimare il valore mancante $t.X$
 - valore medio calcolato sulle tuple del nodo e per la classe $t.C$ se l'attributo è numerico
 - valore più frequente calcolato sulle tuple del nodo e per la classe $t.C$ se l'attributo è categorico