Classificazione I

Prof. Matteo Golfarelli
Alma Mater Studiorum - Università di Bologna

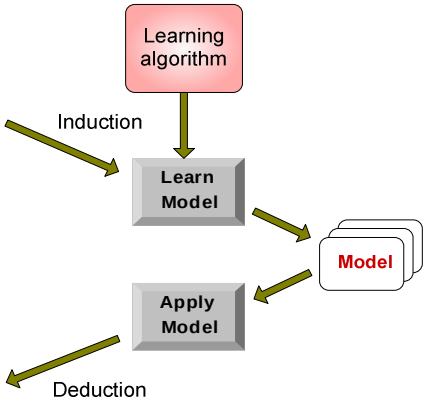
Classificazione: Definizione

- Data una collezione di record (*training set*)
  - ✓ Ogni record è composto da un insieme di *attributi*, di cui uno esprime la *classe* di appartenenza del record.
- Trova un *modello* per l'attributo di classe che esprima il valore dell'attributo in funzione dei valori degli altri attributi.
- Obiettivo: record non noti devono essere assegnati a una classe nel modo più accurato possibile
  - ✓ Viene utilizzato un *test set* per determinare l'accuratezza del modello. Normalmente, il data set fornito è *suddiviso* in training set e test set. Il primo è utilizzato per costruire il modello, il secondo per validarlo.
- I classificatori possono essere utilizzati sia a scopo descrittivo sia a scopo predittivo
- Sono più adatti ad attributi nominali (binari o discreti) poichè faticano a sfruttare le relazioni implicite presenti negli attributi ordinali, numerici o in presenza di gerarchie di concetti (es. scimmie e uomini sono primati)

Classificazione: un esempio

Tid	Attrib1	Attrib2	Attrib3	Class
1	Yes	Large	125K	No
2	No	Medium	100K	No
3	No	Small	70K	No
4	Yes	Medium	120K	No
5	No	Large	95K	Yes
6	No	Medium	60K	No
7	Yes	Large	220K	No
8	No	Small	85K	Yes
9	No	Medium	75K	No
10	No	Small	90K	Yes

Training Set



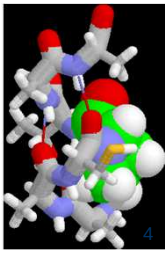
Tid	Attrib1	Attrib2	Attrib3	Class
11	No	Small	55K	?
12	Yes	Medium	80K	?
13	Yes	Large	110K	?
14	No	Small	95K	?
15	No	Large	67K	?

Test Set

Applicazioni

- Predire se una cellula tumorale è benigna o maligna in base alle sue caratteristiche
- Classificare se una transazione con carta di credito sia o meno fraudolenta

- Classificare le strutture proteiche secondarie in alpha-helix, beta-sheet, or random coil
- Classificare le news in base all'argomento:finanza, meteo, sport, intrattenimento, ecc.

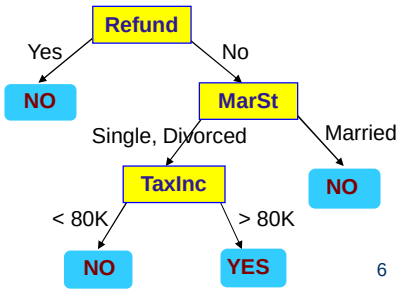


Tecniche di classificazione

- **Alberi decisionali o Decision Tree**
- Regole di decisione
- Nearest-neighbor
- Reti Bayesiane
- Reti neurali
- Support Vector Machines

I Decision Tree

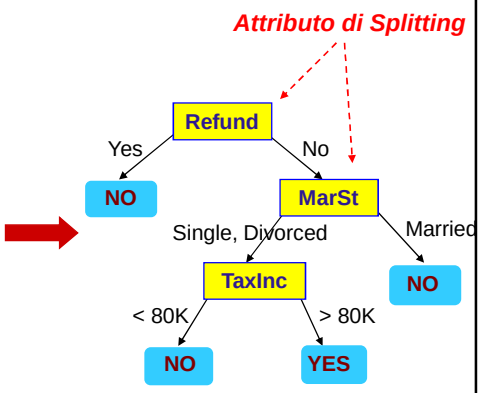
- È una delle tecniche di classificazione maggiormente utilizzate che permette di rappresentare con un albero un insieme di regole di classificazione.
- Struttura gerarchica che consiste di un insieme di nodi, correlati da archi (rami) orientati ed "etichettati". Si hanno due tipi di nodi:
  - ✓ Le classi sono definite nei nodi foglia mentre i rimanenti nodi sono etichettati in base all'attributo che partiziona i record. Il criterio di partizionamento rappresenta l'etichetta degli archi
- Ciascun percorso radice-foglia rappresenta una regola di classificazione



Decision Tree: un esempio

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

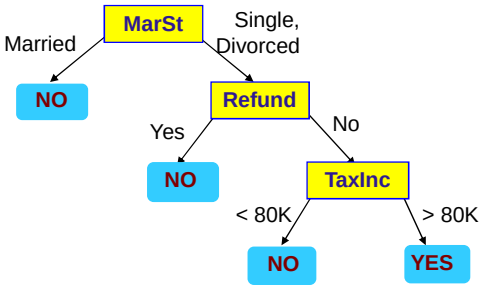
Training Data



Modello: Decision Tree

Decision Tree: un altro esempio

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



Ci possono essere più alberi di decisione per lo stesso data set

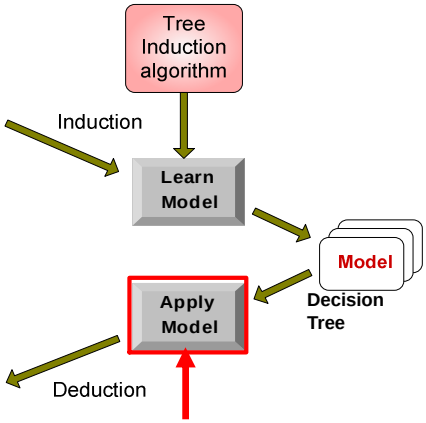
Applicare il modello al data set

Tid	Attrib1	Attrib2	Attrib3	Class
1	Yes	Large	125K	No
2	No	Medium	100K	No
3	No	Small	70K	No
4	Yes	Medium	120K	No
5	No	Large	95K	Yes
6	No	Medium	60K	No
7	Yes	Large	220K	No
8	No	Small	85K	Yes
9	No	Medium	75K	No
10	No	Small	90K	Yes

Training Set

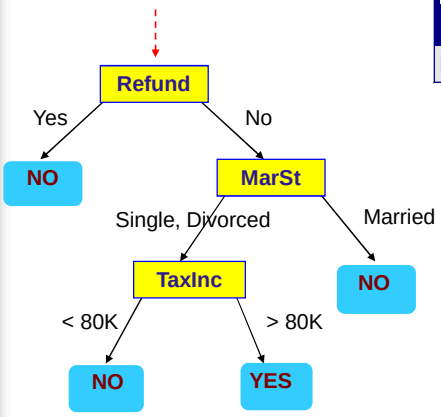
Tid	Attrib1	Attrib2	Attrib3	Class
11	No	Small	55K	?
12	Yes	Medium	80K	?
13	Yes	Large	110K	?
14	No	Small	95K	?
15	No	Large	67K	?

Test Set



Applicare il modello al data set

Si parte dalla radice



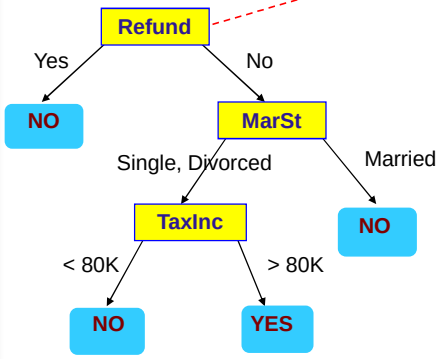
Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Applicare il modello al data set

Test Data

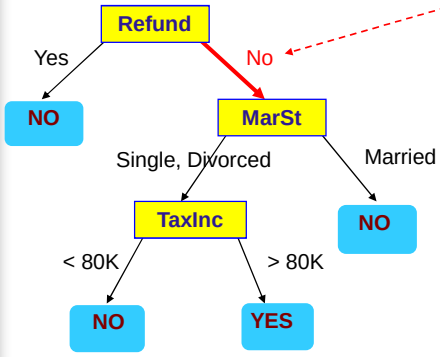
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Applicare il modello al data set

Test Data

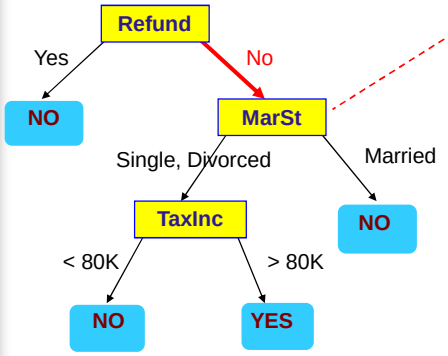
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Applicare il modello al data set

Test Data

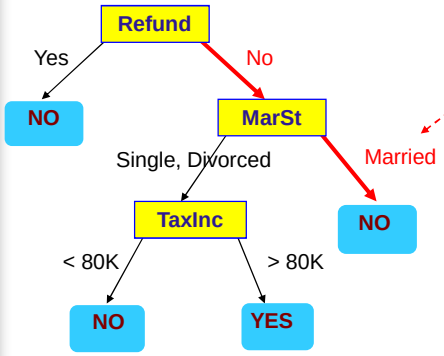
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Applicare il modello al data set

Test Data

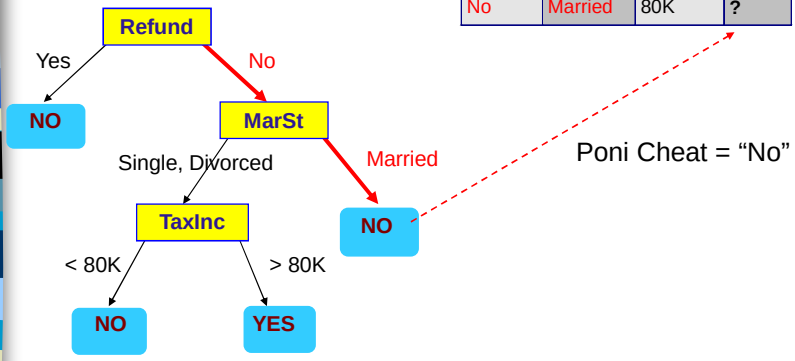
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Applicare il modello al data set

Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



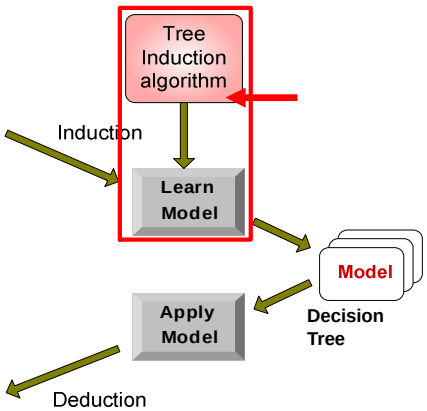
Decision Tree: imparare il modello

Tid	Attrib1	Attrib2	Attrib3	Class
1	Yes	Large	125K	No
2	No	Medium	100K	No
3	No	Small	70K	No
4	Yes	Medium	120K	No
5	No	Large	95K	Yes
6	No	Medium	60K	No
7	Yes	Large	220K	No
8	No	Small	85K	Yes
9	No	Medium	75K	No
10	No	Small	90K	Yes

Training Set

Tid	Attrib1	Attrib2	Attrib3	Class
11	No	Small	55K	?
12	Yes	Medium	80K	?
13	Yes	Large	110K	?
14	No	Small	95K	?
15	No	Large	67K	?

Test Set



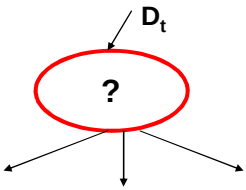
Induzione con Decision Tree

- Il numero di decision tree cresce esponenzialmente con il numero di attributi
- Gli algoritmi utilizzano generalmente tecniche greedy che fanno localmente la scelta “migliore”
- Sono a disposizione molti algoritmi:
  - ✓ Hunt’s Algorithm
  - ✓ CART
  - ✓ ID3, C4.5
  - ✓ SLIQ,SPRINT
- Devono essere affrontati diversi problemi
  - ✓ Scelta del criterio di split
  - ✓ Scelta del criterio di stop
  - ✓ Underfitting
  - ✓ Overfitting
  - ✓ Frammentaizone dei dati
  - ✓ Criterio di ricerca
  - ✓ Espressività
  - ✓ Replicazione degli alberi

Hunt’s Algorithm

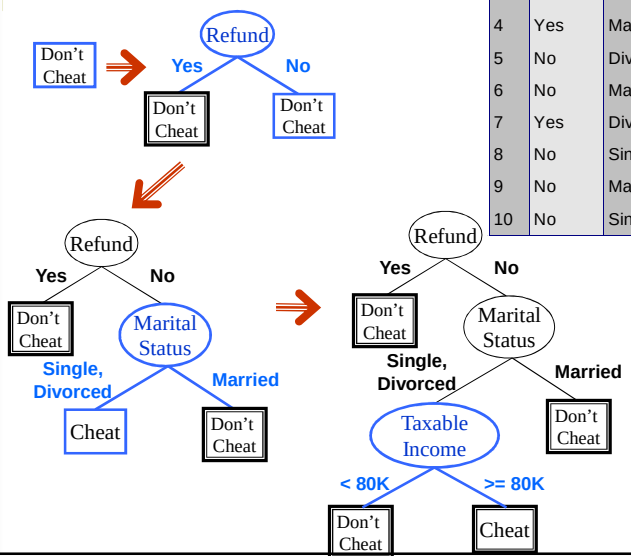
- Approccio ricorsivo che suddivide progressivamente un insieme di record D_t in insiemi di record via via più puri
- Sia D_t l’insieme dei record del training set corrispondenti al nodo t e $y_t=\{y_1,\ldots,y_k\}$ le possibili label di classe
- Procedura generale:
  - ✓ Se D_t contiene record appartenenti alla sola classe y_j, allora t è un nodo foglia con label y_j
  - ✓ Se D_t è un insieme vuoto, allora t è un nodo foglia a cui è assegnata una classe del nodo padre
  - ✓ Se D_t contiene record appartenenti a più classi, si scelga un **attributo e un criterio di split** per partizionare i record in più sottoinsiemi.
  - ✓ Si riapplichi ricorsivamente la procedura generale ai sottoinsiemi

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



Hunt's Algorithm

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



Pseudo-codice di massima

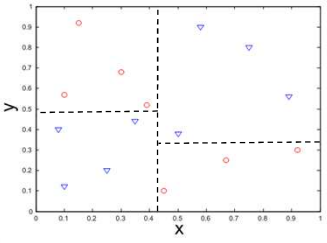
```
// Let E be the training set and F the attributes

result=PostPrune(TreeGrowth(E,F));

TreeGrowth(E,F)
  if StoppingCond(E,F)= TRUE then
    leaf=CreateNode();
    leaf.label=Classify(E);
    return leaf;
  else
   root = CreateNode();
   root.test_cond  = FindBestSplit(E,F);
    let V = {v | v is a possibile outcome of root.test_cond}
    for each v ∈ V do
       E_v = {e | root.test_cond (e)=v and e ∈ E}
       child = TreeGrowth(E_v,F);
       add child as descendants of root and label edge
                (root→child) as v
    end for
  end if
  return root;
end;
```

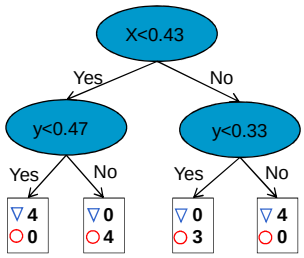
Alcune considerazioni…

- La ricerca di un albero di decisione ottimo è un problema NP-Completo, ma gli algoritmi eurisitci utilizzati sono molto efficienti
  - ✓ La maggior parte degli approcci eseguono una partizione ricorsiva top down basata su criteri greedy
- La classificazione utilizzando un albero decisionale è estremamente veloce e offre una facile interpretazione dei criteri
  - ✓ Il caso peggiore è O(w) dove w è la profondità dell'albero
- Gli alberi di decisione sono sufficientemente robusti rispetto alla presenza di attributi fortemente correlati
  - ✓ Uno dei due attributi non sarà considerato
  - ✓ E' anche possibile cercare di scartare uno degli attributi in fase di preprocessing mediante opportune tecniche di feature selection

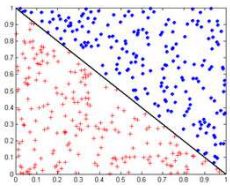
Alcune considerazioni…

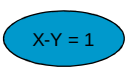
- L'espressività degli alberi decisionali è limitata alla possibilità di effettuare partizionamenti dello spazio di ricerca con condizioni che coinvolgono un solo attributo per volta
  - ✓ Decision boundary paralleli agli assi





- Questa suddivione non è ottenibile con alberi decisionali tradizionali





Elementi caratterizzanti

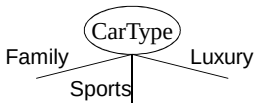
- A parte la logica di base per definire completamente un algoritmo per la costruzione di alberi decisionali è necesario definire:
  - ✓ **La condizione di split**
  - ✓ Il criterio che definisce lo split migliore
  - ✓ Il criterio per interrompere lo splitting
  - ✓ Le modalità per valutare la bontà di un albero decisionale

Come definire la condizione di split

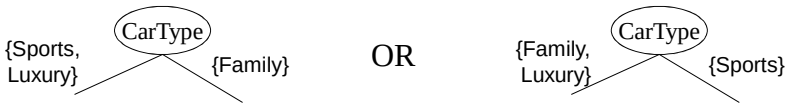
- Dipende dal tipo di attributo
  - ✓ Nominale
  - ✓ Ordinale
  - ✓ Continuo
- Dipende dal numero di split applicabili ai valori dell'attributo
  - ✓ A 2 vie
  - ✓ A più vie

Splitting con attributi nominali

- Split a più vie: crea tante partizioni quanti sono i valori dell'attributo

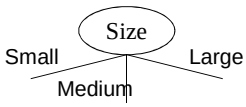


- Split a 2 vie: crea due partizioni e richiede di suddividere i possibili valori dell'attributo in modo ottimale

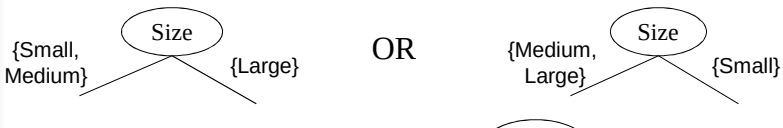


Splitting con attributi ordinali

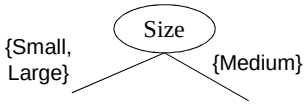
- Il partizionamento non deve violare l'ordinamento dei valori.
- Split a più vie: crea tante partizioni quanti sono i valori dell'attributo



- Split a 2 vie: crea due partizioni e richiede di suddividere i possibili valori dell'attributo in modo ottimale

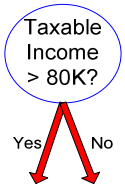


E' sensato questo split?

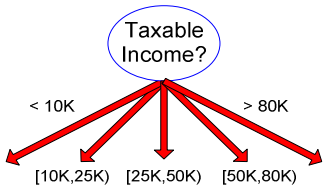


Splitting con attributi continui

- Split a più vie: la condizione di split può esere espressa come un test di comparazione che ha per risultato più range di valori. L'algoritmo deve considerare tutti i possibili range di valori come possibili punti di split
- Split a 2 vie: la condizione di split può esere espressa come un test di comparazione con risultato binario. L'algoritmo deve considerare tutti i valori come possibili punti di split



(i) Binary split



(ii) Multi-way split

Splitting con attributi continui

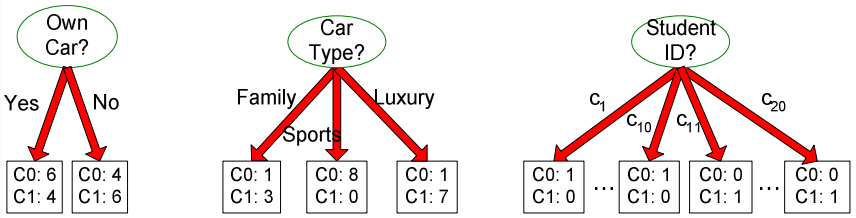
- Per gestire la complessità della ricerca del/i punto/i di split ottimali può essere utilizzata una tecnica di discretizzazione
  - ✓ Statica – si discretizza una sola volta prima di applicare l'algoritmo
  - ✓ Dinamica– si discretizza a ogni passo di ricorsione sfruttando le informazioni sulla distribuzione dei dati in input al nodo D_t.

Elementi caratterizzanti

- A parte la logica di base per definire completamente un algoritmo per la costruzione di alberi decisionali è necesario definire:
  - ✓ La condizione di split
  - ✓ **Il criterio che definisce lo split migliore**
  - ✓ Il criterio per interrompere lo splitting
  - ✓ Le modalità per valutare la bontà di un albero decisionale

Come determinare lo split migliore

- Prima dello split una sola classe con 10 record in classe C0 e 10 record in classe C1



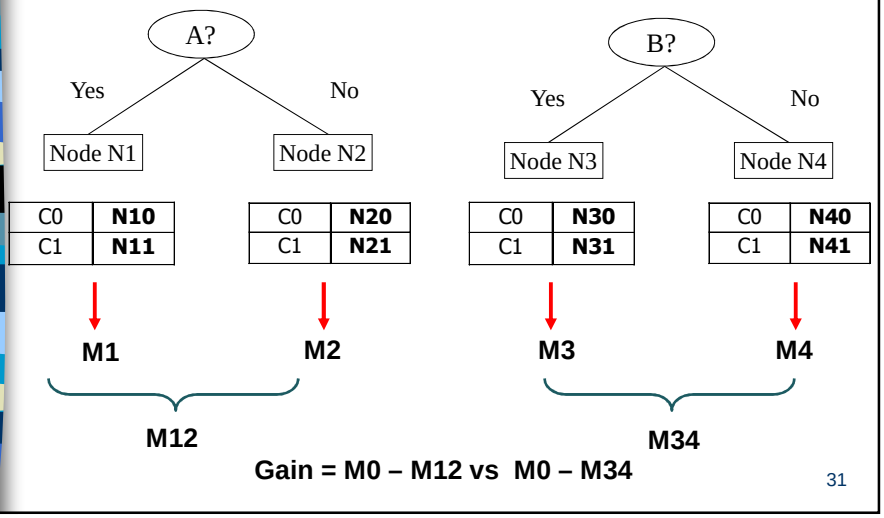
- Il criterio di split deve permettere di determinare classi più pure. Serve una misura di purezza
  - ✓ Gini index
  - ✓ Entropia
  - ✓ Misclassification error

Come determinare lo split migliore

Prima dello Splitting:

C0	N00
C1	N01

→ M0



Gain = M0 – M12 vs M0 – M34

Misure di impurità

- Dato un nodo p con record appartenenti a k classi e un suo partizionamento in n nodi figli
  - ✓ m = numero di record nel padre p
  - ✓ mi = numero di record nel figlio i

ATTENZIONE a non confondere il numero delle classi (k) e quello dei nodi figli (n)

- Gini index: usato in CART, SLIQ, SPRINT.

$$GINI(i) = 1 - \sum_{j=1}^{k} [p(j \mid i)]^2$$

- Entropia usato in ID3 e C4.5

$$Entropy(i) = -\sum_{j=1}^{k} p(j \mid i) \log p(j \mid i)$$

- Errore di classificazione

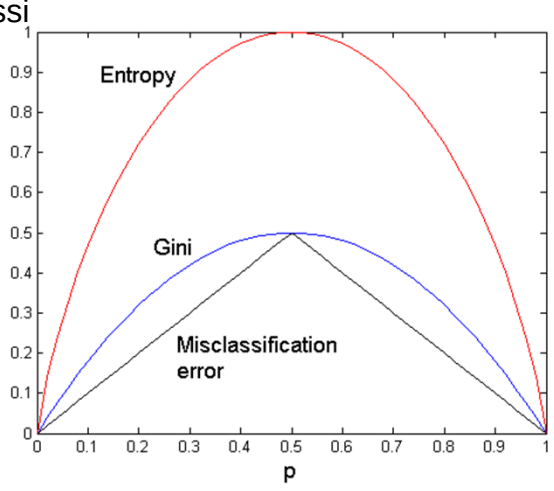
$$Error(i) = 1 - \max_{j \in K} p(j \mid i)$$

- Impurità complessiva dello split è data dalla seguente formula dove meas( ) è una delle misure introdotte

$$Impurity_{split} = \sum_{i=1}^{n} \frac{m_i}{m} meas(i)$$

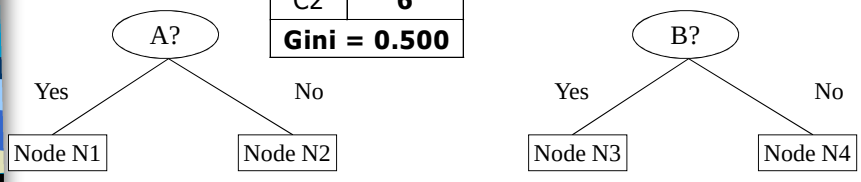
Una comparazione dei criteri di splitting

- Valore dei diversi indici, per un partizionamento in due classi



Calcolo di Gini con attributi binari

	Parent
C1	6
C2	6
Gini = 0.500	



	N1	N2
C1	4	2
C2	3	3
Gini=0.486		

	N3	N4
C1	5	1
C2	2	4
Gini=0.371		

Gini(N3) = 1 – $(5/7)^2$ – $(2/7)^2$ = 0.408

Gini(N4) = 1 – $(1/5)^2$ – $(4/5)^2$ = 0.320

Impurity= 7/12 * 0.408 + 5/12 * 0.320= 0.371

Calcolo di Gini con attributi categorici

- Solitamente è più efficiente creare una "**count matrix**" per ogni valore distinto dell'attributo di classificazione e quindi effettuare i calcoli utilizzando tale matrice

Split a più vie

	CarType		
	Family	Sports	Luxury
C1	1	2	1
C2	4	1	1
Gini	0.393		

Split a 2 vie
(find best partition of values)

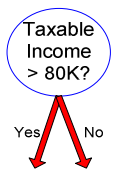
	CarType	
	{Sports, Luxury}	{Family}
C1	3	1
C2	2	4
Gini	0.400	

	CarType	
	{Sports}	{Family, Luxury}
C1	2	2
C2	1	5
Gini	0.419	

Calcolo di Gini con attributi continui

- Richiede di definire il punto di split mediante una condizione binaria
- Il numero delle condizioni possibili è pari al numero dei valori distinti dell'attributo
- Per ogni valore di split è possibile calcolare una count matrix
  - ✓ La matrice conterrà il conteggio degli elementi di ogni classe per valori dell'attributo maggiori o minori del valore di split
- Un approccio naive
  - ✓ Per ogni valore di split v, leggi il DB (con N record) per costruire la count matrix e calcola l'indice Gini
  - ✓ Computazionalmenente inefficiente $O(N^2)$
    - Scansione del DB O(N)
    - Ripetizione per ogni valore di v O(N)

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



Calcolo di Gini con attributi continui

- Una soluzione più efficiente prevede di:
  - ✓ Ordinare i record in base al valore dell'attributo
  - ✓ Leggere i valori ordinati e aggiornare la count matrix, quindi calcolare l'indice di Gini
  - ✓ Scegliere come split point il valore che minimizza l'indice

Valori ordinati

Cheat	No		No		No		Yes		Yes		Yes		No		No		No		No	
Taxable Income																				
	60		70		75		85		90		95		100		120		125		220	
	55		65		72		80		87		92		97		110		122		172		230	
	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>
Yes	0	3	0	3	0	3	0	3	1	2	2	1	3	0	3	0	3	0	3	0	3	0
No	0	7	1	6	2	5	3	4	3	4	3	4	3	4	4	3	5	2	6	1	7	0
Gini	0.420		0.400		0.375		0.343		0.417		0.400		***0.300***		0.343		0.375		0.400		0.420	

Possibili punti di split

Sono possibili ulteriori ottimizzazioni?

Split basato sul GAIN

- Utilizzando misure di impurità delle classi come Gini e Entropy richiede di scegliere il valore di split che massimizza il "guadagno" in termini di riduzione dell'impurità delle classi dopo lo split.
- Per esempio, considerando l'entropia, il guadagno del partizionamento di un nodo p in n nodi figli è:

$$GAIN_{split} = Entropy(p) - \left(\sum_{i=1}^{n} \frac{m_i}{m} Entropy(i) \right)$$

- Selezionare il valore di split che massimizza il GAIN tende a determinare criteri di split che generano un numero molto elevato di classi molto pure e con pochi record.
  - ✓ Partizionare gli studenti in base alla loro matricola garantisce che tutte le classi (formate da un solo studente) siano totalmente pure!!

Split basato sulle INFO

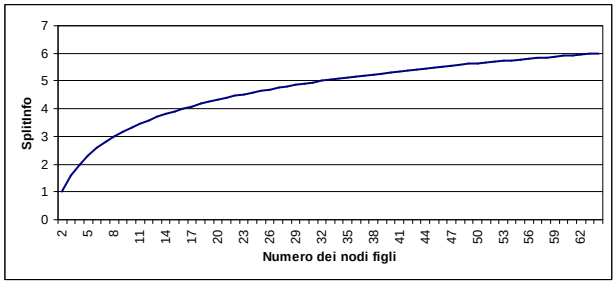
- Per evitare il problema della polverizzazione delle classi è preferibile massimizzare il Gain Ratio:
  - ✓ n = numero di nodi figli
  - ✓ m = numero di record nel padre p
  - ✓ mi = numero di record nel figlio i

$$GainRATIO_{split} = \frac{GAIN_{Split}}{SplitINFO} \qquad SplitINFO = -\sum_{i=1}^{n} \frac{m_i}{m} \log \frac{m_i}{m}$$

  - ✓ Maggiore il numero dei figli, maggiore il valore di SplitInfo con una conseguente riduzione del GainRatio
  - ✓ Per esempio, assumendo che ogni nodo figlio contenga lo stesso numero di record, SplitInfo = log n.
  - ✓ C4.5 utilizza il criterio basato su SplitINFO

Split basato sulle INFO

- Per evitare il problema della polverizzazione delle classi è preferibile massimizzare il Gain Ratio:
  - ✓ n = da 2 a 64
  - ✓ m = 100
  - ✓ mi = m/n



Esercizio

- Calcola gini index e information gain per il seguente problema binario e commenta i risultati

A	B	Classe
T	F	+
T	T	+
T	T	+
T	F	-
T	T	+
F	F	-
F	F	-
F	F	-
T	T	-
T	F	-

Elementi caratterizzanti

- A parte la logica di base per definire completamente un algoritmo per la costruzione di alberi decisionali è necesario definire:
  - ✓ La condizione di split
  - ✓ Il criterio che definisce lo split migliore
  - ✓ **Il criterio per interrompere lo splitting**
  - ✓ Le modalità per valutare la bontà di un albero decisionale

Crtiteri di stop per l'induzione di alberi decisionali

- Interrompere lo split di un nodo quando tutti i suoi record appartengono alla stessa classe
- Interrompere lo split di un nodo quando tutti i suoi record hanno valori similiari su tutti gli attributi
  - ✓ La classificazione sarebbe poco significativa e dipendente da piccole fluttuazioni dei valori
- Interrompere lo split quando il numero dei record nel nodo è inferiore a una certa soglia (*data fragmentation*)
  - ✓ Il criterio selezionato non sarebbe statisticamente rilevante

Metriche per la valutazione del modello

- La Confusion Matrix valuta la capacità di un classificatore sulla base dei seguenti indicatori
  - ✓ TP (true positive): record correttamente classificati come classe Yes
  - ✓ FN (false negative): record **in**correttamente classificati come classe No
  - ✓ FP (false positive): record **in**correttamente classificati come classe Yes
  - ✓ TN (true negative) record correttamente classificati come classe No

		Classe prevista	
		Class=Yes	Class=No
Classe effettiva	Class=Yes	TP	FN
	Class=No	FP	TN

- Se la classificazione utilizza n classi, la matrice di confusione sarà di dimensione n×n

Accuratezza

		Classe prevista	
		Class=Yes	Class=No
Classe effettiva	Class=Yes	TP	FN
	Class=No	FP	TN

- L'accuratezza è la metrica maggiormente utilizzata per sintetizzare l'informazione di una confusion matrix

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

- Equivalentemente potrebbe essere utilizzata la frequenza dell'errore

$$\text{Error rate} = \frac{FP + FN}{TP + TN + FP + FN}$$

Limiti dell'accuratezza

- L'acuratezza non è una metrica adeguata nel caso in cui le classi contengano un numero fortemente diverso di record
  - ✓ Consideriamo un problema di classificazione binario in cui
    - # record della classe 0 = 9990
    - # record della classe 1 = 10
  - ✓ Un modello che che predica sempre l'appartennza alla classe 0 avrà un'accuratezza di 9990/10000 = 99.9 %

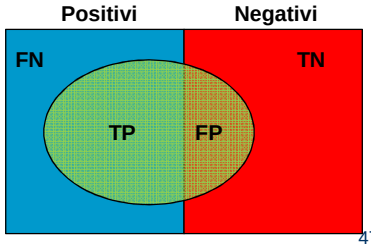
- Nel caso di problemi di classificazione binaria la classe la classe "rara" è anche chiamata *classe positiva*, mentre la classe che include la maggioranza dei record è chiamata *classe negativa*

Precision e Recall

- Precision e Recall sono due metriche utilizzate nelle applicazioni in cui la corretta classificazione dei record della classe positiva riveste una maggiore importanza
  - ✓ Precision misura la frazione di record risultati effettivamente positivi tra tutti quelli che erano stati classificati come tali
    - ✓ Valori elevati indicano che pochi record della classe negativa sono stati erroneamente classificati come positivi.
  - ✓ Recall misura la frazione di record positivi correttamente classificati
    - ✓ Valori elevati indicano che pochi record della classe positiva sono stati erroneamente classificati come negativi.

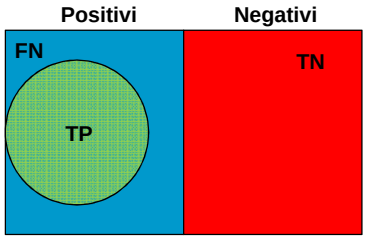
$$\text{Precision, } p = \frac{TP}{TP + FP}$$

$$\text{Recall, } r = \frac{TP}{TP + FN}$$

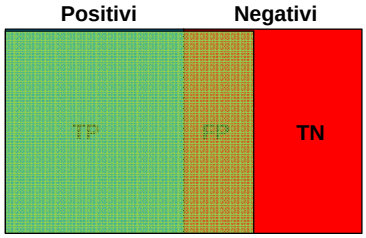


Precision e Recall

- precision = 1 se tutti i record positivi sono stati effettivamente individuati



- recall = 1 se non ci sono falsi positivi



- Se entrambi valgono 1 le classi predette coincidono con quelle reali

F-measure

- Una metrica che riassume precision e recall è denominata F-measure

$$\text{F - measure, F} = \frac{2rp}{r+p} = \frac{2\times TP}{2\times TP + FP + FN}$$

- F-measure rappresenta la media armonica tra precision e recall
  - ✓ La media armonica tra due numeri x e y tende a essere vicina al più piccolo dei due numeri. Quindi se la media armonica è elevata significa che sia precision, sia recall lo sono.
    - ✓ … e quindi non si sono verificati nè falsi negativi nè falsi positivi

Matrice dei costi

- La matrice dei costi codifica la penalità in cui si incorre nel classificare un record in una classe diversa
  - ✓ Una penalità negativa indica il "premio" che si ottiene per una corretta classificazione

C(M)=TP×C(Yes|Yes)+FP×C(Yes|No) +FN×C(No|Yes) +TN×C(No|No)

		Classe prevista j	
Classe effettiva i	C(i|j)	**Class=Yes**	**Class=No**
	Class=Yes	C(Yes|Yes)	C(Yes|No)
	Class=No	C(No|Yes)	C(No|No)

- Un modello costruito struttando, come funzione di purezza, una matrice di costo tenderà a fornire un modello a costo minimo rispetto ai pesi specificati

Calcolo del costo

Cost Matrix	PREDICTED CLASS		
	C(i|j)	+	-
ACTUAL CLASS	+	-1	100
	-	1	0

Model M_1	PREDICTED CLASS		
		+	-
ACTUAL CLASS	+	150	40
	-	60	250

Accuracy = 80%

Cost = 3910

Model M_2	PREDICTED CLASS		
		+	-
ACTUAL CLASS	+	250	45
	-	5	200

Accuracy = 90%

Cost = 4255

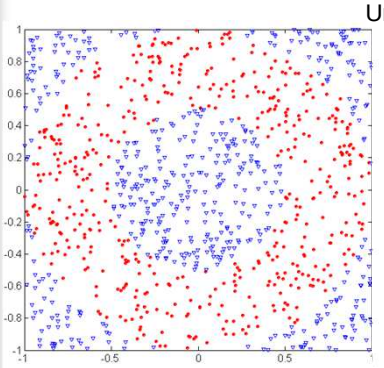
Errori di classificazione

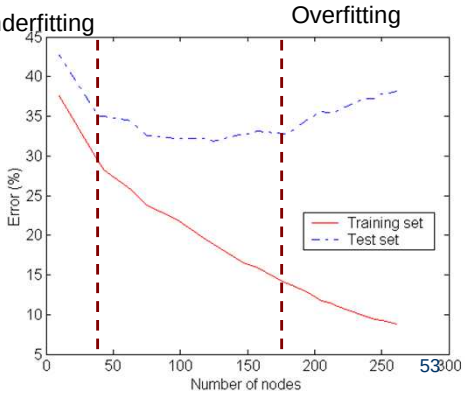
- Training error: sono gli errori che si commettono sul training set
- Generalization error: sono gli errori che si commettono sul test set (record su cui **non** è stato addestrato il sistema).

- Underfitting: il modello è troppo semplice e non consente una buona classificazione nè del training set, nè del test set
- Overfitting: il modello è troppo complesso, consente un'ottima classificazione del training set, ma una pessima classificazione del test set
  - ✓ Il modello non riesce a generalizzare poiché è basato su peculiarità specifiche del training set che non si ritrovano nel test set (es. rumore presente nel training set)

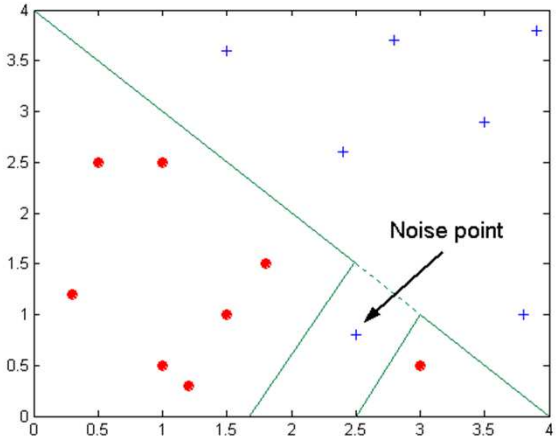
Underfitting e Overfitting

- ✓ 500 cerchi e 500 triangoli
- ✓ Punti circolari: 0.5 ≤ sqrt(x12+x22) ≤ 1
- ✓ Punti triangolari: sqrt(x12+x22) > 0.5 o sqrt(x12+x22) < 1



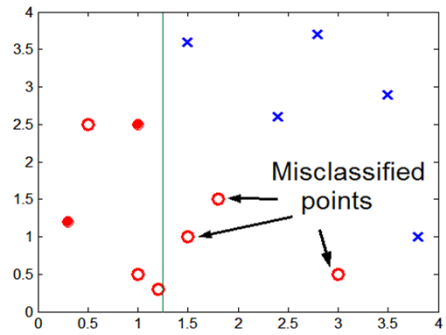


Overfitting dovuto al rumore



- I confini delle aree sono distorte a causa del rumore

Overfitting dovuto alla ridotta dimensione del training set



Punti presenti nel training set ✖ ●

- La mancanza dei punti nella parte bassa del diagramma rende difficile individuare una corretta classificazione per quella porzione di regione

Come gestire l'Overfitting: pre-pruing (Early stopping rule)

- Interrompere lo splitting prima che si arrivi a un albero di massima profondità
- Un nodo non può essere splittato ulteriormente se:
  - ✓ Il nodo non contiene istanze
  - ✓ Tutte le istanze appartengono alla medesima classe
  - ✓ Tutti gli attributi hanno gli stessi valori
- Condizioni più restrittive potenzialmente adottabili sono:
  - ✓ Interrompi lo splitting se il numero di istanze nel nodo è inferiore a una quantità fissata
  - ✓ Interrompi lo splitting se la distribuzione delle istanze tra le classi è indipendente dai valori degli attributi
  - ✓ Interrompi lo splitting se non si migliora la misura di purezza (es. Gini o information gain).

Come gestire l'Overfitting: post-pruning (reduced error pruning)

- Esegui tutti gli split possibili
- Esamina i nodi del decision tree ottenuto con una logica bottom-up
- Collassa un sottoalbero in un nodo foglia se questo permette di ridurre l'errore di generalizzazione (ossia sul validation set)
  - ✓ Scegli di collassare il sottoalbero che determina la massima riduzione di errore (N.B. scelta greedy)
- Le istanze nella nuova foglia possono essere etichettate
  - ✓ In base all'etichetta che compare più frequentemente nel sottoalbero
  - ✓ In base all'etichetta che comapre più frequentemente nelle istanze del training set che appartengono al sottoalbero
- Il post-pruning è più efficace ma implica un maggior costo computazionale
  - ✓ Si basa sull'evidenza del risultato di un albero completo

Note sull'overfitting

- L'overfitting determina alberi decisionali più complessi del necessario
- L'errore di classificazione compiuto sul training set non fornisce stime accurate circa il comportamento dell'albero su record sconosciuti
- Richiede nuove tecniche per stimare gli errori di generalizzazione

Stimare gli errori di generalizzazione

- Un albero di classificazione dovrebbe minimizzare l'errore sul data set reale, purtroppo in fase di costruzione si ha a disposizione solo il training set. Quindi l'errore sul data set reale deve essere stimato.
  - ✓ Re-substitution error: numero degli errori commessi sul training set (Σ e(t))
  - ✓ Generalization error: numero degli errori commessi sul data set reale (Σ e'(t))

- I metodi per stimare l'errore di generalizzazione sono:
  - ✓ Approccio ottimistico: e'(t) = e(t)
  - ✓ Approccio pessimistico
  - ✓ Minimum Description Length (MDL)
  - ✓ Utilizzo del test set: l'errore di generalizzazione è pari all'errore commesso sul test set.
    - Normalmente il test set è ottenuto estraendo dall'iniziale training set 1/3 dei record
    - Offre buoni risultati ma il rischio è quello di operare con un training set troppo piccolo

Occam's Razor

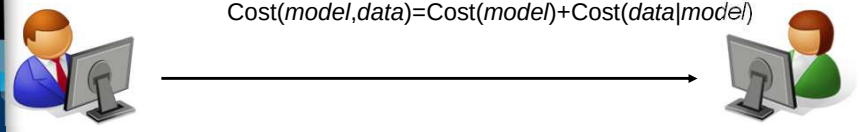
- Dati due modelli con errori di generalizzazioni similari è preferibile quello più semplice
  - ✓ Per modelli complessi c'è maggiore probabilità che il livello di errore sia determinato da condizioni accidentali sui dati
- E' quindi utile considerare la complessità del modello quando si valuta la bontà di un albero decisionale

- Nota: principio metodologico espresso nel XIV secolo dal filoso e frate francescano inglese William of Ockham

Minimum Description Length

- Dati due modelli si sceglie quello che minimizza il costo per descrivere una classificazione
  - ✓ Per descrivere il modello posso:
    - A) Inviare sequenzialmente le label di classe O(n)
    - B) Costruire un classificatore, e inviarne la descrizione assieme a una puntuale descrizione degli errori che esso commette

Cost(*model*,*data*)=Cost(*model*)+Cost(*data*|*model*)



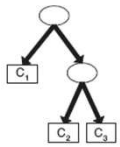
X	y
X1	1
X2	0
X3	1
…	…
Xn	0

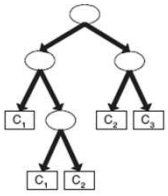
X	y
X1	?
X2	?
X3	?
…	…
Xn	?

MDM: un esempio

- Dataset con n record descritti da16 attributi binari e 3 valori di classe

Tree1
7 errori





Tree2
4 errori

- ✓ Ogni nodo interno è modellato con l'ID dell'attributo usato → log2(16)=4 bit
- ✓ Ogni foglia è modellata con l'ID della classe → log2(3)=2 bit
- ✓ Ogni errore è modellato con la sua posizione nel training set considerando n record→ log2(n)

Cost(Tree1)= 4×2 + 2×3 + 7 × log2(n) =14 + 7 × log2(n)
Cost(Tree2)= 4×4 + 2×5 + 4 × log2(n) =26 + 4 × log2(n)
Cost(Tree1) < Cost(Tree2) se n < 16

Approccio pessimistico

- Stima l'errore di generalizzazione sommando all'errore sul training set una penalizzazione legata alla complessità del modello
  - $e(t_i)$: errori di classificazione commessi nella foglia i
  - $\Omega(t_i)$: penalità associata alla foglia i
  - $n(t_i)$ numero di record del training set appartenenti alla foglia i

$$E(T) = \frac{\sum_{i=1}^{k} e(t_i) + \Omega(t_i)}{\sum_{i=1}^{k} n(t_i)}$$

- Per alberi binari una penalità pari a 0.5 implica che un nodo debba sempre essere espanso nei due nodi figli se migliora la classificazione di almeno un record

Esempio di post-pruning

Class = Yes	20
Class = No	10
Error = 10/30	

Errore di training (Prima dello split) = 10/30

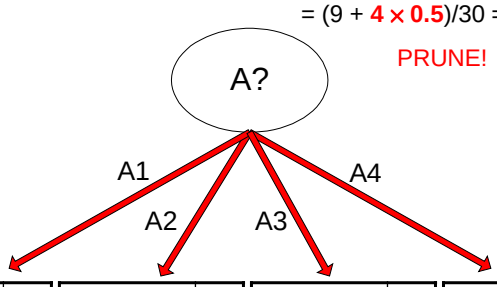
Errore pessimistico = (10 + **0.5**)/30 = 10.5/30

Errore di training (Dopo lo split) = 9/30

Errore pessimistico (Dopo lo split)

= (9 + **4 × 0.5**)/30 = 11/30

PRUNE!



Class = Yes	8	Class = Yes	3	Class = Yes	4	Class = Yes	5
Class = No	4	Class = No	4	Class = No	1	Class = No	1

Esempio di post-pruning

- ✓ Errore ottimistico?

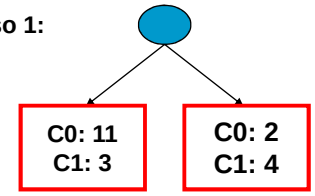
  Non tagliare in nessuno dei casi

- ✓ Errore pessimistico (penalità 0.5)?

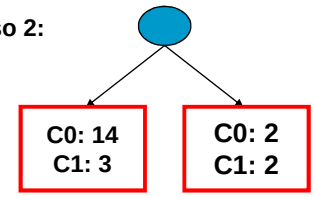
  Non tagliare nel caso 1, taglia nel caso 2

- ✓ Errore pessimistico (penalità 1)?

Caso 1:



Caso 2:



Elementi caratterizzanti

- A parte la logica di base per definire completamente un algoritmo per la costruzione di alberi decisionali è necesario definire:
  - ✓ La condizione di split
  - ✓ Il criterio che definisce lo split migliore
  - ✓ Il criterio per interrompere lo splitting
  - ✓ **Le modalità per valutare la bontà di un albero decisionale**

Costruzione del test set

- Holdout
  - ✓ Utilizzare 2/3 dei record per il training e 1/3 per la validazione
  - ✓ Svantaggi:
    - Opera con un training set ridotto
    - Il risultato dipende dalla composizione del training set e del test set
- Random subsampling
  - ✓ Consiste in una esecuzione ripetuta del metodo holdout in cui il dataset di training è scelto casualmente
- Cross validation
  - ✓ Partiziona i record in k sotto-insiemi distinti
  - ✓ Esegui il training su k-1 partizioni ed il test sulla rimanente
  - ✓ Ripeti il test k volte e calcola l'accuracy media
  - ✓ ATTENZIONE: la cross validation crea k classificatori diversi e quindi la validazione indica quanto il tipo di classificatore e i suoi parametri sono adatti per lo specifico problema
    - I k alberi decisionali costruiti potrebbero avere attributi e condizioni di split diverse a seconda delle caratterisitche del k-esimo training set
- Bootstrap…

Bootstrap

- Differentemente dagli altri approcci prevede il reimbussolamento dei record già selezionati
- Se il dataset iniziale è composto da N record è possibile creare un training set di N record in cui ogni record ha circa il 63.2% di probabilità di comparire (con N sufficientemente grande)

$$1 - (1 - 1/N)^N = 1 - e^{-1} = 0.632$$

  - ✓ I record non utilizzati nemmeno una volta nel training set corrente compongono il validation set
- Si ripete quindi la procedura *b* volte. Comunemente l'accuracy media del modello è calcolata come:

$$Acc_{boot} = \frac{1}{b}\sum_{i=1}^{b} 0.632 \times Acc_i + 0.368 \times Acc_s$$

dove Acc_i è l'accuracy del bootstrap i-esimo, mentre Acc_s è l'accuracy del dataset completo

- Il bootstrap non crea un (nuovo) dataset con più informazioni, ma permette di stabilizzare i risultati ottenibili del dataset a dispozione. E' quindi utilie soprattuto nel caso di dataset di piccole dimensioni.

C4.5

- Algoritmo per la costruzione di alberi decisionale
  - ✓ Estende l'algoritmo ID3 e Hunt
  - ✓ Una sua versione denominata **J48** è implementata in WEKA
- Caratteristiche:
  - ✓ Utilizza il GainRatio come criterio per determinare l'attributo di split
  - ✓ Gestisce gli attributi continui determinando uno split point che divide in due l'intervallo dei valori
  - ✓ Gestisce dati con valori mancanti. Gli attributi mancanti non sono considerati per calcolare il GainRatio.
  - ✓ Può gestire attributi a cui sono associati pesi diversi
  - ✓ Esegue postPruning dell'albero creato
- La costruzione dell'albero si interrompe quando:
  - ✓ Il nodo contiene record appartenenti a una sola classe
  - ✓ Nessun attributo permette di determinare un GainRatio positivo
  - ✓ Il nodo non contiene record

Esercizio

- Utilizzando l'errore di classificazione come misura, identificare quale attributo deve essere scelto per primo e quale per secondo
  - ✓ Calcolare le matrici di contingenza
  - ✓ Calcolare l'information gain

A	B	C	# istanze	
			+	-
T	T	T	5	0
F	T	T	0	20
T	F	T	20	0
F	F	T	0	5
T	T	F	0	0
F	T	F	25	0
T	F	F	0	0
F	F	F	0	25

- Come cambiano i risultati se si utilizza come attributo di split quello peggiore? Commentare il risultato