Tecniche di DM: Alberi di decisione ed algoritmi di classificazione

Vincenzo Antonio Manganaro

vincenzomang@virgilio.it, www.statistica.too.it

Indice

Nella prima parte introdurremo le tecniche di DM relative alla classificazione. Dopo aver descritto la struttura degli alberi di decisione, considereremo i più diffusi algoritmi di classificazione.

1 Concetti preliminari: struttura del dataset negli algoritmi di classificazione, classificazione e raggruppamento.

Negli algoritmi di classificazione i dati in input, chiamati anche *training set*, consistono in records ognuno dei quali avente attributi o caratteristiche multiple. Inoltre ogni record è etichettato con una speciale etichetta di classe. Il seguente schema mostra la struttura di un generico training set:

	ATTRIB. 1(A_1)	ATTRIB. 2 (A_2)	.....	ATTRIB. p (A_p)	CLASSE
Record 1					
Record 2					
Record 3					
.....					
.....					
Record n					

Obiettivo della classificazione è quello di analizzare i dati in imput e sviluppare un'accurata descrizione o un modello per ogni classe, usando le caratteristiche (espresse nell'esempio attraverso gli attributi A_1, A_2, ..., A_p) presenti nei dati. Gli algoritmi di classificazione portano all'identificazione di schemi o insiemi di caratteristiche che definiscono la classe cui appartiene un dato record. In genere, partendo dall'utilizzo di insiemi esistenti e già classificati, si cerca di definire alcune regolarità che caratterizzano le varie classi. Le descrizioni delle classi vengono usate per classificare records, di cui non si conosce la classe di appartenenza, o per sviluppare una migliore conoscenza di ogni classe nel dataset. Non a caso, alcune delle applicazioni di maggiore interesse di questa tecnica di DM includono la ricerca, da parte delle banche, di categorie di clienti ai quali concedere un prestito (in tal caso ogni record è etichettato come buon creditore o cattivo creditore e gli attributi sono i dati dei clienti relativi all'età, al reddito, ecc ...) o applicazioni di target marketing, con cui un'impresa può individuare, sulla base delle caratteristiche dei clienti presenti nel database, un proprio target di mercato allo scopo di rafforzare la propria posizione

in un determinato settore (in tal caso etichettando ogni record del database come cliente fedele e cliente non fedele).
Esiste una sostanziale differenza, che è opportuno chiarire, tra le tecniche di classificazione e le tecniche di raggruppamento o clustering (che avremo modo di considerare più avanti nella tesi). Tramite la classificazione l'utente comunica al tool di DM la caratteristica chiave, di cui i membri del gruppo devono essere dotati, e il tool non si occupa di nessun altro attributo che i membri del gruppo possono avere in comune. Ad esempio, si può dire al tool di DM di creare dei gruppi di clienti sulla base dei diversi livelli di entrate che apportano all'azienda. Utilizzando queste informazioni, l'azienda può investire tempo e denaro per corteggiare potenziali clienti con caratteristiche simili. In altre parole, nella classificazione esiste un attributo madre, il cui numero di modalità rappresenterà il numero dei gruppi che si verranno a formare (questo attributo madre è proprio la classe di appartenenza). Due records appartenenti al medesimo gruppo in un processo di classificazione possono in realtà essere fortemente diversi fra loro. Tale diversità è legata al fatto che il tool di DM ignora gli attributi al di fuori della classe di appartenenza. Nel raggruppamento non esiste un numero di gruppi prefissato. In tal caso il tool di DM crea dei gruppi sulla base di tutti gli attributi presenti nel database, in modo che ogni gruppo sia caratterizzato da elementi "simili" in termini degli attributi descritti nel database e che due elementi appartenenti a gruppi diversi siano sufficientemente "distanti tra di loro".

2 Gli alberi di decisione.

Gli *alberi di decisione* costituiscono il modo più semplice di classificare degli "oggetti" in un numero finito di classi. Essi vengono costruiti suddividendo ripetutamente i records in sottoinsiemi omogenei rispetto alla variabile risposta [8]. La suddivisione produce una gerarchia ad albero, dove i sottoinsiemi (di records) vengono chiamati *nodi* e, quelli finali, *foglie*. In particolare, i nodi sono etichettati con il nome degli attributi, gli archi (i rami dell'albero) sono etichettati con i possibili valori dell'at-

[8] Si ricordi che la variabile risposta è l' attributo classe del dataset e quando si parla di omogeneità rispetto alla variabile risposta ci si riferisce al fatto che le combinazioni di attributi dell'oggetto conducono alla medesima modalità di classe.

tributo soprastante, mentre le foglie dell'albero sono etichettate con le differenti modalità dell'attributo classe che descrivono le classi di appartenenza. Un oggetto è classificato seguendo un percorso lungo l'albero che porti dalla radice ad una foglia. I percorsi sono rappresentati dai rami dell'albero che forniscono una serie di regole. Per capire il meccanismo con il quale, a partire dal training set, si arriva alla costruzione di un albero di decisione, nonché la struttura dell'albero stesso, supponiamo che i dati del training set provengano da una compagnia di assicurazioni, nella quale un esperto ha assegnato ad ogni utente un livello di rischio (A=alto, B=basso) basandosi su esperienze precedenti e incidenti effettivamente avvenuti. Supponiamo che il training set sia il seguente (S sta per sportivo, FA sta per familiare, AU sta per autocarro):

	ETÁ	TIPO DI AUTOVEICOLO	RISCHIO
Utente 1	17	S	A
Utente 2	43	FA	B
Utente 3	68	FA	B
Utente 4	32	AU	B
Utente 5	23	FA	A
Utente 6	18	FA	A
Utente 7	20	FA	A
Utente 8	45	S	A
Utente 9	50	AU	B
Utente 10	64	AU	A
Utente 11	46	FA	B
Utente 12	40	FA	B

La compagnia di assicurazioni vorrebbe una procedura automatica che sia in grado di segnalare se un nuovo cliente può essere rischioso.
L'esempio rappresenta proprio un tipico problema di classificazione che può essere risolto con la costruzione di un albero di decisione associato al training set di cui

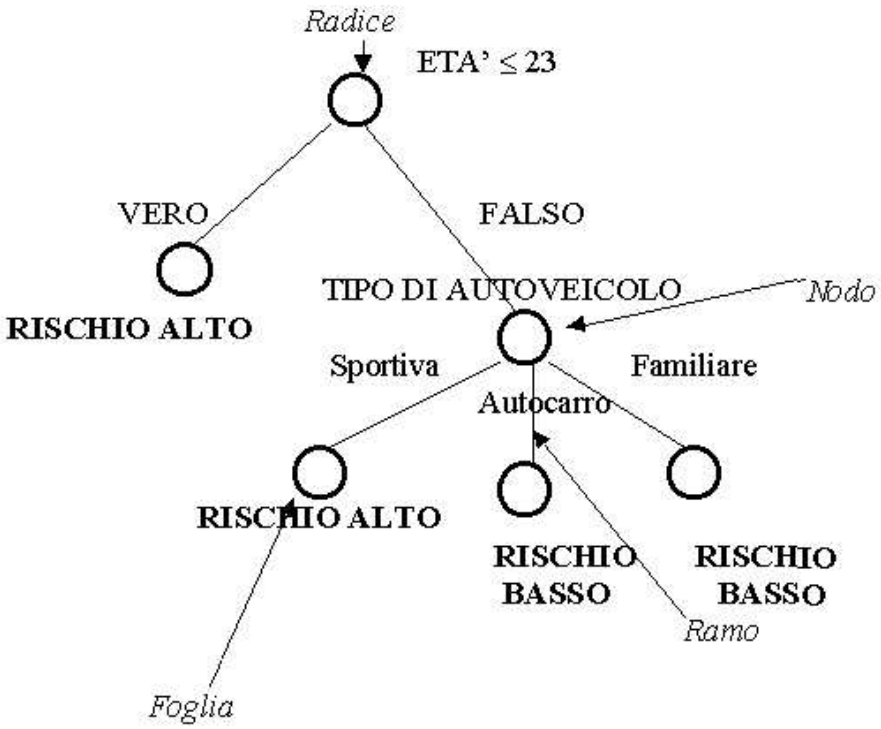

Figura 1: Esempio di un albero di decisione: i nodi rappresentano le combinazioni di attributi presenti, i rami i risultati dei test effettuati sui nodi mentre le foglie rappresentano il risultato della classificazione.

sopra. La figura 3.1 fornisce un esempio di albero associato al training set.
Sebbene gli alberi di decisione forniscano una rappresentazione compatta di una procedura di classificazione, spesso risulta difficile presentare a persone non esperte una struttura del tipo di quella presentata in fig 3.1, per cui si preferisce utilizzare una rappresentazione equivalente, ma più intuitiva, con l'ausilio di "regole di classificazione", che possono essere ricavate facilmente dall'albero e che nel caso dell'esempio in questione potrebbero essere espresse nel modo seguente:

- *SE ETÁ $\leq$ 23 O TIPO DI AUTOVEICOLO É SPORTIVO $\Rightarrow$ IL RISCHIO É ALTO*

- *SE ETÁ > 23 E TIPO DI AUTOVEICOLO É {FAMILIARE, AUTOCARRO} ⇒ IL RISCHIO É BASSO*

Risulta dunque abbastanza agevole comprendere quanto gli alberi di decisione possano essere utili per differenti fini e come essi debbano il loro nome al fatto di essere degli importanti strumenti per il supporto alle decisioni.
Esiste un'importante differenza all'interno della famiglia degli alberi di decisione. Vi sono alberi di decisione usati per predire variabili categoriali (il nuovo cliente della compagnia di assicurazione è un potenziale cliente a rischio o no?) e sono comunemente chiamati *alberi di classificazione,* dal momento che posizionano i records in categorie o classi. Gli alberi di decisione usati per predire variabili di tipo continuo sono invece noti in letteratura con il nome di *alberi di regressione.* Chiaramente esistono molte più difficoltà nel gestire dati di tipo continuo, per cui nella maggior parte dei casi gli algoritmi di classificazione esprimono tali dati sotto forma di intervalli discreti.
L'esempio visto è in realtà abbastanza semplice: l'albero di decisione costruito (si veda fig. 3.1) risulta facile da capire ed interpretare; ma la struttura di un albero di decisione può diventare molto complicata, soprattutto nei casi derivati da database contenenti centinaia di attributi ed una variabile risposta con differenti classi. In situazioni del genere, lasciar "crescere" l'albero senza stabilire un limite di qualsiasi natura può far si che l'albero ottenuto diventi non interpretabile, crei un numero elevato di regole, di fatto sovraadattando i dati. Esistono, quindi, dei criteri di controllo che limitano la crescita degli alberi, basati o sul massimo numero di regole ottenibili dalla classificazione o sulla massima "profondità" raggiungibile dall'albero o ancora sul numero minimo di records che devono essere presenti in ogni nodo per poter effettuare la divisione (*splitting*) in quel nodo. In tale ambito rientra anche la fase di *pruning* dell'albero che considereremo più avanti e che consiste nell'ottenere da un albero il più piccolo "sottoalbero", che di fatto non comprometta l'accuratezza[1] della classificazione/previsione resa possibile dall'albero madre. Ad esempio, un ramo o un sottoalbero che l'utilizzatore giudica irrilevante perché ha un numero esiguo di casi, potrebbe essere rimosso effettuando di fatto un'operazione di "pota-

[1] Ci riferiamo all'accuratezza come alla capacità dell'albero di classificare correttamente i records nelle rispettive classi di appartenenza, così come si vede osservando il training set.

tura". L'algoritmo CART[2] effettua la fase di pruning semplicemente verificando se il miglioramento nell'accuratezza giustifica la presenza aggiuntiva di altri nodi.
Il percorso che porta alla costruzione di alberi di decisione non è certamente semplice; tuttavia nei prossimi paragrafi descriveremo brevemente i principali algoritmi di costruzione di alberi di decisione. Gli algoritmi che affronteremo in tale ambito sono prevalentemente di tipo univariato nel senso che considerano un solo "predittore" alla volta (es. ETÀ ≤ 23 o TIPO DI AUTOVEICOLO con rispettivamente ETÀ e TIPO DI AUTOVEICOLO predittori). Ma vi possono essere algoritmi che generano i meno noti *alberi obliqui*, nei quali il criterio di divisione o anche il predittore potrebbe essere una combinazione lineare di variabili (es. SALARIO $<$ (0.35 $*$ IPOTECA)), o alberi nei quali i predittori siano delle combinazioni logiche di variabili (es. SALARIO $>$ 35.000 O IPOTECA $<$ 150.000).

3 Criteri per la costruzione di alberi di decisione: Entropia ed Information gain.

Dedicheremo questo paragrafo all' esposizione di concetti utili per la costruzione di alberi di decisione. A tale scopo, consideriamo un problema di classificazione con sole due classi, "+" e "-", e sia S l'insieme di records attraverso i quali dobbiamo creare un albero di decisione. Se indichiamo con P_+ la percentuale di esempi classificati con + e con P_- la percentuale di esempi classificati con -, si definisce *entropia* di S, $H(S)$, l'espressione:

$$H(S) = -P_+ log_2 P_+ - P_- log_2 P_-. \tag{1}$$

La figura 3.2 riporta l'andamento dell'entropia al variare delle combinazioni di percentuali P_+ e P_-. Dal grafico si evidenzia come $0 \leq H(S) \leq 1$ ed in particolare $H(S) = 0$ nel caso in cui $P_+ = 100\%$ o $P_- = 100\%$, ovvero nei casi in cui la totalità degli esempi è classificata in una sola delle due classi. D'altro canto $H(S) = 1$ nel caso in cui $P_+ = 50\%$ e conseguentemente $P_- = 50\%$, ovvero nel caso in cui gli esempi sono esattamente divisi nelle due classi.

[2]Classification And Regression Trees. É il più diffuso algoritmo per la costruzione di alberi di classificazione e di regressione.

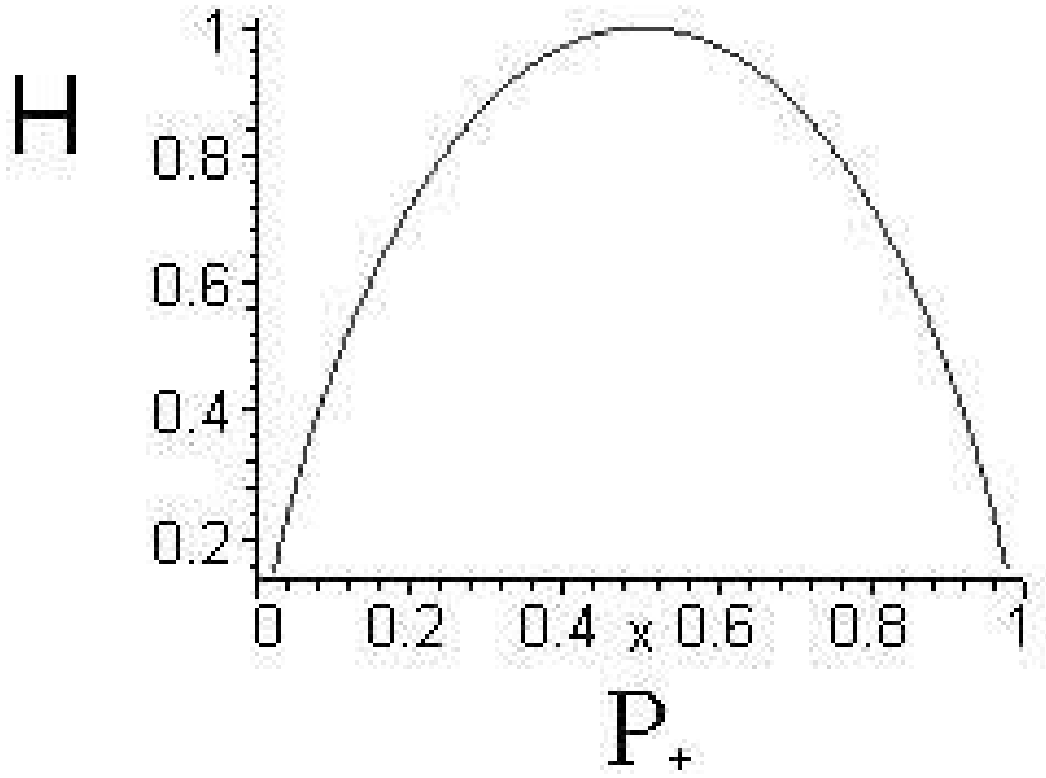

Figura 2: Andamento dell'entropia nel caso di due sole classi. Come è evidente il massimo di entropia si ha in corrispondenza di $P_+ = P_- = 0.5$.

In generale, se i records sono classificati in c classi, l'entropia si definisce come:

$$H(s) = -\sum_{i=1}^{c} P_i log_2 P_i. \tag{2}$$

L'entropia è una misura dell'ordine dello spazio dei records che si considerano per la costruzione degli alberi di decisione. Un valore elevato di entropia esprime il "disordine" che caratterizza lo spazio dei records, ovvero una maggiore difficoltà nell'assegnare ciascun record alla propria classe sulla base degli attributi che caratterizzano la classe: più l'entropia è alta, meno informazione abbiamo sull'attributo classe. Il caso delle sole classi "+" e "-" con un valore di entropia pari a 1, corrispondente a $P_+ = 50\%$ e $P_- = 50\%$, sarebbe il caso più difficile da considerare, se l'obiettivo è quello di individuare gli attributi che caratterizzano le due classi.
In generale, partendo da una situazione di massimo disordine in cui $H(S) = 1$ o da un qualunque valore elevato di entropia, una partizione dei records effettuata rispetto ad un certo attributo A porterebbe ad un nuovo valore $H'(S)$ tale che risulta $H'(S) \leq H(S)$ e quindi ad una diminuzione di entropia. In tale ambito rientra il concetto di *information gain*[3], definito come la diminuzione di entropia che si ottiene partizionando i dati rispetto ad un certo attributo. Se indichiamo con $H(S)$ il valore iniziale di entropia e con $H(S, A)$ il valore dell'entropia dopo aver partizionato i records con l'attributo A, l'information gain, che indicheremo con G, è data da:

$$G = H(S) - H(S, A). \tag{3}$$

Tale quantità è tanto maggiore quanto più elevata è la diminuzione di entropia dopo aver partizionato i dati con l'attributo A. Dunque un criterio di scelta dei nodi di un eventuale albero di classificazione[4] consiste nello scegliere di volta in volta l'attributo[5] A che dà una maggiore diminuzione di entropia o che analogamente massimizza l'information gain.

[3]La traduzione italiana è "guadagno di informazione" ed esprime appunto l'informazione aggiuntiva che si ha sull'attributo classe dopo aver partizionato i records attraverso gli altri attributi del training set.

[4]In realtà gran parte degli algoritmi di classificazione si basano proprio sul criterio dell'information gain.

[5]Se si tratta di costruire alberi di regressione bisogna scegliere oltre all'attributo di tipo continuo anche il valore soglia, cioè il valore in base al quale effettuare la divisione dei records e che inoltre massimizza l'information gain.

L'information gain ha valori molto elevati in corrispondenza di attributi che sono fortemente informativi e che quindi aiutano ad identificare con buona probabilità la classe di appartenenza dei records. Spesso, però, più gli attributi sono informativi, più perdono di generalità; ad esempio, nel database di una compagnia telefonica, il campo codice fiscale è altamente informativo, ha dunque un alto valore di information gain, dal momento che identifica con certezza l'utente, ma non è per nulla generalizzabile. L'ideale è dunque individuare campi altamente informativi con un buon grado di generalizzazione.

4 Accuratezza degli alberi di decisione.

Esaminiamo ora alcuni concetti con i quali è possibile quantificare le potenzialità nel classificare (*accuratezza*) per un albero.
Se indichiamo con $\overline{n}_i$ il numero totale di records del training set che terminano nella foglia i e con n_i il numero di record classificati correttamente in i, è possibile associare ad ogni foglia i un errore ϵ_i dato dalla frazione di records classificati correttamente nel training set:

$$\epsilon_i = \frac{n_i}{\overline{n}_i} \tag{4}$$

L'errore associato a tutto l'albero è dato dalla somma degli errori di tutte le foglie, pesati rispetto alla probabilità che un record finisca su ciascuna foglia. Se il numero totale di records del training set è N, la probabilità che un determinato record finisca nella foglia i è data da:

$$\frac{\overline{n}_i}{N}$$

e dunque l'espressione dell'errore associato a tutto l'albero (E) è la seguente:

$$E = \sum_i \frac{n_i}{N} \epsilon_i \tag{5}$$

È chiaro che la situazione di ottimo si ha quando $E = 1$; in tal caso, infatti, l'albero classifica perfettamente tutti i records del training set, poiché $\epsilon_i = 1$ per ogni i ed E risulta uguale alla probabilità che un record cada in una qualsiasi foglia e cioè ad uno.
Tuttavia, questo modo di valutare le potenzialità di un albero nel classificare non è esente da critiche: basti pensare al fatto che gli errori di classificazione non sono tutti

uguali[6] . Per specificare queste differenze negli errori di classificazione potremmo definire delle matrici di costo, nelle quali ogni cella della matrice specifica quanto "costa" un certo errore di classificazione. Si tratta di matrici che hanno la diagonale principale fatta tutta di zeri (casi in cui l'albero classifica perfettamente i records), mentre altrove i valori sono generalmente $\neq 0$ e rappresentano i costi della errata classificazione dei records. Queste matrici sono in generale non simmetriche, poiché il costo dell'aver classificato un record nella classe i piuttosto che nella vera classe j è, di regola, diverso dal costo di errata classificazione opposto.

5 L'algoritmo CART.

Concentriamoci ora sugli algoritmi di classificazione che fanno uso degli alberi di decisione. Esistono in letteratura diversi algoritmi e tra questi *CART, C4.5, CHAID, SLIQ, SPRINT.* Da parte nostra cercheremo di capire la logica sottostante questi algoritmi, concentrandoci esclusivamente sul primo di questi, che costituisce tra l'altro il più utilizzato, evitando comunque di entrare troppo in dettagli, il cui livello andrebbe oltre gli scopi di questa tesi.
L'algoritmo CART permette di generare alberi binari, ovvero alberi in cui ad ogni nodo corrispondono due soli rami; tuttavia il CART è alla base di altri algoritmi che generano alberi più complessi. Esso si articola in due fasi:

- **GENERAZIONE DELL'ALBERO.**
- **PRUNING DELL'ALBERO.**

Infatti con la prima fase spesso si ottengono alberi completi, ma abbastanza complessi, nel senso che sono costituiti da un numero molto elevato di regole. Per questo motivo, affinché l'albero possa essere effettivamente utile nel classificare records e nel fornire regole efficaci, è necessario sfoltire la ridondanza dell'albero, eliminando i rami meno significativi (pruning sta proprio per potatura).
Consideriamo la fase di generazione dell'albero: si parte dalla totalità dei records

[6]Classificare, ad esempio, un utente "non a rischio" come "a rischio" è un errore che reca meno danno dell'errore contrario; diagnosticare un tumore come benigno quando in realtà questo è maligno risulta sicuramente più pericoloso del caso contrario.

appartenenti al training set e si comincia una divisione binaria in classi. Il criterio con cui viene effettuata questa divisione si basa sull'information gain e consiste nell'individuare l'attributo che mi dà il massimo guadagno di informazione, al fine di stabilire l'appartenenza di un record ad una classe. Nel caso di una divisione binaria in classi, il concetto di information gain può essere particolarizzato nel più specifico concetto di diversità (D). La diversità è una funzione della frazione di records classificati in ciascuna classe. Ad esempio, se N è il numero dei records del training set, una prima divisione binaria dei records porterebbe alla formazione di due classi ognuna delle quali possiede una determinata frazione della totalità dei records. Indicate con P_1 e P_2 tali frazioni, la funzione diversità $D(P_1, P_2)$ può essere pensata come ad una funzione, avente valori in $[0, 1]$, che fornisce una misura di quanto i records sono equamente distribuiti nelle classi. Essa è definita in modo tale che $D = 1$ se i records sono concentrati in una sola classe e $D = 0$ se i records sono divisi in parti perfettamente uguali fra le due classi. Poiché l'obiettivo dell'algoritmo è quello di operare una classificazione posizionando i records in classi prestabilite (si ricordi la sostanziale differenza tra classificazione e clustering dove, nell'ultimo caso, non si ha una specificazione iniziale delle classi), esso cercherà di posizionare i records in maniera "quanto più possibile equa" tra le diverse classi prestabilite e non considererà quei campi o quei valori dei campi che non portano ad una divisione equa dei records. In termini formali, l'algoritmo nelle varie fasi sceglierà dei *campi splitter* (campi divisori presenti nel training set) e proverà diversi valori in modo che in ogni nodo venga minimizzata la funzione diversità. Minimizzare tale funzione coincide col trovare nelle varie fasi i campi e i valori dei campi che in quella fase operano la bipartizione più equa possibile e che quindi c'è da ritenere che danno in quel contesto un'informazione maggiore (in un gruppo di oggetti o individui l'informazione più interessante può essere un qualche attributo che bipartisce perfettamente l'insieme). Il procedimento è chiaramente di tipo iterativo e si arresterà quando non sarà più possibile, modificando le posizioni dei campi e/o i valori dei campi, diminuire le funzioni diversità. Veniamo ora alla fase di pruning (potatura) dell'albero; è necessario, infatti, per applicare efficacemente l'albero ottenuto ad altri datasets eliminare alcuni rami, diminuendo l'errore complessivo dell'albero. A questo punto introduciamo un *tasso di errore aggiustato:*

$$A_t = E_t + \alpha\lambda_t \tag{6}$$

dove l'indice t indica il particolare sottoalbero e λ_t è il numero di foglie nel sottoalbero t. Il tasso di errore aggiustato tiene conto del numero di foglie λ_t, nonché del peso di tale numero (peso indicato con α); esso permette inoltre di effettuare confronti tra i sottoalberi e l'albero completo. Se $t = 0$ si ha l'albero completo e λ_0 è il numero di foglie dell'albero, A_0 è l'errore aggiustato dell'albero completo. Esso coincide con E_0 solo se $\alpha = 0$ ed in generale se $\alpha = 0$ l'errore aggiustato di un qualunque sottoalbero coincide con E_t. L'algoritmo di pruning prevede le seguenti fasi:

1. fissato α si calcola A_t per ogni sottoalbero contenente la radice;
2. si aumenta α da 0 a 1 fino ad ottenere un sottoalbero t_1 per cui risulta $A_{t_1} \leq A_0$ e si eliminano tutti i rami che non sono parte di t_1;
3. si procede allo stesso modo fino ad arrivare alla radice ottenendo in tal modo un set di sottoalberi $\{t_1, t_2, \ldots\ldots\}$;
4. si calcola l'errore di ogni sottoalbero utilizzando un altro insieme di dati (*test set*) e si sceglie il sottoalbero con l'errore minore.

6 Conclusioni.

Non possiamo concludere questo capitolo dedicato alla classificazione ed in particolare agli alberi di decisione senza considerare la critica più comune rivolta all'uso degli alberi di decisione nei problemi di classificazione. La critica sostiene che negli alberi di decisione l'attributo divisore viene scelto quasi sempre da un algoritmo "ingordo", che non tiene assolutamente conto dell'influenza che la scelta di un attributo divisore potrebbe avere sui futuri divisori. In altre parole, la decisione dell'attributo (o campo) divisore avviene ad ogni nodo dell'albero, in un preciso momento durante l'esecuzione dell'algoritmo, e non è mai più riconsiderata in seguito[7]. Conseguenza

[7]Quando l'algoritmo sceglie, in un determinato nodo, un opportuno campo divisore non è assolutamente detto che quello sia (in assoluto) il miglior campo divisore per quel nodo; sicuramente è il miglior campo divisore per quel nodo in quel preciso istante ed alla luce delle informazioni che l'algoritmo possiede in quell'istante; il problema è che, alla luce di nuove informazioni, l'algoritmo non andrà a modificare i campi divisori dei nodi precedenti.

di ciò è che tutti i campi divisori vengono scelti sequenzialmente e ogni campo divisore è di fatto dipendente dai predecedenti. Ciò implica che tutti i futuri campi divisori sono dipendenti dal nodo radice dell'albero, a tal punto che una modifica del campo divisore del nodo radice potrebbe portare alla costruzione di un albero completamente differente. Questa critica, di per sè, potrebbe portare ad una forte limitazione nell'uso di tali strumenti, tuttavia le informazioni che spesso si ricavano dalla costruzione di alberi di decisione e che abbiamo accennato in questa parte della tesi fanno parzialmente dimenticare la correttezza di tale critica.