# Alberi di Decisione (2)

## Corso di AA, anno 2017/18, Padova


Fabio Aiolli

25 Ottobre 2017

# Apprendimento di alberi di decisione: Bias induttivo

Come altri algoritmi induttivi, anche ID3 può essere visto come ricerca di una ipotesi che 'fitta' i dati in uno spazio delle ipotesi.

- Lo spazio delle ipotesi è l'insieme dei possibili alberi di decisione. Nota che le regole corrispondenti sono un sotto-insieme delle possibili DNF definite sugli attributi delle istance
- La ricerca è di tipo *hill climbing*. Si inizia dall'albero vuoto e si procede con alberi via via più elaborati, fermandosi non appena ne troviamo uno consistente con gli esempi. Percui, applicando il principio del guadagno informativo, ne segue che alberi con un numero minore di nodi sono preferiti rispetto ad alberi più grandi.

# Attributi continui

Fino ad ora abbiamo considerato solo attributi a valori discreti.
Cosa succede se uno o più attributi contengono valori continui?

| Day | Outlook | Temperature | Humidity | Wind | PlayTennis |
|---|---|---|---|---|---|
| D1 | Sunny | 90 (Hot) | High | Weak | No |
| D2 | Sunny | 80 (Hot) | High | Strong | No |
| D3 | Overcast | 72 (Hot) | High | Weak | Yes |
| D4 | Rain | 60 (Mild) | High | Weak | Yes |
| D5 | Rain | 40 (Cool) | Normal | Weak | Yes |
| D6 | Rain | 48 (Cool) | Normal | Strong | No |
| D7 | Overcast | 40 (Cool) | Normal | Strong | Yes |
| D8 | Sunny | 60 (Mild) | High | Weak | Yes |
| D9 | Sunny | 40 (Cool) | Normal | Weak | Yes |
| ... | ... | ... | ... | ... | ... |

# Attributi continui

Soluzione: a partire dall'attributo $A$ continuo, creiamo dinamicamente l'attributo (o pseudo-attributo) booleano

$$A_c = \begin{cases} \text{true} & \text{se } A < c \\ \text{false} & \text{altrimenti} \end{cases}$$

Come selezionare il valore "giusto" per $c$?
Una possibilità è scegliere il valore $c$ che corrisponde al guadagno informativo massimo!
È stato dimostrato che il valore ottimale (che massimizza il guadagno) si localizza nel valore di mezzo tra due valori con target diverso.

| **esempi** | {D5,D7,D9} | | {D6} | | {D4,D8} | {D3} | | {D2} | {D1} |
|---|---|---|---|---|---|---|---|---|---|
| **valore** | 40 | | 48 | | 60 | 72 | | 80 | 90 |
| **target** | yes | | no | | yes | yes | | no | no |
| | | ↑ | | ↑ | | | ↑ | | |
| **valore taglio** | | (44) | | (54) | | | (76) | | |

# Attributi continui

Quindi basta calcolare il guadagno per:

$$c = 44 \quad \underbrace{\{D5, D7, D9\}}_{Temp<44} \quad \underbrace{\{D1, D2, D3, D4, D6, D8\}}_{Temp\geq 44} \qquad Guadagno \simeq \boxed{0.379}$$

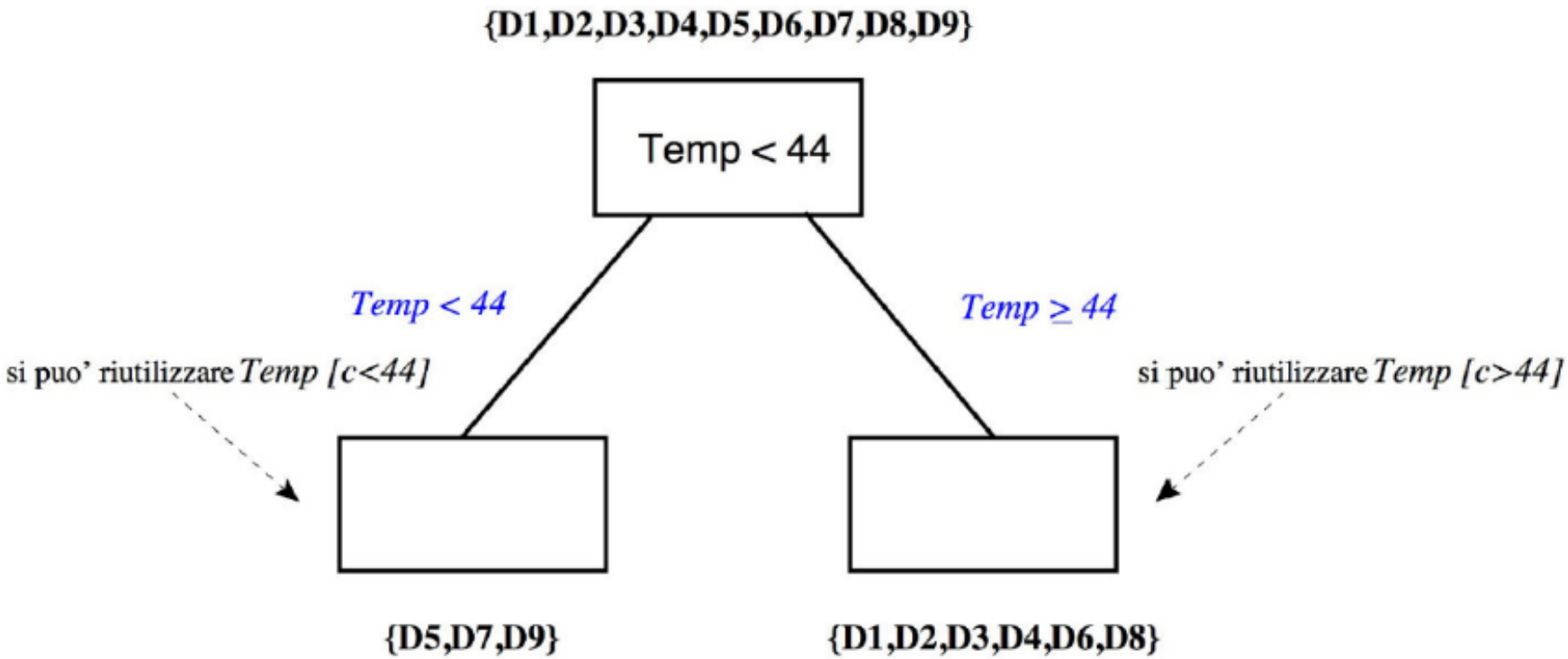
$$c = 54 \quad \underbrace{\{D5, D6, D7, D9\}}_{Temp<54} \quad \underbrace{\{D1, D2, D3, D4, D8\}}_{Temp\geq 54} \qquad Guadagno \simeq 0.091$$

$$c = 76 \quad \underbrace{\{D3, D4, D5, D6, D7, D9\}}_{Temp<76} \quad \underbrace{\{D1, D2, D8\}}_{Temp\geq 76} \qquad Guadagno \simeq 0.093$$

Viene selezionato il taglio $c = 44$ in quanto corrisponde al taglio di guadagno massimo!

# Attributi continui

Attenzione! Nel caso di attributi continui, a differenza di quelli discreti, lo stesso attributo potrà essere riutilizzato sullo stesso cammino (in tal caso ovviamente il valore del taglio ottimo cambierà).



# Attributi con valori mancanti

Problema: Nelle applicazioni pratiche può succedere che, per alcuni esempi, alcuni attributi non abbiano un valore assegnato (valore mancante).

Esempio: Diagnosi medica

- Per il paziente 38 manca il risultato della TAC;
- Per il paziente 45 mancano gli esami del sangue e la radiografia.

Possibili soluzioni: Sia dato un insieme di esempi $\hat{Tr}$, quando per un esempio $(x, y)$ manca il valore di un attributo $A$:

a) Utilizzare per A il valore più frequente in $\hat{Tr}$;

b) Come in a) ma considerando solo esempi con target $y$;

c) Considerando tutti i valori $v_i \in V(A)$, e la loro probabilità di occorrere $p(v_i|\hat{Tr})$, stimata su $\hat{Tr}$. Quindi, sostituire l'esempio $(x, y)$ con $|V(A)|$ "istanze frazionarie", una per ogni valore $v_i$ e peso uguale a $p(v_i|\hat{Tr})$.

# Attributi con valori mancanti (esempio soluzione a)

| Day | Outlook | Temperature | Humidity | Wind | PlayTennis |
|---|---|---|---|---|---|
| D1 | Sunny | Hot | High | Weak | No |
| D2 | Sunny | Hot | High | Strong | No |
| D3 | Overcast | Hot | High | Weak | Yes |
| D4 | Rain | Mild | High | Weak | Yes |
| D5 | - | Cool | Normal | Weak | Yes |
| D6 | Rain | Cool | Normal | Strong | No |
| D7 | Overcast | Cool | Normal | Strong | Yes |
| D8 | Sunny | Mild | High | Weak | No |
| D9 | Sunny | Cool | Normal | Weak | Yes |

Consideriamo l'attributo `Outlook` e l'esempio `D5`:

$$\texttt{D5} \rightarrow (\texttt{[Sunny, Cool, Normal, Weak]},\texttt{Yes})$$

poiché $P(\texttt{Sunny}|\hat{Tr}) = \frac{1}{2}$, $P(\texttt{Overcast}|\hat{Tr}) = P(\texttt{Rain}|\hat{Tr}) = \frac{1}{4}$

# Attributi con valori mancanti (esempio soluzione b)

| Day | Outlook | Temperature | Humidity | Wind | PlayTennis |
|---|---|---|---|---|---|
| D1 | Sunny | Hot | High | Weak | No |
| D2 | Sunny | Hot | High | Strong | No |
| D3 | Overcast | Hot | High | Weak | Yes |
| D4 | Rain | Mild | High | Weak | Yes |
| D5 | - | Cool | Normal | Weak | Yes |
| D6 | Rain | Cool | Normal | Strong | No |
| D7 | Overcast | Cool | Normal | Strong | Yes |
| D8 | Sunny | Mild | High | Weak | No |
| D9 | Sunny | Cool | Normal | Weak | Yes |

Consideriamo l'attributo `Outlook` e l'esempio `D5`:

$$\texttt{D5} \rightarrow (\texttt{[Overcast, Cool, Normal, Weak]},\texttt{Yes})$$

poiché $P(\texttt{O}|y = Yes, \hat{Tr}) = \frac{1}{2}$, $P(\texttt{S}|y = Yes, \hat{Tr}) = P(\texttt{R}|y = Yes, \hat{Tr}) = \frac{1}{4}$

# Attributi con valori mancanti (esempio soluzione b)

| Day | Outlook | Temperature | Humidity | Wind | PlayTennis |
|---|---|---|---|---|---|
| D1 | Sunny | Hot | High | Weak | No |
| D2 | Sunny | Hot | High | Strong | No |
| D3 | Overcast | Hot | High | Weak | Yes |
| D4 | Rain | Mild | High | Weak | Yes |
| D5 | - | Cool | Normal | Weak | Yes |
| D6 | Rain | Cool | Normal | Strong | No |
| D7 | Overcast | Cool | Normal | Strong | Yes |
| D8 | Sunny | Mild | High | Weak | No |
| D9 | Sunny | Cool | Normal | Weak | Yes |

Consideriamo l'attributo `Outlook` e l'esempio `D5`:

$$\texttt{D5} \rightarrow \begin{cases} \texttt{D5}_\texttt{S} = \texttt{[Sunny, Cool, Normal, Weak]} & p_S = 1/2 \\ \texttt{D5}_\texttt{O} = \texttt{[Overcast, Cool, Normal, Weak]} & p_O = 1/4 \\ \texttt{D5}_\texttt{R} = \texttt{[Rain, Cool, Normal, Weak]} & p_R = 1/4 \end{cases}$$

# Attributi con valori mancanti (esempio soluzione c)

| Day | Outlook | Temperature | Humidity | Wind | PlayTennis |
|---|---|---|---|---|---|
| D1 | Sunny | Hot | High | Weak | No |
| D2 | Sunny | Hot | High | Strong | No |
| D3 | Overcast | Hot | High | Weak | Yes |
| D4 | Rain | Mild | High | Weak | Yes |
| D5' | - | Cool | Normal | - | Yes |
| D6 | Rain | Cool | Normal | Strong | No |
| D7 | Overcast | Cool | Normal | Strong | Yes |
| D8 | Sunny | Mild | High | Weak | No |
| D9 | Sunny | Cool | Normal | Weak | Yes |

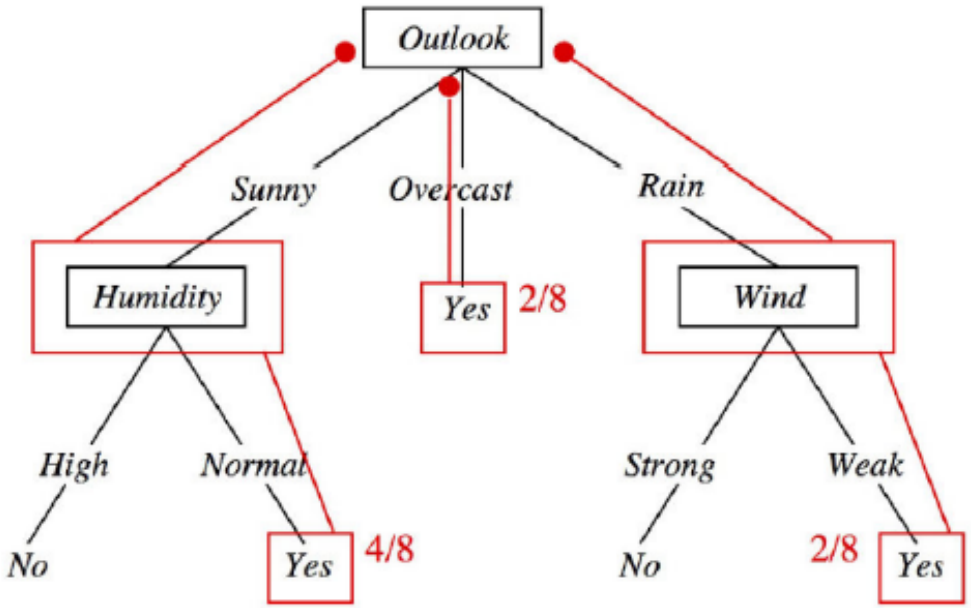
Consideriamo l'attributo `Outlook` e l'esempio `D5'`:

Se più attributi hanno valori mancanti, si continuano a frazionare gli esempi già frazionati ed il peso associato all'esempio è dato dal prodotto dei pesi ottenuti considerando un solo attributo a turno

# Attributi con valori mancanti (esempio soluzione c)

Immaginiamo di dover classificare una nuova istanza con attributi mancanti. In questo caso, per ogni esempio frazionario si esegue la classificazione e per ogni possibile etichetta si sommano i pesi degli esempi che raggiungono foglie corrispondenti a quella etichetta.

Esempio: Classificazione di `D5` con soluzione c




$P(\texttt{Yes}) = 4/8 + 2/8 + 2/8 = 1 \qquad P(\texttt{No}) = 0$

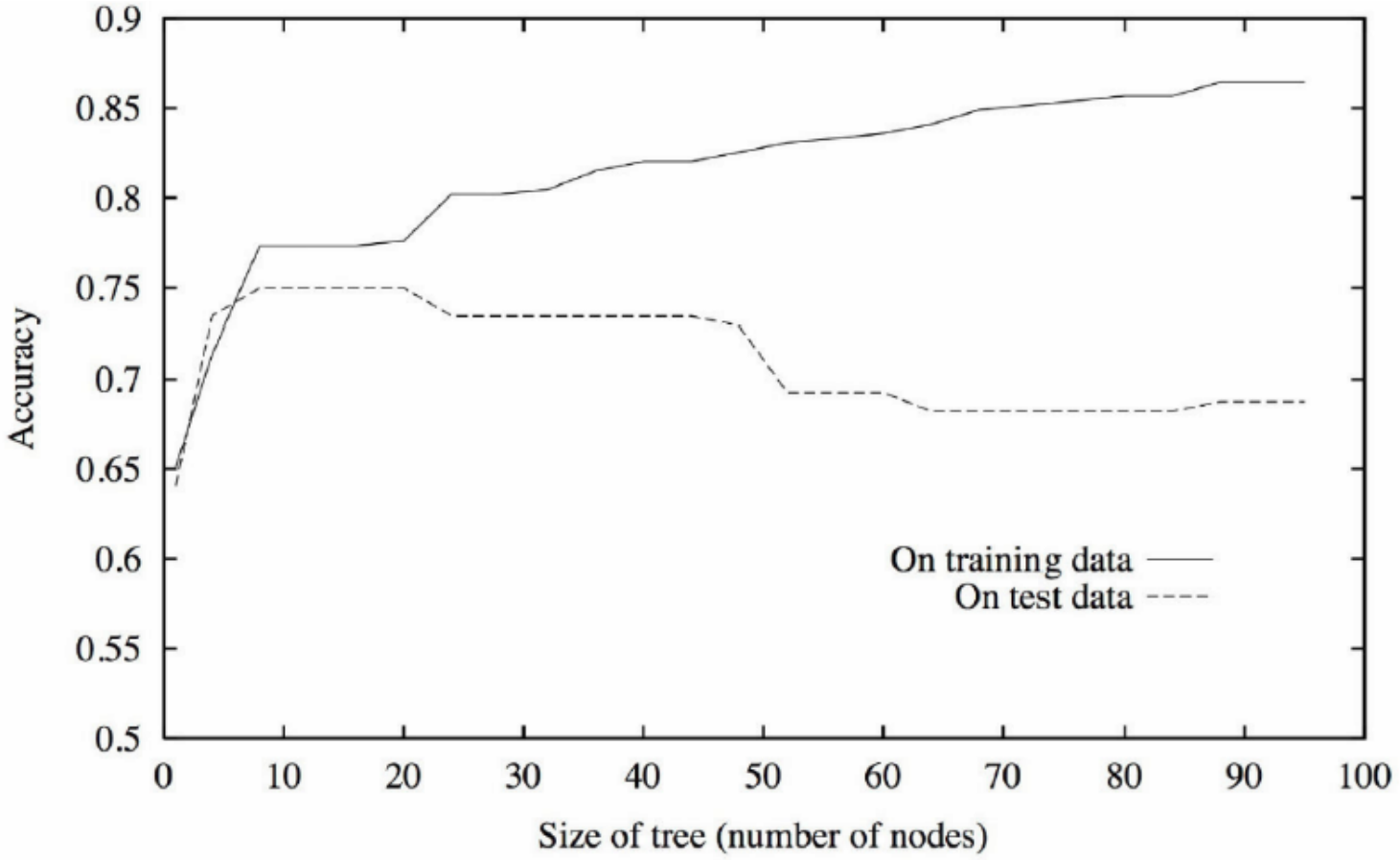
# Attributi con valori mancanti (esempio soluzione c)

## Apprendimento con soluzione c

- Modificare il concetto di cardinalità di un insieme in modo da considerare i pesi frazionari
- Modificare la definizione di Guadagno Informativo conseguentemente

# Apprendimento di alberi di decisione: overfitting

## Problema: Overfitting




## (Parziale) Soluzione: Potatura (o pruning)

# Reduced Error Pruning

- Dividere `Tr` in `Tr'` e `Va` (validation set), `Tr = Tr'` $\cup$ `Va`
- Ripetere fino a quando le prestazioni peggiorano:
  - Per ogni nodo (interno) valutare l'accuratezza su `Va` nel caso il sotto-albero radicato nel nodo venga potato
  - Effettuare la potatura che corrisponde alla miglior performance sul validation set.

In questo caso, effettuare la potatura significa sostituire, al posto di un sotto-albero radicato in un nodo, un nodo foglia etichettato con la etichetta più frequente negli esempi associati a quel nodo.

# Rule-Post Pruning

L'idea di base è trasformare l'albero di decisione in un insieme di regole, e poi effettuare la potatura delle regole.

- Si genera una regola $R_i$ per ogni cammino $path(r, f_i)$ dalla radice `r` alla foglia i-esima $f_i$. $R_i$ sarà della forma:

$$IF(Att_{i_1} = v_{i_1}) \cap (Att_{i_2} = v_{i_2}) \cap \cdots \cap (Att_{i_k} = v_{i_k}) THEN\ label_{f_i}$$

- Si effettua la potatura indipendentemente su ogni $R_i$:
  - Si stimano le prestazioni ottenute usando SOLO $R_i$ come classificatore
  - Si rimuovono le precondizioni (una o più) che conducono ad un aumento della stima delle prestazioni sul validation set, utilizzando un approccio greedy
- Si ordinano le $R_i$ potate in ordine decrescente di prestazione; eventualmente si aggiunge una regola di default che restituisce la classe più frequente

# Classificazione con Post-Rule Pruning

La classificazione avviene seguendo l'ordine stabilito dalle regole:

- La prima regola la cui precondizione è soddisfatta dalla istanza viene usata per generare la classificazione
- Se nessuna regola ha le precondizioni soddisfatte, si utilizza la regola di default per classificare l'istanza (si ritorna la classe più frequente nell'insieme di training)

# Considerazioni su Post-Rule Pruning

Alcune considerazioni su Post-Rule Pruning:

- La stima delle prestazioni necessaria per effettuare la potatura può essere fatta sia usando un insieme di validazione che utilizzando un test statistico sui dati di apprendimento
- La trasformazione Albero $\rightarrow$ Regole permette di generare regole dove si possono considerare contesti per un nodo che non necessariamente contengono i nodi antecedenti (e in particolare la radice): stiamo cambiando lo spazio delle ipotesi in effetti!
- Rispetto agli alberi, di solito le regole sono più semplici da comprendere per un umano
- Di solito, Post-Rule Pruning riesce a migliorare le performance rispetto all'albero su cui è applicato e si comporta meglio del Reduced-Error Pruning