

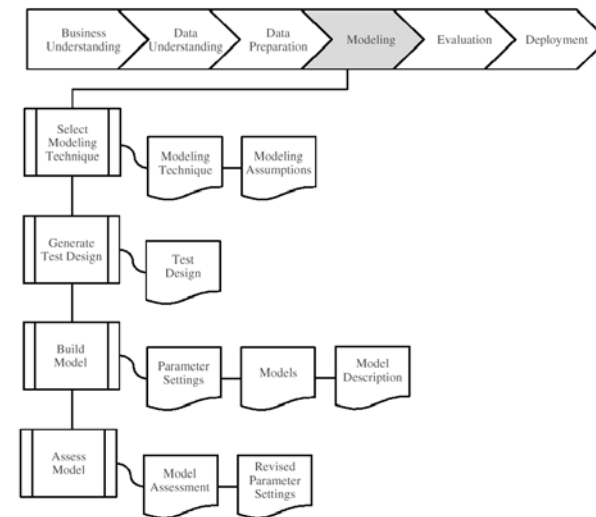
Kapitel VI + VII

Modellierung/ Data-Mining-Verfahren

© Institut AIFB, 2002.
Alle Rechte vorbehalten. Nachdruck oder photomechanische Wiedergabe nur mit Genehmigung des Verfassers.
Zuwendungen unterliegen den strafrechtlichen Bedingungen des Urheberrechtsgesetzes.

1

Kapitel VI + VII: Data-Mining Verfahren



Vorlesung Knowledge Discovery

2

Chapter VI: Supervised Learning Algorithms

Chapter VI

Supervised Learning Algorithms

© Institut AIFB, 2001.
Alle Rechte vorbehalten. Nachdruck oder photomechanische Wiedergabe nur mit Genehmigung des Verfassers.
Zuwendungen unterliegen den strafrechtlichen Bedingungen des Urheberrechtsgesetzes.

3

Chapter VI.1: Decision Trees

Chapter VI.1

Decision Trees

© Institut AIFB, 2001.
Alle Rechte vorbehalten. Nachdruck oder photomechanische Wiedergabe nur mit Genehmigung des Verfassers.
Zuwendungen unterliegen den strafrechtlichen Bedingungen des Urheberrechtsgesetzes.

4

Kapitel VI.1: Konstruktion von Entscheidungsbäumen

VI.1 Konstruktion von Entscheidungsbäumen (Langley 1996, Quinlan 1993)

VI.1.1 Grundlagen

- *Entscheidungsbäume* sind eine spezielle Form von *Konzepthierarchien*

- *Grundidee*:

- aus gegebenen *Trainingsbeispielen* wird Entscheidungsbaum aufgebaut
- Entscheidungsbaum liefert *intensionale Beschreibung* der vorgegebenen Klassen
- Entscheidungsbaum ordnet *neue* Beispiele mit gewisser *Fehlerrate* einer der vorgegebenen Klassen zu

Kapitel VI.1: Konstruktion von Entscheidungsbäumen

Generelle *Voraussetzungen*:

- Beispiele werden in Form von *Attribut-Wert-Paaren* beschrieben
 - Attribut hat *Name* und *Wertebereich*
 - Wertebereich ist nominal, diskret oder kontinuierlich
 - jedes Beispiel wird durch *dieselben* Attribute beschrieben (vgl. relationale Datenbanken)
- Menge von *Klassen* (Konzepten) ist *vordefiniert*; i.a. ist die Anzahl der Klassen sehr viel kleiner als die Zahl der Trainingsbeispiele

Kapitel VI.1: Konstruktion von Entscheidungsbäumen

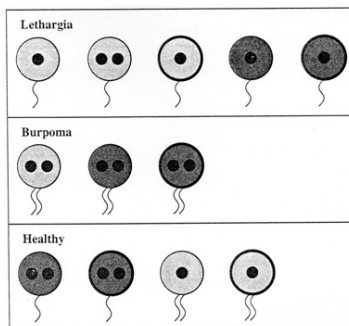


Abbildung VI.1-1: Training instances from a cell domain involving four attributes and three distinct classes. (Langley 1996)

Attributes:

number of nuclei	(values: 1,2)
number of tails	(values: 1,2)
color	(values: light, dark)
wall	(values: thin, thick)

Kapitel VI.1: Konstruktion von Entscheidungsbäumen

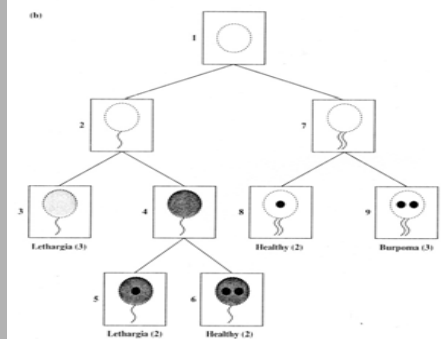


Abbildung VI.1-2: A decision tree generated by the DDT algorithm for the training instances from Abbildung VI.1-1. Each terminal node specifies an associated class name and the number of training cases covered. Numbers next to the notes indicate the order in which the algorithm generated them. The notation (a), which associates attributes with nodes and values with links, is equivalent to (b), which shows the instances that each node covers. (Langley 1996)

Kapitel VI.1: Konstruktion von Entscheidungsbäumen

- ° Von jedem Trainingsbeispiel ist (idealerweise) die *Klassenzugehörigkeit bekannt*
 - überwachtes Lernen (*supervised learning*)
 - ein Attribut spezifiziert jeweils die Klassenzugehörigkeit (vgl. auch Holländer- Klassifizierung)
- ° Klassen müssen durch Entscheidungsbäume oder Produktionsregeln beschreibbar sein
 - typischerweise Beschreibungen der *Form*

$$[(att_{i_1} = value_{i_1}) \wedge \dots \wedge (att_{i_n} = value_{i_n})]$$

Beispiel: Klasse Healthy:

$$[(number_of_tails = 2) \wedge (number_of_nuclei = 1)] \vee [(number_of_tails = 1) \wedge (color = dark) \wedge (number_of_nuclei = 2)]$$

Kapitel VI.1: Konstruktion von Entscheidungsbäumen

Struktur eines *univariaten Entscheidungsbaums*:

- Blattknoten werden mit jeweils *einer Klasse* benannt, alle anderen Knoten mit jeweils einem Test auf *ein einziges Attribut*
- wenn das Attribut eines Knotens *k* verschiedene Werte hat (bzw. in *k* Wertebereiche aufgeteilt wird), dann hat der Knoten *k* Sohnknoten
- Beispielmenge wird an dem Knoten in *k* disjunkte Teilmengen partitioniert in Abhängigkeit vom jeweiligen Wert des Attributs
- Die *Extension* eines Knotens ist immer eine Obermenge der Extension aller Sohnknoten

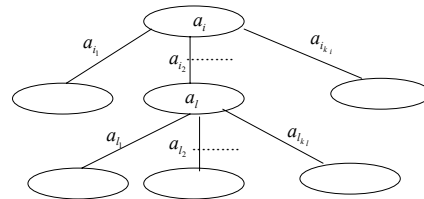
Kapitel VI.1: Konstruktion von Entscheidungsbäumen

- verbinde jeden Knoten mit all seinen Sohnknoten und beschrifte die *Kanten* mit ihren zugehörigen Attributwerten
- *neue Fälle* durchlaufen für ihre Klassifikation den Entscheidungsbaum von der Wurzel ausgehend bis zu einem Blattknoten

Graphisch:

- *m* Attribute a_1, a_2, \dots, a_m
- Wertebereich

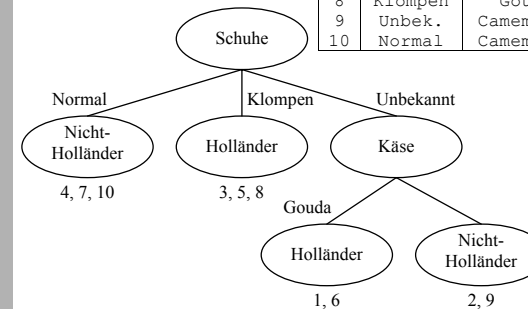
von Attribut a_i : $\{a_{i_1}, \dots, a_{i_k}\}$
 von Attribut a_j : $\{a_{j_1}, \dots, a_{j_l}\}$



Kapitel VI.1: Konstruktion von Entscheidungsbäumen

Wie man in Holland Leute erkennt (ohne Gewähr) ...

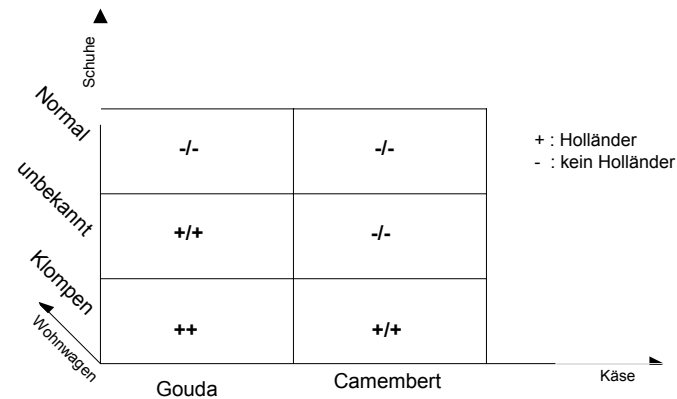
	Schuhe	Lieblings-Käse	Wohnwagen	Holländisch
1	Unbek.	Gouda	Ja	+
2	Unbek.	Camembert	Ja	-
3	Klumpen	Gouda	Ja	+
4	Normal	Gouda	Nein	-
5	Klumpen	Camembert	Ja	+
6	Unbek.	Gouda	Nein	+
7	Normal	Gouda	Ja	-
8	Klumpen	Gouda	Nein	+
9	Unbek.	Camembert	Nein	-
10	Normal	Camembert	Ja	-



Ein Entscheidungsbaum zur Klassifikation

Kapitel VI.1: Konstruktion von Entscheidungsbäumen

Grafische Veranschaulichung der Klassifikationsaufgabe:



Kapitel VI.1: Konstruktion von Entscheidungsbäumen

- Beispielalgorithmus für Aufbau eines univariaten Entscheidungsbaums:
DDT - Algorithmus: Divisive Induction of
 Univariate Decision Trees

- DDT ist *nicht-inkrementell*:
 - alle Trainingsbeispiele müssen verfügbar sein
- DDT ist *Greedy-Algorithmus* (greedy=gierig):
 - lokal optimale Entscheidungen werden getroffen
 - getroffene Entscheidungen können nicht mehr zurückgenommen werden

Kapitel VI.1: Konstruktion von Entscheidungsbäumen

DDT Algorithm: Divisive Induction of Univariate Decision Trees

Inputs: The current node N of the decision tree.
 A set of classified training instances ISET.
 A set of symbolic attributes and their values ASET.

Output: A univariate decision tree.

Top-level call: DDT(root, ASET, ISET).

Kapitel VI.1: Konstruktion von Entscheidungsbäumen

Procedure DDT(N, ASET, ISET)

```

If the training set ISET is empty,
Then label terminal node N as DEFAULT.
Else IF all instances ISET are equal with respect to all attributes in ASET // neuer Blattknoten
    // this is especially the case if ASET is empty // gefunden
Then label terminal node N with the class name.
Else for each attribute A in ASET,
    Evaluate A according to its ability to // wähle „bestes“
    discriminate the classes in ISET. // Attribut B
    Select attribute B with the best evaluation score. //
    For each value V of B,
        Create a new child C of node N. // expandiere
        Label the edge from N to C with V. // aktuellen
        Let JSET be the ISET instances having value V on B. // Knoten
        Let KSET be ASET \ {B}.
        DDT(C, KSET, JSET).
  
```

Kapitel VI.1: Konstruktion von Entscheidungsbäumen

Beispiel "Spielen im Freien":

- 4 Attribute
- 2 Klassen: Play
Don't Play

Outlook	Temp (F)	Humidity (%)	Windy?	Class
sunny	75	70	true	Play
sunny	80	90	true	Don't Play
sunny	85	85	false	Don't Play
sunny	72	95	false	Don't Play
sunny	69	70	false	Play
overcast	72	90	true	Play
overcast	83	78	false	Play
overcast	64	65	true	Play
overcast	81	75	false	Play
rain	71	80	true	Don't Play
rain	65	70	true	Don't Play
rain	75	80	false	Play
rain	68	80	false	Play
rain	70	96	false	Play

Abbildung VI.1-3: A small training set (Quinlan 1993)

Kapitel VI.1: Konstruktion von Entscheidungsbäumen

Partition of cases:

outlook = sunny:				
humidity ≤ 75:				
Outlook	Temp (°F)	Humidity (%)	Windy?	Decision
sunny	75	70	true	Play
sunny	69	70	false	Play
humidity > 75:				
Outlook	Temp (°F)	Humidity (%)	Windy?	Decision
sunny	80	90	true	Don't Play
sunny	85	85	false	Don't Play
sunny	72	95	false	Don't Play
outlook = overcast:				
Outlook	Temp (°F)	Humidity (%)	Windy?	Decision
overcast	72	90	true	Play
overcast	83	78	false	Play
overcast	64	65	true	Play
overcast	81	75	false	Play
outlook = rain:				
windy = true:				
Outlook	Temp (°F)	Humidity (%)	Windy?	Decision
rain	71	80	true	Don't Play
rain	65	70	true	Don't Play
windy = false:				
Outlook	Temp (°F)	Humidity (%)	Windy?	Decision
rain	75	80	false	Play
rain	68	80	false	Play
rain	70	96	false	Play

Corresponding decision tree:

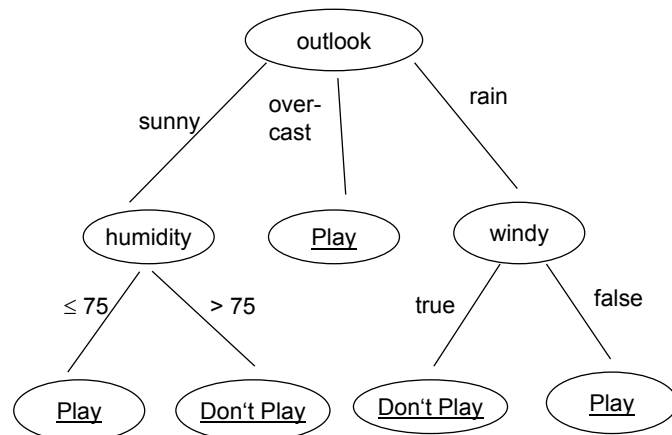
```

outlook = sunny:
| humidity ≤ 75: Play
| humidity > 75: Don't Play
outlook = overcast: Play
outlook = rain:
| windy = true: Don't Play
| windy = false: Play
    
```

Abbildung VI.1-4: Final partition of cases and corresponding decision tree (Quinlan 1993)

Kapitel VI.1: Konstruktion von Entscheidungsbäumen

Graphische Darstellung des Entscheidungsbaumes zum Beispiel:



Kapitel VI.1: Konstruktion von Entscheidungsbäumen

VI.1.2 C4.5 - Algorithmus

(Quinlan 1993)

- weitverbreiteter Algorithmus zum Aufbau von Entscheidungsbäumen (oder entsprechenden Produktionsregeln)
- Spezielle Variante des DDT-Algorithmus:
 - Verwendung einer speziellen *Bewertungsfunktion* für Auswahl des besten Attributs: *gain ratio*
 - beinhaltet zusätzlich Verfahren zur nachträglichen *Vereinfachung* des erzeugten Entscheidungsbaums: *pruning*
 - aus Entscheidungsbaum können entsprechende *Produktionsregeln* erzeugt werden
⇒ alternative lesbarere Darstellung
 - *Windowing* - Technik für Handhabung einer großen Anzahl von Beispielen

Kapitel VI.1: Konstruktion von Entscheidungsbäumen

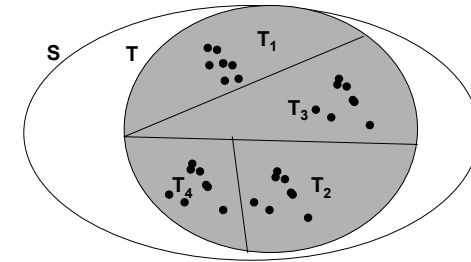
a) Bewertungsfunktion für Attributauswahl

- für gegebene Menge von Trainingsbeispielen ist Anzahl der möglichen Entscheidungsbäume i.a. *sehr groß*
- Daher ist Generierung *aller* Entscheidungsbäume und dann Auswahl des Besten (exhaustive search) *nicht möglich*
- Daher wird in jedem Expansionsschritt das *vielversprechendste* Attribut ausgewählt (Greedy Algorithmus): verwende das Attribut, das *am meisten Information* liefert (im Sinne der Informationstheorie)
- C4.5 verwendet *gain ratio* als Kriterium. Es ist eine Abwandlung des im folgenden beschriebenen *gain criterion*.

Kapitel VI.1: Konstruktion von Entscheidungsbäumen

gain criterion

- Bezeichnungen: S: Gesamtbeispielmenge
- T: Menge der Trainingsbeispiele ($T \subset S$)
- T_1, T_2, \dots, T_n : Partition von T



Kapitel VI.1: Konstruktion von Entscheidungsbäumen

- Informationsgehalt einer Botschaft b_i :

hat b_i die Wahrscheinlichkeit p_i ,

so ist der Informationsgehalt von $b_i \equiv -\log_2(p_i)$ bits

Beispiel: 8 gleichwahrscheinliche Botschaften b_1, \dots, b_8 :

jedes b_i hat den Informationsgehalt

$$-\log_2\left(\frac{1}{8}\right) \text{ bits} = 3 \text{ bits}$$

Kapitel VI.1: Konstruktion von Entscheidungsbäumen

- Botschaft, die mitteilt, dass beliebig ausgewähltes Beispiel aus Menge T in Klasse c_i liegt, hat den Informationsgehalt

$$-\log_2\left(\frac{|C_i \cap T|}{|T|}\right) \text{ bits}$$

- Erwartungswert für den Informationsgehalt dieser Botschaften (bei k Klassen C_1, \dots, C_k):

$$\text{info}(T) = - \sum_{j=1}^k \frac{|C_j \cap T|}{|T|} * \log_2\left(\frac{|C_j \cap T|}{|T|}\right) \text{ bits}$$

(Entropie von T)

z.B.: k = 2

C_1 : play

C_2 : don't play

Kapitel VI.1: Konstruktion von Entscheidungsbäumen

- Sei X ein Test, der T in T_1, \dots, T_n partitioniert.

Dann ist der Erwartungswert für den Informationsgehalt einer Botschaft, die mitteilt, dass ein beliebig ausgewähltes Beispiel aus T_i in der Klasse C_j liegt (bei k Klassen C_1, \dots, C_k):

$$info(T_i) = - \sum_{j=1}^k \frac{|C_j \cap T_i|}{|T_i|} * \log_2 \left(\frac{|C_j \cap T_i|}{|T_i|} \right) \text{ bits}$$

Damit ist der Erwartungswert über alle T_1, \dots, T_n :

$$info_X(T) = \sum_{i=1}^n \frac{|T_i|}{|T|} * info(T_i)$$

z.B.: k = 2

C_1 : play

C_2 : don't play

z.B.: X: outlook; n = 3

T_1 : sunny

T_2 : overcast

T_3 : rain

Kapitel VI.1: Konstruktion von Entscheidungsbäumen

- *gain criterion* :

$$gain(x) = info(T) - info_X(T)$$

Maß für den Informationsgewinn durch Partitionierung von T durch Test X:

- wähle Test X, so dass gain (X) maximiert wird, d.h. möglichst großer Informationsgewinn durch aussagekräftiges Attribut X
- mit info(T) fix wähle ein Attribut X mit möglichst kleinem $info_X(T)$, d.h. Erwartungswert für noch benötigte Informationen zur Klassifikation ist möglichst klein

Kapitel VI.1: Konstruktion von Entscheidungsbäumen

- *gain criterion* :

$$gain(x) = info(T) - info_X(T)$$

Bem.: Dies ist verwandt zum gemittelten Information Gain in III.3.5(iv):

$$\bar{I}_{gain}(c, A) = \frac{1}{s} \sum_{i=1}^s I_{gain}(c, A_i) = \frac{1}{s} \sum_{i=1}^s (H(c) - H(c|A_i)) = H(c) - \frac{1}{s} \sum_{i=1}^s H(c|A_i)$$

nur dass dort nur die Vorhersage einer Klasse C und nicht die Vorhersage in die Einteilung der Klassen C_1, \dots, C_k betrachtet wurde.

Kapitel VI.1: Konstruktion von Entscheidungsbäumen

k = 2

C_1 : play

C_2 : don't play

Fortführung des Beispiels "Spielen im Freien" :

2 Klassen : Play, Don't Play

Klasse Play: 9 Fälle

Klasse Don't Play : 5 Fälle

damit:

$$\begin{aligned} info(T) &= - \sum_{j=1}^2 \frac{|C_j \cap T|}{|T|} * \log_2 \left(\frac{|C_j \cap T|}{|T|} \right) \text{ bits} \\ &= - \left(\frac{9}{14} * \log_2 \left(\frac{9}{14} \right) + \frac{5}{14} * \log_2 \left(\frac{5}{14} \right) \right) \text{ bits} \\ &= 0.940 \text{ bits} \end{aligned}$$

(durchschnittlich benötigte Information zur Identifizierung der Klassenzugehörigkeit eines Beispiels aus T)

Kapitel VI.1: Konstruktion von Entscheidungsbäumen

k = 2

C₁: play

C₂: don't play

- Attribut 'outlook' erzeugt 3 Teilmengen:

$$T_1: \text{outlook}=\text{sunny} \rightarrow |T_1|=5$$

$$T_2: \text{outlook}=\text{overcast} \rightarrow |T_2|=4$$

$$T_3: \text{outlook}=\text{rain} \rightarrow |T_3|=5$$

Und liefert nach Definition damit:

$$\begin{aligned} \text{info}_{\text{outlook}}(T) &= \sum_{i=1}^3 \frac{|T_i|}{|T|} * \text{info}(T_i) \\ &= \frac{5}{14} * \left(-\frac{2}{5} * \log_2\left(\frac{2}{5}\right) - \frac{3}{5} * \log_2\left(\frac{3}{5}\right) \right) \\ &\quad + \frac{4}{14} * \left(-\frac{4}{4} * \log_2\left(\frac{4}{4}\right) - \frac{0}{4} * \log_2\left(\frac{0}{4}\right) \right) \\ &\quad + \frac{5}{14} * \left(-\frac{3}{5} * \log_2\left(\frac{3}{5}\right) - \frac{2}{5} * \log_2\left(\frac{2}{5}\right) \right) \\ &= 0.694 \text{ bits} \end{aligned}$$

X: outlook; n = 3

T₁: sunny

T₂: overcast

T₃: rain

Kapitel VI.1: Konstruktion von Entscheidungsbäumen

k = 2

C₁: play

C₂: don't play

Daraus folgt:

$$\begin{aligned} \text{gain}(\text{outlook}) &= \text{info}(T) - \text{info}_{\text{outlook}}(T) \\ &= (0.940 - 0.694) \text{ bits} = 0.246 \text{ bits} \end{aligned}$$

Vergleich mit dem Attribut 'windy':

windy erzeugt 2 Teilmengen:

$$T_1: \text{windy} = \text{true} \rightarrow |T_1|=6$$

$$T_2: \text{windy} = \text{false} \rightarrow |T_2|=8$$

$$\circ \text{info}_{\text{windy}}(T) = \sum_{i=1}^2 \frac{|T_i|}{|T|} * \text{info}(T_i) = \dots = 0.892 \text{ bits}$$

◦ damit

$$\text{gain}(\text{windy}) = (0.940 - 0.892) \text{ bits} = 0.048 \text{ bits}$$

X: windy; n = 2

T₁: true

T₂: false

Also wird 'outlook' als stärker diskriminierendes Merkmal weiter oben im Baum verwandt als 'windy'.

Kapitel VI.1: Konstruktion von Entscheidungsbäumen

Das gain criterion bevorzugt Tests mit vielen verschiedenen Testwerten

(Attribut mit großem Wertebereich), wie das folgende Beispiel zeigt.

⇒ Bildung von Teilmengen mit wenigen Fällen

⇒ im Extremfall einelementige Teilmengen

$$\Rightarrow \text{info}_x(T) = 0$$

Aber für Klassifizierungszwecke ist die Bildung derartiger Teilmengen unerwünscht (Overfitting)! Deswegen wird (nach dem Beispiel) die Variante *gain ratio* des gain criterion vorgestellt.

Kapitel VI.1: Konstruktion von Entscheidungsbäumen

Fortführung des Beispiels "Spielen im Freien" :

- füge neues Attribut "Datum" hinzu

- Attribut mit vielen verschiedenen Werten

- Attribut "Datum" liefert sehr hohen Informationsgewinn

- Datumsangabe bestimmt eindeutig Wert des Zielattributes, d.h. Klassenzugehörigkeit
- Attribut "Datum" würde als Wurzelattribut gewählt werden

- Das Attribut "Datum" ist sehr gut geeignet für die Beschreibung der Trainingsdaten, aber nicht geeignet für Klassifikation neuer, bisher unbekannter Beispiele.

Kapitel VI.1: Konstruktion von Entscheidungsbäumen

- Idee: 'normalisiere' 'gain criterion' durch
'split info': Erwartungswert für Informationsgehalt einer Botschaft,
die mitteilt, daß beliebig ausgewähltes *Beispiel in Teilmenge* T_i liegt
(T wird durch Test X in Teilmengen T_1, \dots, T_n partitioniert)

$$\text{split info}(x) = - \sum_{i=1}^n \frac{|T_i|}{|T|} * \log_2 \left(\frac{|T_i|}{|T|} \right) \text{ bits}$$

$$\text{damit: } \text{gain ratio}(x) = \frac{\text{gain}(x)}{\text{split info}(x)}$$

- da 'split info' für wenige große T_i *sehr klein* ist:
maximiere 'gain ratio' unter Nebenbedingung, dass gain
wenigstens so groß ist wie der *Durchschnittswert* von
gain über alle möglichen Tests X .

Kapitel VI.1: Konstruktion von Entscheidungsbäumen

Anwendung auf das Beispiel:

Attribut 'outlook' erzeugt 3 Teilmengen
mit 5, 4 und 5 Beispielen; damit

$$\begin{aligned} \text{split info}(\text{outlook}) &= - \left(\frac{5}{14} \right) * \log_2 \left(\frac{5}{14} \right) - \left(\frac{4}{14} \right) * \log_2 \left(\frac{4}{14} \right) \\ &\quad - \left(\frac{5}{14} \right) * \log_2 \left(\frac{5}{14} \right) = 1.577 \text{ bits} \end{aligned}$$

damit

$$\begin{aligned} \text{gain ratio}(\text{outlook}) &= \frac{\text{gain}(\text{outlook})}{\text{split info}(\text{outlook})} \\ &= \frac{0.246}{1.577} = 0.156 \end{aligned}$$

Kapitel VI.1: Konstruktion von Entscheidungsbäumen

b) Pruning

Problem der **Überspezialisierung** von Trainingsbeispielen:

I.a. ist der aus Trainingsdaten erzeugte Entscheidungsbaum zu komplex
und hat eine zu hohe Fehlerrate bei neuen Beispielen.

Def.: **Overfitting (Überspezialisierung)** (Mitchell, 1997)

Eine Klassenbeschreibung L ist in Bezug auf eine Menge von
Trainingsdaten **überspezialisiert**, wenn es eine alternative Klassen-
beschreibung L' gibt, so daß L in Bezug auf die Trainingsdaten
eine geringere Fehlerrate hat als L' , aber in Bezug auf alle
möglichen Beispiele, d.h. insbesondere in Bezug auf bisher
nicht bekannte Beispiele eine größere Fehlerrate hat als L' .

Kapitel VI.1: Konstruktion von Entscheidungsbäumen

- Da Attribute von Trainingsbeispielen zu einem gewissen Grad *zufällig*
sind und *nicht alle* für die Klassifikation *wichtigen* Informationen
enthalten, ist *Gefahr des Overfitting* immer gegeben.

- Lösung: vereinfache erzeugten Entscheidungsbaum durch Pruning
(Abschneiden).

- **Pruning speziell bei C4.5:**

Ersetze *Teilbaum*

◦ durch einen *Blattknoten*

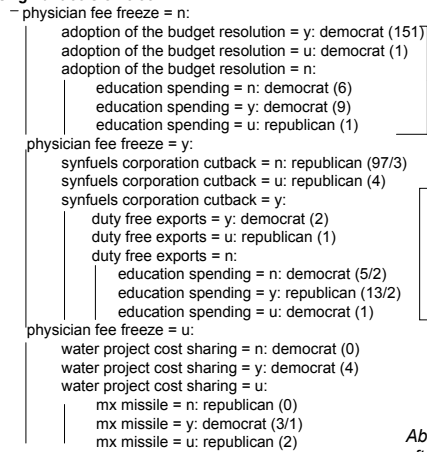
◦ bzw. durch den *besten Zweig* des Teilbaums

sofern sich dadurch die erwartete Fehlerrate für neue Beispiele
verringert

- Bem.: Pruning *erhöht* stets die Fehlerrate für
gegebene Trainingsbeispiele!

Kapitel VI.1: Konstruktion von Entscheidungsbäumen

Original decision tree:



After pruning:

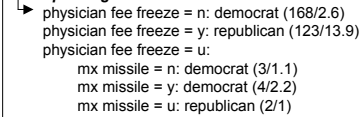


Abbildung VI.1-5: Decision tree before and after pruning (Quinlan 1993)

Kapitel VI.1: Konstruktion von Entscheidungsbäumen

Beispiel (vgl. Abbildung VI.1-5)

- 1. Teilbaum ‚physician fee freeze‘ = n :
- trennt 1 Republikaner von 167 Demokraten
- Pruning zum Blattknoten ‚democrat‘ :
- 1 Trainingsbeispiel wird jetzt falsch klassifiziert

Vorgehensweise zum Schätzen der Fehlerrate:

- ° **Alternative 1:**
- trenne geg. Menge von Beispielen in *Trainingsbeispiele* und *Testbeispiele* und verwende Testbeispiele für die Abschätzung der Fehlerrate für neue Fälle
- Problem: Die Menge der Trainingsbeispiele wird kleiner.
- *Cross validation* als Abhilfe

Kapitel VI.1: Konstruktion von Entscheidungsbäumen

k-fache Kreuzvalidierung (k-fold cross validation)

- teile vorhandene Beispiele in k gleich große Teilmengen auf
- jeweils (k-1) Teilmengen werden als Trainingsbeispiele verwendet, verbleibende Teilmenge als Testbeispiele
- führe Lernvorgang und anschließende Validierung k-fach durch, mit jeweils anderer Menge von Testbeispielen
- wähle Pruning-Schritt derart, daß Fehlerrate für Testbeispiele möglichst klein ist

Kapitel VI.1: Konstruktion von Entscheidungsbäumen

° Alternative 2:

- ° verwende als Grundlage den aus allen Trainingsbeispielen erzeugten Entscheidungsbaum
- Vorteil: Alle vorhandenen Daten werden zum Lernen genutzt. Wichtig bei wenigen Ausgangsdaten
- Nachteil: Fehlerrate kann nur abgeschätzt werden für neue Beispiele
- C4.5 verwendet *Alternative 2* mit folgender Idee:
- ° Blattknoten deckt N Trainingsbeispiele ab (Wie repräsentativ ist N?), davon sind E falsch zugeordnet
- Fehlerrate E/N ist bekannt
- Schließe auf Fehlerrate für alle (unbekannten) Beispiele, die durch Blatt abgedeckt werden, per Binomialverteilung. (Eigentlich darf dies nicht gemacht werden, da die geforderten Annahmen nicht zutreffen, in der Praxis funktioniert es aber doch häufig.)
- *Pruning* darf geschätzte Fehlerrate *nicht erhöhen*

Kapitel VI.1: Konstruktion von Entscheidungsbäumen

Evaluation on training data (300 items):

Before Pruning		After Pruning		
Size	Errors	Size	Errors	Estimate
25	8 (2.7%)	7	13 (4.3%)	(6.9%)

Evaluation on test data (135 items):

Before Pruning		After Pruning		
Size	Errors	Size	Errors	Estimate
25	7 (5.2%)	7	4 (3.0%)	(6.9%)

Abbildung VI.1-6: Results with congressional voting data
(Quinlan 1993)

Kapitel VI.1: Konstruktion von Entscheidungsbäumen

c) Windowing (for dealing with *large* datasets)

- start the construction of the decision tree with a subset of the training examples- called a window
- select subset in a way that the distribution of classes in the initial window is as uniform as possible
- first decision tree is built from the initial window
- classify remaining training examples with the new decision tree
- some of these examples are misclassified
 - include subset of them in the next window
 - (C4.5 chooses at least 50%)
- stop iteration if
 - all training examples are correctly classified or
 - sequence of trees is not becoming more accurate

Kapitel VI.1: Konstruktion von Entscheidungsbäumen

d) Erzeugung von Produktionsregeln

- Komplexe Entscheidungsbäume sind schwer zu verstehen, da jeder Test im *Kontext* aller vorhergehenden Tests zu interpretieren ist.
- Lösung:
 - Betrachte alle Tests auf *Pfad* von Wurzel zu aktuellem Knoten
 - und transformiere die Pfadtests in Bedingungen für *Produktionsregeln* der Form

IF test₁ ∧ test₂ ∧ .. ∧ test_n
THEN class = C1

- Eine Klasse wird als *Default-Klasse* verwendet.

Kapitel VI.1: Konstruktion von Entscheidungsbäumen

good, bad.

duration:	continuous.
wage increase first year:	continuous.
wage increase second year:	continuous.
wage increase third year:	continuous.
cost of living adjustment:	none, tcf, tc.
working hours:	continuous.
pension:	none, ret_allw, empl.contr.
standby pay:	continuous.
shift differential:	continuous.
education allowance:	yes, no.
statutory holidays:	continuous.
vacation:	below average, average, generous.
longterm disability assistance:	yes, no.
contribution to dental plan:	none, half, full.
bereavement assistance:	yes, no.
contribution to health plan:	none, half, full.

Abbildung VI.1-7: File defining labor-neg classes and attributes

(Quinlan 1993)

Kapitel VI.1: Konstruktion von Entscheidungsbäumen

C4.5 [release 5] decision tree generator Fri Dec 6 13:33:54 1991

Options:
File stem <labor-neg>
Trees evaluated on unseen cases

Read 40 cases (16 attributes) from labor-neg.data

Decision Tree:

```
wage increase first year <= 2.5 :
  working hours <= 36 : good (2.0/1.0)
  working hours > 36 :
    contribution to health plan = none : bad (5.1)
    contribution to health plan = half : good (0.4/0.0)
    contribution to health plan = full : bad (3.8)
wage increase first year > 2.5 :
  statutory holidays > 10 : good (21.2)
  statutory holidays <= 10 :
    wage increase first year <= 4 : bad (4.5/0.5)
    wage increase first year > 4 : good (3.0)
```

Simplified Decision Tree:

```
wage increase first year <= 2.5 : bad (11.3/2.8)
wage increase first year > 2.5 :
  statutory holidays > 10 : good (21.2/1.3)
  statutory holidays <= 10 :
    wage increase first year <= 4 : bad (4.5/1.7)
    wage increase first year > 4 : good (3.0/1.1)
```

Tree saved

Evaluation on training data (40 items):

Before Pruning		After Pruning		
Size	Errors	Size	Errors	Estimate
12	1 (2.5%)	7	1 (2.5%)	<<

Evaluation on test data (17 items):

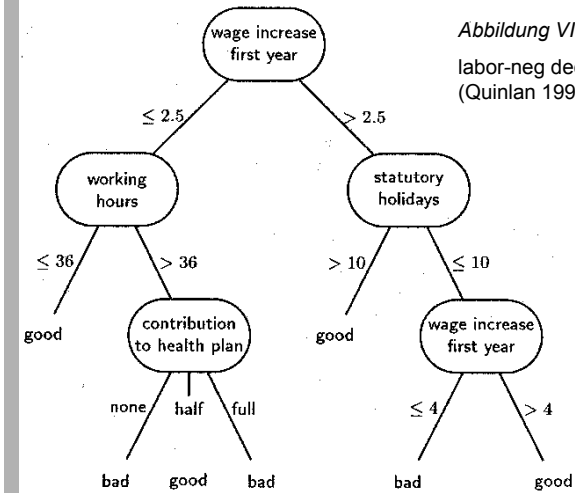
Before Pruning		After Pruning		
Size	Errors	Size	Errors	Estimate
12	3 (17.6%)	7	3 (17.6%)	<<

(a) (b) <<-classified as
10 1 (a): class good
2 4 (b): class bad

Vorlesung Knowledge Discovery 45

Abbildung VI.1-8:
Output of C4.5 on labor-neg data
(Quinlan 1993)

Kapitel VI.1: Konstruktion von Entscheidungsbäumen



Kapitel VI.1: Konstruktion von Entscheidungsbäumen

C4.5 [release 5] rule generator Fri Dec 6 13:34:20 1991

Options:
File stem <labor-neg>
Rulesets evaluated on unseen cases

Read 40 cases (16 attributes) from labor-neg.

Processing tree 0

Final rules from tree 0:

```
Rule 5:
wage increase first year > 2.5
statutory holidays > 10
-> class good [93.0%]
Rule 4:
wage increase first year > 4
-> class good [90.6%]
Rule 3:
wage increase first year <= 4
statutory holidays <= 10
-> class bad [87.1%]
Rule 2:
wage increase first year <= 2.5
working hours > 36
-> class bad [85.7%]
```

Default class: good

Evaluation on training data (40 items):

Rule	Size	Error	Used	Wrong	Advantage	
5	2	7.0%	19	0	0 (0.0%)	good
4	1	9.4%	3	0	0 (0.0%)	good
3	2	12.9%	10	0	0 (0.0%)	bad
2	2	14.3%	4	0	4 (4.0%)	bad

Tested 40, errors 0 (0.0%) <<

(a) (b) <<-classified as
26 14 (a): class good
(b): class bad

Evaluation on test data (17 items):

Rule	Size	Error	Used	Wrong	Advantage	
5	2	7.0%	9	1	0 (0.0%)	good
4	1	9.4%	3	1	0 (0.0%)	good
3	2	12.9%	1	0	1 (1.0%)	bad
2	2	14.3%	3	0	3 (3.0%)	bad

Tested 17, errors 2 (11.8%) <<

(a) (b) <<-classified as
11 4 (a): class good
2 4 (b): class bad

Vorlesung Knowledge Discovery 47

Abbildung VI.1-10:
Result of postprocessing to rules
(Quinlan 1993)

Kapitel VI.1: Konstruktion von Entscheidungsbäumen

- Regeln können gegebenenfalls vereinfacht werden durch Entfernen von Tests aus dem Bedingungsteil:
Beispiel: In Regel 4 in Abbildung VI.1-10 ist die Bedingung „statutory holidays <= 10“ entfernt worden.
 - Vorgehensweise für Klassifikation eines neuen Beispiels:
 - o gehe Regeln der Reihe nach durch (Reihenfolge ist relevant)
 - o erste Regel, deren Bedingungsteil erfüllt ist, wird ausgewählt
 - o rechte Seite der Regel bestimmt Klassenzugehörigkeit für betrachtetes Beispiel
 - o ist von keiner Regel der Bedingungsteil erfüllt, wird Default-Klasse gewählt
- Vorlesung Knowledge Discovery 48

e) Fazit

- C4.5 weitverbreitetes Verfahren zur Erzeugung von Entscheidungsbäumen und zugehörigen Produktionsregeln

- Verfahren abhängig von
 - Bewertungsfunktion für Attributauswahl
 - Schätzung der Fehlerrate bei Pruning
 - Schätzung der Fehlerrate bei Erzeugung und Vereinfachung von Produktionsregeln

- vergleichbare Verfahren
 - ID3 (Quinlan 1983)
 - CN2 (Clark/Niblatt 1989)