Decision Trees*

von Julia Heise, Philipp Thoms,
Hans-Martin Wulfmeyer

*Entscheidungsbäume

Gliederung

1. Einführung
2. Induktion
3. Beispiel
4. Fazit

Einführung

1. **Einführung**
    a. Was sind Decision Trees?
    b. Wo und Wann setzt man sie ein?
    c. Warum Decision Trees?
    d. Herausforderungen
2. Induktion
3. Beispiel
4. Fazit

Was sind Decision Trees?

- Instanzen die durch Attribute beschrieben werden als Baum dargestellt
- gerichtet und nach “Aussagekraft” der Attribute geordnet
- Der Baum t sagt die Werte c1,...,cn eines Attributes für die Instanz X voraus
    - t: X -> {c1,...,cn}
- Regeln für die Zuordnung in Form von **if … ,then…** Aussagen
    - abgelesen von den Pfaden des Baumes

Wo und wann setzt man sie ein?

- Klassifikation und Regression
- Bedingungen:
  - Instance Space: Instanzen werden durch Attribute-Werte Paare beschrieben
  - Target Attribute: Sollen zugeordnet werden
  - Target Attribut muss diskret sein
- klassische Einsatzbeispiele: Test auf Kreditwürdigkeit oder medizinische Diagnosen

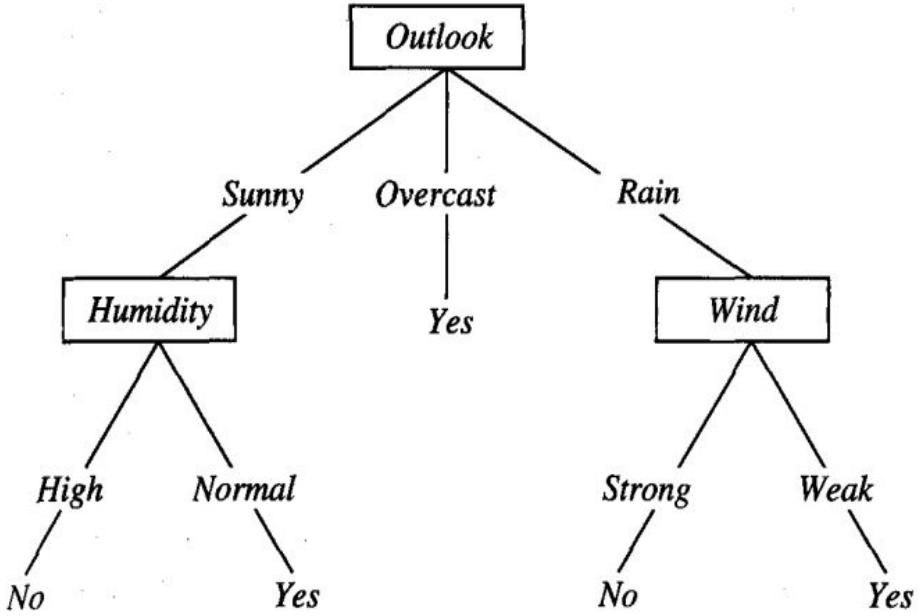
Warum Decision Trees?

- Regeln in kürzester Form leicht ablesbar
- Evaluieren der Attribute läuft bedeutend schneller ab
    - Entscheidungsprozess wird verkürzt

Outlook	Humidity	Wind	Decision
sunny	high	strong	no
rain	high	strong	no
rain	normal	strong	no
sunny	normal	strong	yes
overcast	high	strong	yes
overcast	normal	strong	yes
sunny	high	weak	no
sunny	normal	weak	yes
rain	high	weak	yes
rain	normal	weak	yes
overcast	high	weak	yes
overcast	normal	weak	yes

Outlook	Humidity	Wind	Decision
overcast	high	strong	yes
overcast	high	weak	yes
overcast	normal	strong	yes
overcast	normal	weak	yes
rain	high	strong	no
rain	normal	strong	no
rain	high	weak	yes
rain	normal	weak	yes
sunny	high	strong	no
sunny	high	weak	no
sunny	normal	strong	yes
sunny	normal	weak	yes

Outlook	Humidity	Wind	Decision
overcast	high	strong	yes
overcast	high	weak	yes
overcast	normal	strong	yes
overcast	normal	weak	yes
rain	high	strong	no
rain	normal	strong	no
rain	high	weak	yes
rain	normal	weak	yes
sunny	high	strong	no
sunny	high	weak	no
sunny	normal	strong	yes
sunny	normal	weak	yes



Regelbeispiel:

if (Outlook = sunny **AND** Humidity =Normal)

OR (Outlook = Overcast)

OR (Outlook = Rain **AND** Wind = Weak)

then Yes

Herausforderungen

- Kann man die Attribute so wählen, dass der Baum möglichst klein ist?
    - Algorithmen um Attribute zu priorisieren
- Kontinuierliche Attributwerte : Wie viele Bereiche und wie groß?
- Wie behandelt man Fehler oder fehlende Attributwerte?

Induktion

1. Einführung
2. **Induktion**
    a. Algorithmen zur Induktion
    b. Bewertungsmaße zum Splitting
3. Beispiel
4. Fazit

Algorithmen zur Induktion

- ID3 (Iterative Dichotomiser 3)
- C4.5

ID3 Algorithmus (Quinlan, 1979)

- lernt den Baum von oben nach unten (Top-down)
- wählt das “beste” Merkmal zum “spalten” aus (greedy)
  - beste = “Information gain” des splits
- rekursive Anwendung auf Kindknoten

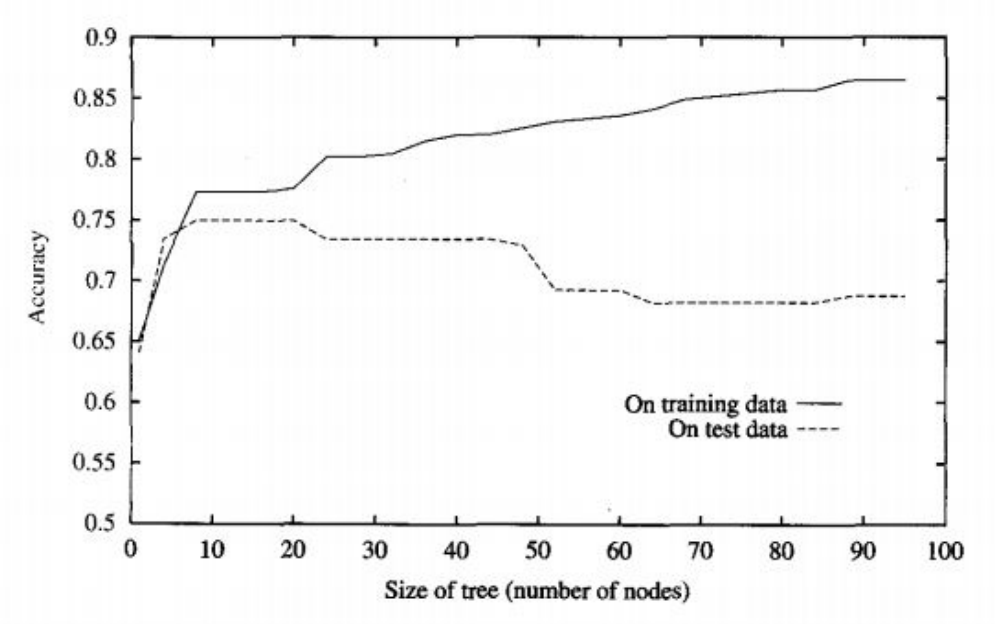
ID3 Algorithmus - Pseudocode

1. X ← das “beste” Attribute aus allen Attributen
2. X = “Spalt”-Attribut vom derzeitigen Knoten
3. FÜR ALLE Werte von X erstelle Kindknoten mit zugehörigen Daten
4. Beginne von vorne falls noch Merkmale übrig

ID3 Algorithmus - Nachteile

- lernt meist nur das lokale Optima
- kann nur mit diskreten Daten arbeiten
- Problem des Overfitting bzw. Rauschanfälligkeit

ID3 Algorithmus - Overfitting



ID3 Algorithmus - Overfitting Ansätze

- Wachsen des Baums ab bestimmter Tiefe stoppen
    - Problem: Wann?

- Baum induzieren, dann Beschneiden (pruning) des Baums
    - reduced-error pruning
    - post-pruning
        - ➔ C4.5 Algorithmus

C4.5

"a landmark decision tree program that is probably the machine learning workhorse most widely used in practice to date".

- Ian H. Witten; Eibe Frank; Mark A. Hall (2011)

C4.5 Algorithmus (Quinlan, 1993)

- Erweiterung des ID3 Algorithmus
- löst das Problem des Overfittings
- behandelt fehlende Attribute in Trainingsbeispielen
- kann mit kontinuierlichen Werten umgehen

C4.5 Algorithmus - Regeln

- unbekannte Attributwerte:
  - Wahrscheinlichkeit & Gewichtung
- kontinuierlichen Werte:
  - alle möglichen Splits in Attributwerten erfassen und besten selektieren

- Post-Pruning (Bottom-Up):
  - Ast im Baum durch Blatt ersetzen wenn erwartete Fehler durch Blatt geringer
  - Ast im Baum durch Teil-Ast ersetzen wenn erwartete Fehler durch Teil-Ast geringer

Weitere DT Algorithmen

- C5.0/See5
  - kommerzielle & patentgeschützte Weiterentwicklung von C4.5
- CART (Classification And Regression Trees)
  - induziert nur binäre Bäume, Gini Index zur Bewertung
  - andere pruning Methode -> cost-complexity model
- CHAID (chi-square automatic interaction detector)
  - nur diskrete Werte, stoppen des Wachsens ab bestimmter Tiefe

Bewertungsmaße

- Entropie
- Gini

Entropie

- misst die “unreinheit” der Menge unserer Beispiele
- für boolesche Klassifikation: $$Entropy(S) \equiv -p_{\oplus} \log_2 p_{\oplus} - p_{\ominus} \log_2 p_{\ominus}$$
- für mehr als zwei Klassen: $$Entropy(S) \equiv \sum_{i=1}^{c} -p_i \log_2 p_i$$

Entropie - Information Gain

- beschreibt wie gut ein Merkmal unsere Menge klassifiziert
- Vergleich der Entropie vor und nach dem Spalten mit Merkmal A
- genauer beschreibt es die reduzierung der Entropie nach dem Spalten

$$Gain(S, A) \equiv Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

S = Trainingsbeispiele
A = ausgewähltes Merkmal/Attribut
Values(A) = alle möglichen Werte die A annehmen kann
S_v = Untermenge von S in der alle Beispiele Merkmal=v haben

Information Gain - Gain Ratio

- normalisiert den Information Gain

$$GainRatio(a_i, S) = \frac{InformationGain(a_i, S)}{Entropy(a_i, S)}$$

- Probleme:
  - nicht definiert wenn der Nenner null ist
  - Gain Ratio favorisiert Attribute mit sehr geringer Entropie

- Lösung:
  - Wähle alle Attribute bei denen Information Gain ≥ Ø des Information Gain
  - von dieser Menge selektieren wir das beste Attribut anhand des Gain Ratio

Gini Impurity

- misst die homogenität oder “Reinheit” der Trainingsbeispiele
- auch als Missklassifikationsrate beschreibbar
- kleiner = besser

$$Gini(S) = 1 - \sum_i p_i^2$$

P_i = relative Häufigkeit der Elemente mit Klasse i
J = Menge der möglichen Klassen

Gini Impurity ≠ Gini Koeffizient

Gini Impurity vs Information Gain(Entropie)

- Gini eher für kontinuierliche Attribute, Entropie eher für diskrete Attribute
- Unterschied in nur 2% der Fälle im Vergleich
- Entropie ist etwas langsamer das der Logarithmus berechnet werden muss

Ergebnis: *macht kaum einen Unterschied welchen wir verwenden*

Beispiel

1. Einführung
2. Induktion
3. **Beispiel**
4. Fazit

Beispiel

PlayTennis mithilfe von ID3

Day	Outlook	Temperature	Humidity	Wind	Play
D1	Sun	Hot	High	Low	No
D2	Sun	Hot	High	High	No
D3	Overcast	Hot	High	Low	Yes
D4	Rain	Sweet	High	Low	Yes
D5	Rain	Cold	Normal	Low	Yes
D6	Rain	Cold	Normal	High	No
D7	Overcast	Cold	Normal	High	Yes
D8	Sun	Sweet	High	Low	No
D9	Sun	Cold	Normal	Low	Yes
D10	Rain	Sweet	Normal	Low	Yes
D11	Sun	Sweet	Normal	High	Yes
D12	Overcast	Sweet	High	High	Yes
D13	Overcast	Hot	Normal	Low	Yes
D14	Rain	Sweet	High	High	No

Entropy vom Set:

Entropy (S)= $-9/14*\log_2(9/14)$ $-5/14*\log_2(5/14)$

= 0.94

Information Gain von Outlook:

Gain(S,Outlook) = Entropy(S) - $5/14*$Entropy(S_{Sun})

- $4/14*$Entropy(S_{Rain})

- $5/14*$Entropy($S_{Overcast}$)

= 0.94 - $5/14*0.971$ - $4/14*0$ - $5/14*0,971$

Gain(S,Outlook) = 0.246

Entropy vom Set:

Entropy (S)= $-9/14*\log_2(9/14)$ $-5/14*\log_2(5/14)$

= 0.94

Information Gain:

Gain(S, Outlook)	**= 0.246**
Gain(S, Humidity)	**= 0.151**
Gain(S, Wind)	**= 0.048**
Gain(S, Temperature)	**= 0.029**

Information Gain von Outlook:

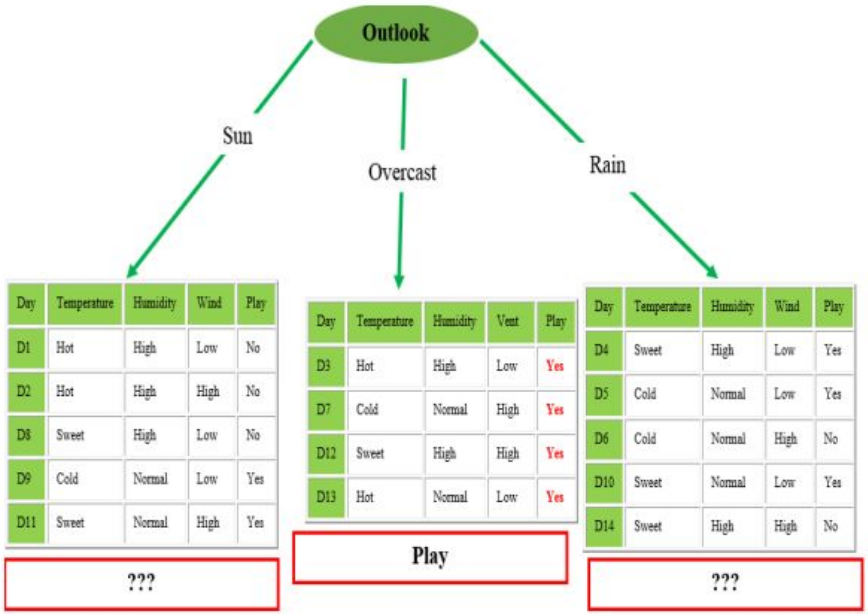
Gain(S,Outlook) = Entropy(S) - $5/14*\text{Entropy}(S_{Sun})$

- $4/14*\text{Entropy}(S_{Rain})$

- $5/14*\text{Entropy}(S_{Overcast})$

= 0.94 - 5/14*0.971 - 4/14*0 - 5/14*0,971

Gain(S,Outlook) = 0.246



- Overcast ist eindeutig
- Information gain muss für Sun und Rain neu berechnet werden

Bsp. für den linken Ast:

S_{Sun} = {D1,D2,D8,D9,D11}

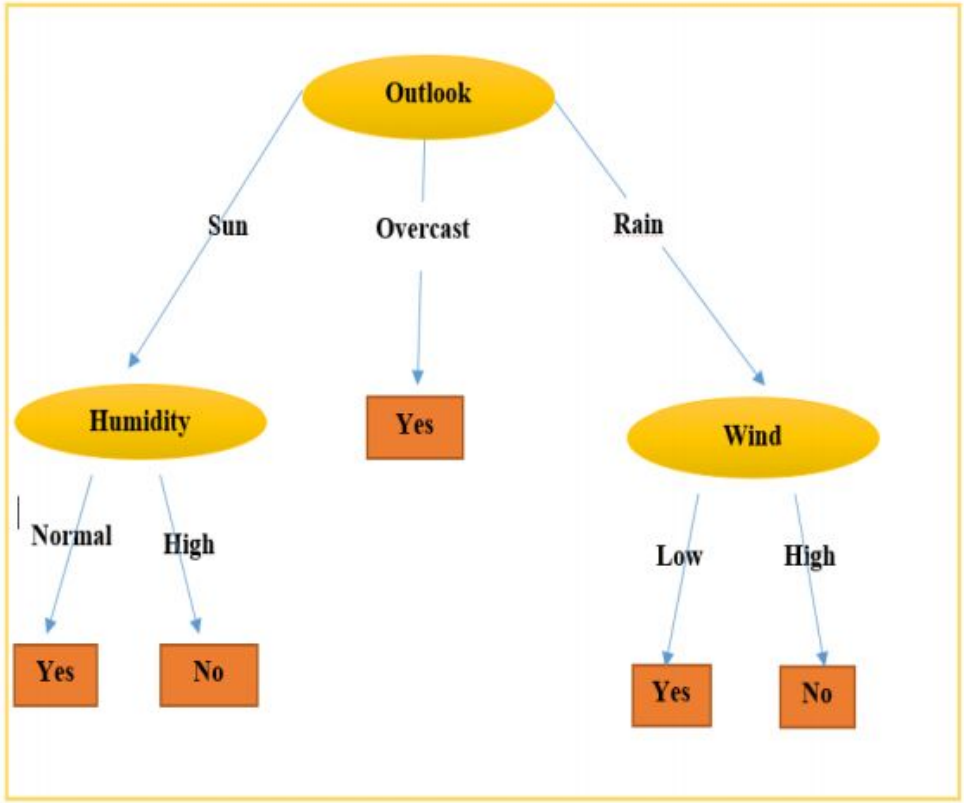
$Gain(S_{Sun},Humidity) = 0.97 - (\frac{3}{5})*0 - (\frac{2}{5})*0 = 0.97$

$Gain(S_{Sun},Temperature) = 0.97 - (\frac{2}{5})*0 - (\frac{2}{5})*1 - (\frac{1}{5})*0 = 0.57$

$Gain(S_{Sun},Wind) = 0.97 - (\frac{2}{5})*1 - (\frac{3}{5})*0.981 = 0.019$

fertiger Baum wenn:

- Alle Attribute abgelaufen
- festes Ergebnis (Entropy = 0)



Fazit

1. Einführung
2. Induktion
3. Beispiel
4. **Fazit**

Fazit

Vorteile	Nachteile
Regeln werden einfach abgeleitet	Keine gute Klassifizierung für kontinuierliche Targetfunktionen
Robust, sowohl gegenüber Fehlern in der Klassifikation im Training, als auch bei fehlerhaften Attributwerten	Kontinuierliche Attribute erhöhen den Rechenaufwand stark
Kann einfach zu “Random Forest” erweitert werden	Kann sehr groß werden → Beschneidung ist notwendig
	Kann an Overfitting leiden

Danke für die Aufmerksamkeit!

Seht ihr den Wald vor lauter Bäumen nicht mehr,

oder habt ihr alles verstanden?

Quellen

- Machine Learning - A Guide to Current Research (1986)
  Tom Mitchell et. al.
- Induction of Decision Trees (1985)
  J.R. Quinlan
- A comparative study of decision tree ID3 and C4.5 (2014)
  Badr Hssina et. al.
- Top 10 algorithms in data mining (2008)
  Wu, X., Kumar, V., Ross Quinlan, J. et al.
- Theoretical Comparison between the Gini Index and Information Gain Criteria (2004)
  Laura Elena Raileanu, Kilian Stoffel