



DHBW

Duale Hochschule
Baden-Württemberg

Duale Hochschule Baden - Württemberg Mannheim

Seminararbeit

Entscheidungsbäume

Am Beispiel des ID3 Algorithmus

Studiengang Angewandte Informatik

Studienrichtung Informatik

Autor:

Martin Pretz

Matrikelnummern:

7060026

Kurs:

TINF18AI1

Bearbeitungszeitraum:

19.05.2021 - 10.06.2021

Inhaltsverzeichnis

Abbildungsverzeichnis	3
Tabellenverzeichnis	4
1 Abstract	5
2 Einführung	6
3 Was sind Entscheidungsbäume?	7
3.1 Aufbau und Funktionsweise	7
3.2 Erzeugung eines Entscheidungsbaums	7
4 Der ID3 Algorithmus	8
4.1 Grundlagen	8
4.1.1 Entropie	8
4.1.2 Informationsgewinn	9
4.1.3 Funktionsweise	11
4.2 Anwendung an einem Beispiel Datensatz	12
4.3 Implementation	17
4.3.1 Berechnung der Entropie	17
4.3.2 Berechnung des Informationsgewinns	17
4.3.3 Berechnung des Modalwertes	18
4.3.4 ID3 - Hauptfunktion	18
5 Zusammenfassung	20
Literatur	21

Abbildungsverzeichnis

Abbildung 4.1	Definition der Entropie nach Shannon[6]	8
Abbildung 4.2	Exmeplarishe Berechnung der Entropie	9
Abbildung 4.3	Allgemeine Definition des Informationsgewinns [3, 9, 10]	9
Abbildung 4.4	Vorläufiger Teilbaum nachdem <i>HUMIDITY</i> klassifiziert wurde .	14
Abbildung 4.5	Vorläufiger Teilbaum nachdem <i>WIND</i> klassifiziert wurde	15
Abbildung 4.6	Finaler Entscheidungsbaum	16
Abbildung 4.7	Funktion zu Berechnung der Entropie eines Attributes[13, 7] . .	17
Abbildung 4.8	Funktion zur Berechnung des Informationsgewinns [13, 7] . . .	18
Abbildung 4.9	Berechnung des Modalwertes [14]	18
Abbildung 4.10	Hauptfunktion des ID3 Algorithmus [14, 12, 13]	19

Tabellenverzeichnis

Tabelle 4.1	Beispiel Datensatz S zur Berechnung des Informationsgewinns [8]	9
Tabelle 4.2	Beispiel Datensatz S	12
Tabelle 4.3	Teilmenge S_1	13
Tabelle 4.4	Teilmenge $S_{1_{HUMIDITY=High}}$	13
Tabelle 4.5	Teilmenge $S_{1_{HUMIDITY=Normal}}$	13
Tabelle 4.6	Teilmenge S_2	14
Tabelle 4.7	Teilmenge $S_{2_{WIND=Weak}}$	14
Tabelle 4.8	Teilmenge $S_{2_{WIND=Strong}}$	15
Tabelle 4.9	Teilmenge S_3	15

1 Abstract

2 Einführung

3 Was sind Entscheidungsbäume?

3.1 Aufbau und Funktionsweise

Ein Entscheidungsbaum ist ein geordneter und gerichteter [1] Baum welcher aus Knoten und Kanten besteht. Dabei unterteilt man Knoten in zwei verschiedene Typen. Zum einen gibt es die Entscheidungsknoten [2]. Dabei handelt es sich um eine Überprüfung einer Eigenschaft eines Datenobjektes [3] während eine Kante gerade das Ergebnis jener Überprüfung ist. Dabei gibt es den Wurzelknoten als besondere Ausprägung eines Entscheidungsknoten. Er unterscheidet sich von einem normalen Entscheidungsknoten nur in soweit, als dass der Wurzelknoten der Ursprung des gesamten Entscheidungsbaums ist. Zum anderen gibt es die Endknoten [2] welche üblicherweise als Blätter bezeichnet werden. Bei einem Blatt handelt es sich um einen Endzustand welcher das Ergebnis bzw. die Klassifikation angibt. [1]

Möchte man nun ein Datenobjekt klassifizieren so folgt man beginnend mit dem Wurzelknoten dem Entscheidungsbaum abwärts. Dabei wird bei jedem Entscheidungsknoten eine Eigenschaft bzw. ein Attribut des Datenobjektes überprüft. Basierend auf dem Ergebnis dieser Überprüfung wird dann entlang einer der Kanten der nachfolgende Entscheidungsknoten ausgewählt. [1] Dies "[...] wird so lange fortgesetzt bis man ein Blatt erreicht" [1] und somit das Datenobjekt klassifiziert ist.

TODO: Sollte der Aufbau vielleicht mathematisch genau beschrieben werden?

3.2 Erzeugung eines Entscheidungsbaums

Die Erzeugung eines Entscheidungsbaums erfolgt in der Regel nach dem Top-Down Prinzip, indem in jedem Prozessschritt das beste Attribut ausgewählt wird um daraus einen Knoten zu erzeugen und den Datensatz aufzuteilen. [4] Dabei wird ein Trainingsdatensatz mit Objekten verwendet für die bereits eine Klassifikation vorliegt. [1] Dabei nutzen die verschiedenen Algorithmen unterschiedliche Methoden um das beste Attribut zu ermitteln, z.B. den Informationsgewinn, den Gini-Index oder den Misclassification Error. [3]

4 Der ID3 Algorithmus

In diesem Kapitel wird die Erzeugung eines Entscheidungsbaums am Beispiel des ID3 Algorithmus erläutert. Bei ID3 (Iterative Dichotomiser 3) handelt es sich um einen Algorithmus zur Erstellung eines Entscheidungsbaumes welcher von Ross Quinlan entwickelt wurde. [5]

4.1 Grundlagen

Der ID3-Algorithmus macht sich zwei Konzepte der Informationstheorie zu nutze. Es handelt sich zum einen um die Entropie und zum anderen um den Informationsgewinn. Beide Konzepte werden im nachfolgenden erläutert. Im Anschluss daran wird die eigentliche Funktionsweise des ID3 Algorithmus erläutert.

4.1.1 Entropie

In der Informationstheorie wird mit der Entropie H die Sicherheit bzw. Unsicherheit einer Variablen X angegeben. Dementsprechend ist x_i eine mögliche Ausprägung der Variablen X und $P(x_i)$ die Wahrscheinlichkeit mit der die Variable X die Ausprägung x_i hat. [6]

$$H(X) = - \sum_{i=1}^n P(x_i) \log_b P(x_i)$$

Abbildung 4.1: Definition der Entropie nach Shennon[6]

Beispiel: Sei D ein Datensatz in dem das Attribut X mit den möglichen Ausprägungen x_1, x_2 und x_3 vorkommt. Weiterhin gelte, dass x_1 neun mal, x_2 drei mal und x_3 fünf mal in D vorhanden ist. Zur Bestimmung der Entropie von X ergibt sich die nachfolgende Berechnung. Für den ID3-Algorithmus wird üblicherweise der Logarithmus zur Basis $b=2$ verwendet. [7]

$$\begin{aligned}
H(X) &= - \sum_{i=1}^3 P(x_i) \log_2 P(x_i) \\
&= -(P(x_1) \log_2 P(x_1) + P(x_2) \log_2 P(x_2) + P(x_3) \log_2 P(x_3)) \\
&= -\frac{9}{17} \cdot \log_2 \left(\frac{9}{17} \right) - \frac{3}{17} \cdot \log_2 \left(\frac{3}{17} \right) - \frac{5}{17} \cdot \log_2 \left(\frac{5}{17} \right) \\
&\approx 0,485755 + 0,441618 + 0,519275 \\
&\approx \underline{\underline{1,446648}}
\end{aligned}$$

Abbildung 4.2: Exmeplarische Berechnung der Entropie

4.1.2 Informationsgewinn

In der Informationstheorie beschreibt der Informationsgewinn IG das Maß an Informationen das über eine Zufallsvariablen X durch Beobachtung einer anderen Zufallsvariablen Y gewonnen werden kann. [3] Konkret ergibt sich der Informationsgewinn aus der Differenz der Entropie $H(X)$ und der bedingten Entropie $H(X|Y)$. [8]

$$IG(X, Y) = H(X) - H(X|Y) = H(X) - \sum_{y \in Y} P(y) H(X|Y = y)$$

Abbildung 4.3: Allgemeine Definition des Informationsgewinns [3, 9, 10]

Beispiel: Sei S ein Datensatz mit den in Tabelle 4.1 dargestellten Werten. Außerdem seien A , B , C und T Attribute von S mit den möglichen Ausprägungen *True* und *False*. Sei weiterhin das Attribut T das Zielattribut gegen das der Informationsgewinn der übrigen Attribute ermittelt werden soll.

ID	Attribut A	Attribut B	Attribut C	Attribut T
1	True	True	True	False
2	True	False	True	True
3	False	False	True	True
4	False	True	True	False
5	False	True	False	True

Tabelle 4.1: Beispiel Datensatz S zur Berechnung des Informationsgewinns [8]

Im nachfolgenden wird der Informationsgewinn für das Attribut A berechnet. Dabei wird zu erst die Entropie des Datensatz S bestimmt, welche durch die Entropie des Zielattributes T charakterisiert wird. Es gilt also $H(S) = H(T)$.

$$\begin{aligned}
 H(S) = H(T) &= - \sum_{i=1}^2 P(x_i) \log_2 P(x_i) \\
 &= -(P(\text{True}) \log_2 P(\text{True}) + P(\text{False}) \log_2 P(\text{False})) \\
 &= -\frac{3}{5} \cdot \log_2 \left(\frac{3}{5} \right) - \frac{2}{5} \cdot \log_2 \left(\frac{2}{5} \right) \\
 &\approx 0,970951
 \end{aligned}$$

Im nächsten Schritt wird die bedingte Entropie für das Attribut A berechnet. [10, 9]

$$\begin{aligned}
 H(T|A) &= \sum_{a \in A} P(a) \cdot H(T|A = a) \\
 &= P(\text{True}) \cdot H(T|A = \text{True}) + P(\text{False}) \cdot H(T|A = \text{False}) \\
 &= \frac{2}{5} \cdot \left(-\frac{1}{2} \cdot \log_2 \left(\frac{1}{2} \right) - \frac{1}{2} \cdot \log_2 \left(\frac{1}{2} \right) \right) + \frac{3}{5} \cdot \left(-\frac{1}{3} \cdot \log_2 \left(\frac{1}{3} \right) - \frac{2}{3} \cdot \log_2 \left(\frac{2}{3} \right) \right) \\
 &= \frac{2}{5} \cdot 1 + \frac{3}{5} \cdot 0,918296 \\
 &\approx 0.950978
 \end{aligned}$$

Nachdem nun sowohl die Entropie als auch die bedingte Entropie berechnet sind kann final der Informationsgewinn bestimmt werden.

$$\begin{aligned}
 IG(T, A) &= H(T) - H(T|A) \\
 &= 0,970951 - 0.950978 \\
 &= \underline{\underline{0,019973}}
 \end{aligned}$$

Analog können die Informationsgewinne für die Attribute B und C berechnet werden. Diese liegen bei $IG(T, B) = 0,419973$ und bei $IG(T, C) = 0,170951$.

4.1.3 Funktionsweise

Der ID3 Algorithmus startet mit einem Datensatz S mit Objekten O_1, \dots, O_n welche die Attribute A_1, \dots, A_k und die Klassifizierung C besitzen. [7, 5] Die Werteausprägungen der Attribute sind dabei normalerweise endlich und diskret. [11]

Zu Beginn muss ein (Wurzel-) Knoten bestimmt werden. Dazu wird der Informationsgewinn $IG(S)$ für jedes Attribut A_1, \dots, A_k berechnet. Das Attribut mit dem höchsten Informationsgewinnungswert A_H wird dann als Knoten gewählt. Dabei gilt dass A_H die möglichen Ausprägungen a_1, \dots, a_v haben kann. Basierend auf A_H wird S in S_1, \dots, S_v Teilmengen zerlegt wobei die Teilmenge S_i die Objekte aus S beinhaltet deren Wert in A_H die Ausprägungen a_i hat. Anders ausgedrückt gilt $S_i = \{S | A_H = a_i\}$. Für jede Teilmenge S_i wird dann rekursiv der vorher beschriebene Prozess für die verbleibenden Attribute $\{A_1, \dots, A_k\} \setminus A_H$ durchlaufen wodurch entsprechende Teilbäume erzeugt werden. [5] Dabei wird die Rekursion beendet sofern,

1. alle Objekte aus S_i die gleiche Ausprägung C_i der Klassifizierung C besitzen. In diesem Fall wird der zu diesem Zeitpunkt ausgewählte Knoten zu einem Blatt mit der Ausprägung C_i . [5, 12]
2. bereits alle Attribute A_1, \dots, A_k für den Entscheidungsbaum verwendet worden sind, es also keine Attribute mehr gibt mittels derer ein Objekt aus S_i klassifiziert werden könnte, aber nicht alle Objekte der selben Klassifizierung C_i zugeordnet werden können. In diesem Fall wird der aktuelle Knoten zu einem Blatt, welchem der Modalwert von C zugeordnet wird. [5, 12]
3. die Teilmenge S_i keine Objekte mehr beinhaltet, sie also leer ist. Dies tritt ein wenn sich in der übergeordneten Menge keine Objekte befinden welche die Ausprägung a_i eines Attributes A_M besitzen. In diesem Fall wird ein Blatt erzeugt welchem der Modalwert von C aus der übergeordneten Menge zugeordnet wird. [12]

TODO: Mathematische Korrektheit prüfen!!!

4.2 Anwendung an einem Beispiel Datensatz

Zur Veranschaulichung des ID3 Algorithmus wurde ein Datensatz S gewählt welcher in Tabelle 4.2 dargestellt ist. [7]. Dieser Datensatz beinhaltet die Attribute $WEATHER$, $TEMP$, $HUMIDITY$ und $WIND$ sowie eine Klassifizierung $PLAY$.

Hierbei ist die Aufgabe den ID3 Algorithmus zu nutzen um einen Entscheidungsbaum zu erstellen um vorherzusagen ob das Wetter ermöglicht dass Fußball gespielt werden kann.

DAY	WEATHER	TEMP	HUMIDITY	WIND	PLAY
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Cloudy	Hot	High	Weak	Yes
4	Rainy	Medium	High	Weak	Yes
5	Rainy	Cold	Normal	Weak	Yes
6	Rainy	Cold	Normal	Strong	No
7	Cloudy	Cold	Normal	Strong	Yes
8	Sunny	Medium	High	Weak	No
9	Sunny	Cold	Normal	Weak	Yes
10	Rainy	Medium	Normal	Weak	Yes
11	Sunny	Medium	Normal	Strong	Yes
12	Cloudy	Medium	High	Strong	Yes
13	Cloudy	Hot	Normal	Weak	Yes
14	Rainy	Medium	High	Strong	No

Tabelle 4.2: Beispiel Datensatz S

Als erstes werden die Informationsgewinne für die Attribute aus dem Datensatz S berechnet. Dabei ergibt sich folgende Übersicht.

$$\begin{aligned}IG(S, WEATHER) &= 0.246749 \\IG(S, TEMP) &= 0.029222 \\IG(S, HUMIDITY) &= 0.151835 \\IG(S, WIND) &= 0.048127\end{aligned}$$

Da das Attribut $WEATHER$ den höchsten Informationsgewinn hat wird es als Knoten bzw. Wurzelknoten gewählt. $WEATHER$ besitzt die drei möglichen Ausprägungen $Sunny$, $Cloudy$ und $Rainy$ für die nun im jeweiligen Iterationschritt eine Teilmenge von S erzeugt wird. Daraus ergibt sich zunächst für die Ausprägung $Sunny$ folgende Teilmenge $S_1 = S_{WEATHER=Sunny}$.

DAY	WEATHER	TEMP	HUMIDITY	WIND	PLAY
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
8	Sunny	Medium	High	Weak	No
9	Sunny	Cold	Normal	Weak	Yes
11	Sunny	Medium	Normal	Strong	Yes

Tabelle 4.3: Teilmenge S_1

Für S_1 wird nun rekursiv ein Teilbaum erzeugt wobei die Teilmenge keine der Abbruchbedingungen erfüllt, sodass man nun mit der Berechnung der Informationsgewinne für die Attribute aus S_1 fortfahren kann. Dabei ergeben sich folgende Werte.

$$\begin{aligned}
IG(S, TEMP) &= 0.570951 \\
IG(S, HUMIDITY) &= 0.970951 \\
IG(S, WIND) &= 0.019973
\end{aligned}$$

Nun wird das Attribut *HUMIDITY* als Knoten gewählt da es nun den höchsten Informationsgewinn besitzt. Dabei hat *HUMIDITY* die zwei mögliche Ausprägungen *High* und *Normal*. Somit ergeben sich die Teilmengen $S_{1_{HUMIDITY=High}}$ und $S_{1_{HUMIDITY=Normal}}$ aus S_1 .

DAY	WEATHER	TEMP	HUMIDITY	WIND	PLAY
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
8	Sunny	Medium	High	Weak	No

Tabelle 4.4: Teilmenge $S_{1_{HUMIDITY=High}}$

DAY	WEATHER	TEMP	HUMIDITY	WIND	PLAY
9	Sunny	Cold	Normal	Weak	Yes
11	Sunny	Medium	Normal	Strong	Yes

Tabelle 4.5: Teilmenge $S_{1_{HUMIDITY=Normal}}$

Betrachtet man oben genannten Teilmengen fällt sofort auf dass alle Objekte dieser Teilmengen die jeweils die selbe Klassifizierung besitzen. Für $S_{1_{HUMIDITY=High}}$ nämlich die Klassifizierung *No* und für $S_{1_{HUMIDITY=Normal}}$ die Klassifizierung *Yes*. Daher wird für beide Ausprägungen des Knotens *Humidity* ein Blatt erstellt welches jeweils die entsprechende Klassifizierung zugewiesen bekommt. Daraus ergibt sich folgender vorläufiger Teilbaum.

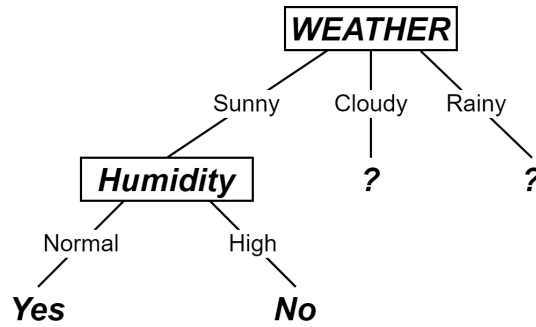


Abbildung 4.4: Vorläufiger Teilbaum nachdem *HUMIDITY* klassifiziert wurde

Da nun die Ausprägungen des Attributes *HUMIDITY* klassifiziert worden sind fahren wir mit der zweiten Ausprägung von *WEATHER* fort, nämlich *Rainy*. Für diese wird die Teilmenge $S_2 = S_{WEATHER=Rainy}$ erstellt.

DAY	WEATHER	TEMP	HUMIDITY	WIND	PLAY
4	Rainy	Medium	High	Weak	Yes
5	Rainy	Cold	Normal	Weak	Yes
6	Rainy	Cold	Normal	Strong	No
10	Rainy	Medium	Normal	Weak	Yes
14	Rainy	Medium	High	Strong	No

Tabelle 4.6: Teilmenge S_2

Für S_2 wird nun erneut der Informationsgewinn der verbleibenden Attribute berechnet. Zu diesem Zeitpunkt bleiben nur noch die Attribute *TEMP* und *WIND* übrig, da die anderen Attribute bereits verwendet worden sind. Es ergeben sich folgende Werte für den Informationsgewinn.

$$\begin{aligned}
 IG(S, TEMP) &= 0.019973 \\
 IG(S, WIND) &= 0.970951
 \end{aligned}$$

Basierend auf den Informationsgewinnen wird *WIND* als Knoten gewählt. Dieses Attribut besitzt die zwei Ausprägungen *Weak* und *Strong*. Somit ergeben sich die Teilmengen $S_{2_{WIND=Weak}}$ und $S_{2_{WIND=Strong}}$ aus S_2 .

DAY	WEATHER	TEMP	HUMIDITY	WIND	PLAY
4	Rainy	Medium	High	Weak	Yes
5	Rainy	Cold	Normal	Weak	Yes
10	Rainy	Medium	Normal	Weak	Yes

Tabelle 4.7: Teilmenge $S_{2_{WIND=Weak}}$

DAY	WEATHER	TEMP	HUMIDITY	WIND	PLAY
6	Rainy	Cold	Normal	Strong	No
14	Rainy	Medium	High	Strong	No

Tabelle 4.8: Teilmenge $S_{2_{WIND=Strong}}$

Bei beiden Teilmengen fällt auf dass alle Objekte aus diesen Teilmengen die selbe Klassifizierung aufweisen. An dieser Stelle wird analog zum Vorgehen beim Attribut *HUMIDITY* ein Blatt für die Ausprägungen von *WIND* erstellt. Daraus ergibt sich nun folgender Teilbaum.

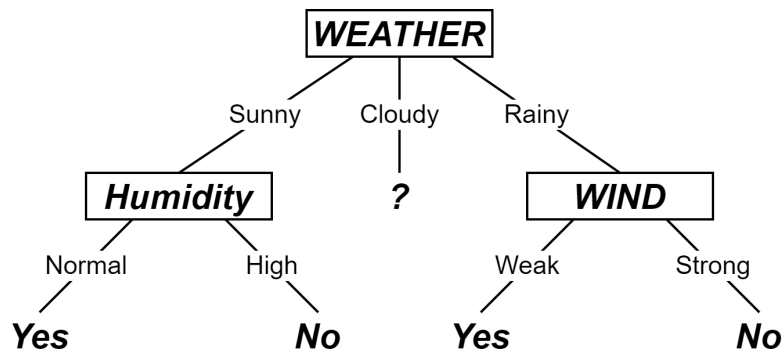


Abbildung 4.5: Vorläufiger Teilbaum nachdem *WIND* klassifiziert wurde

Nachdem also nun die Klassifizierung für das Attribut *WIND* und somit auch für die Ausprägung *Rainy* abgeschlossen ist, bleibt nur noch die Ausprägung *Cloudy* zur Klassifizierung übrig. Basierend darauf ergibt sich folgende Teilmenge $S_3 = S_{WEATHER=Cloudy}$.

DAY	WEATHER	TEMP	HUMIDITY	WIND	PLAY
3	Cloudy	Hot	High	Weak	Yes
7	Cloudy	Cold	Normal	Strong	Yes
12	Cloudy	Medium	High	Strong	Yes
13	Cloudy	Hot	Normal	Weak	Yes

Tabelle 4.9: Teilmenge S_3

Bei der Betrachtung von S_3 fällt auf dass alle beinhalteten Objekte die selbe Klassifizierung *Yes* besitzen. Aus diesem Grund wird analog zu der Vorgehensweise bei *WIND* und *HUMIDITY* ein Blatt mit der entsprechenden Klassifizierung erstellt. Somit wird der ID3 Algorithmus beendet und es entsteht der folgende finale Entscheidungsbaum.

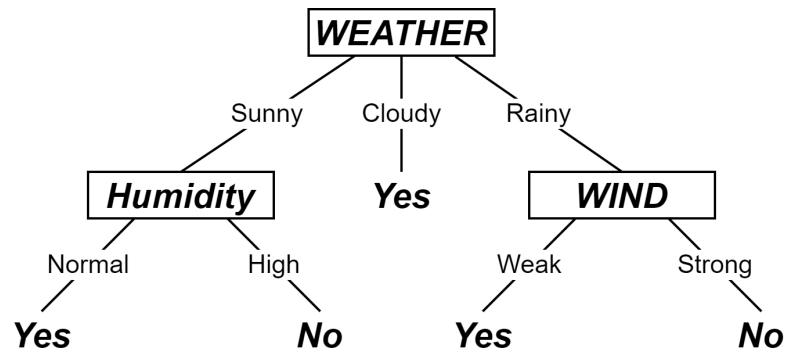


Abbildung 4.6: Finaler Entscheidungsbaum

Aus dem oben dargestellten Entscheidungsbaum ergeben sich folgende Regeln für das Spielen.

1. Wenn das Wetter sonnig ist und die Humidität normal, dann kann Fußball gespielt werden. Wenn die Humidität allerdings hoch ist, dann kann kein Fußball gespielt werden. [7]
2. Wenn das Wetter bewölkt ist, dann kann Fußball gespielt werden. [7]
3. Wenn das Wetter regnerisch ist und der Wind schwach, dann kann Fußball gespielt werden. Falls aber der Wind stark ist, dann kann kein Fußball gespielt werden. [7]

4.3 Implementation

Diese persönliche Implementation besteht im wesentlichen aus vier Funktionen. Zum einen aus Nebenfunktionen wie der Berechnung des Informationsgewinns, der Entropie und des Modalwerts, zum anderen aus der Hauptfunktion bzw. dem eigentlichen ID3 Algorithmus. **TODO: Sollen die einzelnen Funktionen wirklich in separate Kapitel oder eher zusammengefasst werden?**

4.3.1 Berechnung der Entropie

Die hier vorliegende Implementation zur Berechnung der Entropie erwartet als Eingabeparameter ein Attribut eines Datensatzes. Im Anschluss daran wird über die Zuweisung in Zeile 3 die Anzahl aller Ausprägungen des Attributes ermittelt. Dabei kommt die Bibliotheksfunktion `np.unique` zum Einsatz. Diese Funktion gibt zwei Listen zurück. Die erste Liste beinhaltet dabei alle möglichen Ausprägungen des Attributes während die zweite Liste die Anzahl eben jener Ausprägungen beinhaltet. Im Anschluss daran wird über die Länge der Liste `count` iteriert. Dabei wird in jedem Durchlauf die Wahrscheinlichkeit einer Ausprägung des Attributes berechnet. Außerdem wird die Entropie partiell für die vorliegende Ausprägung berechnet und mit dem vorherigen partiellen Entropiewert addiert.

```
1     def entropy(attribute):
2         entropy = 0.0
3         values, count = np.unique(attribute, return_counts=True)
4         for index in range(len(values)):
5             probability = count[index] / sum(count)
6             entropy += (-probability * np.log2(probability))
7         return entropy
```

Abbildung 4.7: Funktion zu Berechnung der Entropie eines Attributes[13, 7]

4.3.2 Berechnung des Informationsgewinns

Die vorliegende Implementation des Informationsgewinns erwartet drei Eingabeparameter. Diese sind der Datensatz, ein Attribut dieses Datensatzes und das Zielattribut gegen welches der Informationsgewinn bestimmt werden soll. Zu Beginn werden unter Zuhilfenahme der Funktion `np.unique` alle möglichen Ausprägungen des Attributes sowie deren Anzahl bestimmt.

Im Hauptteil dieser Funktion wird über die möglichen Ausprägungen `values` des

Attribute iteriert. Dabei wird in Zeile 5 zunächst eine Teilmenge `subdata` des ursprünglichen Datensatzes gebildet (vgl. Kapitel 4.1.2). Im Anschluss daran wird die Wahrscheinlichkeit der Ausprägung `value`, sowie die Entropie der Teilmenge `subdata` berechnet. Dabei werden die Entropieen der verschiedenen Ausprägungen addiert. Im letzten Schritt wird der Informationsgewinn aus der Differenz der Entropie des Zielattributes und der bedingten Entropie berechnet (vgl. Kapitel 4.1.2).

```

1      def information_gain(data, attribute, target_attribute):
2          values, count = np.unique(data[attribute], return_counts=True)
3          entropy_val = 0.0
4          for index, value in enumerate(values):
5              subdata = data[data[attribute] == value][target_attribute]
6              probability = count[index] / sum(count)
7              entropy_val += ( probability * entropy(subdata) )
8          target_entropy = entropy(data[target_attribute])
9          information_gain = target_entropy - entropy_val
10         return information_gain

```

Abbildung 4.8: Funktion zur Berechnung des Informationsgewinns [13, 7]

4.3.3 Berechnung des Modalwertes

Die Funktion `modal` erwartet eine Liste von Elementen als Eingabeparameter. Zu Beginn der Funktion wird zunächst mit Hilfe der Funktion `np.unique` die Menge aller vorkommenden Werte sowie deren Anzahl in der Anfangsliste bestimmt. Im Anschluss daran wird aus beiden Informationen ein Dictionary erstellt. Im letzten Schritt wird mittels einer Lambda Funktion der Schlüssel des Dictionary ermittelt dessen Wert am höchsten ist.

```

1      def modal(attribute):
2          values, count = np.unique(attribute, return_counts=True)
3          total = dict(zip(values, count))
4          return max(total, key=lambda k: total[k])

```

Abbildung 4.9: Berechnung des Modalwertes [14]

4.3.4 ID3 - Hauptfunktion

Die Hauptfunktion erwartet als Eingabeparameter den Ursprungsdatensatz, das Zielattribut und eine Liste aller möglichen Attribute des Datensatzes. Bei dem Zielattribut handelt es sich um die Klassifizierung.

Zu Beginn der Funktion werden zwei Abbruchbedingungen geprüft (vgl. Kapitel 4.1). Als erstes wird untersucht, ob die sich in dem Datensatz `data_set` ausschließlich Objekte mit der gleichen Klassifizierung befinden. Wenn dem so ist, wird genau diese Klassifizierung zurück gegeben. Als zweites wird untersucht, ob bereits alle Attribute verwendet worden sind. Sofern dies zutrifft wird der Modalwert des Zielattributes zurück gegeben.

Im Anschluss daran wird mit der Funktion `calculate_all_IG` der Informationsgewinn für jedes Attribut berechnet, wobei das Attribut mit dem höchsten Wert (`best_attribute`) als (Wurzel-) Knoten gewählt wird. Im Anschluss daran wird in Zeile 10 `best_attribute` aus der Menge aller Attribute entfernt.

Nun beginnt der iterative Teil der Funktion. Dabei wird über alle Ausprägungen des Attributes `best_attribute` eine Teilmenge `subset` des ursprünglichen Datensatzes erstellt. Danach wird die dritte Abbruchbedingung für die Rekursion geprüft, nämlich ob die Teilmenge `subset` leer ist. In diesem Fall wird ein Blatt erstellt welchem der Modalwert `modal_value` des Zielattributes zugeordnet wird. Wenn allerdings die Teilmenge `subset` nicht leer ist, so wird in Zeile 18 rekursiv ein Teilbaum erstellt. Dieser wird in Zeile 19 der Ausprägung `value` des Attributes `best_attribute` zugewiesen.

```

1      def ID3(data_set, target_attribute, attributes):
2          if len(np.unique(data_set[target_attribute])) <= 1:
3              return np.unique(data_set[target_attribute])[0]
4          if len(attributes) <= 1:
5              return modal(data_set[target_attribute])
6
7          IG = calculate_all_IG(data_set, target_attribute, attributes)
8          best_attribute = max(IG, key=lambda k: IG[k])
9          tree = {best_attribute: {}}
10         attributes = [x for x in attributes if x != best_attribute]
11
12         for value in np.unique(data_set[best_attribute]):
13             subset = data_set[data_set[best_attribute] == value]
14             if subset.empty:
15                 modal_value = modal(data_set[target_attribute])
16                 tree[best_attribute][value] = modal_value
17             else:
18                 subtree = ID3(subset, target_attribute, attributes)
19                 tree[best_attribute][value] = subtree
20         return tree

```

Abbildung 4.10: Hauptfunktion des ID3 Algorithmus [14, 12, 13]

5 Zusammenfassung

Entscheidungsbäume bieten diverse Vorteile gegenüber anderen Methoden. Einer der offensichtlichsten Vorteile ist die einfache Verständlichkeit und Interpretierbarkeit von Entscheidungsbäumen. [3] Sie können visualisiert werden und ermöglichen es somit auch Laien mit ihnen zu Arbeiten. [15]

Aber natürlich besitzen Entscheidungsbäume auch Einschränkungen. Sie sind relativ inkonsistent, da bereits kleine Änderungen innerhalb des Trainingsdatensatzes zu weitreichenden Veränderungen des Entscheidungsbaums führen können. [15]

Eine weitere Herausforderung ist dass die Entscheidungsbäume nicht zwangsweise optimal sind. Die Erstellung eines garantiert optimalen Entscheidungsbaums ist ein NP-vollständiges Problem [3], weshalb auch viele Implementationen einen gierigen Ansatz wählen. Dies bedeutet dass die verschiedenen Implementationen nach der lokal optimalen Entscheidung für jeden Knoten suchen. [15]

Sofern der Trainingsdatensatz zu speziell ist kann es bei einem Entscheidungsbäume zu sogenanntem Overfitting kommen. Dies bedeutet, dass der Entscheidungsbaum nicht gut generalisiert und zu stark von dem Trainingsdatensatz abhängig ist. In diesem Fall würden reale Daten die signifikant von dem Trainingsdatensatz abweichen falsch klassifiziert werden. [3] Diesem Problem kann mittels Pruning entgegengewirkt werden.

Literatur

- [1] *Entscheidungsbaum – Wikipedia*, <https://de.wikipedia.org/wiki/Entscheidungsbaum>, (Accessed on 05/20/2021).
- [2] P. S. B. Kamiński M. Jakubczyk, *A framework for sensitivity analysis of decision trees*, 2017.
- [3] M. Schinck, *Data Mining Vorlesung 2: Classification 1, Einführung, Validierung und Decision Trees*, 2021.
- [4] O. M. L. Rokach, *Top-down induction of decision trees classifiers - a survey*, 2005.
- [5] J. Quinlan, *Induction of Decision Trees*, 1986.
- [6] „Entropy (information theory) - Wikipedia.“ (), Adresse: [https://en.wikipedia.org/wiki/Entropy_\(information_theory\)](https://en.wikipedia.org/wiki/Entropy_(information_theory)).
- [7] S. V. Rupali Bhardwaj, *Implementation of ID3 Algorithm*, CSE, Bahra Univerity India, 2013.
- [8] *Information gain in decision trees - Wikipedia*, (Accessed on 05/27/2021). Adresse: https://en.wikipedia.org/wiki/Information_gain_in_decision_trees.
- [9] *Conditional entropy - Wikipedia*, https://en.wikipedia.org/wiki/Conditional_entropy, (Accessed on 05/27/2021).
- [10] „Bedingte Entropie – Wikipedia.“ (), Adresse: https://de.wikipedia.org/wiki/Bedingte_Entropie.
- [11] K. Jearanaitanakij, *Classifying Continuous Data Set by ID3 Algorithm*, 2005.
- [12] „ID3 algorithm - Wikipedia.“ (), Adresse: https://en.wikipedia.org/wiki/ID3_algorithm.
- [13] *Machine Learning with Python: Decision Trees in Python*, https://www.python-course.eu/Decision_Trees.php, (Accessed on 05/21/2021).
- [14] *Python max()*, <https://www.programiz.com/python-programming/methods/built-in/max>, (Accessed on 05/21/2021).
- [15] „Decision tree learning - Wikipedia.“ (), Adresse: https://en.wikipedia.org/wiki/Decision_tree_learning.