

Entscheidungsbäume als Datenstruktur

Prof. Dr. Jörg Frochte

Maschinelles Lernen

...

$$\|x\| = 0 \Rightarrow x = \mathbf{0}$$

$$\|\alpha \cdot x\| = |\alpha| \cdot \|x\|$$

$$\|x + y\| \leq \|x\| + \|y\|$$

...

Hochschule Bochum
Bochum University
of Applied Sciences
Campus **Velbert/Heiligenhaus**



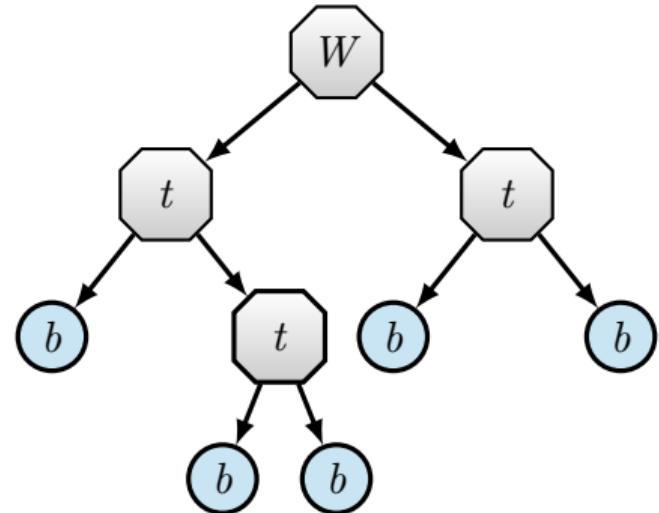
- Entscheidungsbäume basieren auf der – in der Informatik bekannten und grundlegenden – Datenstruktur eines Baumes.
- Wie ein solcher Baum aussehen soll, kann man mit Hilfe des maschinellen Lernens bestimmen.
- Die entsprechenden Bäume können dann sowohl für die Klassifikation als auch für die Regression eingesetzt werden.
- Etwas ungewohnt mag es sein, dass der Baum in der Informatik quasi um 180° gedreht dargestellt wird.



Metrosideros excelsa (New Zealand Christmas Tree)
Eigene Aufnahme Nordinsel Neuseeland

Ein binärer Baum als Datenstruktur

- Ein Entscheidungsbaum hat eine Wurzel W (engl. root), an der die Auswertung beginnt.
- Am Ende steht ein Blatt b (engl. leaf), welches der einzige Knoten ist, an dem keine Entscheidung mehr gefällt, sondern ein Zustand klassifiziert bzw. ein Regressionswert ausgegeben wird.
- An jedem anderen Knoten t – ebenso wie an W – wird eine Entscheidung getroffen und der Baum entsprechend durchlaufen.
- Ein solcher Baum heißt binär, wenn von jedem Knoten genau zwei Äste abgehen.
- Diese Form ist besonders einfach zu programmieren, jedoch können auch andere Bäume gelernt werden.

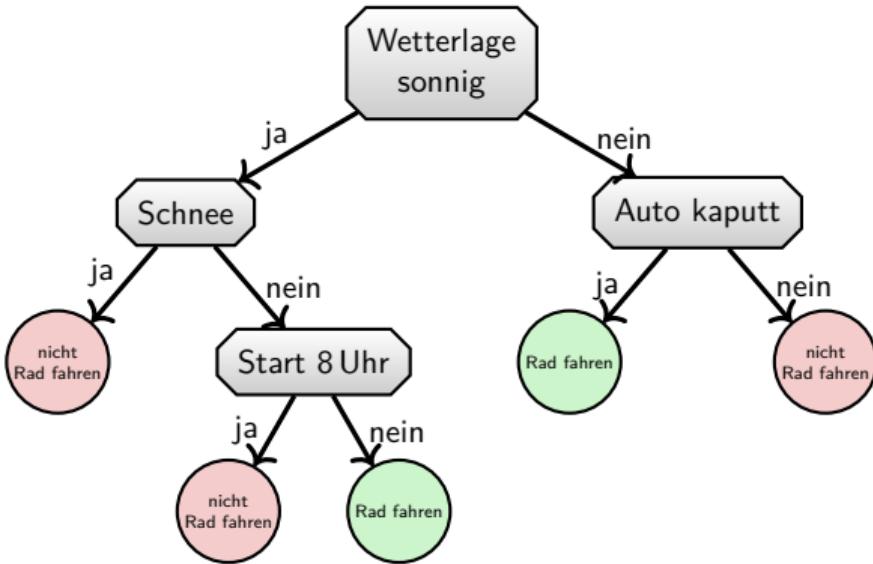


Beispieldatensätze für den Entscheidungsbaum

Nr.	Wetterlage sonnig?	Schnee	Auto kaputt	Start 08:00	Rad fahren
1	sonnig	Nein	Nein	Nein	Ja
2	Regen	Nein	Nein	Nein	Nein
3	Regen	Nein	Ja	Nein	Ja
4	Regen	Nein	Nein	Nein	Nein
5	sonnig	Nein	Ja	Nein	Ja
6	Regen	Nein	Ja	Ja	Ja
7	sonnig	Nein	Nein	Ja	Nein
8	sonnig	Ja	Nein	Nein	Nein
9	sonnig	Ja	Ja	Nein	Nein
:	:	:	:	:	:

Beispielanwendung

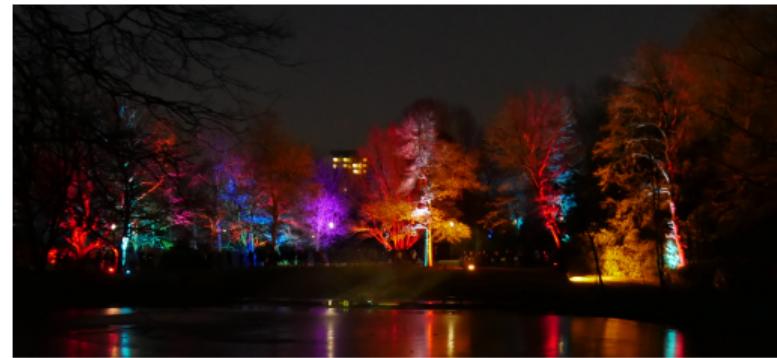
- Es gibt vier Merkmale für den Baum:
 $x = [\text{Wetterlage sonnig}, \text{Auto kaputt}, \text{Schnee}, \text{Start 8 Uhr}]$
- Jedes ist binär, also mit true, false codiert.
- Das Ziel ist es immer, einen Baum zu bauen, der möglichst klein ist und gleichzeitig möglichst viele Fälle der Datenbank richtig entscheidet.



Entscheidungsbaum für die Radfahren-Entscheidung

Wann ist ein Baum optimal?

- Die Frage ist, was versteht man bei einem Baum unter *klein*?
- Hierzu gibt es vier recht populäre Eigenschaften:

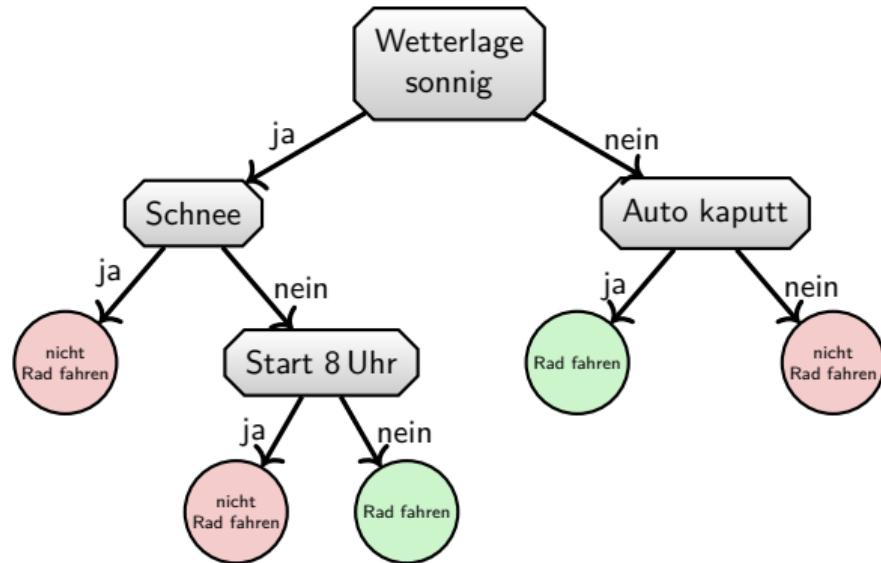


Parkleuchten im Grugapark (Essen)
Eigene Aufnahme

- ❶ **Anzahl der Blätter in einem Baum**
- ❷ **Baumhöhe**, also der längste Pfad im Baum, den es von der Wurzel bis zu einem Blatt geben könnte.
- ❸ **Pfadlängensumme** oder **External Path Length**
- ❹ **Gewichtete Pfadlängensumme** oder **Weighted external path length**

Pfadlängensumme

- Zur Berechnung der *External Path Length* beginnen wir an jedem Blatt und laufen von hier zur Wurzel. Dabei zählen wir die Kanten, die wir passieren.
- Die Summe über alle bei diesen Rückwärtsläufen ermittelten Zahlen ist eben die *External Path Length*.
- In unserem Beispiel aus der Abbildung rechts ist der Wert dieses Merkmals 12.
- Etwas, worum sich die *External Path Length* bzw. *Pfadlängensumme* nicht kümmert, ist, wie oft ein Pfad durchlaufen wird.

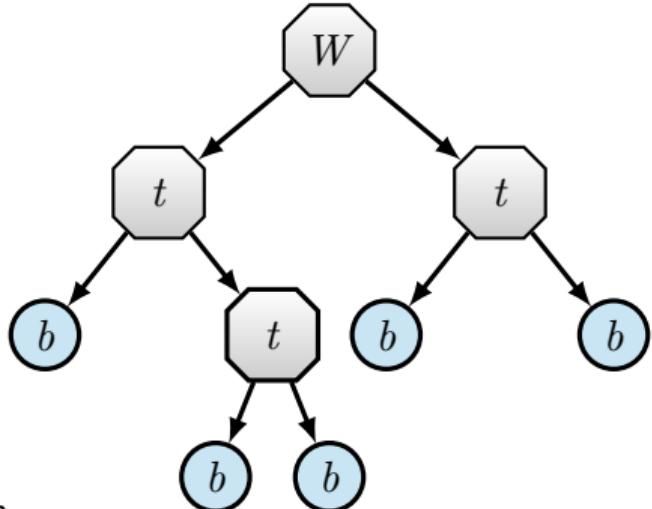


Gewichtete Pfadlängensumme

- Die Idee der *gewichteten Pfadlängensumme* ist, dass es auch auf den Anwendungsfall ankommt, wie schwer z. B. ein sehr tiefer Seitenarm wiegt.
- Wenn dieser für einen sehr seltenen Fall steht und kaum durchlaufen wird, ist es weniger wichtig, als wenn der Regelfall so tief ist.
- Um das beurteilen zu können, nimmt man die Trainingsmenge und hofft, dass diese repräsentativ für die spätere Anwendung ist.
- Wir nehmen hier einmal die ersten neun Einträge aus der Tabelle, um die gewichtete Pfadlängensumme zu bestimmen.
- Damit erhalten wir für unser Beispiel eine gewichtete Pfadlängensumme 21, eine Pfadlängensumme von 8, eine Baumhöhe von 3 und 5 Blätter.

Welche Methoden braucht man?

- Wichtige Bestandteile der Datenstruktur sind die hinterlegten Entscheidungen an den Knoten und die Werte der Leaf-Knoten.
- Leider kennt Python keine generische Baum-Klasse, sodass wir diese selber umsetzen müssen.
- Wenn man diese Klasse möglichst einfach halten will, benötigt man folgende Methoden:
 - addNode – die Möglichkeit, einen neuen Knoten hinzuzufügen
 - trace – die Möglichkeit zu verfolgen, wie eine Entscheidung entsteht bzw. wie der Weg durch den Baum war
 - eval – um den Ausgabewert des Baumes zu ermitteln.



Was tun wir nun damit?

- Wir werden uns im Folgenden den CART-Algorithmus ansehen.
- Mit diesem Ansatz ist es möglich, Bäume für Klassifikation und Regression zu lernen.
- Großer Vorteil dieser Methoden ist die Transparenz der Entscheidungen.
- Anschließend werden wir mehrere Bäume zu einem Random Forest kombinieren und so eine sehr gute Methode für strukturierte Daten bzgl. der Genauigkeit zu erhalten.
- Schauen Sie sich im Buch genau an, wie ein Baum in Python umgesetzt wird.



*Parkleuchten im Grugapark (Essen)
Eigene Aufnahme*