

# CART-KLASSIFIKATOR

PATTERN MATCHING & MACHINE LEARNING

F. FRETER, E. KIRCHBERGER,  
S. SYMHOVEN & J. WUSTL

SOMMERSEMESTER 2023

20. JUNI 2022



Hochschule München  
University of Applied Sciences  
Fakultät für Informatik und Mathematik

- 1 Training und Aufbau des Baumes
- 2 Bewertungsmaße für einen Split
- 3 Overfitting und Pruning
- 4 Vor- und Nachteile
- 5 Verbesserungsmöglichkeiten & Ausblick

# **TRAINING UND AUFBAU DES BAUMES**

# CART: CLASSIFICATION AND REGRESSION TREES

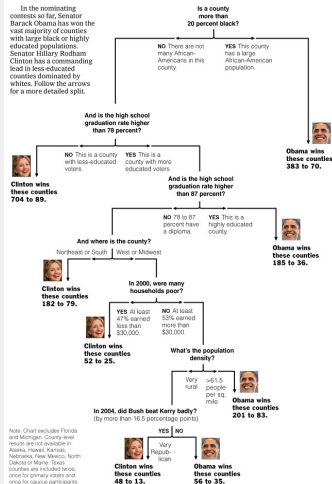
**CART-Algorithmen:** sind Binary-Decission Tree Verfahren, welches für **Klassifizierung** (kategorisch) und **Regression** (kontinuierlich) verwendet werden kann.

## Classification Trees

Im Folgenden fokussieren wir uns auf die **Classification Trees**.

### Decision Tree: The Obama-Clinton Divide

In the nominating contests so far, Senator Barack Obama has won the vast majority of counties with large black or highly educated populations. Senator Hillary Rodham Clinton has a commanding lead in less-educated counties dominated by whites. Follow the arrows for a more detailed split.



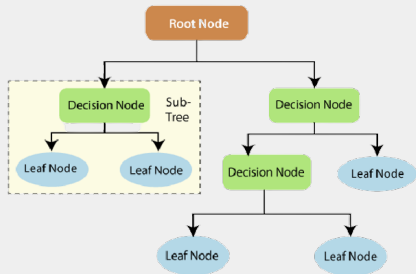
Sources: Election results via The Associated Press; Census Bureau; Dave Leip's Atlas of U.S. Presidential Elections

AMERICAN OVERSIGHT

Abbildung: Decision Tree [3]

# AUFBAU EINES CLASSIFICATION TREES

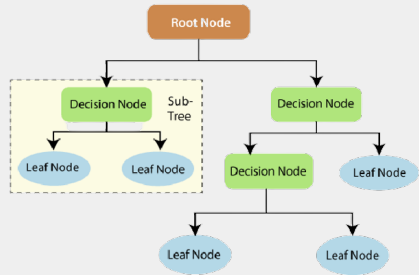
- **Root Node:** Startpunkt, enthält alle Daten und startet die Unterteilung (basierend auf Merkmalen mit Informationsgewinn).
- **Decision Node:** Teilt Daten weiter auf, basierend auf den restlichen Merkmalen.
- **Leaf Node:** Endpunkte repräsentieren finale Vorhersagen (basierend auf Merkmalen des gegebenen Datenpunkts). Keine weiteren sinnvollen Teilungen möglich.



**Abbildung:** Decision Tree [1]

# AUFBAU EINES CLASSIFICATION TREES

- **Root Node:** Startpunkt, enthält alle Daten und startet die Unterteilung (basierend auf Merkmalen mit Informationsgewinn).
- **Decision Node:** Teilt Daten weiter auf, basierend auf den restlichen Merkmalen.
- **Leaf Node:** Endpunkte repräsentieren finale Vorhersagen (basierend auf Merkmalen des gegebenen Datenpunkts). Keine weiteren sinnvollen Teilungen möglich.



**Abbildung:** Decision Tree [1]

## Ziel

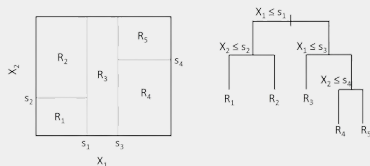
Optimale Vorhersagen auf Basis von Eingangsmerkmalen.

**Allgemeine Strategie:** Eingangsdaten werden in  $P$  disjunkte Regionen  $R_1, \dots, R_P$  aufgeteilt, wobei jede Region  $R_p$  eine Entscheidungsklasse  $Y_p$  repräsentiert. **Binary Splitting**, Beispiel:

$$x_i \leq a$$

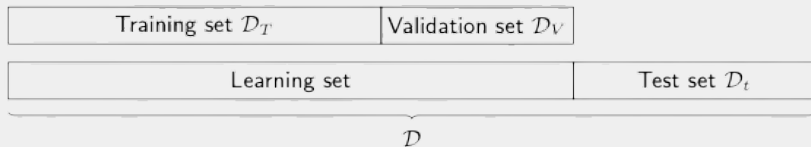
## Trainings Methode:

- Aufteilung des Ausgangsraums  $R$  in  $R_1$  und  $R_2$
- Suche nach der besten Aufteilung für  $R_1$  und  $R_2$
- Wiederhole für alle erzeugten Regionen



**Abbildung:** Rekursive Teilung [3]

# VORGEHEN BEI KLASSIFIKATION



**Abbildung:** Training- und Validation Set [?]

■ Do it



Um ein neues Sample  $X$  zu klassifizieren:

- Test der Attribute von  $X$  um die zutreffende Region zu finden für die Klassenverteilung  $y_r = (y_{c_1}, \dots, y_{c_k})$ .
- Die Wahrscheinlichkeit das ein Punkt  $X \in \mathcal{R}$  zu einer Gruppe gehört, ist definiert durch

$$p(y = c|\mathcal{R}) = \frac{y_{c, \mathcal{R}}}{\sum_{c_i \in \mathcal{C}} y_{c_i, \mathcal{R}}}.$$

- Ein neues Sample bekommt die Zuteilung welche am häufigsten in der jeweiligen Region ist.

$$\hat{y} = \arg \max_c p(y = c|x) = \arg \max_c p(y = c|\mathcal{R}) = \arg \max_c y_{c, \mathcal{R}}$$

## Problem: Wie finde ich den besten Split?

Direkte Optimierung ist schwer umsetzbar, da die iterative Überprüfung aller Bäume bei komplexeren Daten schnell explodiert.

Mögliche Lösung:

**Greedy Heuristic:** Bei jedem Schritt wird die aktuell optimale Entscheidung getroffen

Hierbei wird eine Node aufgeteilt, wenn sie den Misclassification Error (ME)  $i_E$  an Node  $t$  verbessert.

$$i_E(t) = 1 - \max_c p(y = c|t)$$

Die Verbesserung bei Durchführung eines Splitts  $s$  von  $t$  zu  $t_R$  und  $t_L$  für  $i(t) = i_E(t)$  ist wie folgt definiert:

$$\Delta i(s, t) = i(t) - p_L \cdot i(t_L) - p_R \cdot i(t_R)$$

# PROBLEME MIT MISCLASSIFICATION ERROR

**Problem 1:** Kein Splitt durchgeführt bei  $i_E(t) = \frac{40}{200}$ , obwohl verbesserte Klassifikation möglich, durch Kombination von Tests.

$$x_1 \leq 5 : p_L \cdot i_E(t_L) - p_R \cdot i_E(t_R) = \frac{40}{200}$$

$$x_2 \leq 3 : p_L \cdot i_E(t_L) - p_R \cdot i_E(t_R) = \frac{40}{200}$$

**Problem 2:** Schlechte Sensitivität zur Veränderung der Klassenwahrscheinlichkeiten. Verteilung vor Split: (400, 400)

$$\text{Split } a : \{(100, 300), (300, 100)\} \rightarrow i_E(t, a) = 0.25$$

$$\text{Split } b : \{(200, 400), (200, 0)\} \rightarrow i_E(t, b) = 0.25$$

**Lösung:** Ein Kriterium welches als Maß für die Reinheit der Klassenverteilung an Node  $t$  verwendet werden kann.

# BEWERTUNGSMASSE FÜR EINEN SPLIT

# BEWERTUNGSMASSE: GINI-INDEX, INFORMATIONSGEWINN & MISSCLASSIFICATION ERROR

## ■ Gini-Index

- ▶ Maß der Unreinheit einer Gruppe
- ▶  $i_G(t) = 1 - \sum_{i=1}^k \pi_i^2$ , wobei  $\pi_i$  die Wkt. der Klasse  $i$  ist.
- ▶ **Ziel:** Minimierung des gewichteten Gini-Indexes.

## ■ Informationsgewinn: Entropy

- ▶ Reduktion der Entropie durch den Split
- ▶  $IG = H(\text{parent}) - \sum_{j=1}^m \frac{n_j}{n} H(\text{child}_j)$ , wobei  $H$  die Entropie ist.
- ▶ **Ziel:** Maximierung des Informationsgewinns.

## ■ Missclassification Error

- ▶ Der Misclassification Error (ME) ist ein Maß für die Fehlklassifizierung.
- ▶  $i_E(t) = 1 - \max_c p(y = c|t)$
- ▶ ME kann als Bewertungsmaß für die Baumkonstruktion verwendet werden.

Misst, wie oft eine zufällig ausgewählte Instanz falsch klassifiziert würde, wenn sie gemäß der Klassenverteilung zufällig klassifiziert wird.

- Die Gini-Unreinheit ist ein Wert zwischen 0 (vollständig rein, alle Elemente gehören zur gleichen Klasse) und 1 (maximal unrein, gleichmäßige Verteilung der Klassen).
- Die Gini-Unreinheit für einen Knoten  $t$  mit  $K$  Klassen kann wie folgt berechnet werden:

$$i_G(t) = \sum_{c_i \in C} \underbrace{\pi_{c_i}}_{\text{Wahrscheinlichkeit, ein Element auszuwählen}} \cdot \underbrace{(1 - \pi_{c_i})}_{\text{Wahrscheinlichkeit, dass es falsch klassifiziert wird}}$$

# OVERFITTING UND PRUNING



# OVERFITTING IN DECISION TREES

Daten werden rekursiv aufgeteilt und lassen den DT dadurch wachsen. Wann sollte man das Wachstum stoppen um overfitting zu vermeiden?

## Mögliche Stop- (oder Pruning-) Kriterien:

- Verteilung im Ast ist rein, d.h.  $i(t) = 0$
- Maximale Tiefe erreicht
- Anzahl der Proben in jedem Ast unterhalb eines bestimmten Schwellenwerts  $t_n$
- Nutzen der Aufteilung ist unterhalb eines bestimmten Schwellenwerts  $\Delta i(s, t) < t\Delta$
- Genauigkeit auf dem Validierungsset
- **Ziel:** Erstellung eines Modells, das gut auf neue, ungesehene Daten verallgemeinert und somit Overfitting vermeidet.

Alternativ kann der Baum zunächst vollständig wachsen und anschließend beschnitten werden (**Post-Pruning**).

## Verschiede Pruning-Methoden:

- Reduced Error Pruning
- Minimum Description Length Pruning
- Cost-Complexity Pruning

# COST-COMPLEXITY PRUNING

- **Ziel:** Verhindern von Overfitting durch Entfernen von Zweigen, die wenig zur Vorhersageleistung beitragen
- **Kostenkomplexitätspruning:** Gleichgewicht zwischen Baumgröße und Trainingsfehler
- **Kostenkomplexitätskriterium:**

$$C_{\alpha}(T) = C(T) + \alpha|T|, \text{ mit}$$

- ▶  $C(T)$  ist der Misclassification Error des Baumes  $T$ .
  - ▶  $|T|$  ist die Anzahl der terminalen Knoten des Baumes  $T$ .
  - ▶  $\alpha$  ist ein Komplexitätsparameter, der die Präferenz zwischen Baumgröße und Trainingsfehler steuert.
- Durch Variieren von  $\alpha$  kann eine Sequenz optimaler Bäume ermittelt werden.
  - Kreuzvalidierung kann verwendet werden, um den optimalen Wert von  $\alpha$  zu bestimmen.

# **VOR- UND NACHTEILE**

## **Vorteile:**

- leicht zu trainieren
- leicht zu interpretieren
- einfach zu visualisieren
- können mit verschiedenen Prädiktoren umgehen  
→ keine Dummies erforderlich

## **Nachteile:**

- nicht die besten Lerner
- reagieren empfindlich auf sich ändernde Trainingsdaten
- werden von den oben genannten Splits dominiert  
→ erster Split beeinflusst stark die Form des gesamten Baums

# **VERBESSERUNGSMÖGLICHKEITEN & AUSBLICK**

- **Stacking:** Ensemble-Lern-Technik. Mehrere CART-Modelle kombiniert werden. Ausgaben der einzelnen Modelle werden als Eingabe für ein Meta-Modell verwendet.
- **Bayesian Model Averaging:** Modellselektion. Mehrere Modelle auf der Grundlage von Bayes'schen Wahrscheinlichkeiten kombiniert werden.
- **Bagging:** Ensemble-Lern-Technik. Mehrere CART-Modelle werden auf unterschiedlichen Stichproben der Daten trainiert.
- **Random Forests:** Ensemble-Lern-Modell. Besteht aus vielen unkorrelierten Entscheidungsbäumen, die auf zufälligen Untergruppen der Daten trainiert werden.
- **Boosting:** Ensemble-Lern-Technik. Sequentielle Anordnung von schwachen CART-Modellen, wobei jeder Baum versucht, die Fehler des vorherigen Baums zu korrigieren.

# REFERENCES I



BAHZAD CHARBUTY AND ADNAN MOHSIN ABDULAZEEZ.  
**CLASSIFICATION BASED ON DECISION TREE ALGORITHM FOR MACHINE LEARNING.**

*Journal of Applied Science and Technology Trends, 2021.*



PROF. DR. STEPHAN GÜNNEMANN.  
**MACHINE LEARNING. LECTURE 2: DECISION TREES, 2021.**



**TREVOR HASTIE, ROBERT TIBSHIRANI, AND JEROME FRIEDMAN.**  
**THE ELEMENTS OF STATISTICAL LEARNING, 2009.**



FRAGEN, KRITIK ODER ANREGUNGEN?