Entscheidungsbäume

Seminar Künstliche Neuronale Netze

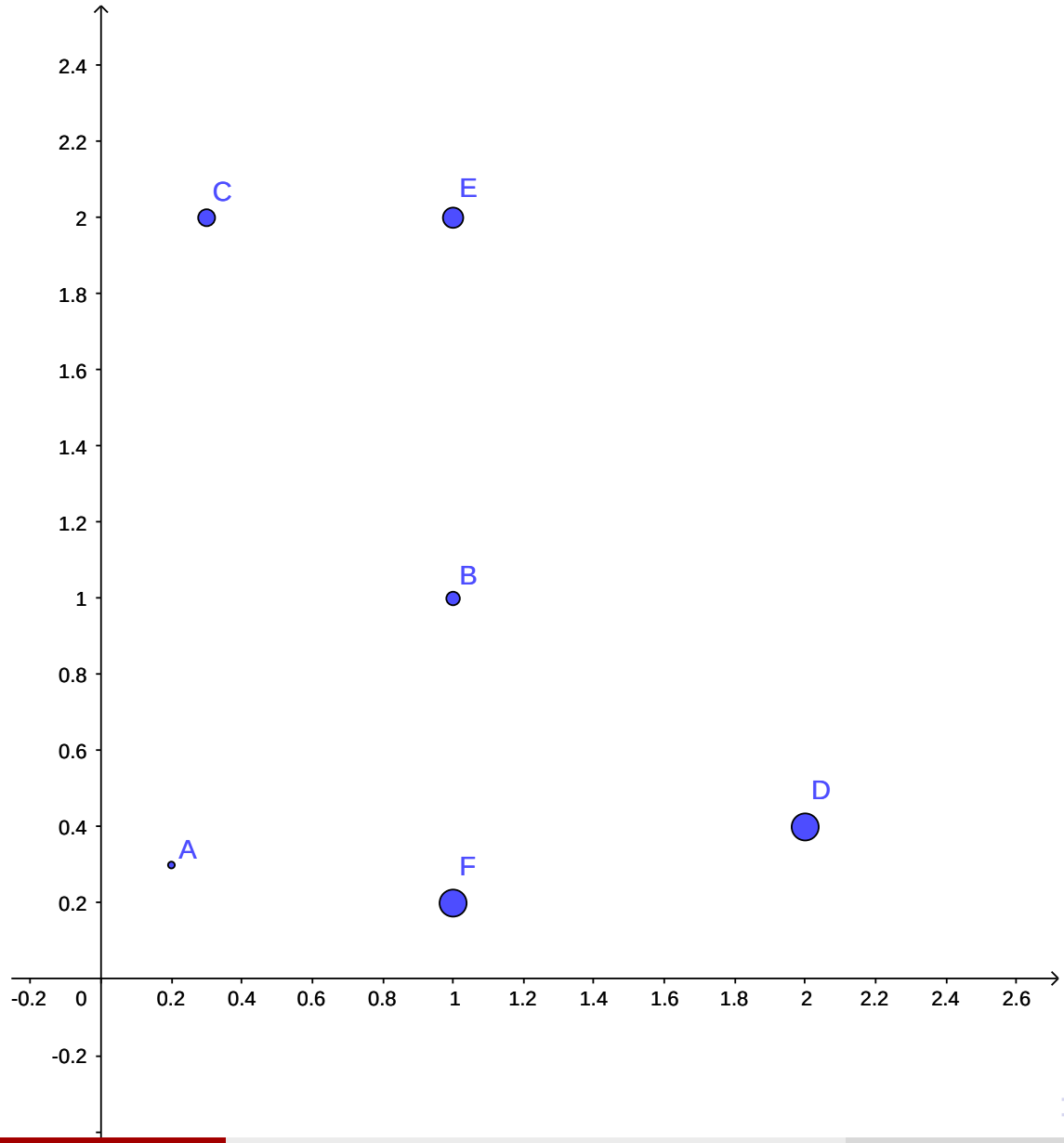
Ina Geller

16. November 2018

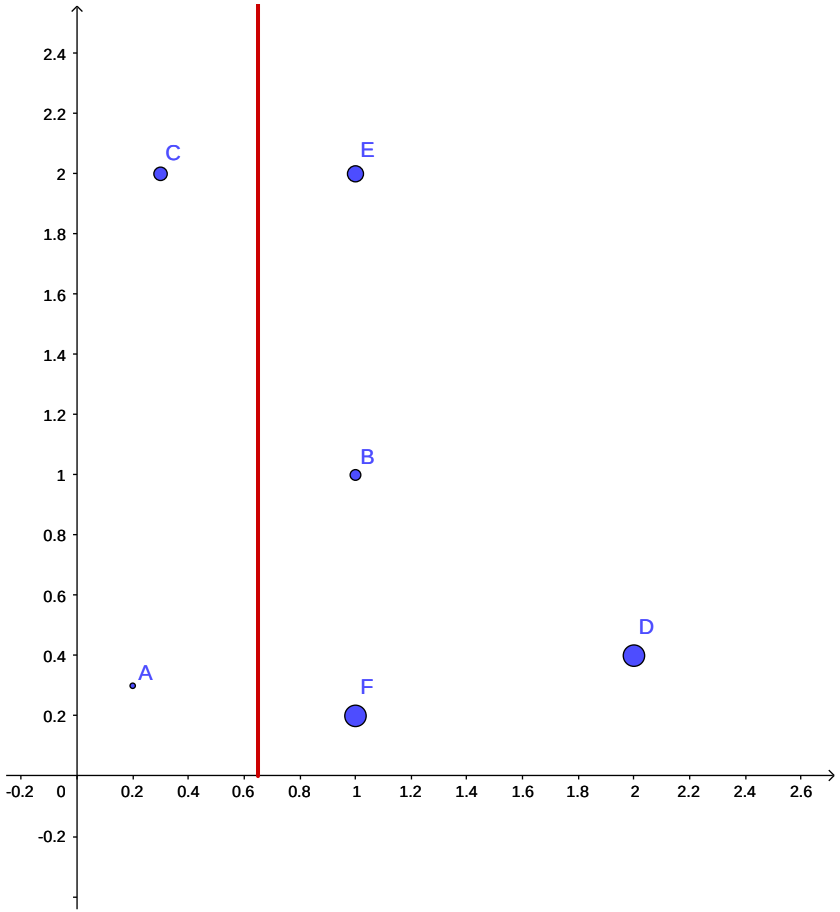
Motivation

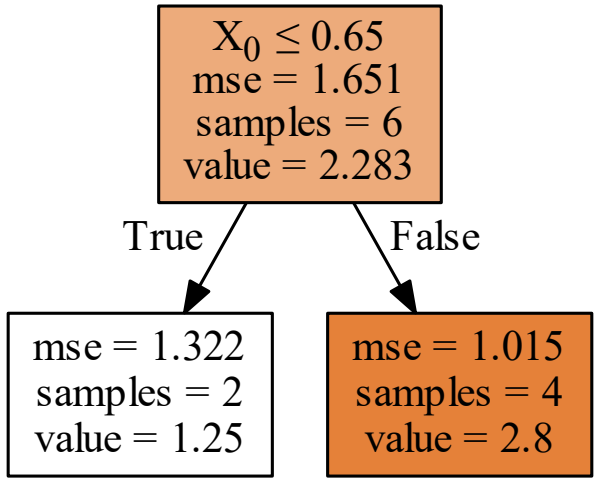
- Anwendung bei Regressions- und Klassifizierungsproblemen
- sehr einfach zu verstehen und zu programmieren
- Ergebnisse leicht zu interpretieren
- Funktionsweise: Restriktionen auf den Einflussgrößen unterteilen den Vorhersageraum in einfache Bereiche, Mittelwerte der Zielgrößen in diesen Bereichen dienen als Vorhersagewerte

Zusammenhang Bäume & Bereiche im Vorhersageraum

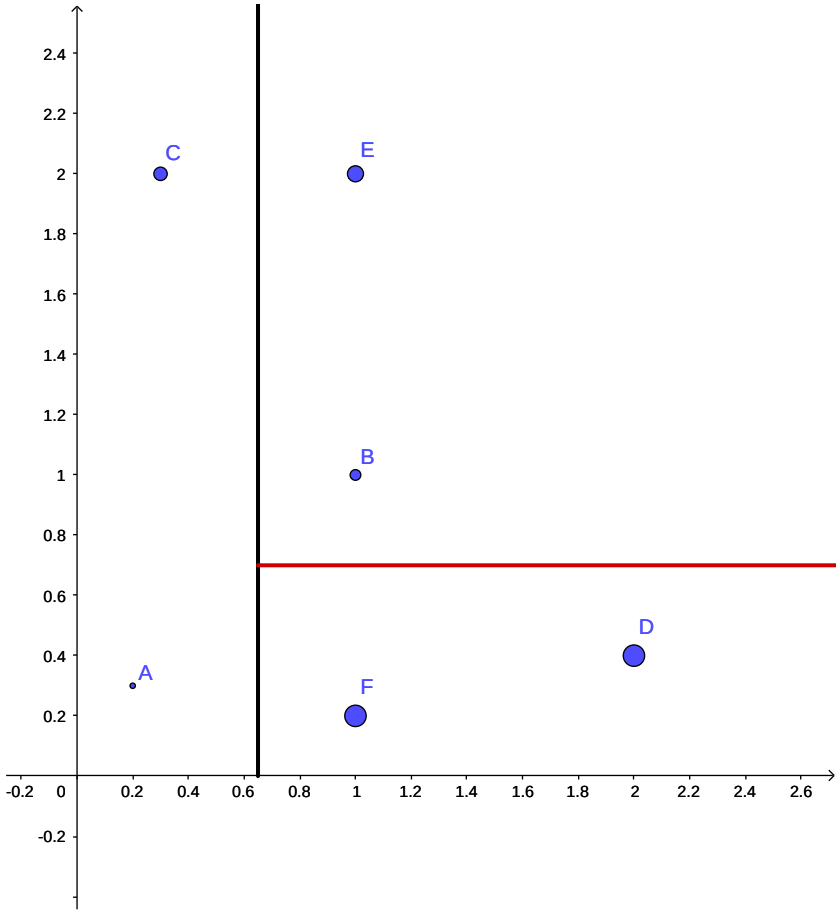


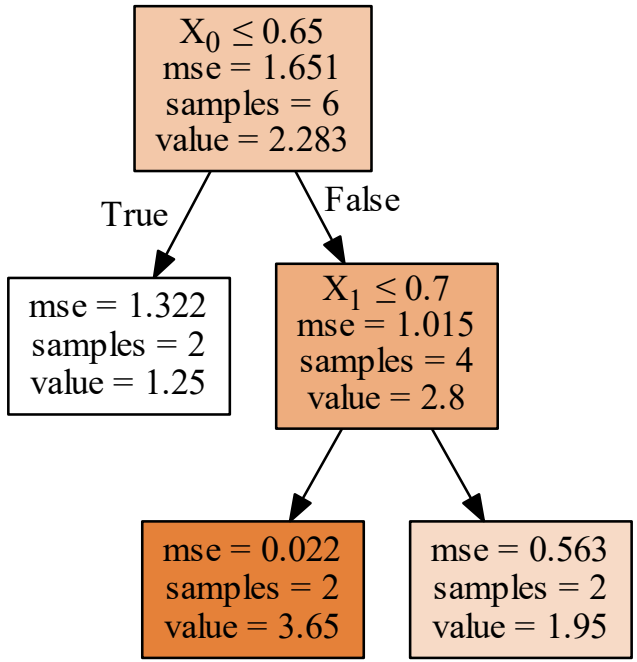
Zusammenhang Bäume & Bereiche im Vorhersageraum



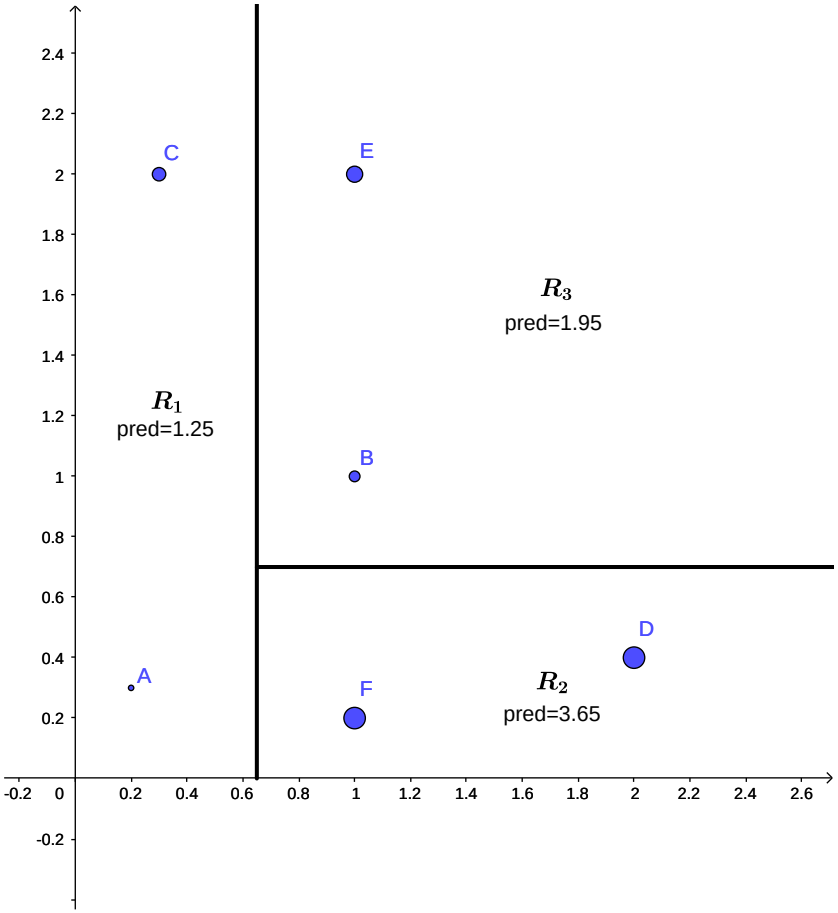


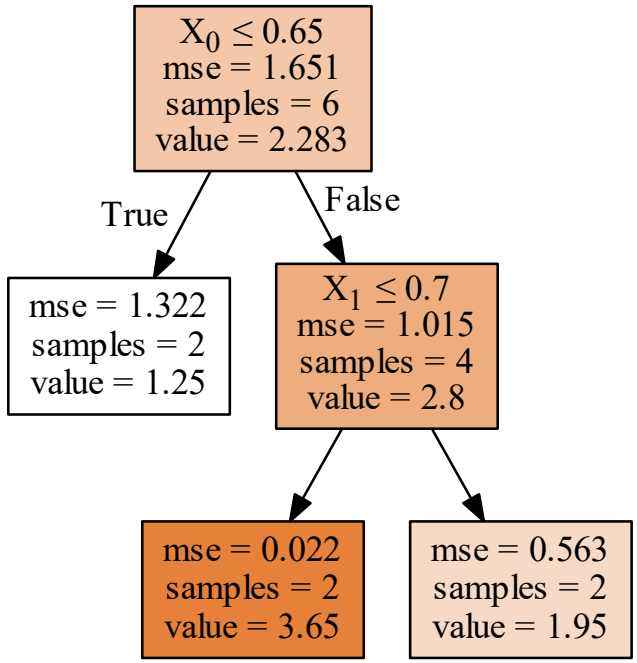
Zusammenhang Bäume & Bereiche im Vorhersageraum





Zusammenhang Bäume & Bereiche im Vorhersageraum





Bestimmung der Restriktionen

- Finde für jede Einflussgröße X_j
- die beste Schnittgerade durch s
- Wähle j und s mit der besten Vorhersage
- Wiederhole dieses Verfahren in den beiden gefundenen Bereichen

Bestimmung der besten Vorhersage

Wie bei linearen Modellen:

$$(j, s) = \underset{j,s}{\text{argmin}} \sum_{i:x_i \in R_1(j,s)} (y_i - \hat{y}_{R_1})^2 + \sum_{i:x_i \in R_2(j,s)} (y_i - \hat{y}_{R_2})^2$$

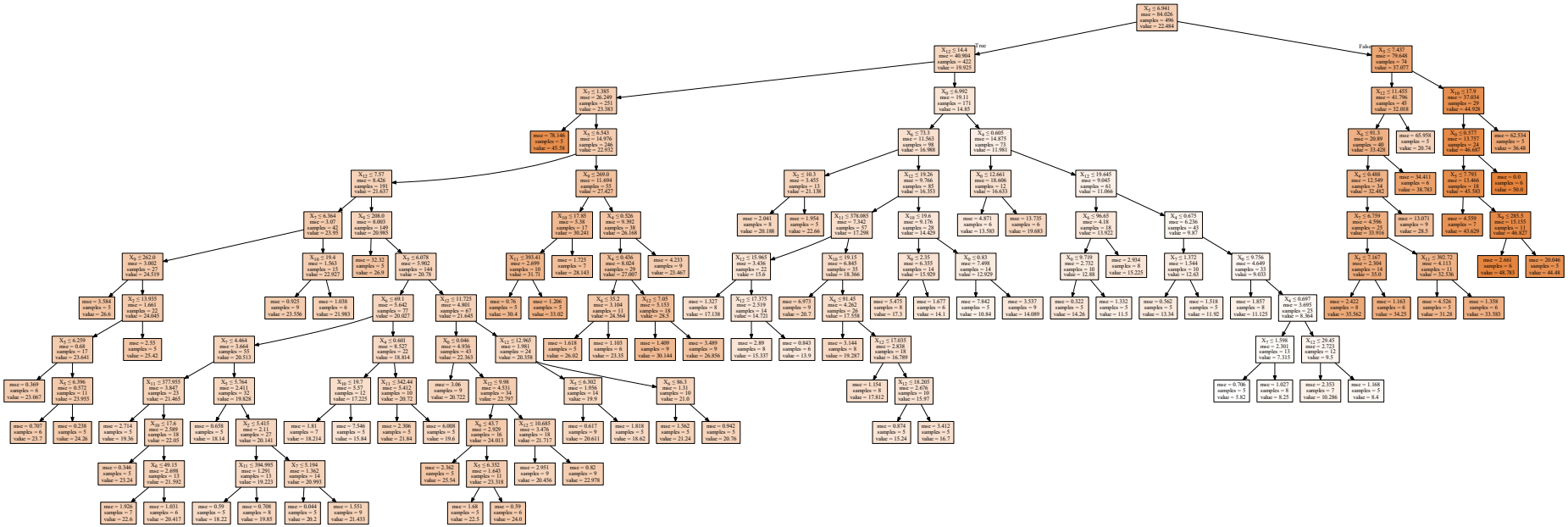
Man erhält die Halbebenen $R_1(j, s) = \{X | X_j \leq s\}$ und $R_2(j, s) = \{X | X_j > s\}$.

$\hat{y}_{R_n}$ ist der Mittelwert der Zielgrößen in Region n

Eigenschaften des Algorithmus

- Top-down-Prinzip
- Greedy
- rekursive binäre Aufteilung

Überangepasster Baum



Nachteile

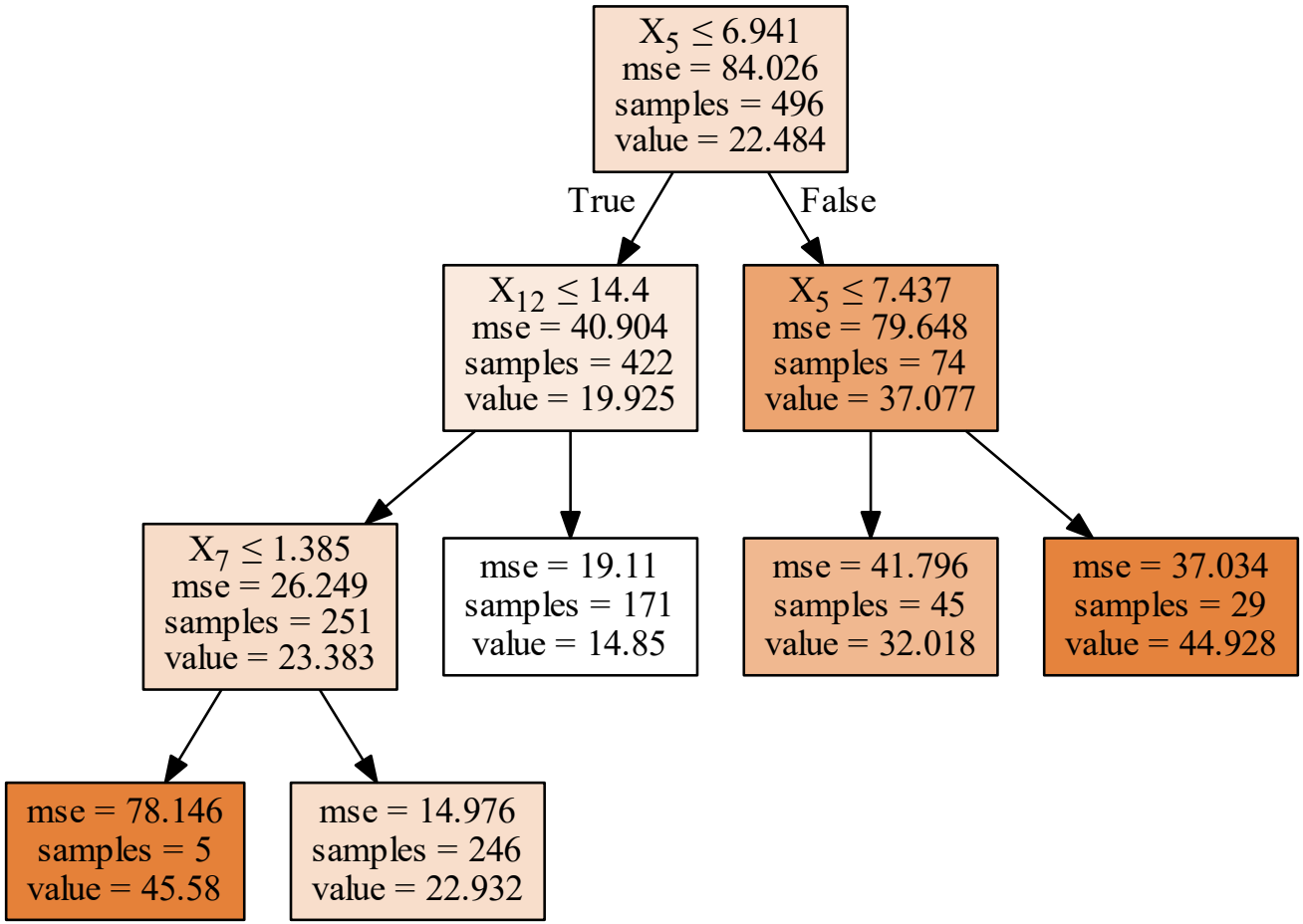
- Überanpassung an die Trainingsdaten, schlechte Vorhersagen auf dem Testset
- schlechte Performance, zu viele Berechnungen um ein Blatt zu erreichen
- Unübersichtlichkeit und schlecht zu interpretieren

Möglichkeiten

- Stoppkriterien wie maximale Tiefe, Mindestanzahl an Beobachtungen pro Blatt etc.
- nur verzweigen, wenn RSS-Verbesserung einen Grenzwert überschreitet
- zunächst den ganzen Baum berechnen und nachträglich ineffiziente Äste wieder abschneiden (pruning)

Kürzere Bäume verringern die Komplexität, aber erhöhen die systematische Verzerrung (Bias-Variance Trade-off)

Kürzerer Baum



Algorithmus Pruning

1. Bilde großen Baum auf den Trainingsdaten.
2. Wende Cost Complexity Pruning an, um eine Folge an besten Teilbäumen in Abhängigkeit von α zu erhalten.
3. Wende Kreuzvalidierung an, um das beste α zu bestimmen.
4. Gib den Teilbaum aus 2. aus, der zu dem in 3. bestimmten α gehört.

Cost Complexity Pruning

Jedem $\alpha > 0$ entspricht ein Teilbaum $T_\alpha \subset T_0$. Wir bestimmen eine Folge an Teilbäume über

$$\min \sum_{m=1}^{|T_\alpha|} \sum_{i:x_i \in R_m} (y_i - \hat{y}_{R_m})^2 + \alpha |T_\alpha|$$

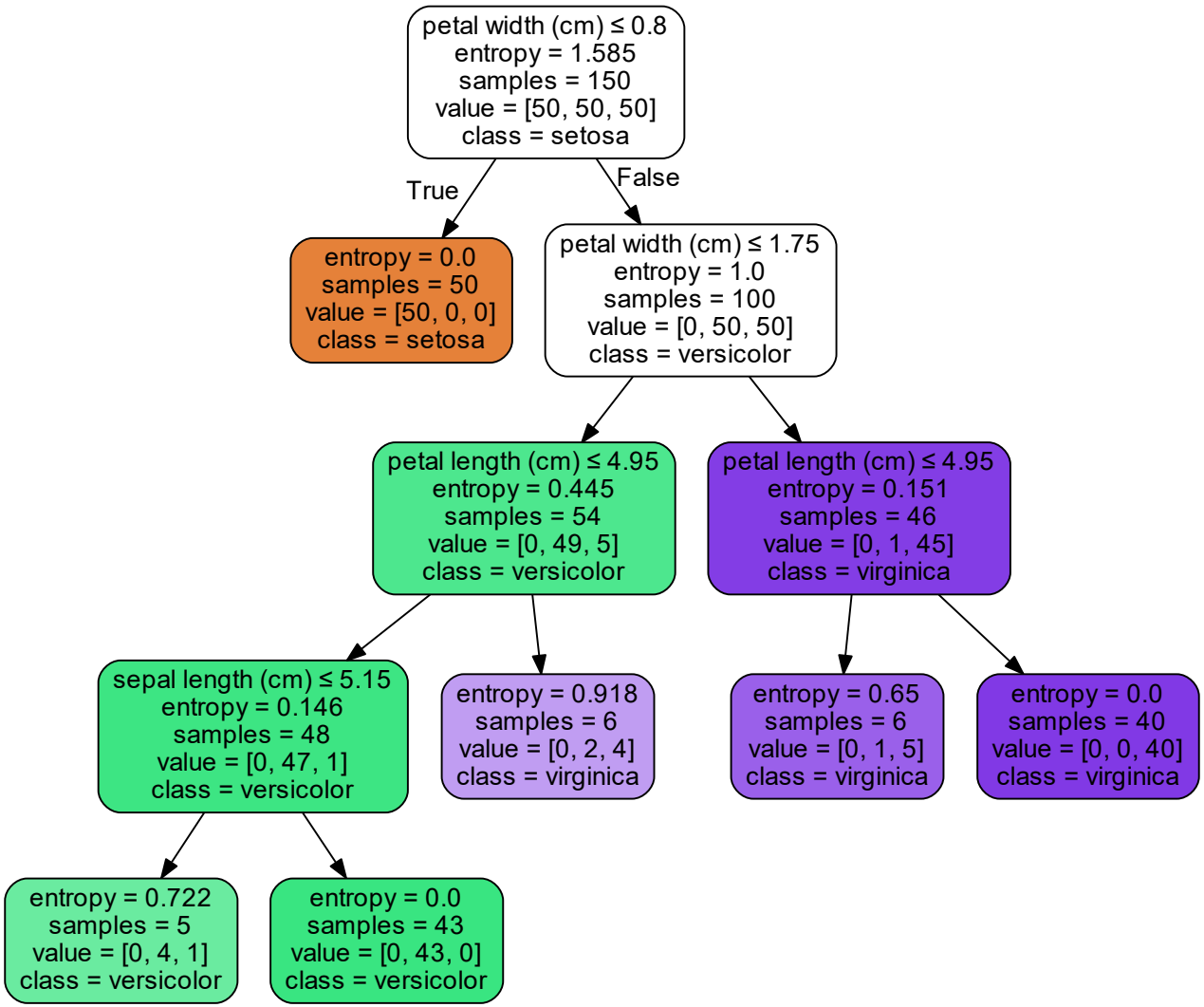
$|T_\alpha|$ ist die Anzahl an Blättern in T_α
R_m ist das Rechteck, das dem m-ten Blatt entspricht
$\hat{y}_{R_m}$ ist die Vorhersage für alle Daten, die in R_m fallen

$\Rightarrow$ Wenn man α von 0 an wachsen lässt, werden nach und nach einzelne Äste weggeschnitten.

Klassifizierungsbäume

- ähnlich wie Regressionsbäume, nur für qualitative Zielgrößen
- Vorhersage ist die in der Region am häufigsten vorkommende Klasse
- Nicht nur Vorhersage, sondern auch Aufteilung der Klassen in den einzelnen Regionen interessant

Irisblumenklassifizierung



Definitionen

Anzahl Beobachtungen pro Klasse:

$$\hat{b}_{mk} = \sum_{b \in m} \epsilon_{bk}$$

Blatt m
mit zugehöriger Region R_m
Beobachtung b
Klasse k und

$$\epsilon_{bk} = \begin{cases} 1 & ,wenn\ \ b\ \ in\ \ k \\ 0 & ,sonst \end{cases}$$

Definitionen

Anteil von Klasse k in m ist:

$$\hat{p}_{mk} = \frac{p_{mk}}{\sum_k \hat{b}_{mk}}$$

Vorhersage für Region m:

$$k = \underset{k}{\operatorname{argmax}}\, \hat{b}_{mk}$$

Methoden zum Finden der besten Schnittgeraden

- Klassifikationsfehlerrate: Anteil falsch klassifizierter Beobachtungen pro Blatt
- Gini-Index: totale Varianz
- Entropy: Informationsgehalt

Gini-Index

$$G = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk})$$

- G klein, wenn alle $\hat{p}_{mk}$ entweder 0 oder 1
- Berechnet totale Varianz über alle Blätter
- Gibt die Reinheit (purity) des Knotens an

Entropy

$$D = -\sum_{k=1}^{K} \hat{p}_{mk} \; log(\hat{p}_{mk})$$

- D nimmt sehr kleine Werte nahe 0 an, wenn die Klassen möglichst rein sind
- Spaltet Bereiche auch nach Reinheit der Knoten
- führt zu ähnlichen Ergebnissen wie Gini-Index

Vergleich zu linearen Modellen