

4.2) Bagging, Random Forests & Boosting

Decision Trees haben viele Vorteile, aber auch ein paar Nachteile!

Einer davon ist, daß sie eine große Varianz haben, das heißt:

☒ Das Ergebnis hängt sehr stark vom
(zufälligen) Train-Test-Split ab

- ⇒
 - Modell hängt an den Daten (rein datengetrieben)
 - keine funktionale Modellstruktur

Eine Strategie um dieses Problem abzuwenden
ist die sogenannte

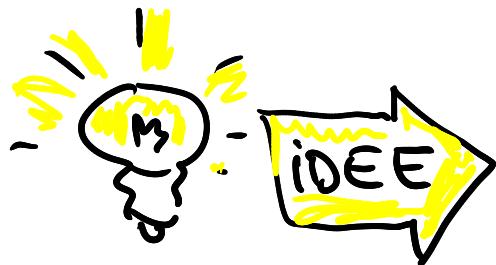
BOOTSTRAP AGGREGATION auch

BAGGING genannt

4.2.1) Bagging

254

Bagging kann nicht nur für Decision Trees verwendet werden, sondern funktioniert auch bei anderen Verfahren



Aus der Statistik:

Betrachte unabhängige Zufallsvariablen Z_1, \dots, Z_n , die alle die gleiche Varianz σ^2 haben.

$$\Rightarrow \frac{1}{n} \sum_{i=1}^n Z_i = \bar{Z}$$

ist wieder eine ZV mit Varianz

$$\frac{1}{n} \sigma^2 < \sigma^2$$

⇒ Mittelwertbildung reduziert die Varianz! 

"Übertragung auf ML-Vorfahren:

155

- ziehe viele verschiedene Trainingsmengen
- bilde für jede dieser Mengen ein Modell
- mittle die resultierenden Prognosen

In der Praxis heißt das



Dieser Algorithmus wird
BAGGING genannt (allgemein)

$$\hat{f}_{\text{bag}}(x) = \frac{1}{B} \sum_{b=1}^B \hat{f}^{*b}(x)$$

The final step shows the formula for the "Finales Modell": $\hat{f}_{\text{bag}}(x) = \frac{1}{B} \sum_{b=1}^B \hat{f}^{*b}(x)$. A small fire icon is positioned below the formula, and a red circle with a cross is placed above it.

Bemerkung:

- ① Für Decision Trees ist es sinnvoll, beim Bagging tiefe Strukturen zu erlauben.
Jedes der Modelle hat dann einen kleinen Bias und eine hohe Varianz. Die Varianz wird aber durch die Mittelwertbildung deutlich reduziert!
 - ② Die Mittelwertbildung im ~~*~~ funktioniert bei Regression Trees. Bei Klassifikationsproblemen tritt eine Wahrheitentscheidung an die Stelle des Mittelwerts.
 - ③ Die Anzahl B der Einzelmödelle muss festgelegt werden.
Es zeigt sich aber, dass ein großes B nicht zu overfitting führt
⇒ unkritisch
- Fäustregel: B so groß, dass sich keine weitere Verbesserung auf der Testmenge zeigt
signifikante!



Wie lässt sich der Testfehler eines Bagging-Modells sinnvoll abschätzen?

Grossvalidation wäre eine Möglichkeit.
... es geht aber auch billiger!

Überlegung: Ziehen mit Zurücklegen \Rightarrow

\Rightarrow Die Wahrscheinlichkeit nicht ausgewählt zu werden beträgt für ein Sample

$$p = 1 - \frac{1}{m}$$

Bootstrapping

Stichprobenwiederholung
für eine Stichprobe der Größe n

⇒ n mal nicht ausgewählt: $(1 - \frac{1}{n})^n$

258

⇒ Grenzwert $n \rightarrow \infty$:

$$\lim_{n \rightarrow \infty} \left(1 - \frac{1}{n}\right)^n = e^{-1} \approx 0,368 = 1 - 0,632$$

Das bedeutet im Kontext:

Eine Stichprobenwiederholung enthält
(im Schnitt) nur ungefähr 63% der
zugrunde liegenden Datenpunkte

← wegen
"Zurücklegen"

... erstaunlich, aber wahr!!

Für uns heißt das:

jeder Tree $\hat{f}^{*b}(x)$ ($b=1, \dots, B$) sieht nur
63% der verfügbaren Datenpunkte !!

~~Def:~~: Die restlichen 37% der Datenpunkte heißen
Out-of-Bag (OOB) samples.

... die kann man üblicherweise zum Testen benutzen!



- 1) Berechne Modellprognosen für das i -te Sample
für diejenigen Modelle $\hat{f}^{*b}(x)$, für die genau
dieses Sample OOB ist $\approx \frac{1}{3} B$ Prognosen
für das i -te Sample
 - 2) bilde für dieses Sample den Mittelwert aus den
 $\frac{1}{3} B$ Prognosen (bzw. Mehrheitsentscheidung)
 - 3) wiederhole den Prozeß für $i=1\dots,n$ und
berechne damit den globalen RMSE (accuracy)
des Modells
- ⇒ OOB - Testfehler

Bemerkung: Wenn B groß genug ist, dann ist der OOB-Fehler bzw. die OOB-accuracy praktisch äquivalent zum Ergebnis einer LOOCV!

Vorteil: Man kriegt das Ganze quasi "geschenkt" ohne zusätzlich Splitten, modellieren zu müssen!

... das klingt so, als wäre das eine Top-Idee

ABER ...

Nachteile des Bagging:

- ① Decision Tree:
- einfach zu interpretieren
 - Rangierung der Input-Features möglich

Das ist beim Bagging jetzt nicht mehr ganz so einfach ...

⇒ Zunahme der Prognosegüte geht zu Lasten der Interpretierbarkeit

⇒ Für ein Ranking muss über alle B Spalten gemittelt werden

(Abnahme des GINI/Gini Index beim Split für eine bestimmte Variable ist ein Kriterium beim Ranking)

2

Die einzelnen Bäume sind natürlich hochgradig korreliert.

263

Das kann problematisch sein, denn

$$Z_1, \dots, Z_n \text{ UNABHÄNGIG} \Rightarrow \frac{1}{n} \sum_{i=1}^n Z_i = \bar{Z} \text{ hat Varianz } \frac{1}{n} \sigma^2 < \sigma^2$$

$$\text{Var}(Z_i) = \sigma^2$$

bei der Mittelwertbildung von stark korrelierten Größen wird die Varianz nicht mit $\frac{1}{n} \sigma^2 < \sigma^2$ kleiner werden, sondern deutlich weniger.

Also: gute Idee (theoretisch), aber in der Praxis nicht so grandios wie erhofft!

Entscheidende Frage jetzt ..



Lassen sich die Fälsche dekorrelieren?

Dazu müssten sich die einzelnen Modelle offensichtlich hinsichtlich ihrer Struktur stärker unterscheiden.

2 Möglichkeiten

- ① Unterschiedliche Prädiktoren
- ② Unterschiedliche Modellstruktur
(z.B. in Bezug auf Komplexität)

4.2.2) Random Forests

265



Benötigt für einen Split im Tree immer nur
einen Teil der verfügbaren
Prädiktoren!

Das erhöht bei vielen Bäumen die
Unterschiedlichkeit der Modelle

⇒ **DEKORRELATION !**

Algorithmus:

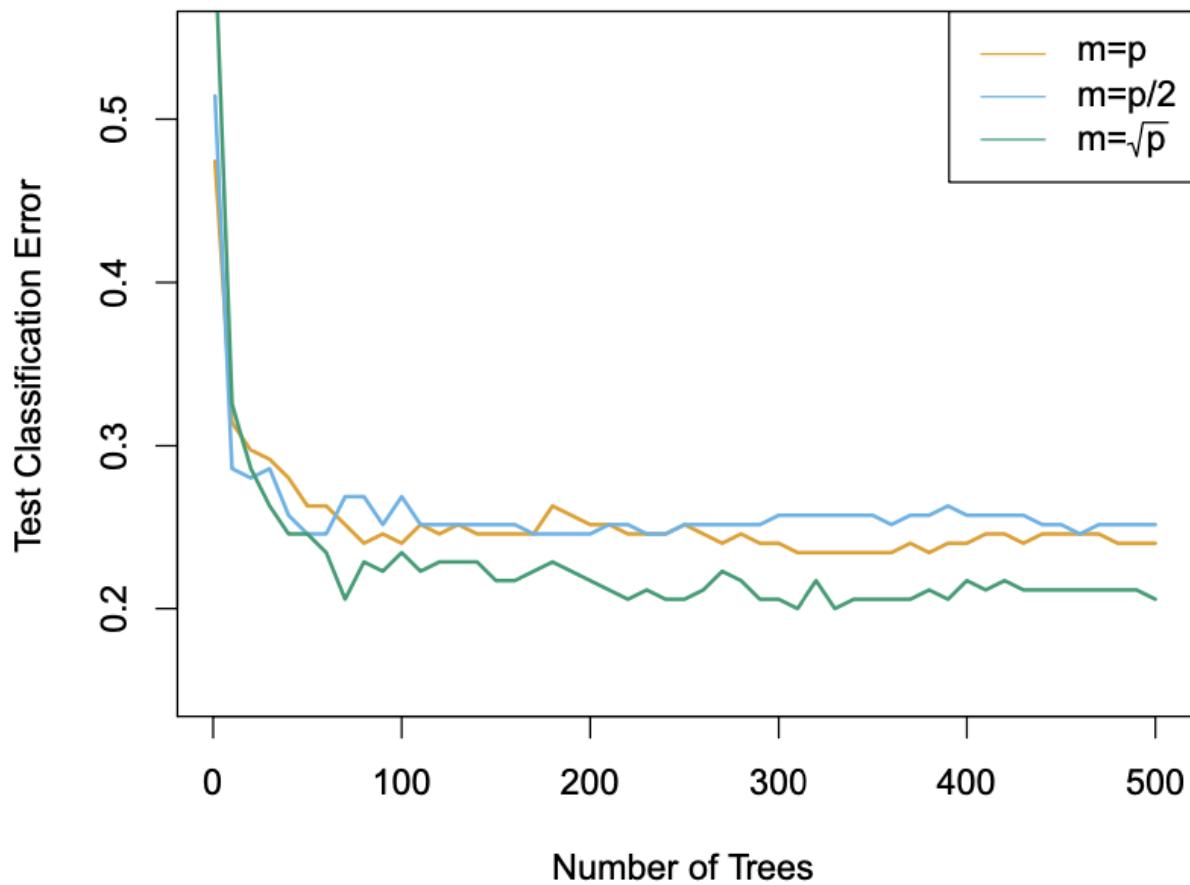
- ① bilde eine Anzahl von Bäumen auf Bootstrap-Trainingsmengen
- ② wähle für jeden Split im einem Baum eine zufällige (!) Auswahl von m Kandidaten X_1, \dots, X_m aus der Menge aller p Input-Features X_1, \dots, X_p
- ③ für den Split darf nur eines der X_1, \dots, X_m verwendet werden
- ④ wähle die Größe der Teilmenge so, dass $m \approx \sqrt{p}$ gilt

Bemerkung: für $m=p$:

Random Forest = Bagging

Bsp.: Random Forest

267



- Es werden bis zu 500 Prädiktoren zugelassen.
- $m=p$: bagging
⇒ Reduktion des Fehlers für $m=\sqrt{p}$ deutlich!

- $m \hat{=} \text{Anzahl zugelassener Prädiktoren für einen Split}$
- $m=p$: bagging

Einzelner Baum: Fehler $\approx 45,7\%$

VORSTELLUNG:

268

Angenommen, wir hätten einen "Super-Prädiktor" X_j aus X_1, \dots, X_p , dann

BAGGING

alle Bäume bewerten natürlich X_j für den ersten Split ... usw.

⇒ alle Bäume treffen ihre Entscheidung auf Basis maximal ähnlicher Regelwerke

⇒ alle Bäume sind sich sehr ähnlich, die Modelle sind hochgradig korreliert

⇒ geringe Reduktion der Varianz durch die Mittelwertbildung !!

RANDOM FOREST

Es werden immer nur Teilmengen von X_1, \dots, X_p zugelassen

269

⇒ im vielen dieser Teilmengen ist beim 1. Split der "Super-Predictor" X_j gar nicht enthalten

$\frac{P-m}{P}$ der Bäume sehen (im Mittel) X_j beim 1. Split nicht usw.

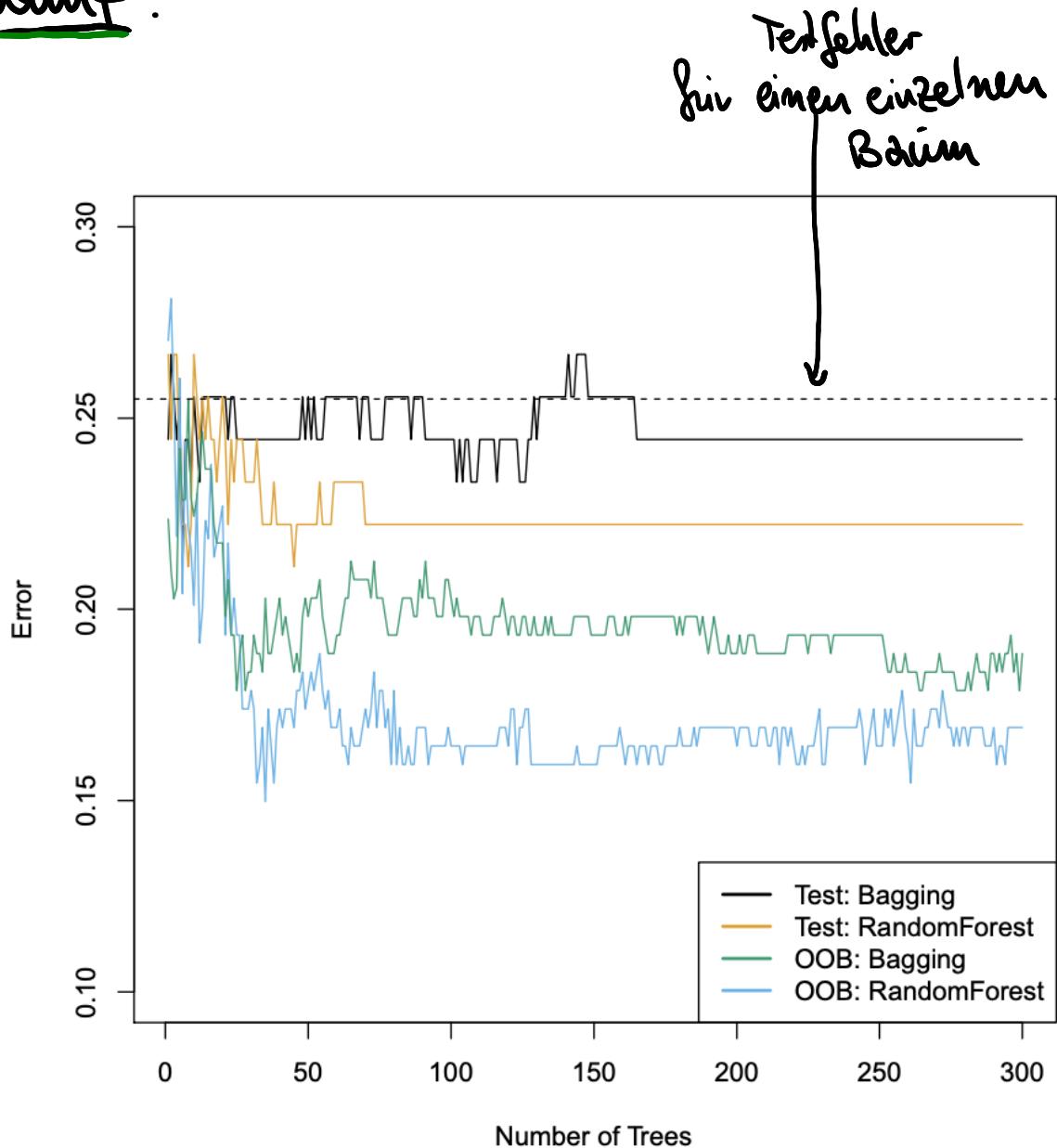
⇒ die Einzelbäume haben ganz unterschiedliche Regelwerke hinterlegt

⇒ stark unterschiedlich d.h. dekorreliert

⇒ am Ende durch die Mittelwertbildung sehr starke Reduktion der Varianz möglich



insgesamt:



(Tibshirani, 2014)

Auf einer Testmenge schneidet der Random Forest etwas besser ab.

OOB-Fehler: gibt realistischere Einschätzung ohne Cross-Validation

- Anzahl der Bäume ist inkritisch!
- ⇒ B genügend groß wählen, so dass der Fehler konvergiert ist.

Hier: $B \approx 100$ genügt!

→ Sensitivitätsanalyse

Ein einzelner Baum hat eine klare Struktur und die Entscheidungsregeln liegen offen

⇒ es lässt sich leicht erkennen, welche Variablen für die Entscheidung wichtig sind

JETZT: problematisch, weil im Ensemble entschieden wird.

⇒ verschiedene Bäume kommen mit ganz unterschiedlichen Regelwerken zum Ergebnis



Zeigt sich die Feature-Importance trotzdem angeben?



Nutze den RSS (RMSE) oder den Gini-Index

Bsp.: Bagging

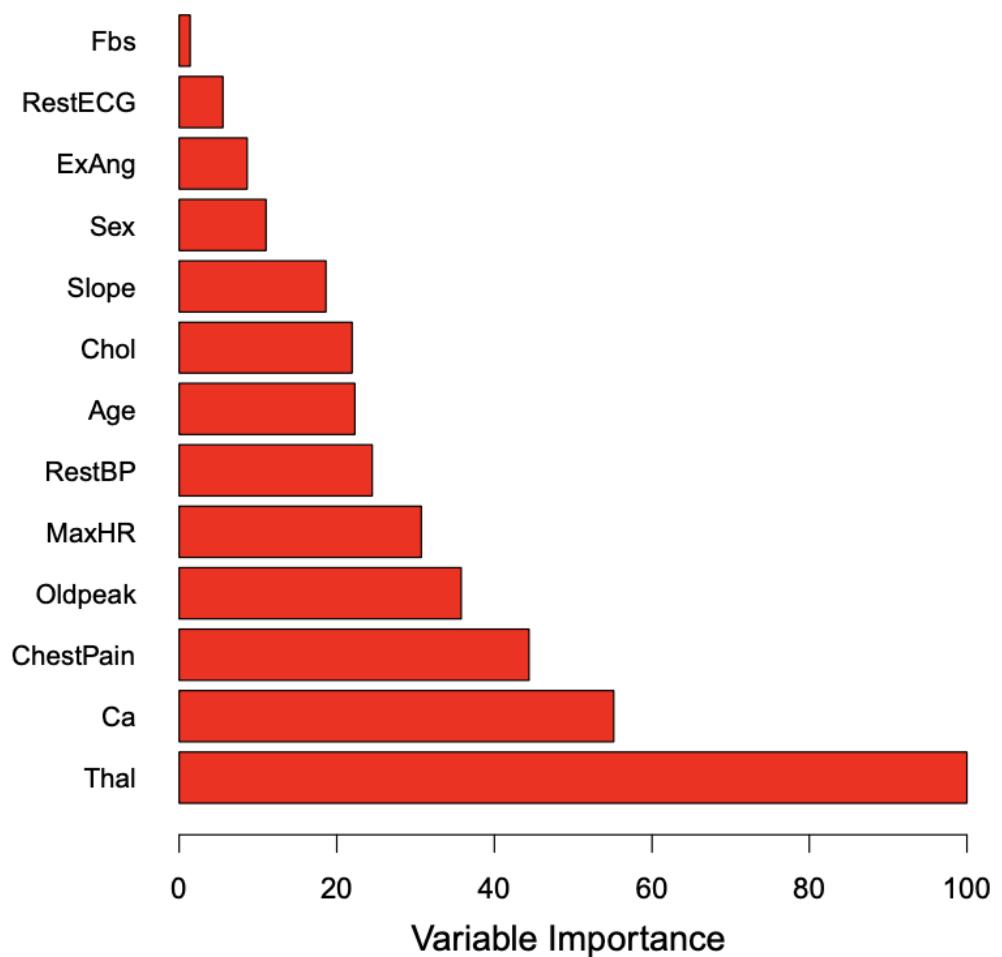
- Fülle die Summe der Reduktion des RSS bezüglich Split_j mit einem gegebenen Prädiktor x_j für einen Baum
- Mittle die Summe über alle B Bäume

Also: großer Wert für ein bestimmtes x_j
 \Rightarrow wichtiger Prädiktor

Analog: Abnahme des Gini-Index bei Klassifikation

Bsp.: Tibshirani (Heart Disease)

273



← Die Importance der Einflussgrößen ist durch Mittelwertbildung über alle B Modelle ermittelt worden

- Die Werte sind im Bezug auf das Maximum ($\text{Thal} \approx 100\%$) skaliert

↑ Klassifikator!

Hg) ist Abnahme des Gini-Index!

4.2.3) Boosting

Zum Boosting handelt es sich um ein Verfahren, das für viele verschiedene ML-Verfahren eingesetzt werden kann.

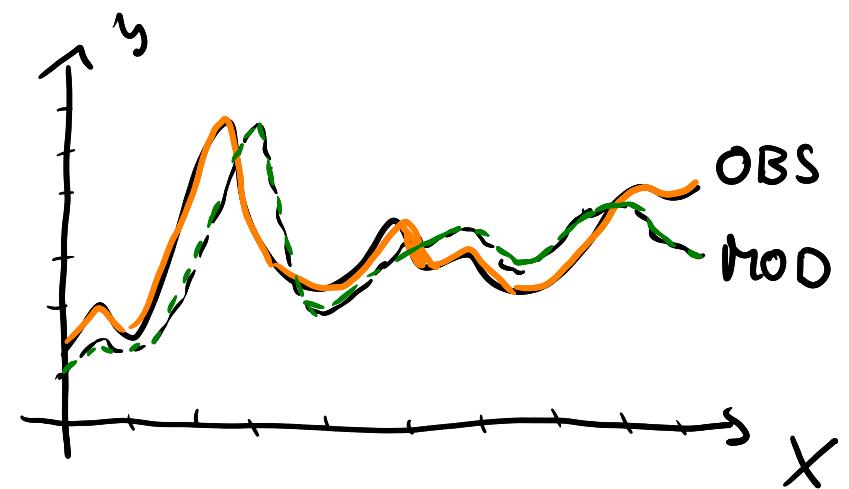


→ Betrachte ein Modell

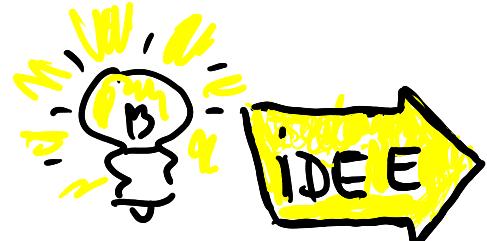
$$\hat{f}_1(x) = Y + \varepsilon$$

Dann gilt immer:

- Das Modell kann die beobachteten Daten nicht vollständig sondern nur zum Teil abbilden



⇒ Fehler $y_i - \hat{y}_i = r_i$ Residuum



Man könnte jetzt ja einfach versuchen mit
einem zweiten Modell den Fehler zu modellieren!

⇒

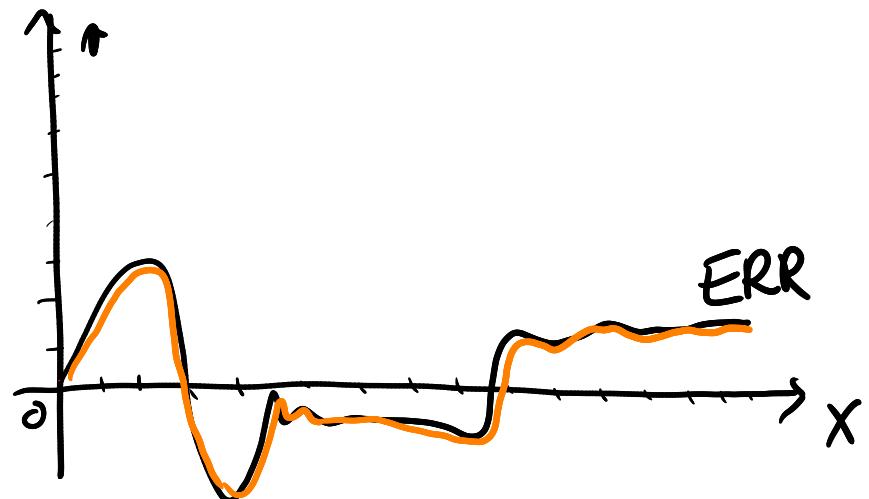
$$f_2(x) = \varepsilon + \dots$$

⇒ insgesamt ergibt sich
dann:

$$f_1(x) + f_2(x) = Y + \tilde{\varepsilon}$$



Das zusammengesetzte Modell hat hoffentlich einen
kleineren Restfehler !!





Wiederholt man diesen Prozess öfter, dann sollte man (langsam) immer bessere Prognosen hinkriegen.

276

Erinnerung: In der Praxis gewichtet man das Residualmodell noch mit einem sogenannten Shrinkage Parameter λ .

Was sorgt dafür, dass der Prozess der Anpassung verlangsamt wird und mehr verschiedene Modelle zum Zug kommen (besser für die Gesamtperformance).

Für Regression Trees sieht der Algorithmus wie folgt aus:

Def.: Boosting (Regression Trees)



① Setze $\hat{f}(x) = 0$ und $r_i = y_i$, $\forall i \in \text{Trainingsmenge}$

② Für $b=1, \dots, B$ wiederhole

a) Fülle Baum \hat{f}^b mit d Split's an (X, r)

b) erweitere \hat{f} durch

$$\hat{f} \leftarrow \hat{f}(x) + \lambda \hat{f}^b(x)$$

c) bestimme neue Restfehler

$$r_i \leftarrow r_i - \lambda \hat{f}^b(x_i)$$

③ Gib das finale Modell $\hat{f}(x)$ aus :

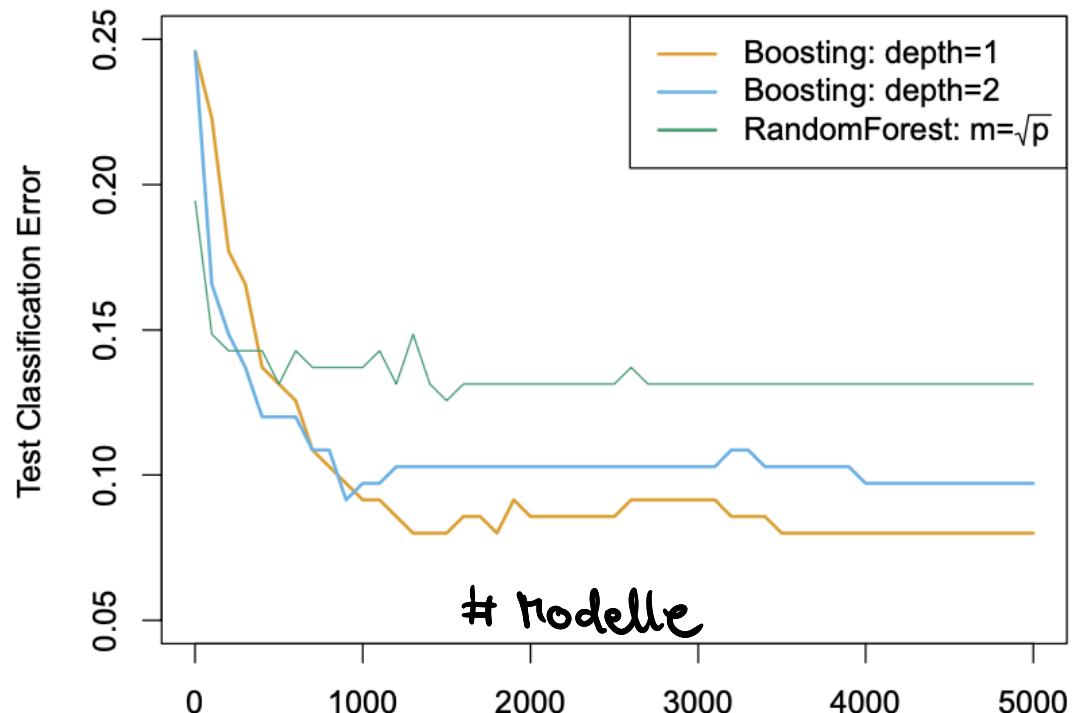
$$\boxed{\hat{f}(x) = \sum_{b=1}^{B} \lambda \hat{f}^b(x)}$$

Wir haben beim Boosting 3 Meta-Parameter zu bestimmen:

278

- ① Anzahl der Bäume B (cross validation)
- ② Shrinkage Parameter λ (typisch: 0,01 ; 0,001)
bei sehr kleinem λ sind mehr Bäume nötig
- ③ Anzahl der Splits d (oft $d=1$, generell eher klein)
→ Kleine Bäume werden bevorzugt, da das Verfahren auf der Trainingsmenge sonst zu schnell konvergiert ...

Bsp.: Boosting



- $\text{depth} \hat{=} \text{Anzahl der Spalten, die in jedem Baum zugelassen sind}$ ($d \hat{=} \text{Variable pro Modell}$)

- $d=1$: Stump interaction !
 - ⇒ additives Modell, weil jeder Baum nur 1 Variable zählt!
 - ⇒ viele Terme nötig !!
- ⇒ d heißt durch "Interaktions-tiefe" d splits involvieren d Variablen ...
- Top-Modell für $d=1$
- Random Forest im Vergleich das schlechteste Modell !

ZUSAMMENFASSUNG : TREE ENSEMBLE METHODEN

280

BAGGING

- Bäume werden unabhängig voneinander auf Bootstrap-Samples aufgebaut

→ Modelle sind sehr ähnlich → lokale Minima des Fehlers

→ Deterioration der Teilbäume notwendig

RANDOM FOREST

- ebenfalls unabhängig auf Bootstrap-Samples aufgebaut
- Deterioration der Teilbäume durch zufällige Subsets von Prädiktoren bei jedem Split

BOOSTING

- kein Random Sampling - komplette Datenmenge
- Bäume werden sukzessiv aufgebaut (→ residual models)
- jeder neue Baum lernt das verbleibende Fehlersignal der Vorgänger