trees & forest

agenda:

1. decision trees: introduction
2. decision trees: how does it work?
3. random forest: introduction
4. rondom forest: how does it work?
5. hands on

Decision Tree (DT)

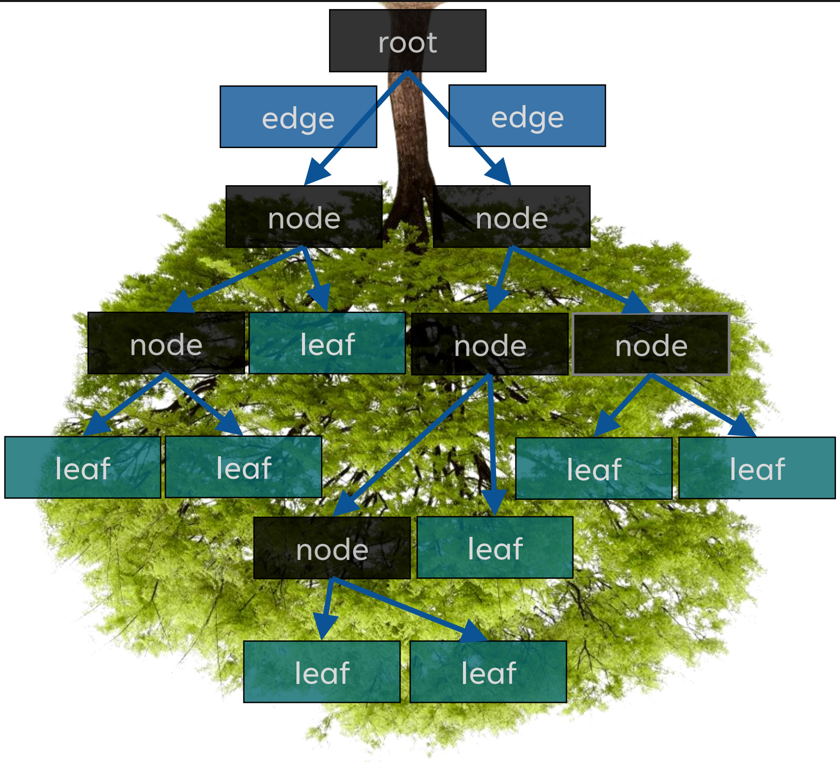
metaphoric basis:
a standard tree... well... upside down

with all the bells and whistles:
- root
- branching (nodes)
- branches (edges)
- leafs, terminal nodes

the main principle

„question-and-answer“

root/node: interrogate the data
edge: if „yes“: go left, else: go right
leaf: contains final decision / statement



Decision Tree (DT)

metaphoric basis:
a standard tree... well... upside down

with all the bells and whistles:

- root
- branching (nodes)
- branches (edges)
- leafs, terminal nodes

the main principle

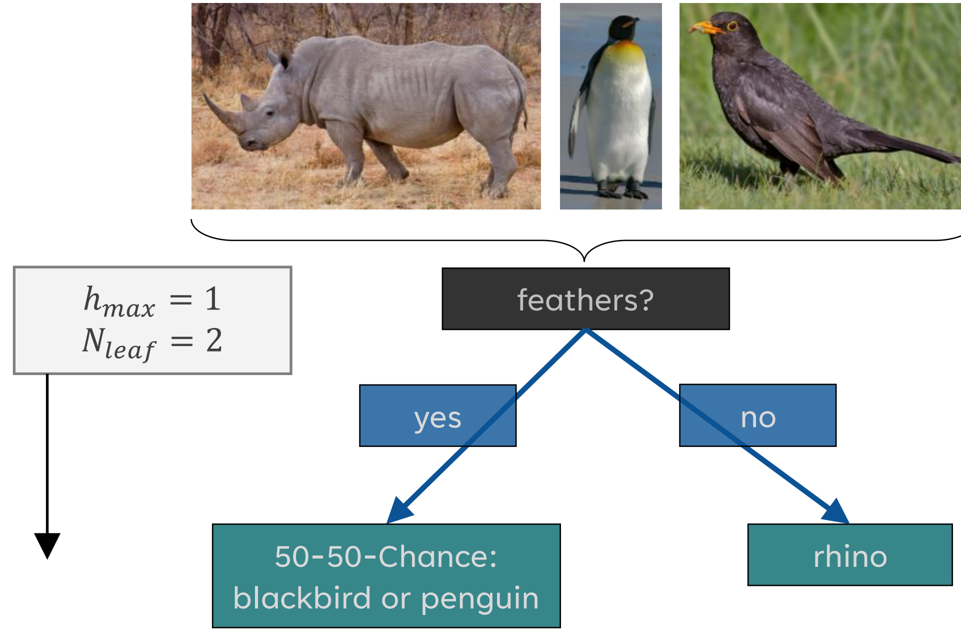
„question-and-answer“

root/node: interrogate the data
edge: if „yes“: go left, else: go right
leaf: contains final decision / statement

example

classification of animals

pruned DT:
multiple statements in terminal node („impurity“)



Decision Tree (DT)

metaphoric basis:
a standard tree… well… upside down

with all the bells and whistles:
- root
- branching (nodes)
- branches (edges)
- leafs, terminal nodes

the main principle

„question-and-answer“

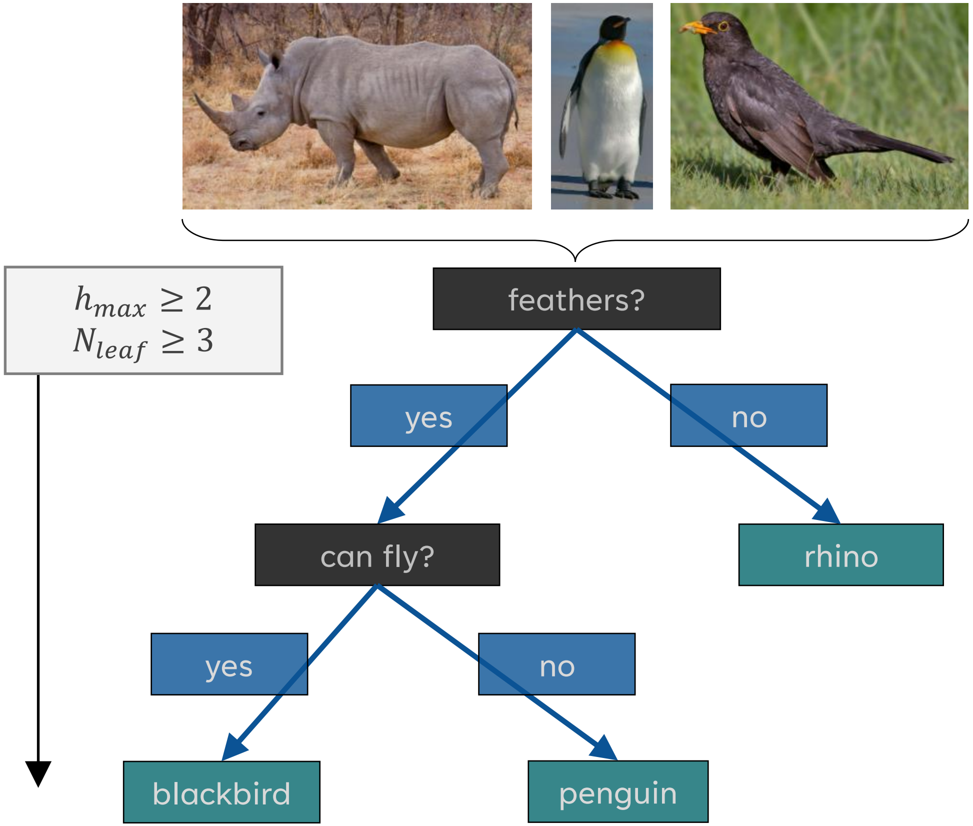
root/node: interrogate the data
edge: if „yes“: go left, else: go right
leaf: contains final decision / statement

example

classification of animals

unpruned DT:
single statement per leaf („purity“)

➔ important **hyperparameter**: h_{max} or N_{leaf}



mathematical formulation of Q&A

regression & classification:

1. a DT splits the domain into M subdomains
2. constant value c_m in every subdomain R_m

decision to split a domain:

➔ metric: **node impurity** Q

metric for regression

N: data points, y: true value, $\hat{y}$: predicted value

Mean Squared Error (MSE):

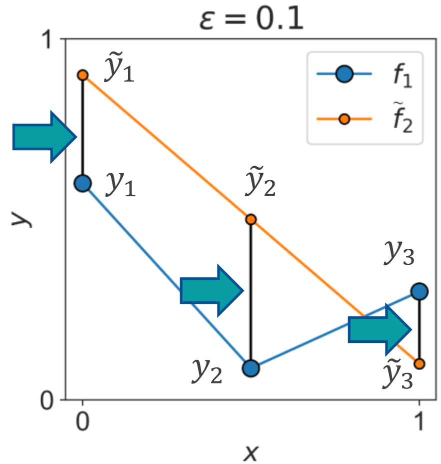
$$Q = \varepsilon_{MSE}(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \frac{1}{N}\sum_{i=0}^{N-1}(y_i - \hat{y}_i)^2$$

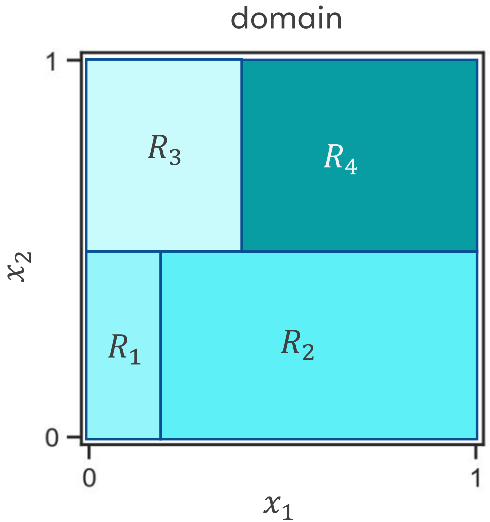
metric for classification

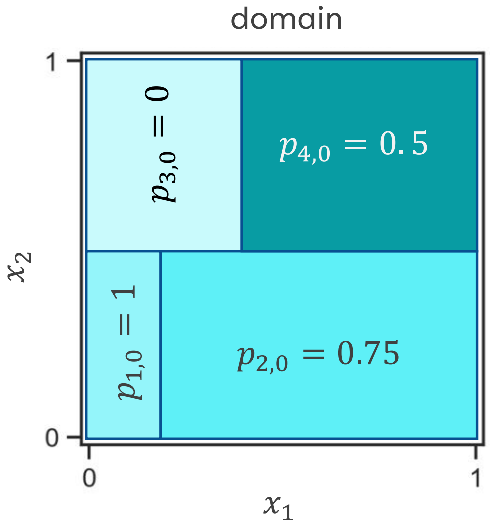
$p_{m,k}$: probability of occurrence of class $k \in K$ in node $m \in M$

Gini (impurity) Index:

$$Q_m = \sum_{k=1}^{K} p_{m,k}(1 - p_{m,k})$$







mathematical formulation of Q&A

regression & classification:
1. a DT splits the domain into M subdomains
2. constant value c_m in every subdomain R_m

decision to split a domain:
➔ metric: **node impurity** Q

metric for regression

N: data points, y: true value, $\hat{y}$: predicted value
Mean Squared Error (MSE):

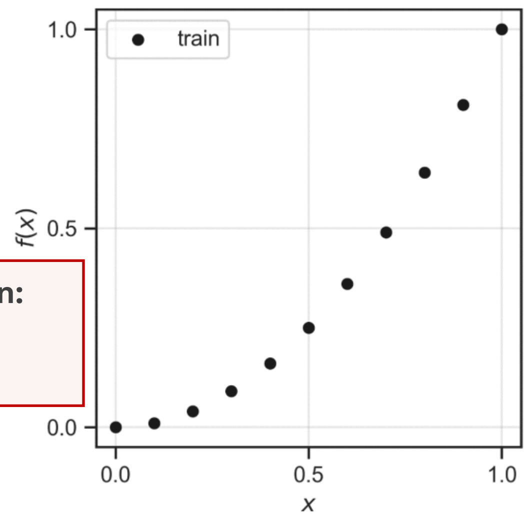
$$Q = \varepsilon_{MSE}(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \frac{1}{N}\sum_{i=0}^{N-1}(y_i - \hat{y}_i)^2$$

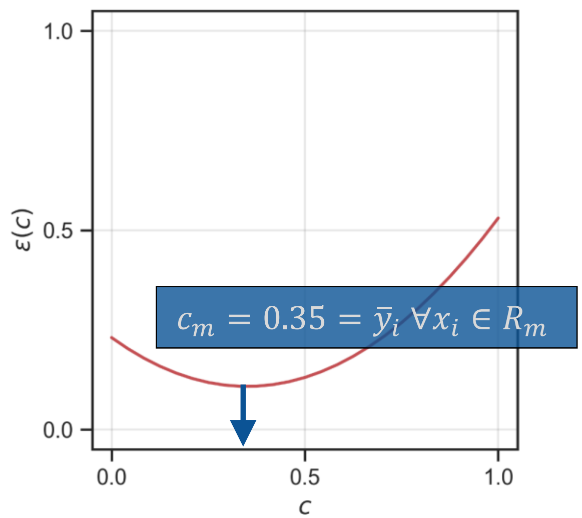
example

$f: \mathbb{R} \to \mathbb{R},\ f(x) = x^2 = y$
mean is „best guess" for each subdomain
$\hat{y}_i = c_m = \bar{y}_i\ \forall x_i \in R_m$

$n = 11$
$\hat{y} = ?$
$\varepsilon_{MSE} = ?$

R_m is currently whole domain:
what is the best
$c_m = \hat{y}_i \forall x_i \in R_m$





mathematical formulation of Q&A

regression & classification:

1. a DT splits the domain into M subdomains
2. constant value c_m in every subdomain R_m

decision to split a domain:
➔ metric: **node impurity** Q

metric for regression

N: data points, y: true value, $\hat{y}$: predicted value
Mean Squared Error (MSE):

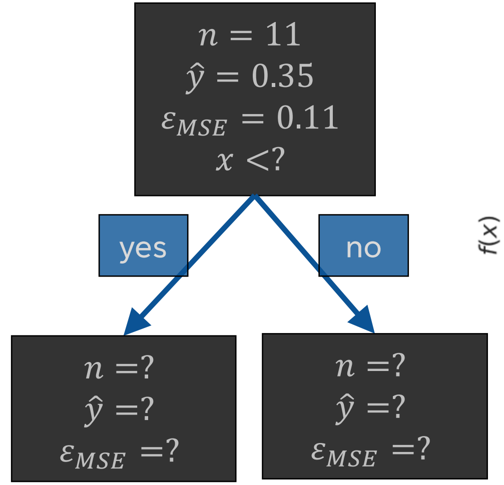
$$Q = \varepsilon_{MSE}(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \frac{1}{N}\sum_{i=0}^{N-1}(y_i - \hat{y}_i)^2$$

example

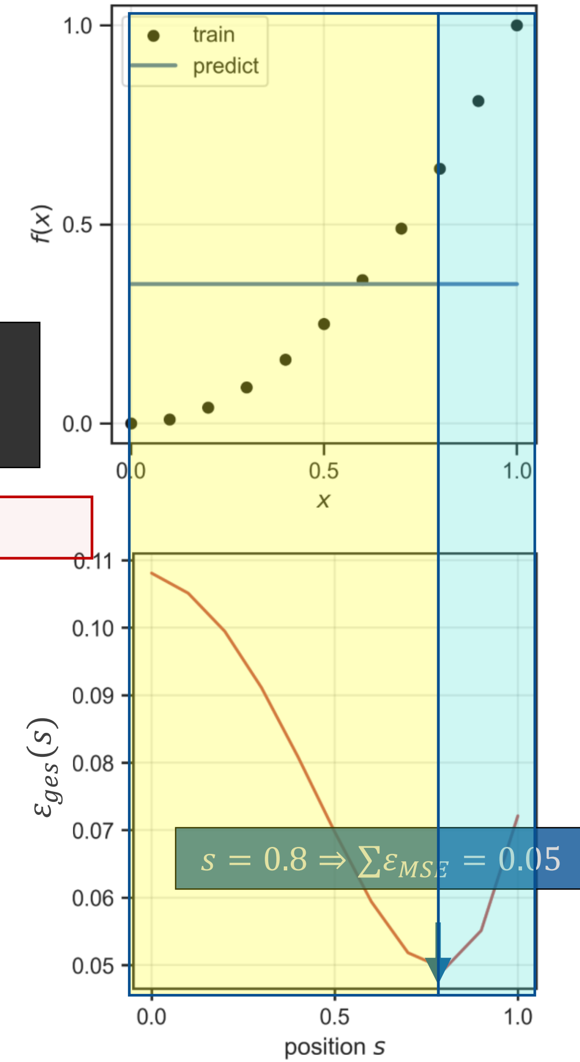
$f: \mathbb{R} \to \mathbb{R},\ f(x) = x^2 = y$
mean is „best guess“ for each subdomain
$\hat{y}_i = c_m = \bar{y}_i\ \forall x_i \in R_m$

minimize ε_{ges} by splitting domain at position s:

$$\varepsilon_{ges}(s) = \sum_{m=1}^{M=2} \varepsilon_m(s)$$



best split-position?



mathematical formulation of Q&A

regression & classification:

1. a DT splits the domain into M subdomains
2. constant value c_m in every subdomain R_m

decision to split a domain:

➔ metric: **node impurity** Q

metric for regression

N: data points, y: true value, $\hat{y}$: predicted value

Mean Squared Error (MSE):

$$Q = \varepsilon_{MSE}(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \frac{1}{N}\sum_{i=0}^{N-1}(y_i - \hat{y}_i)^2$$

example

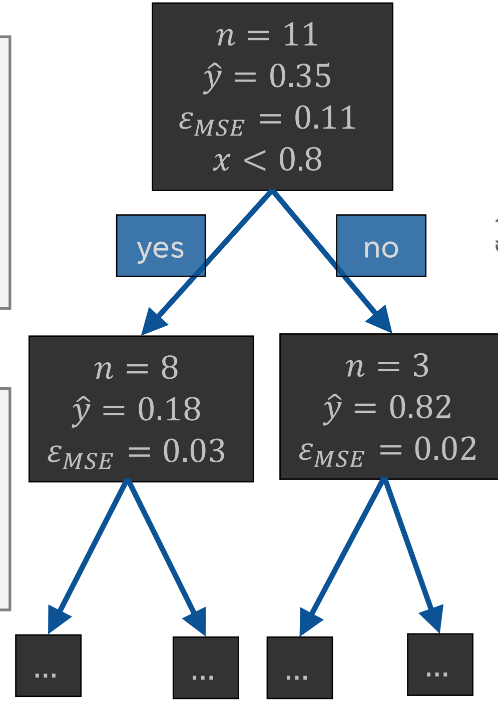
$f: \mathbb{R} \to \mathbb{R}$, $f(x) = x^2 = y$

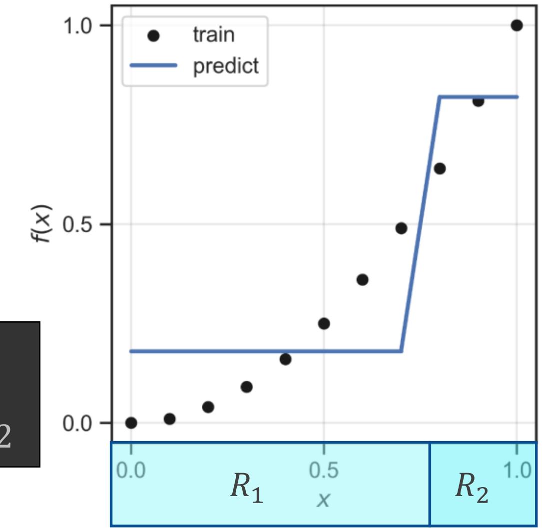
mean ist „best guess“ for each subdomain

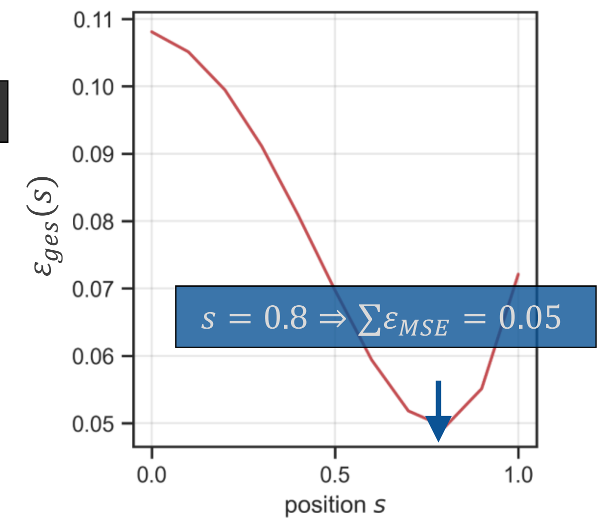
$\hat{y}_i = c_m = \bar{y}_i \; \forall x_i \in R_m$

minimize ε_{ges} by splitting domain at position s:

$$\varepsilon_{ges}(s) = \sum_{m=1}^{M=2} \varepsilon_m(s)$$







mathematical formulation of Q&A

regression & classification:

1. a DT splits the domain into M subdomains
2. constant value c_m in every subdomain R_m

decision to split a domain:
➔ metric: **node impurity** Q

metric for classification

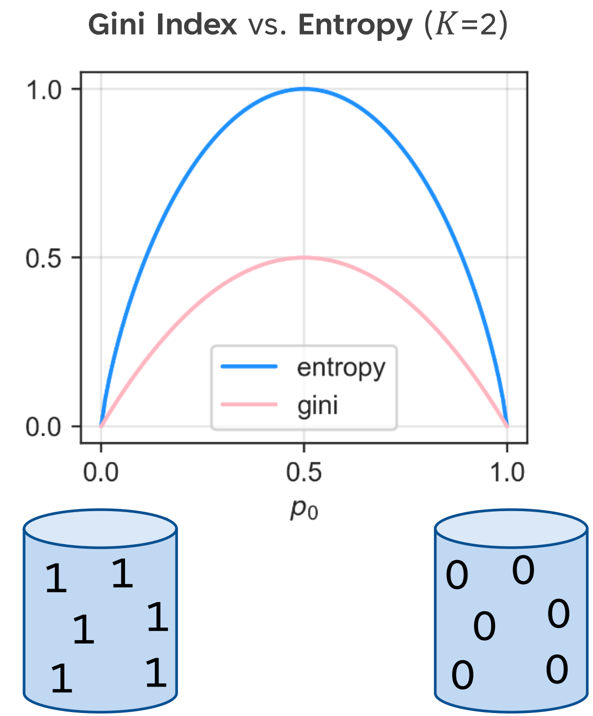
$p_{m,k}$: probability of occurrence of class $k \in K$ in node $m \in M$

Gini (impurity) Index:

$$Q_m = \sum_{k=1}^{K} p_{m,k}(1 - p_{m,k})$$

Entropy H:

$$H = -\sum_{k=1}^{K} p_{m,k} \ln(p_{m.k})$$



mathematical formulation of Q&A

regression & classification:

1. a DT splits the domain into M subdomains
2. constant value c_m in every subdomain R_m

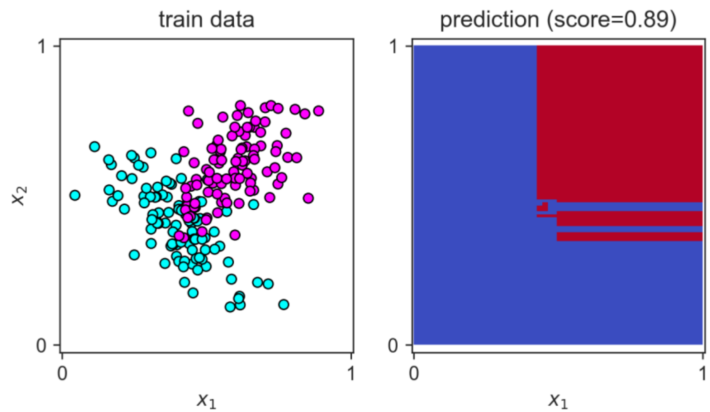
decision to split a domain:
➔ metric: **node impurity** Q

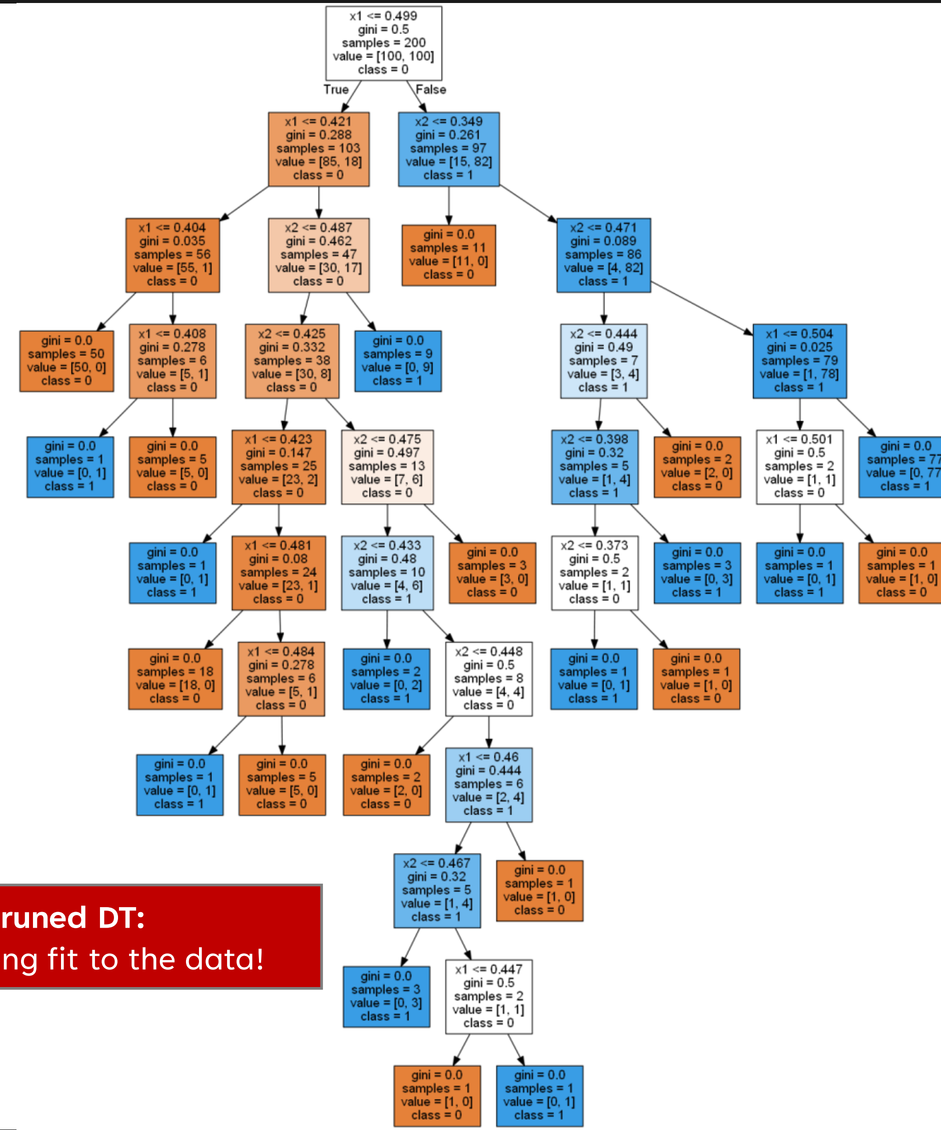
metric for classification

$p_{m,k}$: probability of occurrence of class $k \in K$ in node $m \in M$

Gini (impurity) Index:

$$Q_m = \sum_{k=1}^{K} p_{m,k}(1 - p_{m,k})$$





unpruned DT:
strong fit to the data!

bias & variance

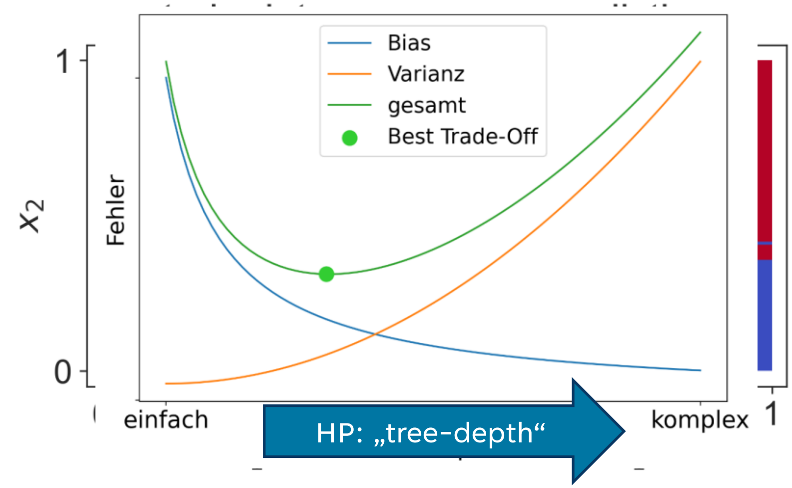
general definition of a function for approximation:

$$\hat{f}\colon \mathbb{R}^{n,m} \to \mathbb{R}^{m}, \hat{f}(\boldsymbol{X}) = \hat{\boldsymbol{y}} = \boldsymbol{y} + \boldsymbol{\varepsilon}$$

the **approximation error** ε contains of:

- unknown influences
- **model-bias**:
  simplifications, high when underfitting
- **model-variance**
  high komplexity, high when overfitting

➔ **IMPORTANT: bias-variance trade-off**



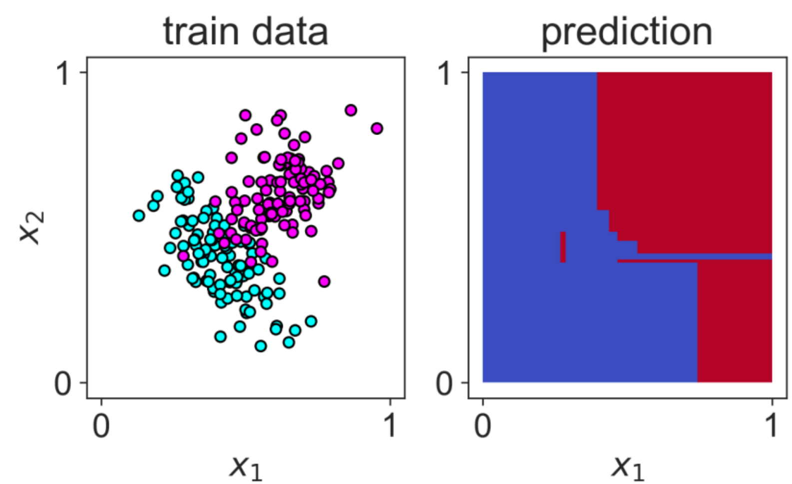
unpruned DT: pros & cons

pros:

- can handle mixed and redundant variables
- **small Bias**
- …

cons:

- **high varianz**
- **usually prediction is not very good**
- …



bias & variance

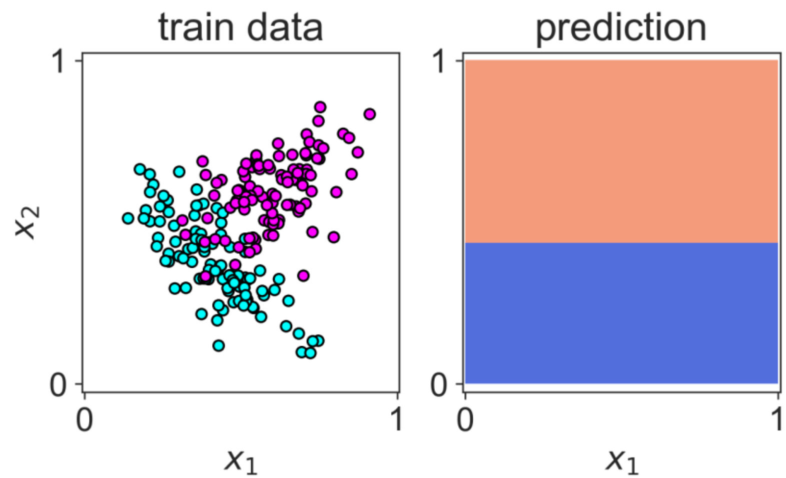
general definition of function for approximation:

$$\hat{f}: \mathbb{R}^{n,m} \rightarrow \mathbb{R}^{m}, \hat{f}(\boldsymbol{X}) = \widehat{\boldsymbol{y}} = \boldsymbol{y} + \boldsymbol{\varepsilon}$$

the **approximation error** ε contains of:

- unknown influences
- **modell-Bias**:
  simplifications, high when underfitting
- **modell-Variance**
  high komplexity, high when overfitting

➔ **IMPORTANT: bias-variance trade-off**



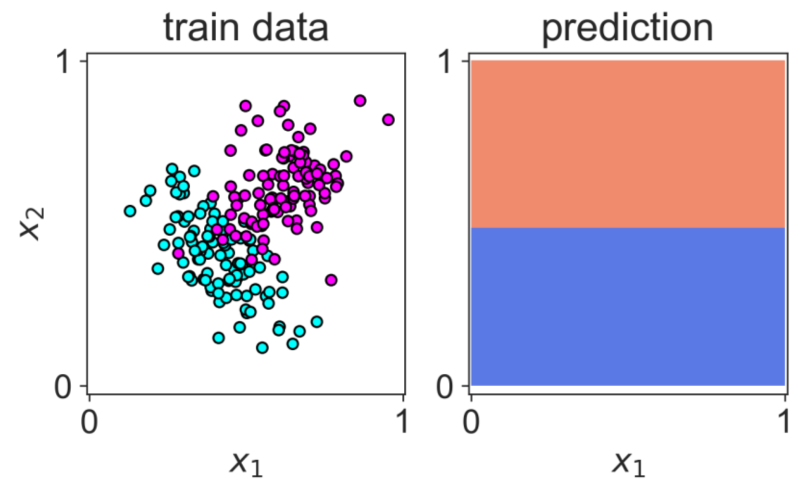
pruned DT: pros & cons

pros:

- can handle mixed and redundant variables
- **small variance**
- ...

cons:

- **high bias**
- **usually prediction is not very good**
- ...

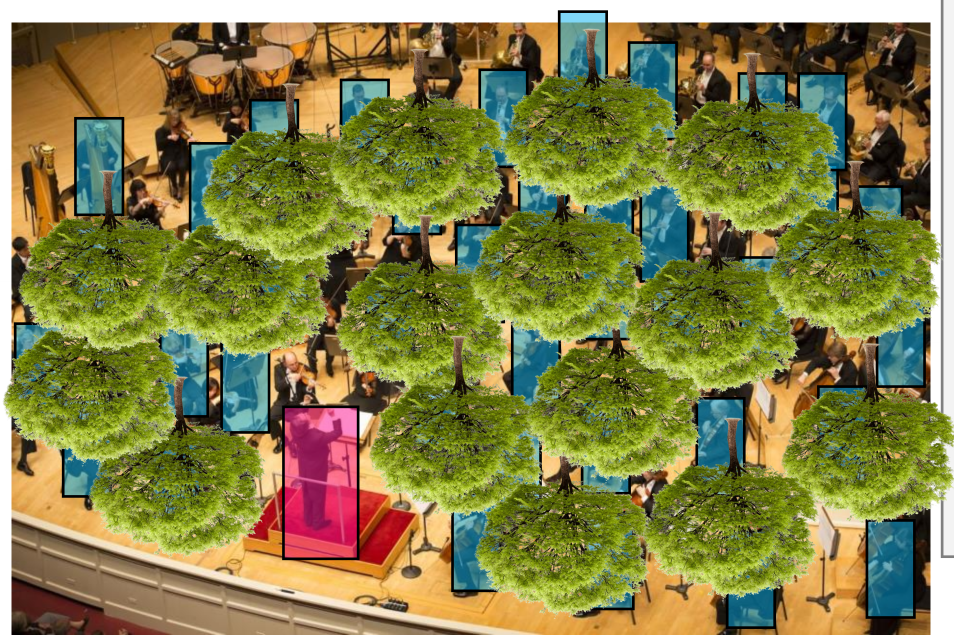


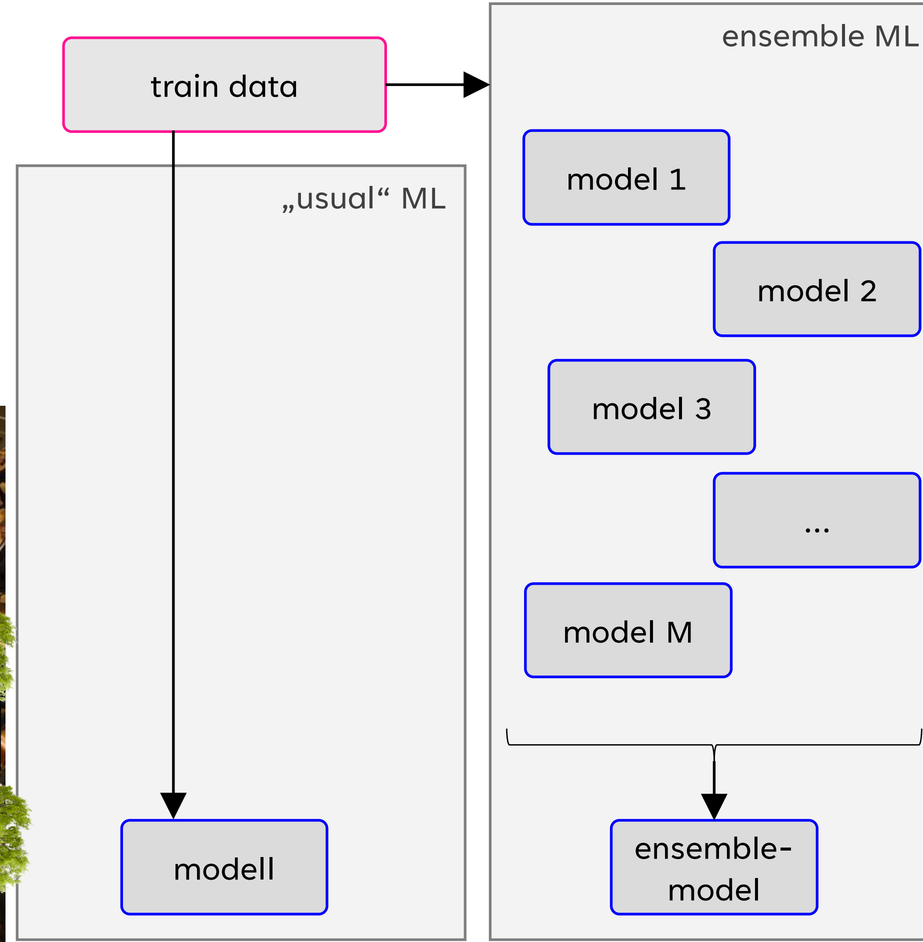
what is a random forest?

it‘s an ensemble

what is an ensemble?

it‘s an aggregation…
- of multiple models (usually DTs)
- to exploit pros
- to avoid cons





what is a random forest?

it‘s an ensemble

what is an ensemble?

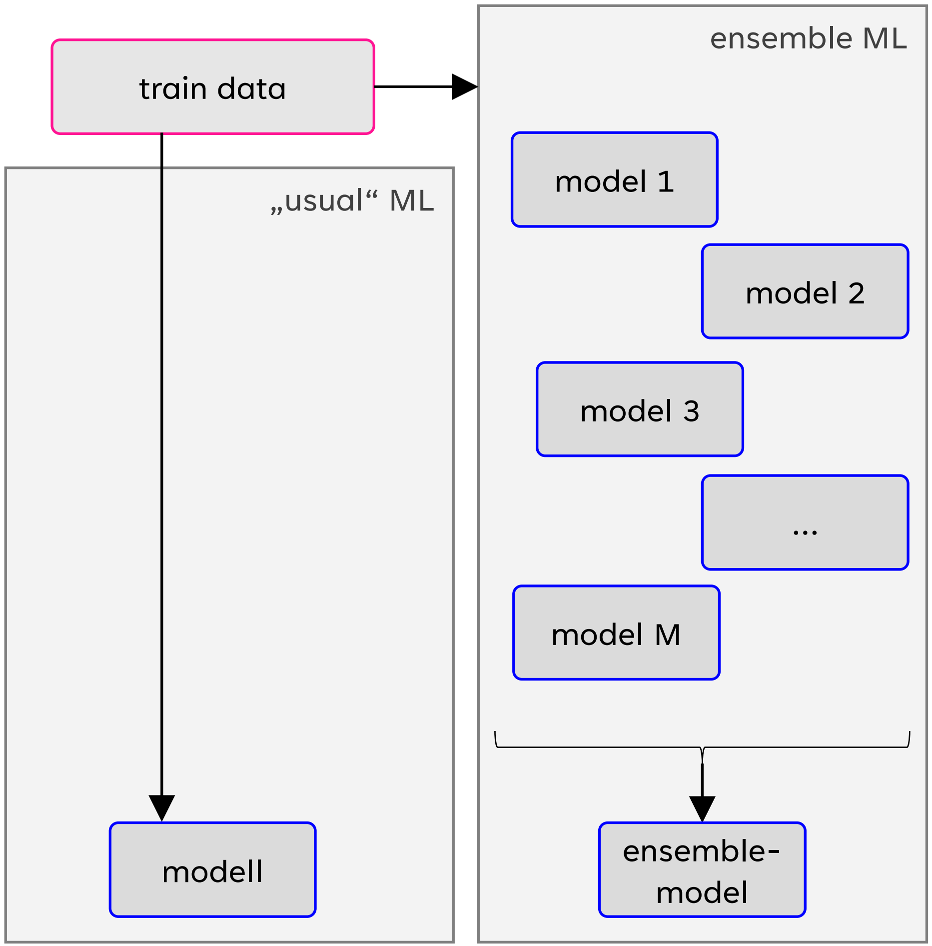
it‘s an aggregation…
- of multiple models (usually DTs)
- to exploit pros
- to avoid cons

different types of ensembles

Bagging [Breiman, 1996]
- **Random Forest** [Breiman, 2001]

Boosting:
- AdaBoost [Freund & Shapire, 1996]
- Gradient Boosting [Friedman, 1999]
  (*Extreme Gradient Boosting* [Chen & Guestrin, 2016])

Stacking



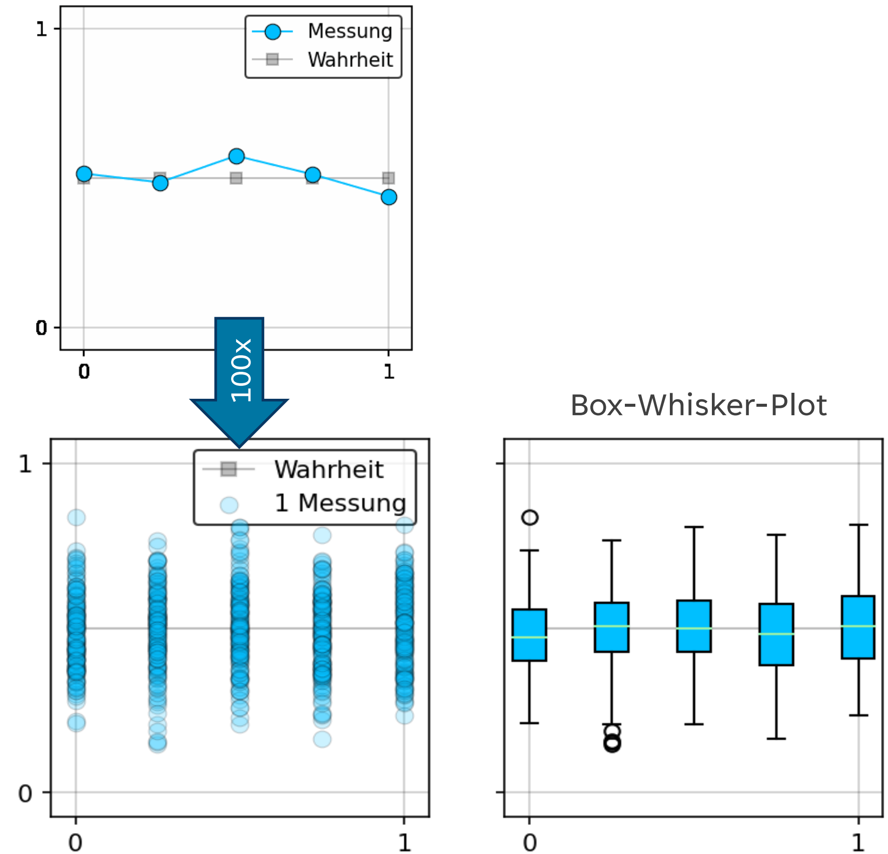
starting point

unpruned DT*
- pro: small bias
- con: high variance

bagging

short for „booststrap aggrigation“

idea:
build an ensemble of unpruned DTs
- exploit pro of **small bias** (strong adaption)
- avoid con of high variance by **boostrapping** (averaging **random resamples**)



* bagging works with different types of models too

starting point

unpruned DT*
- pro: small bias
- con: high variance

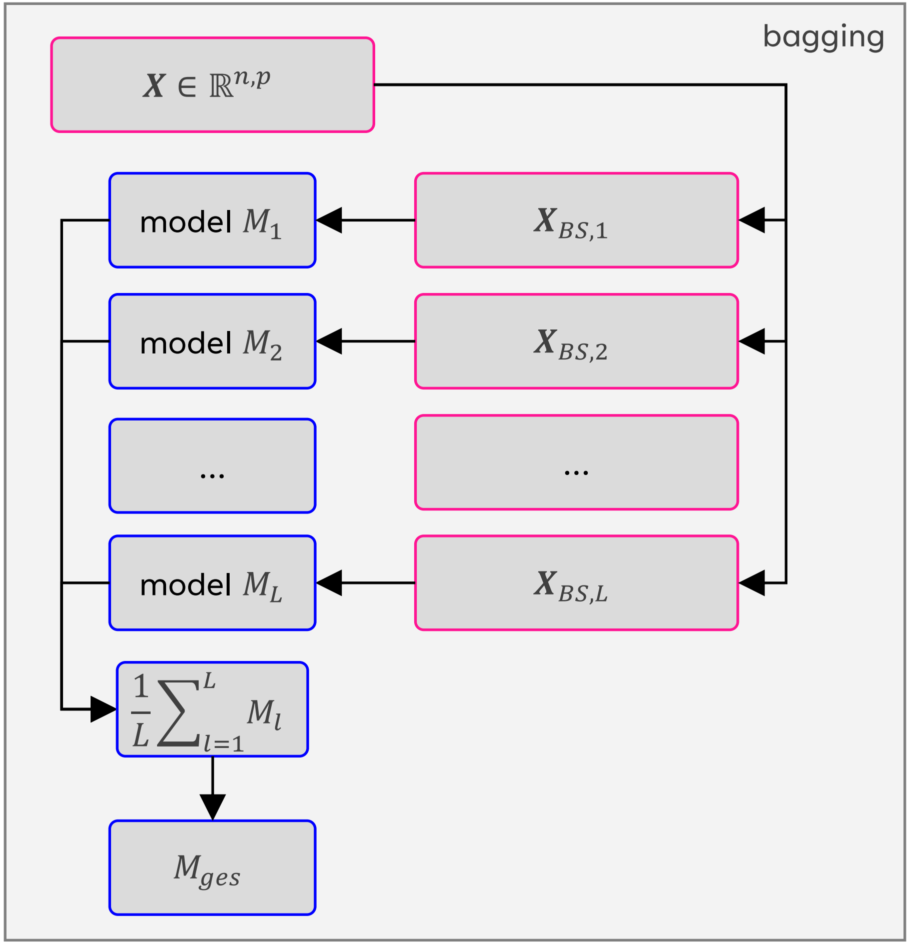
bagging

short for „booststrap aggrigation“

idea:
build an ensemble of unpruned DTs
- exploit pro of **small bias** (strong adaption)
- avoid con of high variance by **boostrapping** (averaging **random resamples**)

process

1. resample data $\boldsymbol{X} \in \mathbb{R}^{n,p}$ by bootstrapping $L \in \mathbb{N}$ times:
$$\boldsymbol{X} \rightarrow \{\boldsymbol{X}_{BS,1}, \dots, \boldsymbol{X}_{BS,L}\} \text{ with } \boldsymbol{X}_{BS,l} \in \mathbb{R}^{n,p}$$
2. train L models using the L new datasets
$$\{\boldsymbol{X}_{BS,1}, \dots, \boldsymbol{X}_{BS,L}\} \rightarrow \{M_1, \dots, M_L\}$$
3. answer of the whole ensemble M_{ges} : averaging all ensemble-members
$$M_{ges} = \frac{1}{L}\sum_{l=1}^{L} M_l$$



starting point

unpruned DT*
- pro: small bias
- con: high variance

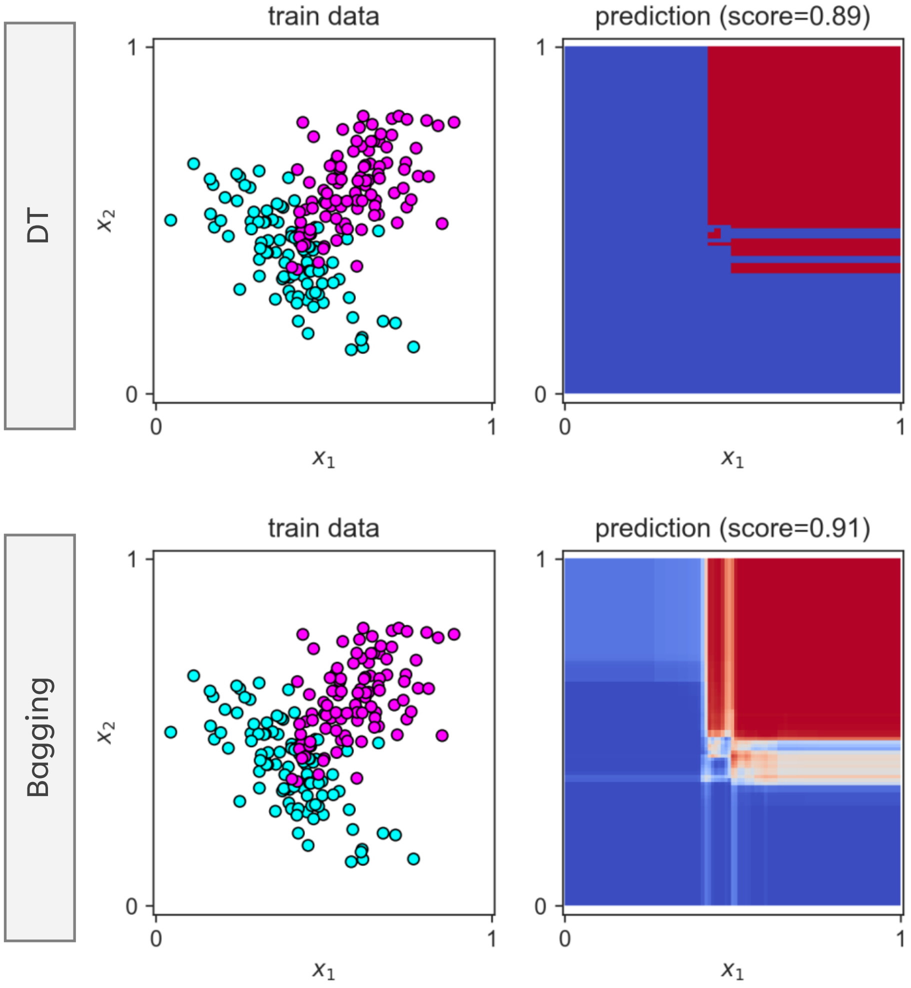
bagging

short for „booststrap aggrigation“

idea:
build an ensemble of unpruned DTs
- exploit pro of **small bias** (strong adaption)
- avoid con of high variance by **boostrapping** (averaging **random resamples**)

process

1. resample data $\boldsymbol{X} \in \mathbb{R}^{n,p}$ by bootstrapping $L \in \mathbb{N}$ times:
$$\boldsymbol{X} \to \{\boldsymbol{X}_{BS,1}, \ldots, \boldsymbol{X}_{BS,L}\} \text{ with } \boldsymbol{X}_{BS,l} \in \mathbb{R}^{n,p}$$
2. train L models using the L new datasets
$$\{\boldsymbol{X}_{BS,1}, \ldots, \boldsymbol{X}_{BS,L}\} \to \{M_1, \ldots, M_L\}$$
3. answer of the whole ensemble M_{ges} : averaging all ensemble-members
$$M_{ges} = \frac{1}{L} \sum_{l=1}^{L} M_l$$



* bagging works with different types of models too

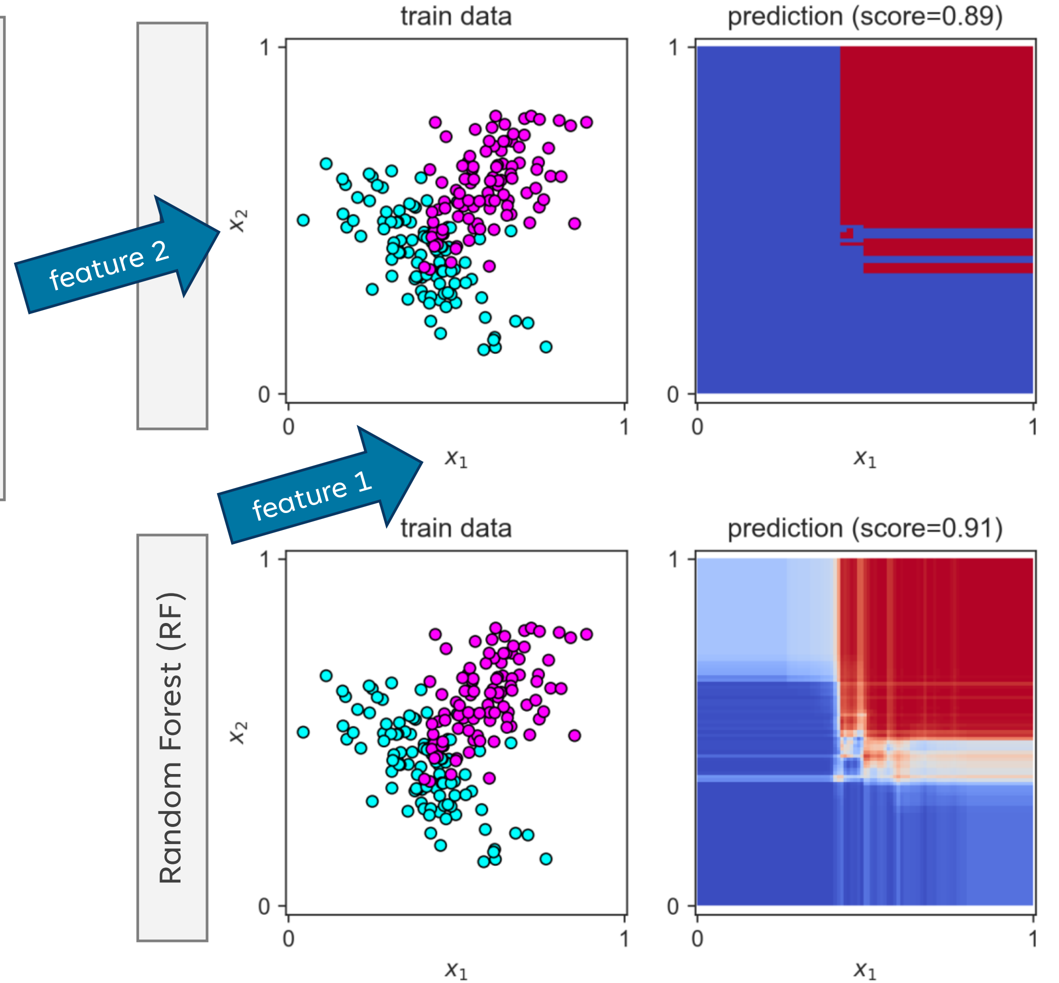
extended bagging

decreasing variance even further by...
uncorrelated DTs

$\boldsymbol{X} \in \mathbb{R}^{n,p}$ training data, n samples, p features

random forest:
train DTs with randomly selected $m < p$ features
$\boldsymbol{X} \in \mathbb{R}^{n,m}$

e.g.:
$m = \sqrt{p}$ or $\log(p)$



code available*

clone or download GitHub-Repository
https://github.com/saifedias/tree_randomForest.git

online Notebook via Binder
https://mybinder.org/v2/gh/saifedias/tree_randomForest.git/HEAD

would you like to know more? – a short outline

bagging, boosting, stacking
https://towardsdatascience.com/ensemble-methods-bagging-boosting-and-stacking-c9214a10a205

ensemble learning
https://www.kaggle.com/discussions/general/263786

* at least for a few days from now (07.04.2022)

BTU ML-Group

contact:
marlon.lehmann@b-tu.de