Ensembling methods

Stacking, bagging, boosting

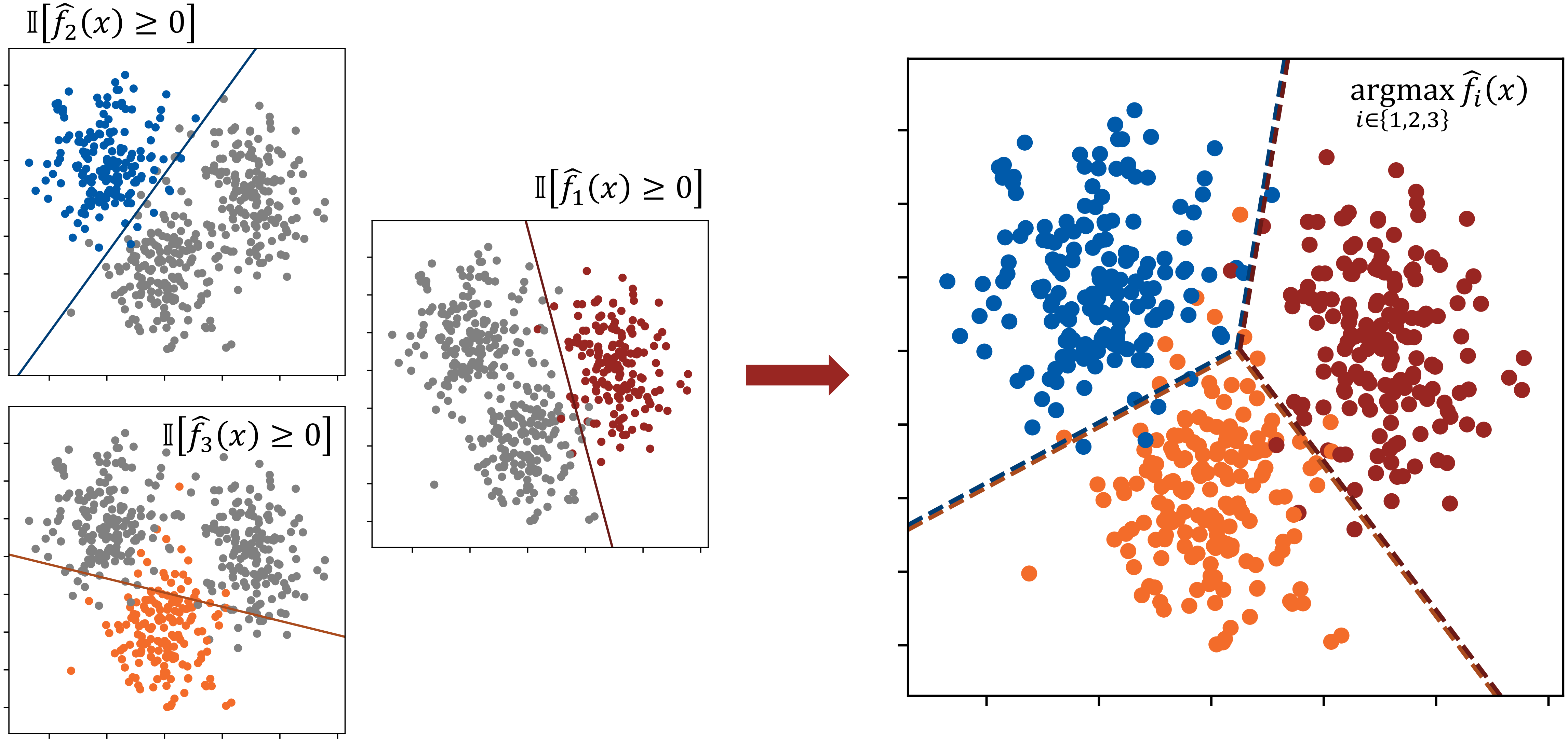
Machine Learning and Data Mining, 2024

Majid Sohrabi
National Research University Higher School of Economics

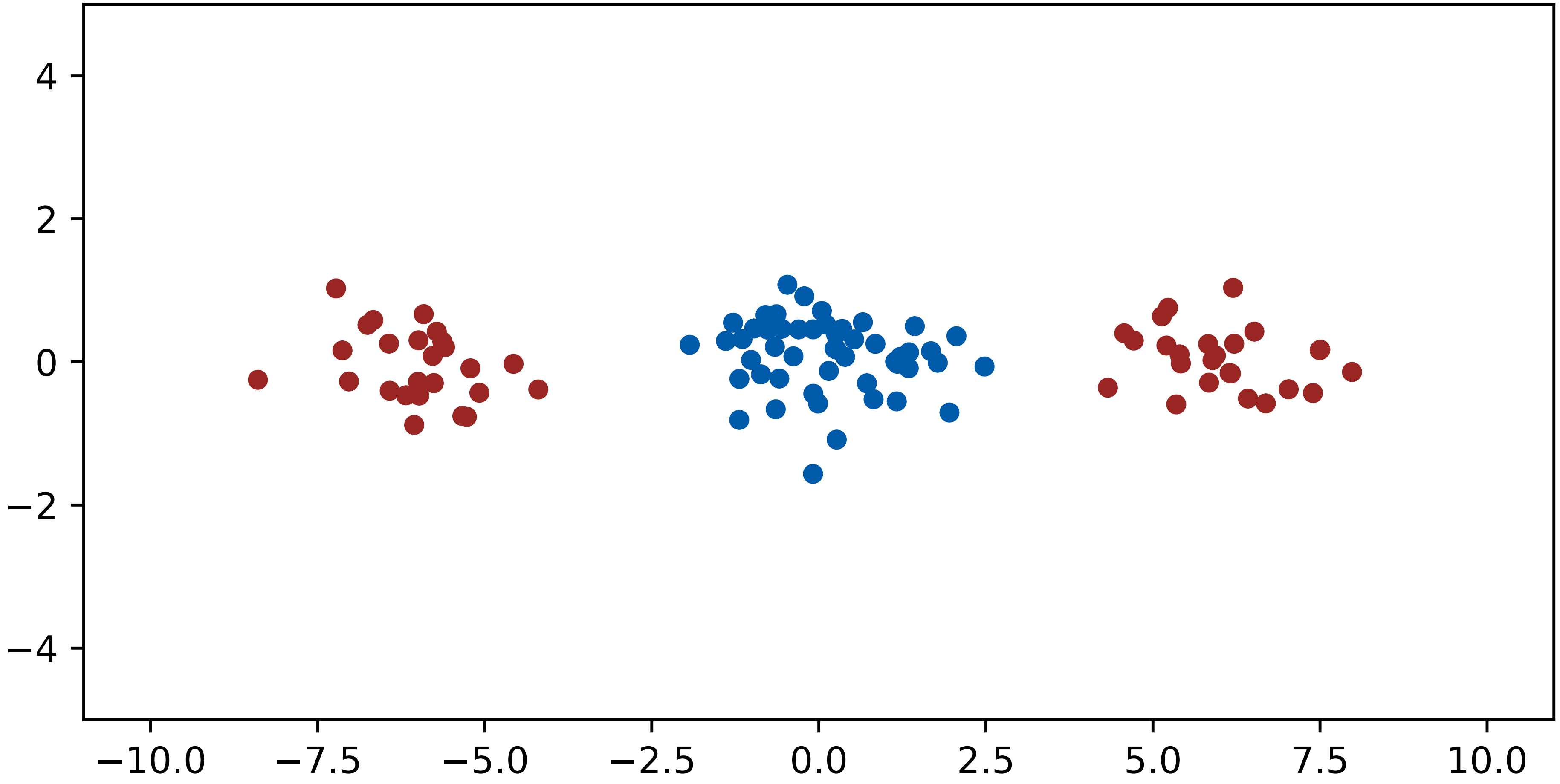
November 06, 2024

Why ensembles?

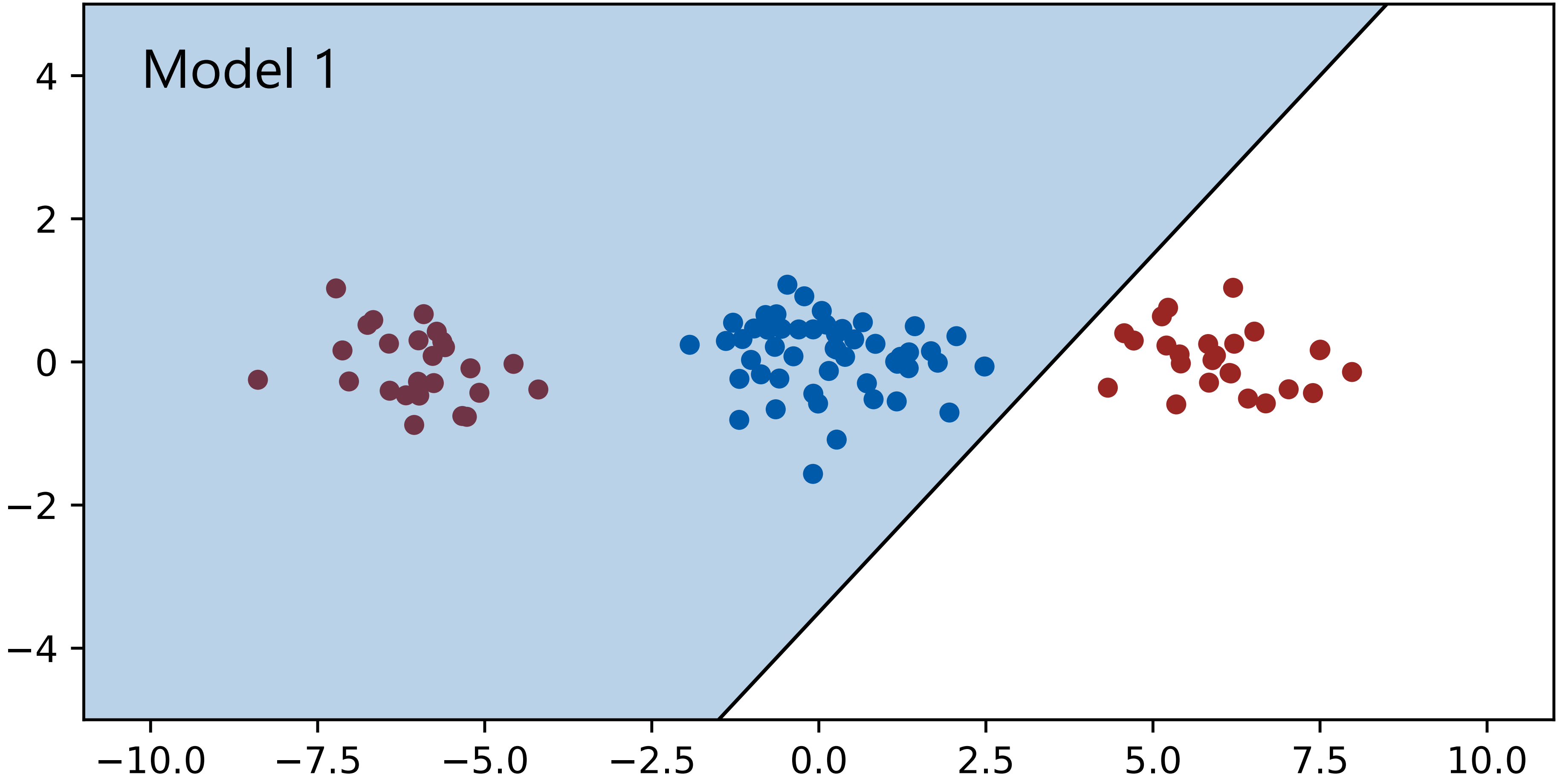
One-vs-rest classification scheme



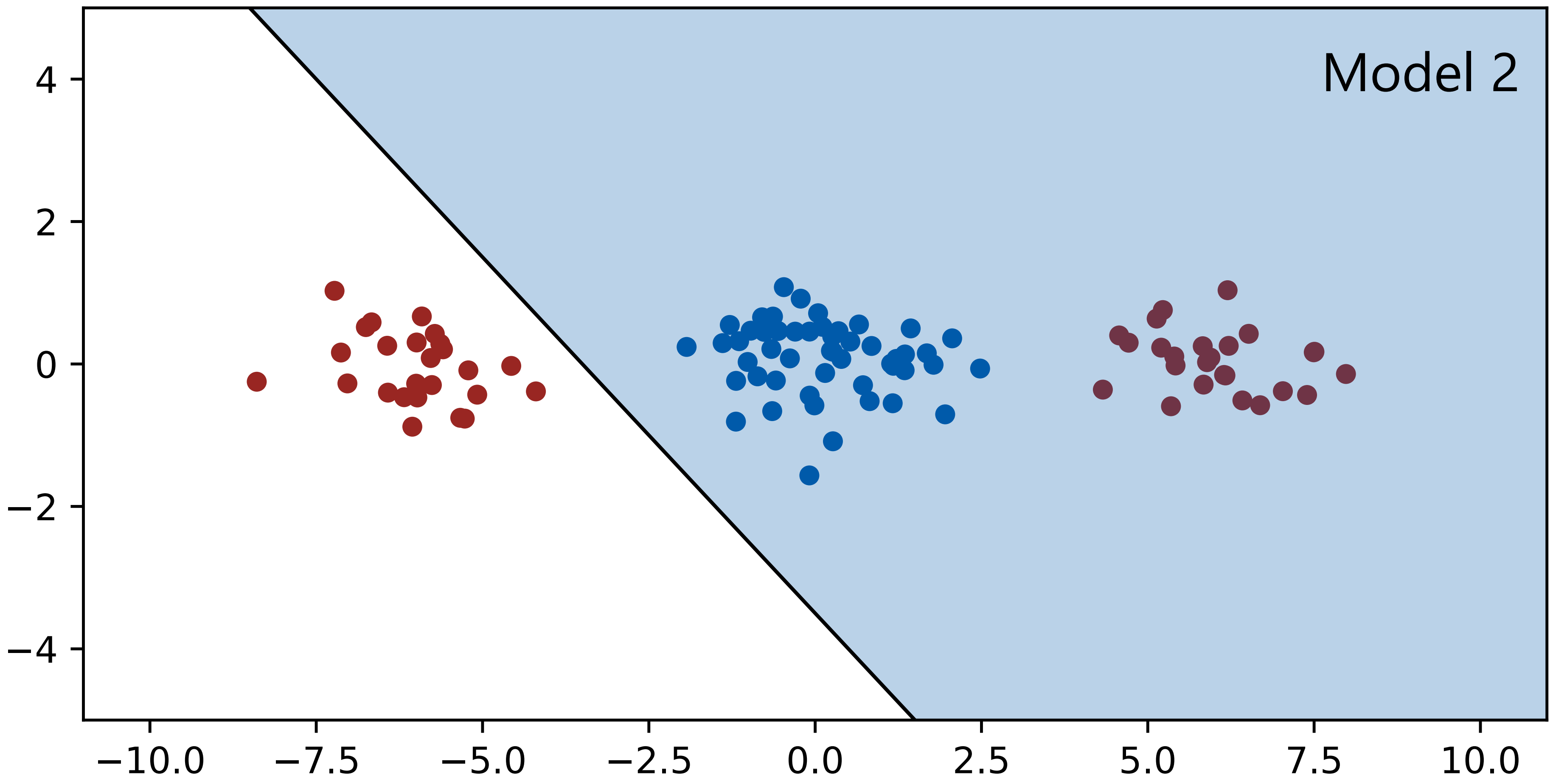
Increasing complexity of simple underfitting models



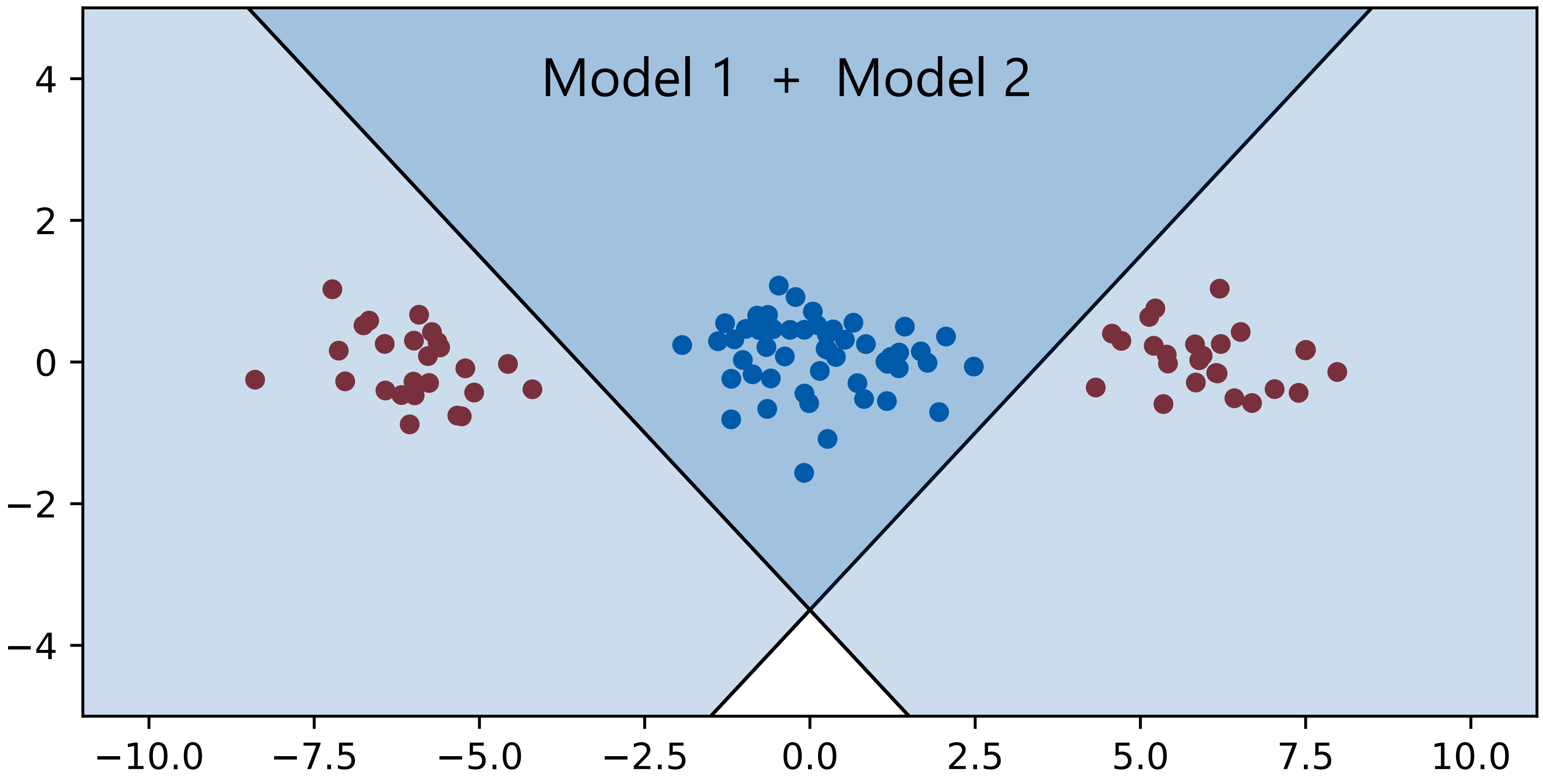
Increasing complexity of simple underfitting models



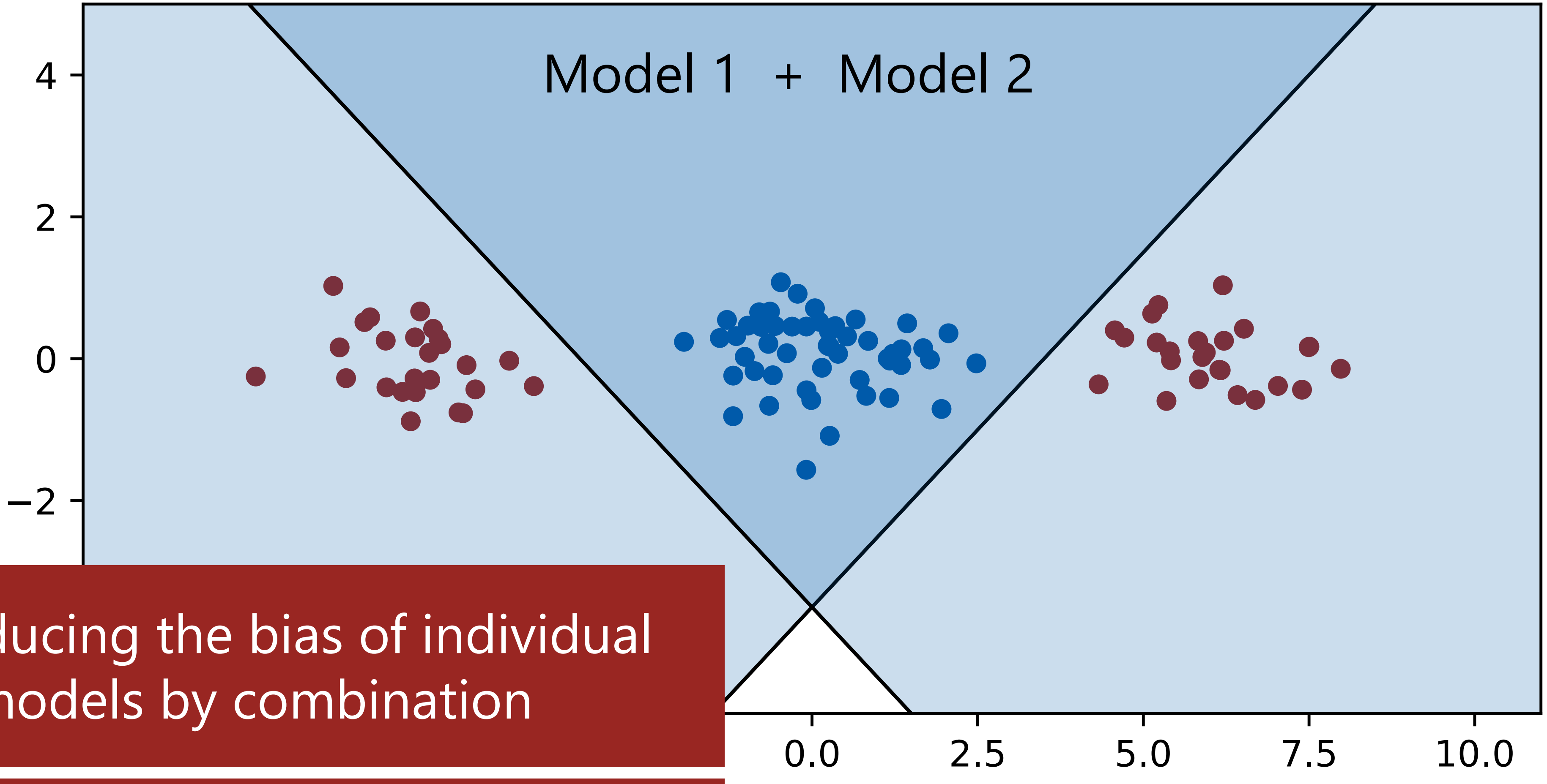
Increasing complexity of simple underfitting models



Increasing complexity of simple underfitting models



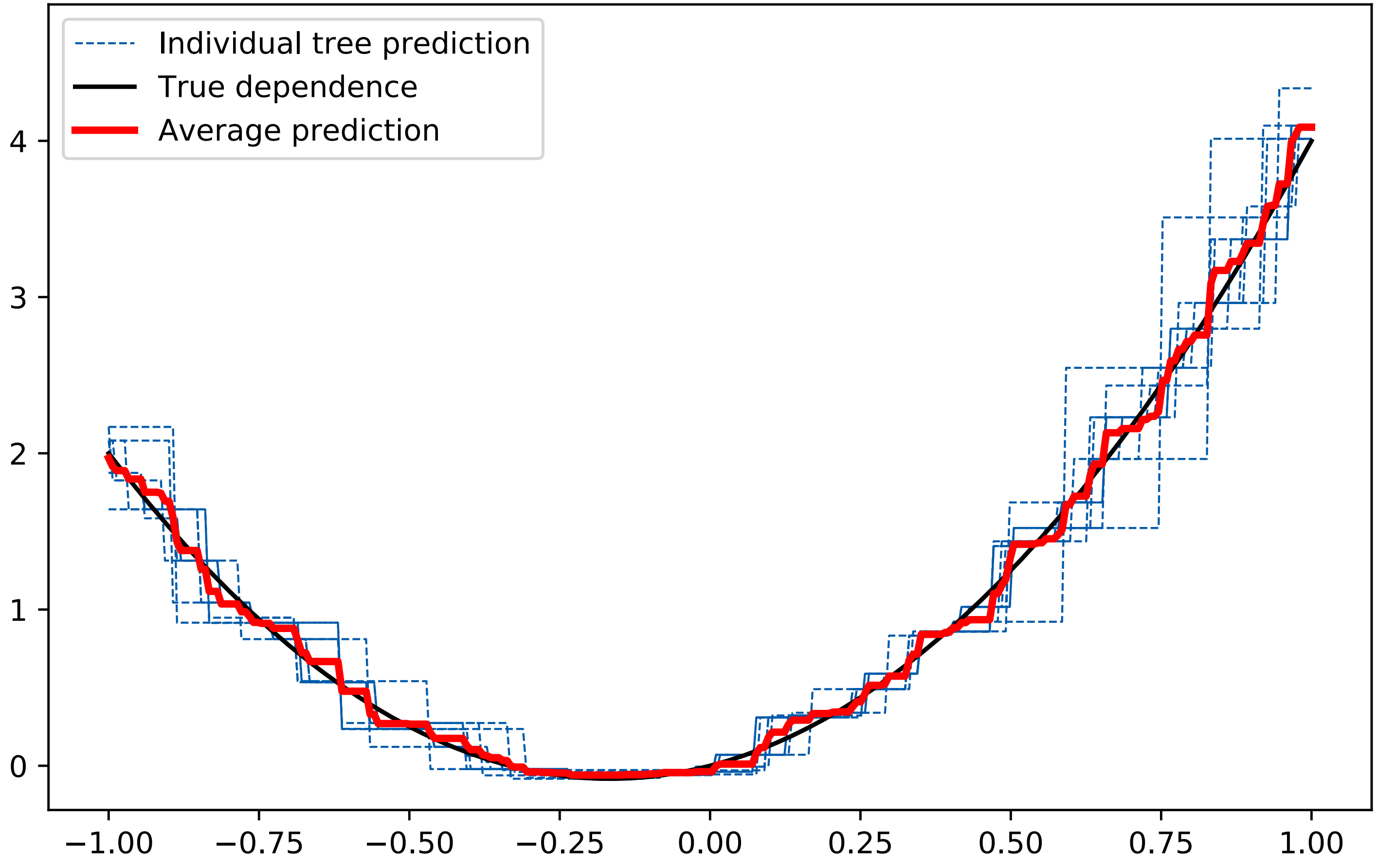
Increasing complexity of simple underfitting models



I.e. reducing the bias of individual models by combination

May increase variance

Reducing variance by averaging



Take a set of overfitting (high-variance) models $\widehat{f_1}(x), \dots, \widehat{f_M}(x)$

- E.g. decision trees trained on different training sets

For regression – average their predictions:

$$F(x) = \frac{1}{M}\sum_{i=1}^{M} \widehat{f_i}(x)$$

For classification – majority voting

Ambiguity decomposition

Consider the following ensemble:

$$F(x) = \sum_i w_i f_i(x), \qquad w_i \geq 0, \qquad \sum_i w_i = 1$$

Then it's easy to show that:

$$\underbrace{(F(x) - y)^2}_{\textbf{Ensemble error}} = \underbrace{\sum_i w_i (f_i(x) - y)^2}_{\textbf{Base learner error}} - \underbrace{\sum_i w_i \big(f_i(x) - F(x)\big)^2}_{\textbf{Ambiguity}}$$

Disagreement reduces the error!

[Ambiguity decomposition proof]

$$F(x) = \sum_i w_i f_i(x), \qquad w_i \geq 0, \qquad \sum_i w_i = 1$$

$$(F(x) - y)^2 = \left(\sum_i w_i f_i(x) - y\right)^2 = \sum_i w_i f_i(x) \cdot F(x) - 2\sum_i w_i f_i(x) \cdot y + \sum_i w_i y^2$$

$$= \sum_i w_i [f_i(x) \cdot F(x) - 2 f_i(x) \cdot y + y^2]$$

$$= \sum_i w_i \left[f_i^2(x) - 2 f_i(x) \cdot y + y^2 - f_i^2(x) + 2 f_i(x) \cdot F(x) - F^2(x) + F^2(x) - f_i(x) \cdot F(x)\right]$$

$$= \sum_i w_i \left[(f_i(x) - y)^2 - \left(f_i(x) - F(x)\right)^2 + F^2(x) - f_i(x) \cdot F(x)\right]$$

$$= \sum_i w_i (f_i(x) - y)^2 - \sum_i w_i \left(f_i(x) - F(x)\right)^2$$

Majority voting classifier error

Assume we have M classifiers with prediction error probability $p < \frac{1}{2}$ each

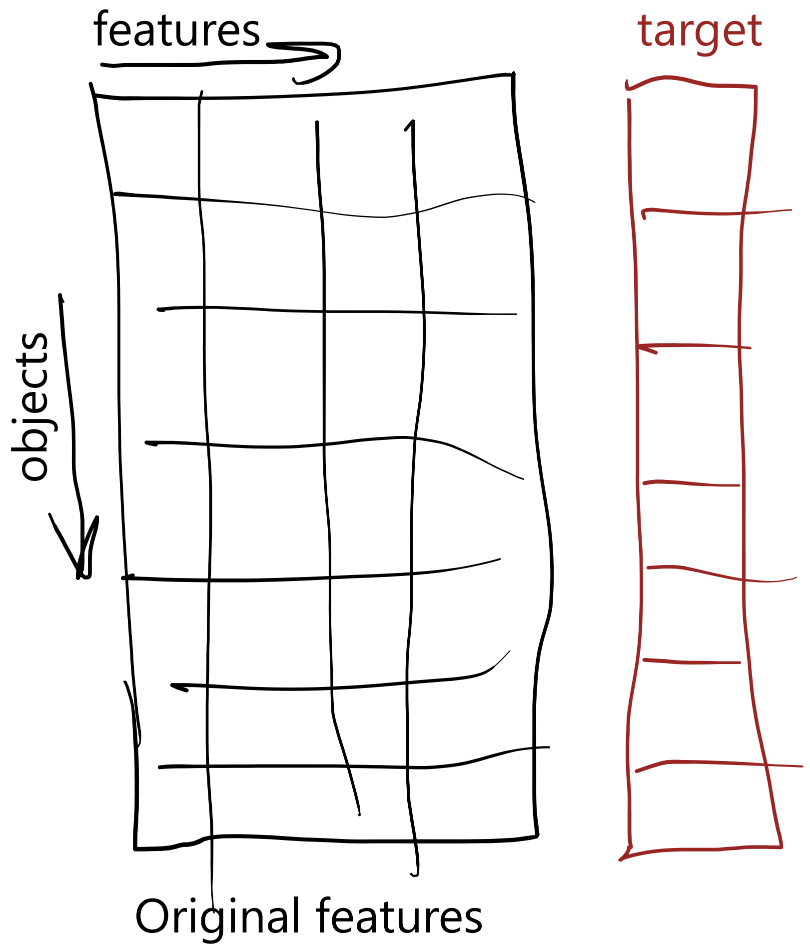
Suppose the guesses are wrong or correct independently of each other

Then major vote error probability $\rightarrow 0$ as $\mathrm{M} \rightarrow +\infty$.

Stacking

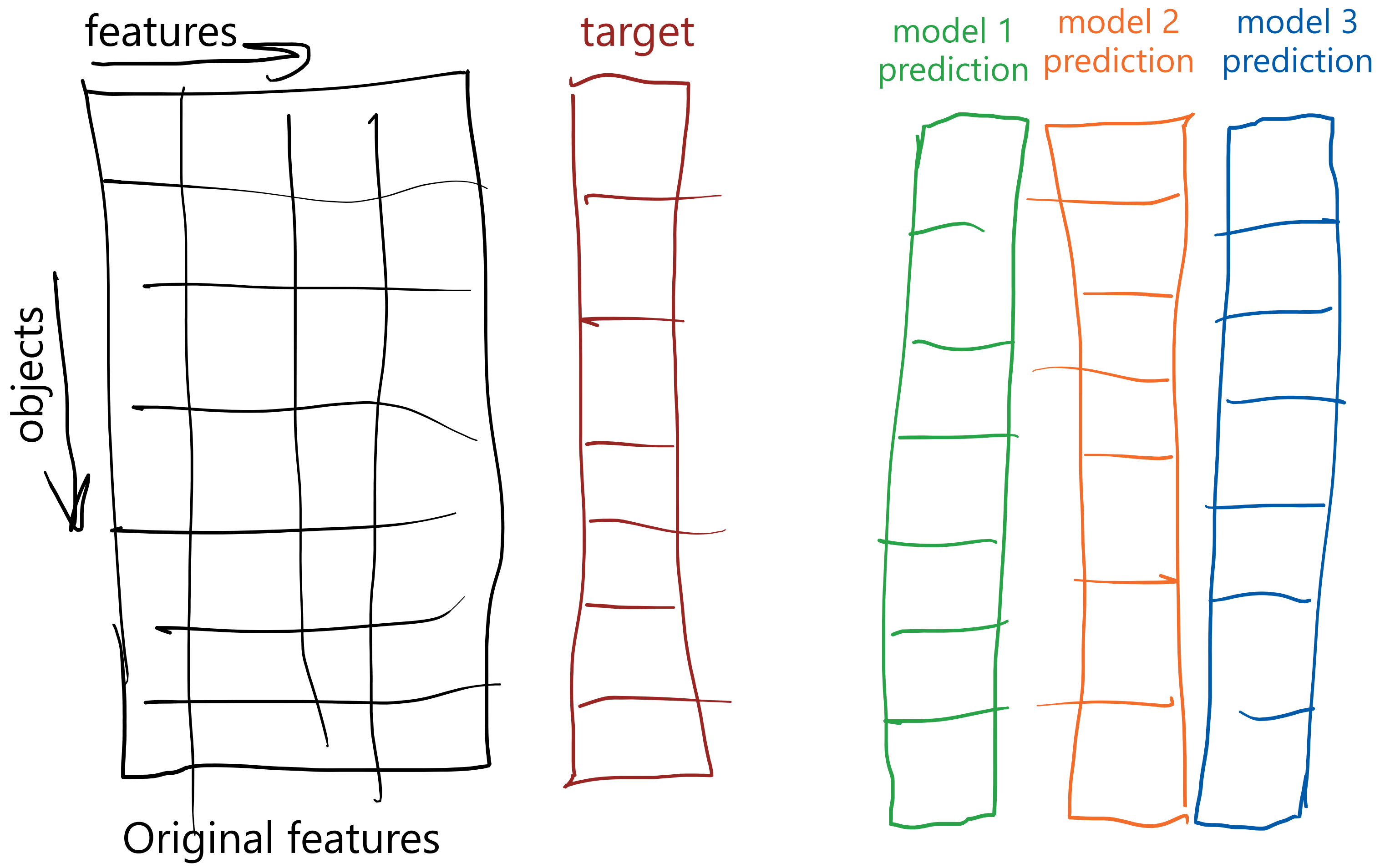
Stacking

Using output of initial models as features for the aggregation model



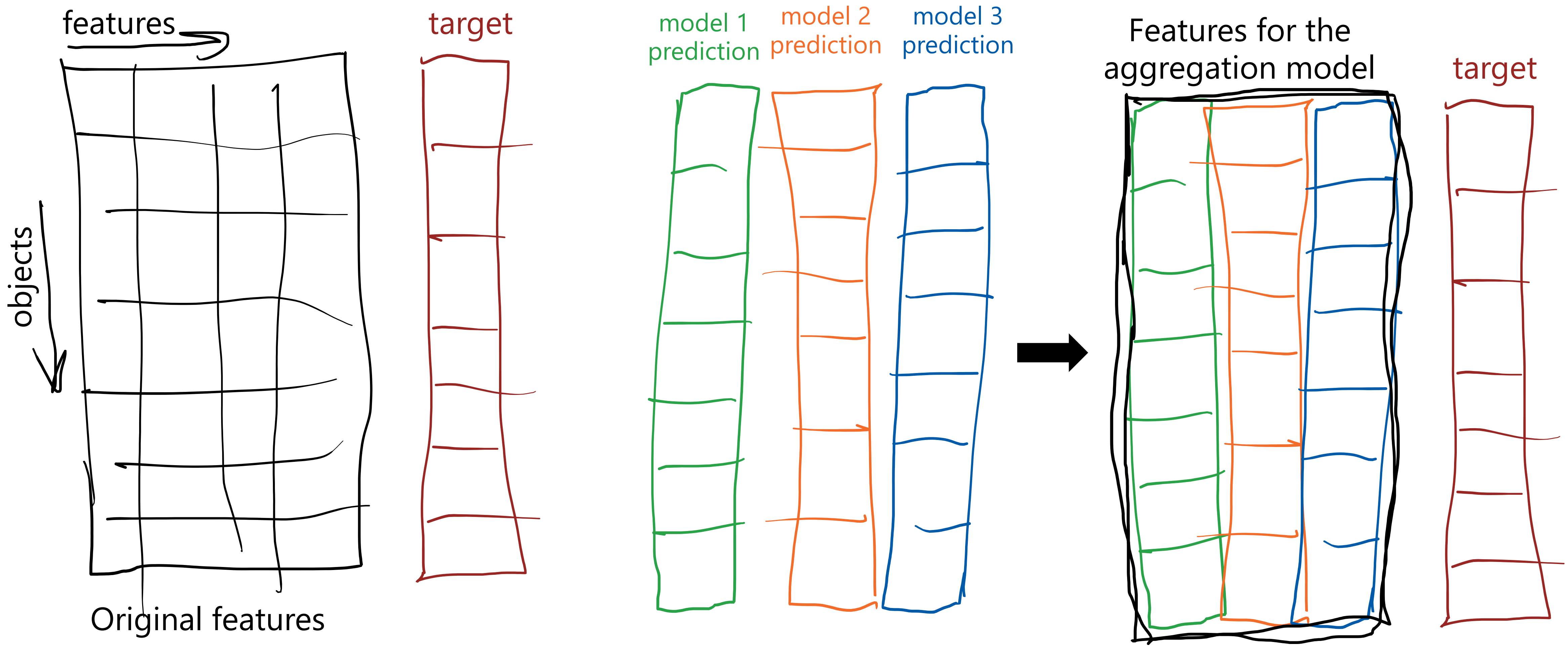
Stacking

Using output of initial models as features for the aggregation model



Stacking

Using output of initial models as features for the aggregation model



Stacking

Using output of initial models as features for the aggregation model

Easy to overfit if using the base models’ predictions from the train data

Solution: use cross-validated estimates

Bagging

Bagging

Bagging = bootstrap aggregating

Generate bootstrap samples (re-sample your data with replacement)

Train independent base models on those samples

Combine their predictions

Example

- Bagged decision trees
- Random Forest (= bagged trees + feature subsampling)
- Extra random trees (= Random Forest + randomized splits)

Boosting

Forward stagewise additive modeling (FSAM)

Loss function: $L(f(x), y)$

Base learners $\widehat{f_{\mathrm{m}}}$

Approximate the output as:

$$F_M(x) = \widehat{f_0}(x) + \sum_{m=1}^{M} \widehat{c_m} \cdot \widehat{f_m}(x)$$

Do so in steps:

- Start from 0, constant or just fit $\widehat{f_0}$ to data
- At each step solve:

$$\left(\widehat{c_m}, \widehat{f_m}\right) = \underset{c,f}{\operatorname{argmin}} \left[\sum_{n=1}^{N} L(F_{m-1}(x_n) + c \cdot f(x_n), y_n) \right]$$

AdaBoost

AdaBoost = FSAM with exponential loss

$$L(f(x), y) = \exp[-y \cdot f(x)]$$

$$y \in \{-1, +1\},$$

– and base learners being binary classifiers: $\widehat{f_m}(x) \in \{-1, +1\}$

Minimization can be done analytically

– if individual learners allow for weighted samples

AdaBoost

Loss at step m:

$$\sum_{n=1}^{N} \exp\left[-y_n\left(F_{m-1}(x_n) + c \cdot f(x_n)\right)\right]$$

AdaBoost

Loss at step m:

$$\sum_{n=1}^{N} \exp\left[-y_n\left(F_{m-1}(x_n) + c \cdot f(x_n)\right)\right]$$

$$= \sum_{n=1}^{N} \exp[-y_n \cdot F_{m-1}(x_n)] \cdot \exp[-y_n \cdot c \cdot f(x_n)] = \sum_{n=1}^{N} w_n \cdot \exp[-y_n \cdot c \cdot f(x_n)]$$

AdaBoost

Loss at step m:

$$\sum_{n=1}^{N} \exp\left[-y_n\left(F_{m-1}(x_n) + c \cdot f(x_n)\right)\right]$$

$$= \sum_{n=1}^{N} \exp[-y_n \cdot F_{m-1}(x_n)] \cdot \exp[-y_n \cdot c \cdot f(x_n)] = \sum_{n=1}^{N} w_n \cdot \exp[-y_n \cdot c \cdot f(x_n)]$$

$$= \sum_{y_n = f(x_n)} w_n \cdot e^{-c} + \sum_{y_n \neq f(x_n)} w_n \cdot e^{c}$$

AdaBoost

Loss at step m:

$$\sum_{n=1}^{N} \exp\left[-y_n\left(F_{m-1}(x_n) + c \cdot f(x_n)\right)\right]$$

$$= \sum_{n=1}^{N} \exp[-y_n \cdot F_{m-1}(x_n)] \cdot \exp[-y_n \cdot c \cdot f(x_n)] = \sum_{n=1}^{N} w_n \cdot \exp[-y_n \cdot c \cdot f(x_n)]$$

$$= \sum_{y_n = f(x_n)} w_n \cdot e^{-c} + \sum_{y_n \neq f(x_n)} w_n \cdot e^{c}$$

$$= \sum_{n=1}^{N} w_n \cdot e^{-c} + \sum_{y_n \neq f(x_n)} w_n \cdot (e^{c} - e^{-c})$$

AdaBoost

So at step m we optimize:

$$\left(\widehat{c_m}, \widehat{f_m}\right) = \underset{c,f}{\operatorname{argmin}} \left[\sum_{n=1}^{N} w_n \cdot e^{-c} + \sum_{y_n \neq f(x_n)} w_n \cdot \left(e^{c} - e^{-c}\right) \right]$$

$$w_n = \exp[-y_n \cdot F_{m-1}(x_n)]$$

In other words, find $\widehat{f_m}$, s.t. $\sum_{\widehat{f_m}(x_n) \neq y_n} w_n$ is minimized

Easy to show that for c:

$$c_m = \frac{1}{2} \ln \frac{\sum_{\widehat{f_m}(x_n) = y_i} w_n}{\sum_{\widehat{f_m}(x_n) \neq y_i} w_n}$$

AdaBoost

So at step m we optimize:

$$(\widehat{c_m}, \widehat{f_m}) = \underset{c,f}{\operatorname{argmin}} \left[\sum_{n=1}^{N} w_n \cdot e^{-c} + \sum_{y_n \neq f(x_n)} w_n \cdot (e^c - e^{-c}) \right]$$

$$w_n = \exp[-y_n \cdot F_{m-1}(x_n)]$$

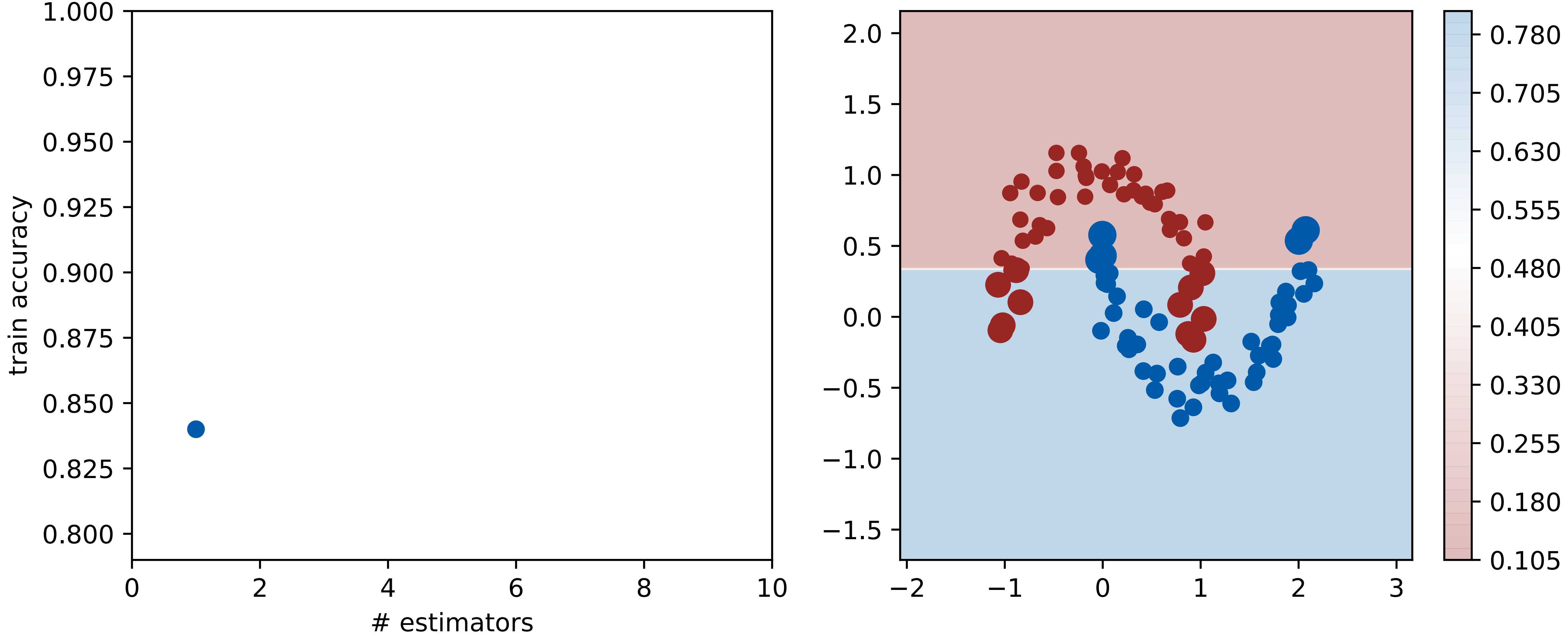
pay more attention to objects predicted wrongly

In other words, find $\widehat{f_m}$, s.t. $\sum_{\widehat{f_m}(x_n) \neq y_n} w_n$ is minimized

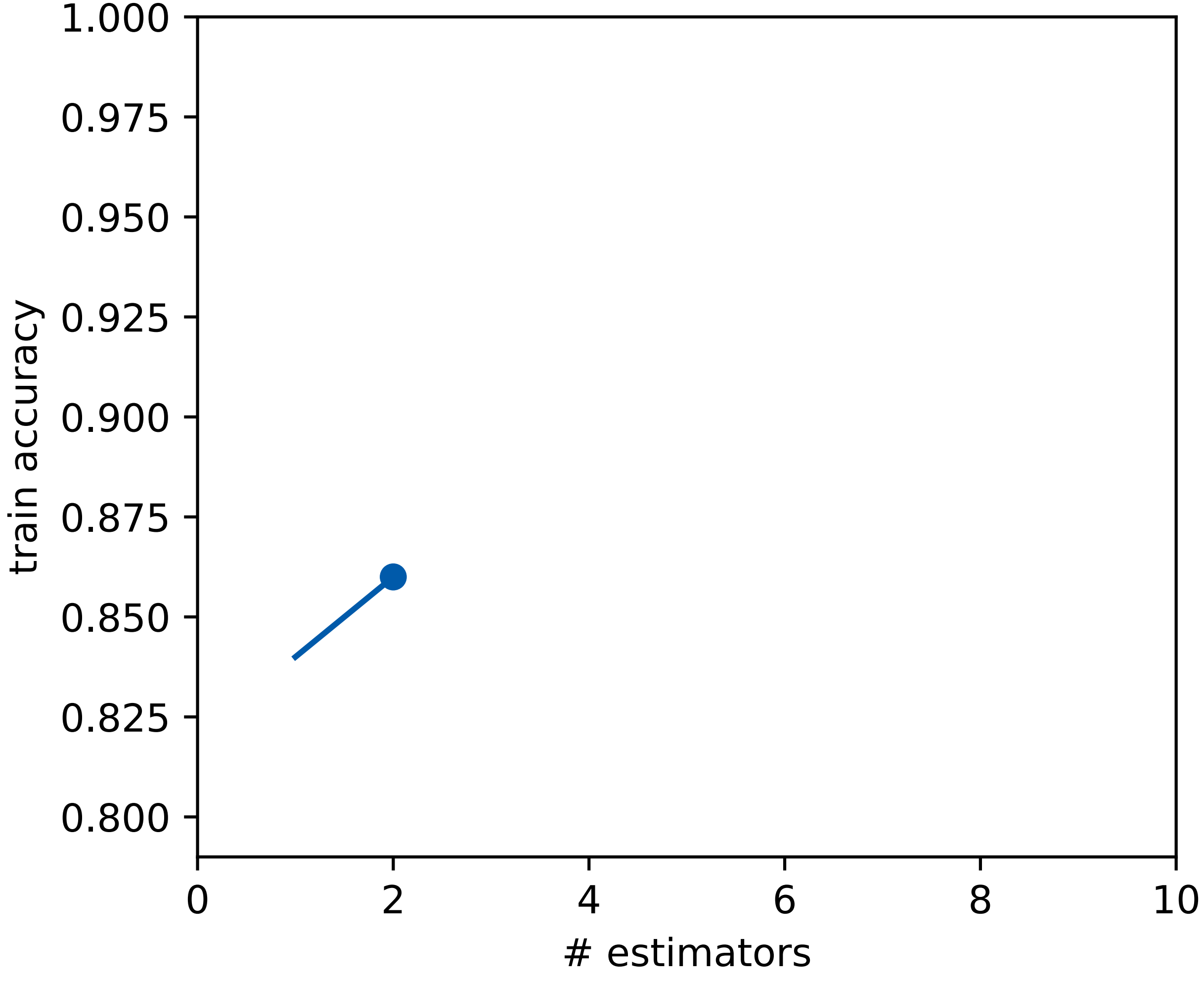
Easy to show that for c:

$$c_m = \frac{1}{2} \ln \frac{\sum_{\widehat{f_m}(x_n) = y_i} w_n}{\sum_{\widehat{f_m}(x_n) \neq y_i} w_n}$$

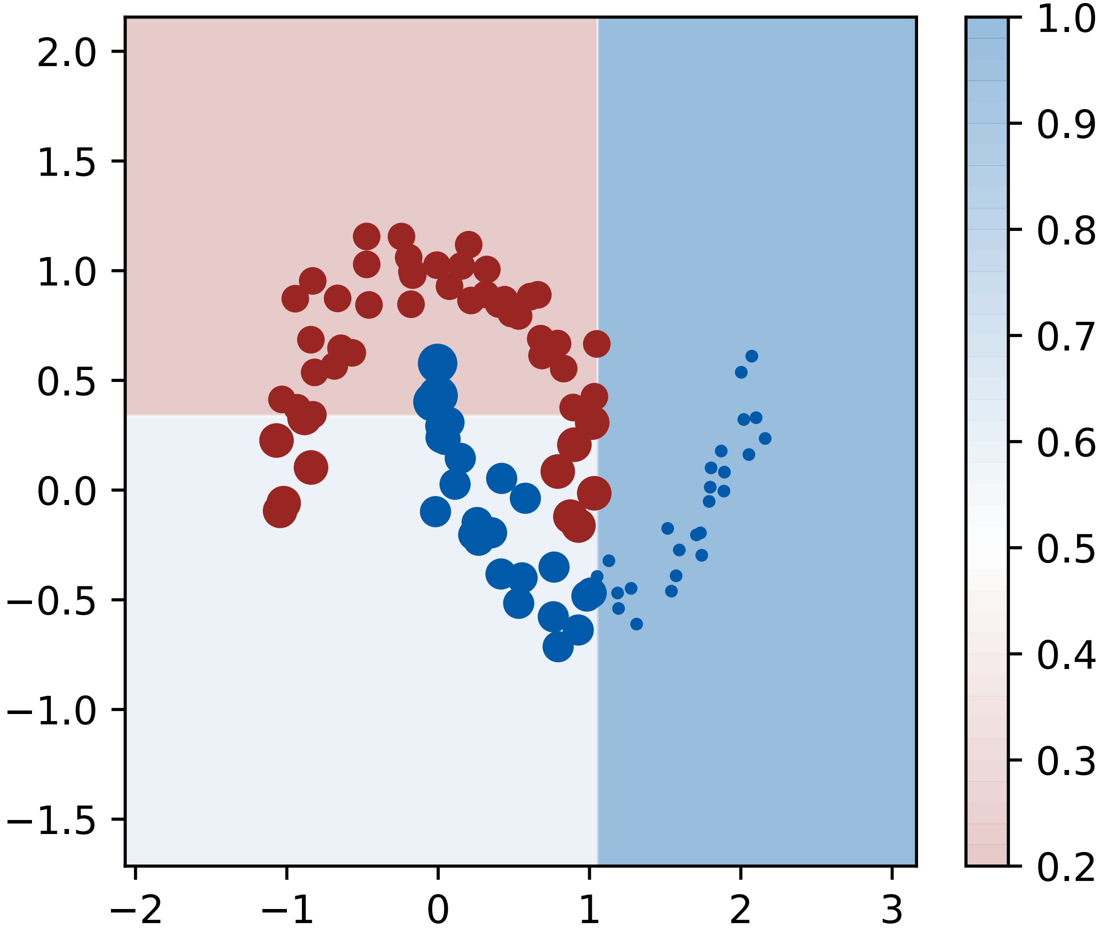
Example


train accuracy
1.000
0.975
0.950
0.925
0.900
0.875
0.850
0.825
0.800
0
2
4
6
8
10
estimators
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2
−1
0
1
2
3
0.780
0.705
0.630
0.555
0.480
0.405
0.330
0.255
0.180
0.105

Example

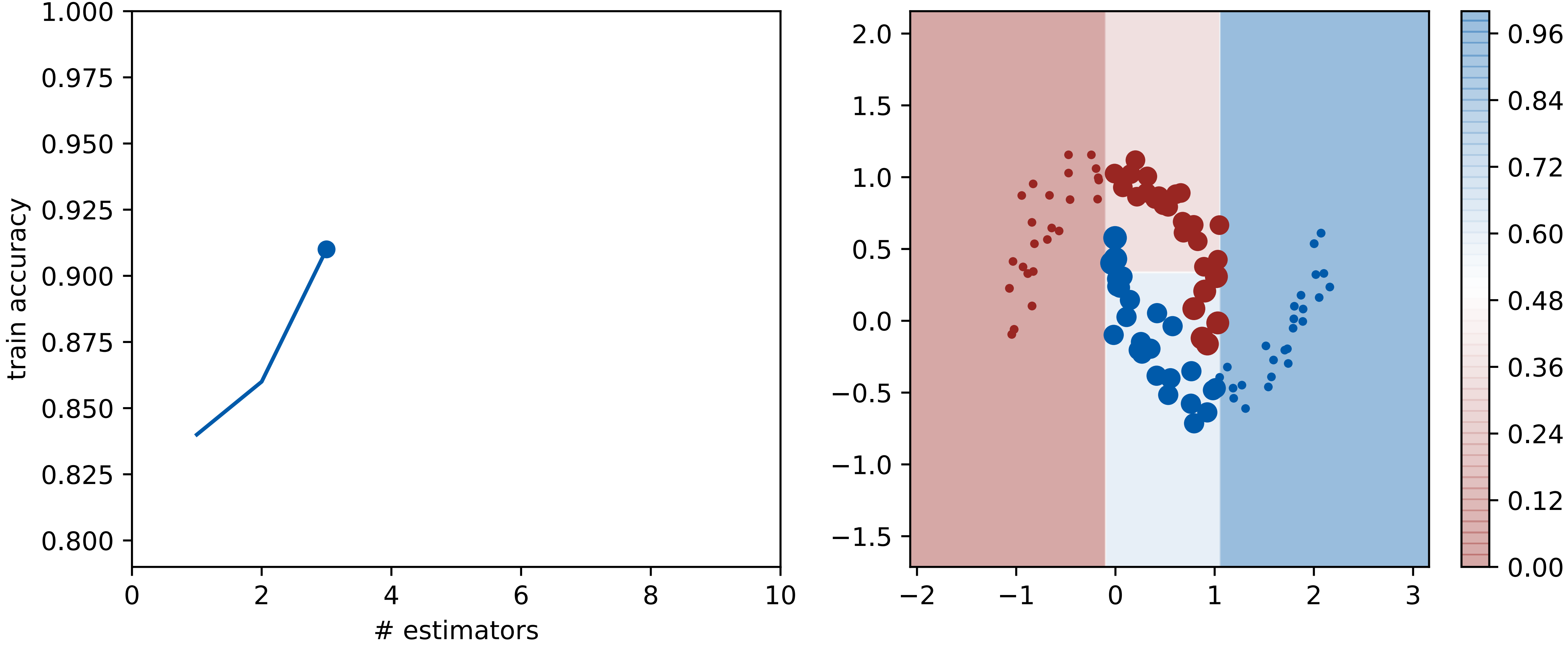


1.000
0.975
0.950
0.925
0.900
0.875
0.850
0.825
0.800
train accuracy
0
2
4
6
8
10
estimators

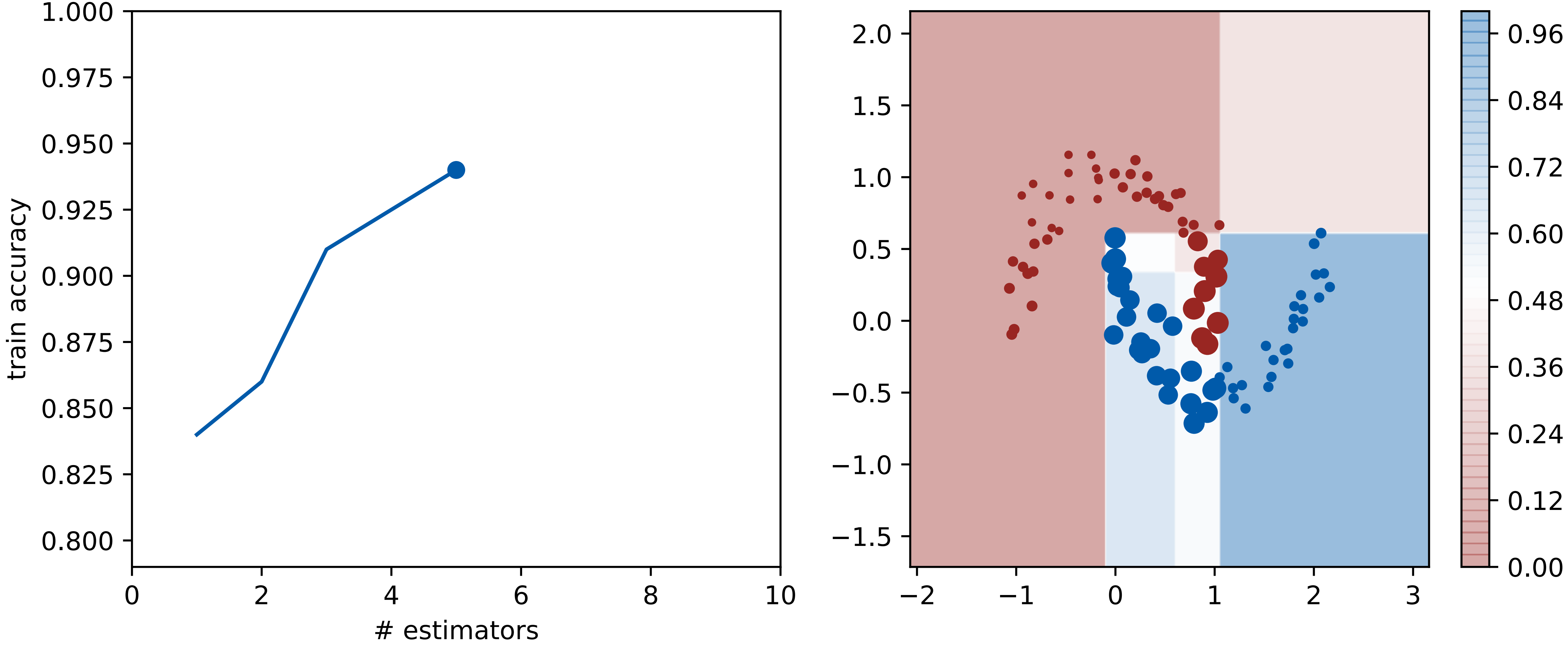


2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2
−1
0
1
2
3
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2

Example

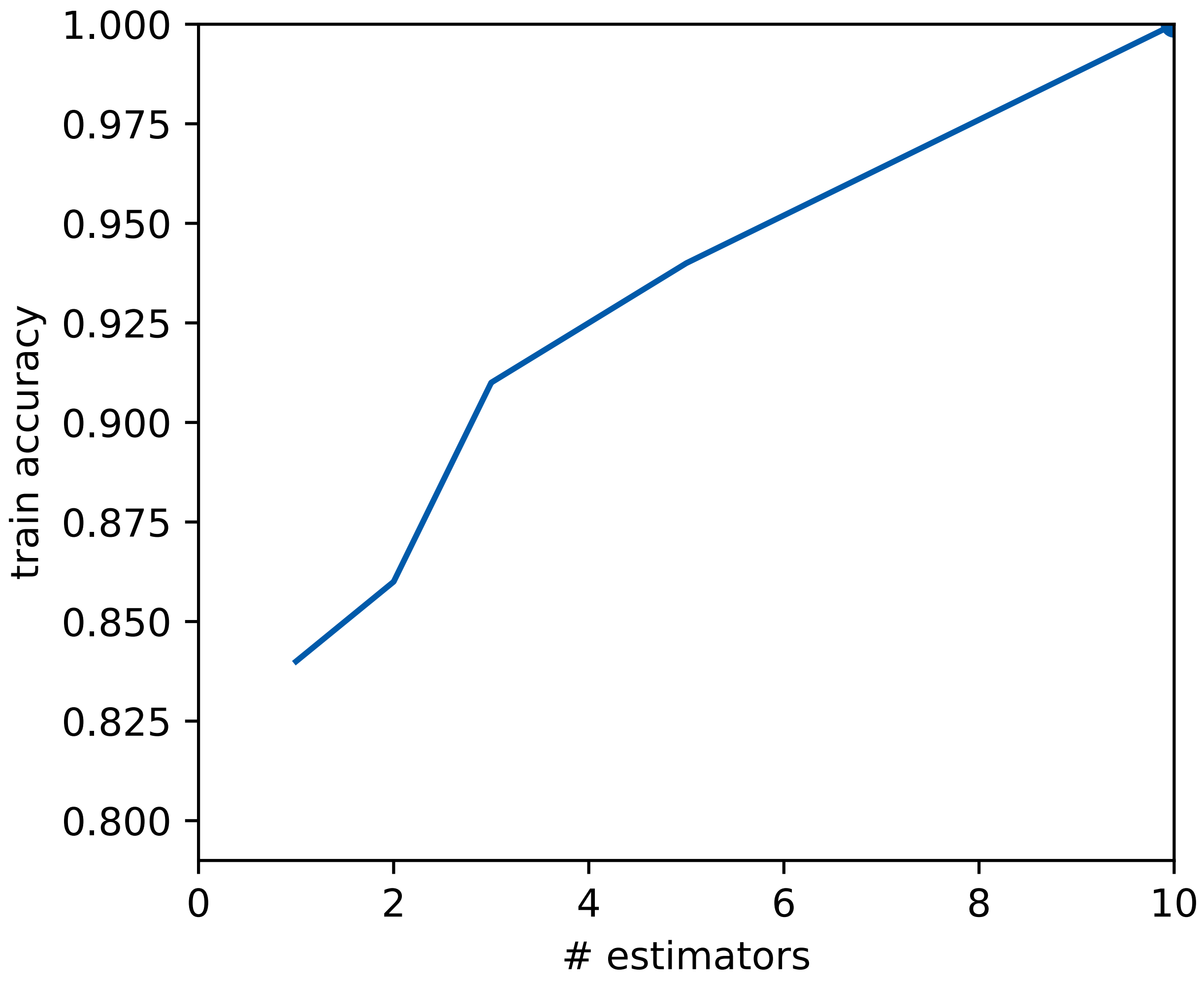

1.000
0.975
0.950
0.925
0.900
0.875
0.850
0.825
0.800
train accuracy
0
2
4
6
8
10
estimators
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2
−1
0
1
2
3
0.96
0.84
0.72
0.60
0.48
0.36
0.24
0.12
0.00

Example

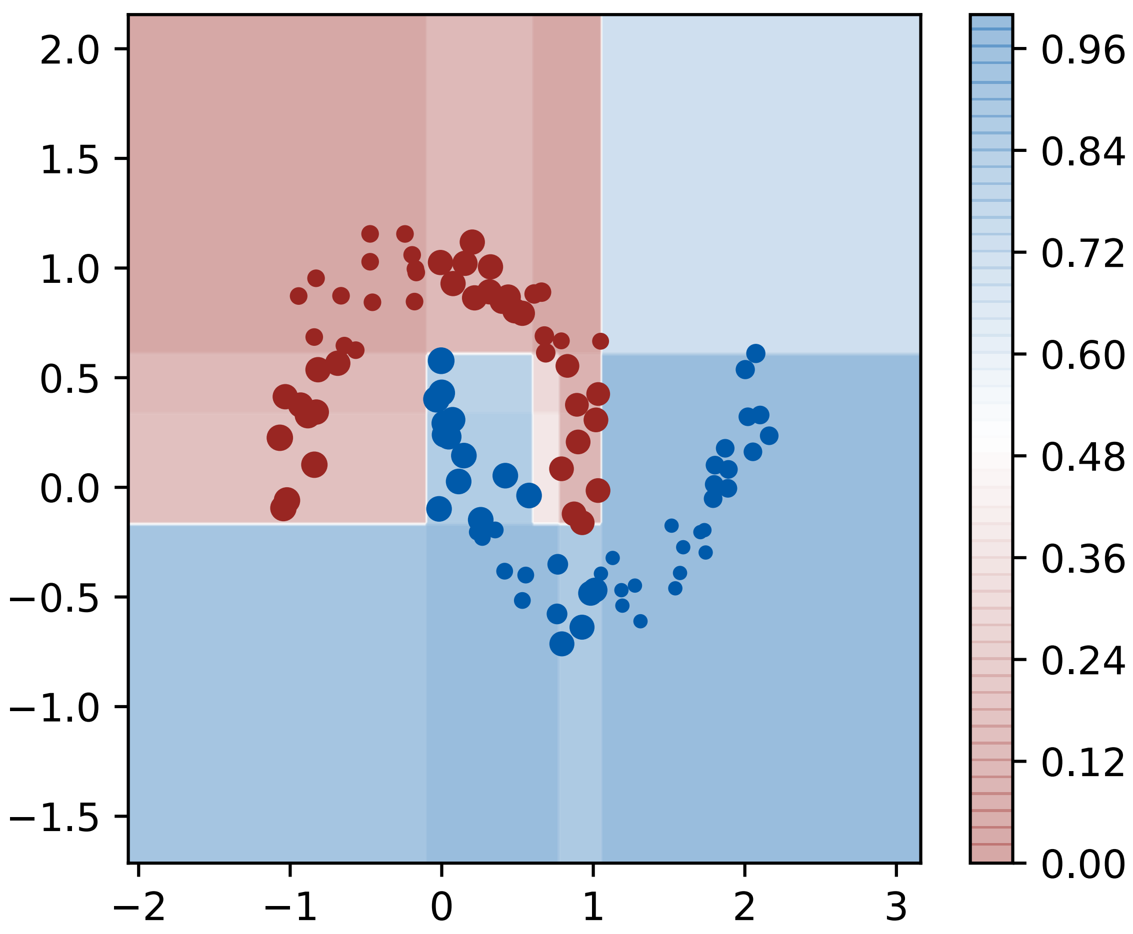


1.000
0.975
0.950
0.925
0.900
0.875
0.850
0.825
0.800
train accuracy
0
2
4
6
8
10
estimators
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2
−1
0
1
2
3
0.96
0.84
0.72
0.60
0.48
0.36
0.24
0.12
0.00

Example



1.000
0.975
0.950
0.925
0.900
0.875
0.850
0.825
0.800
train accuracy
0
2
4
6
8
10
estimators



2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2
−1
0
1
2
3
0.96
0.84
0.72
0.60
0.48
0.36
0.24
0.12
0.00

Gradient Boosting

FSAM minimization cannot be solved analytically for a general loss function

Find approximation (linear):

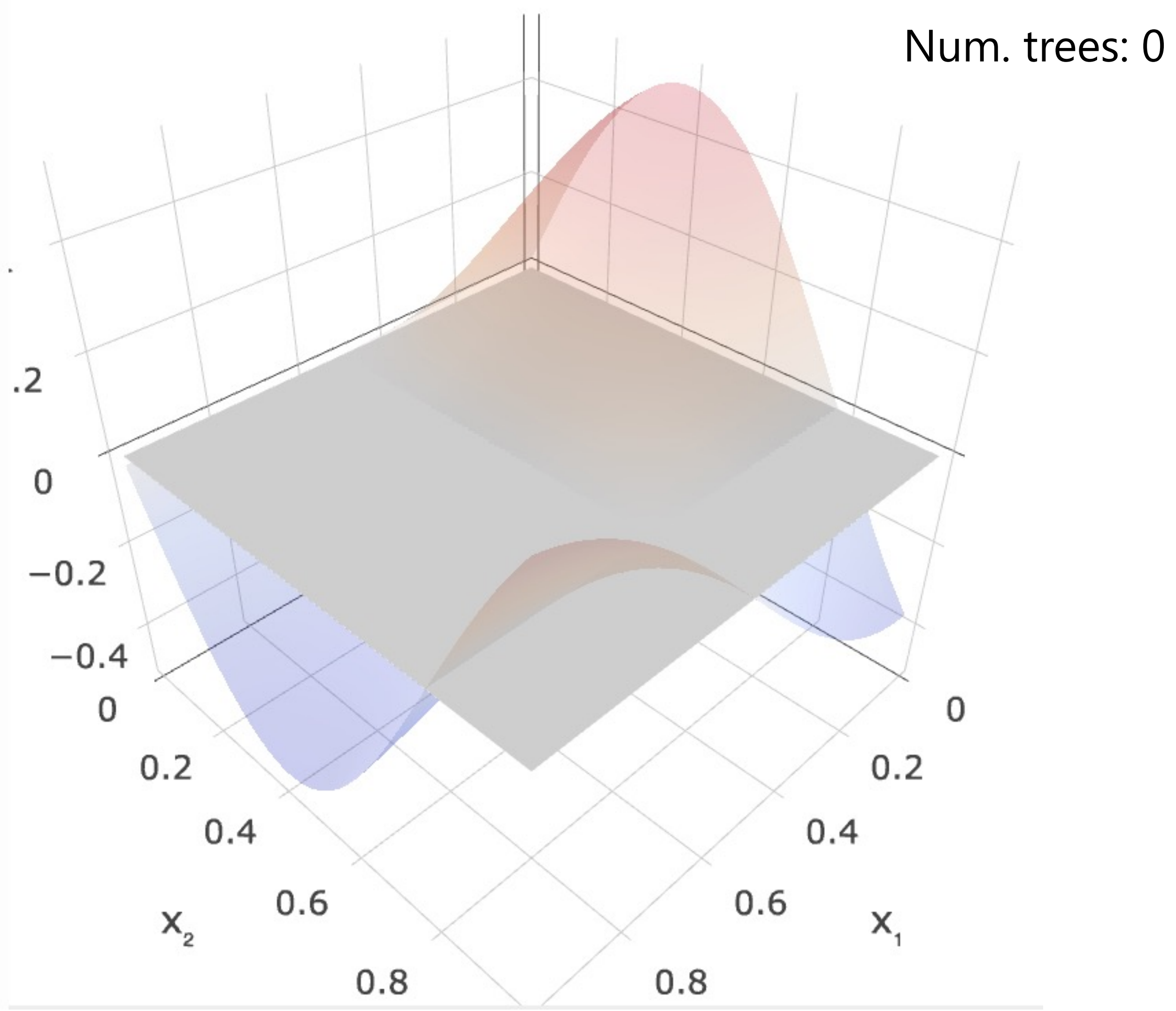
$$L(F(x) + f(x), y) \approx L(F(x), y) + \frac{\partial L(F, y)}{\partial F} f(x)$$

Gradient shows the direction of maximal increase $\Rightarrow$ fit $f(x)$ to the negative of the gradient

Then solve for c:

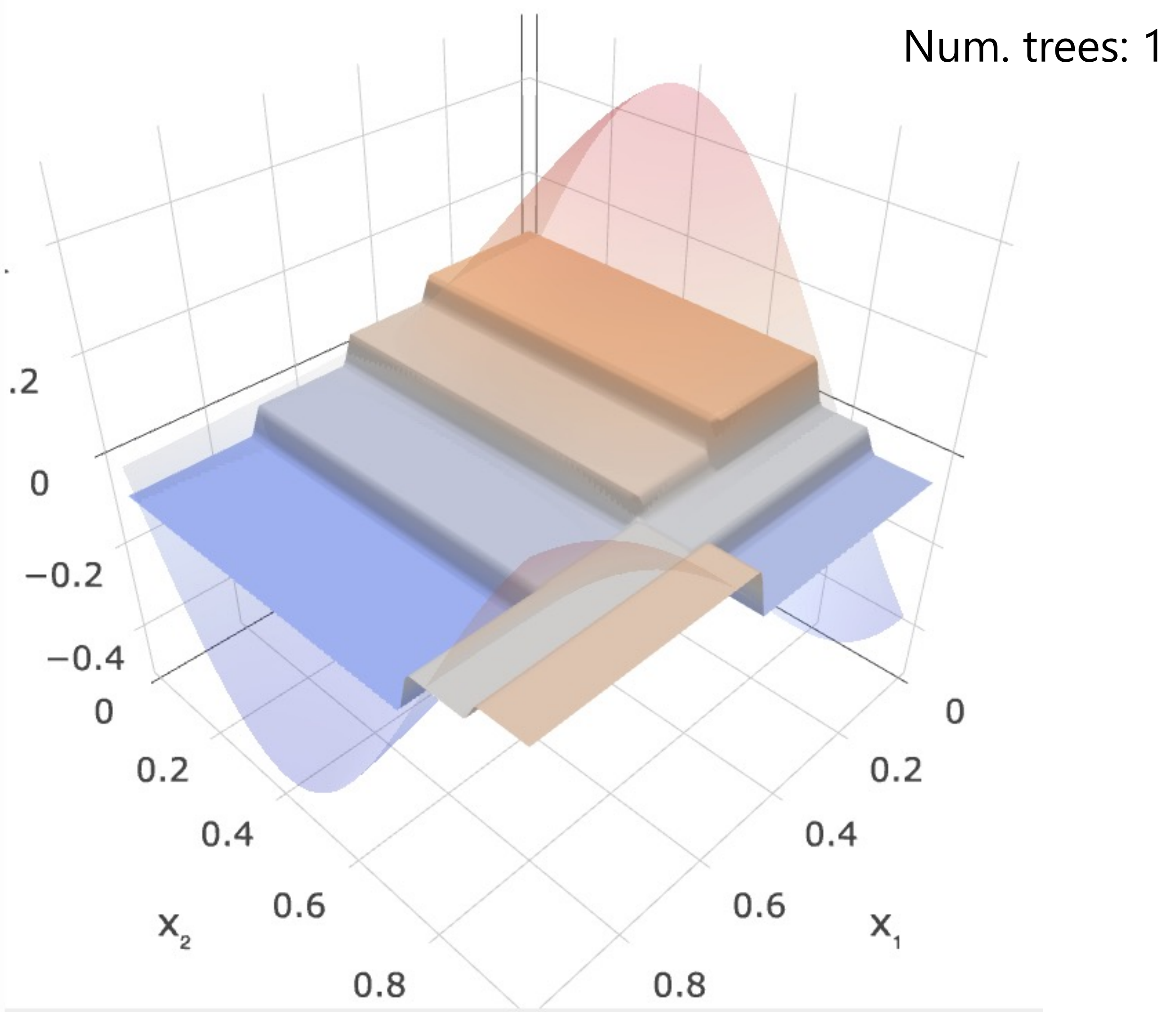
$$L(F(x) + c \cdot f(x), y) \rightarrow \min_c$$

Example



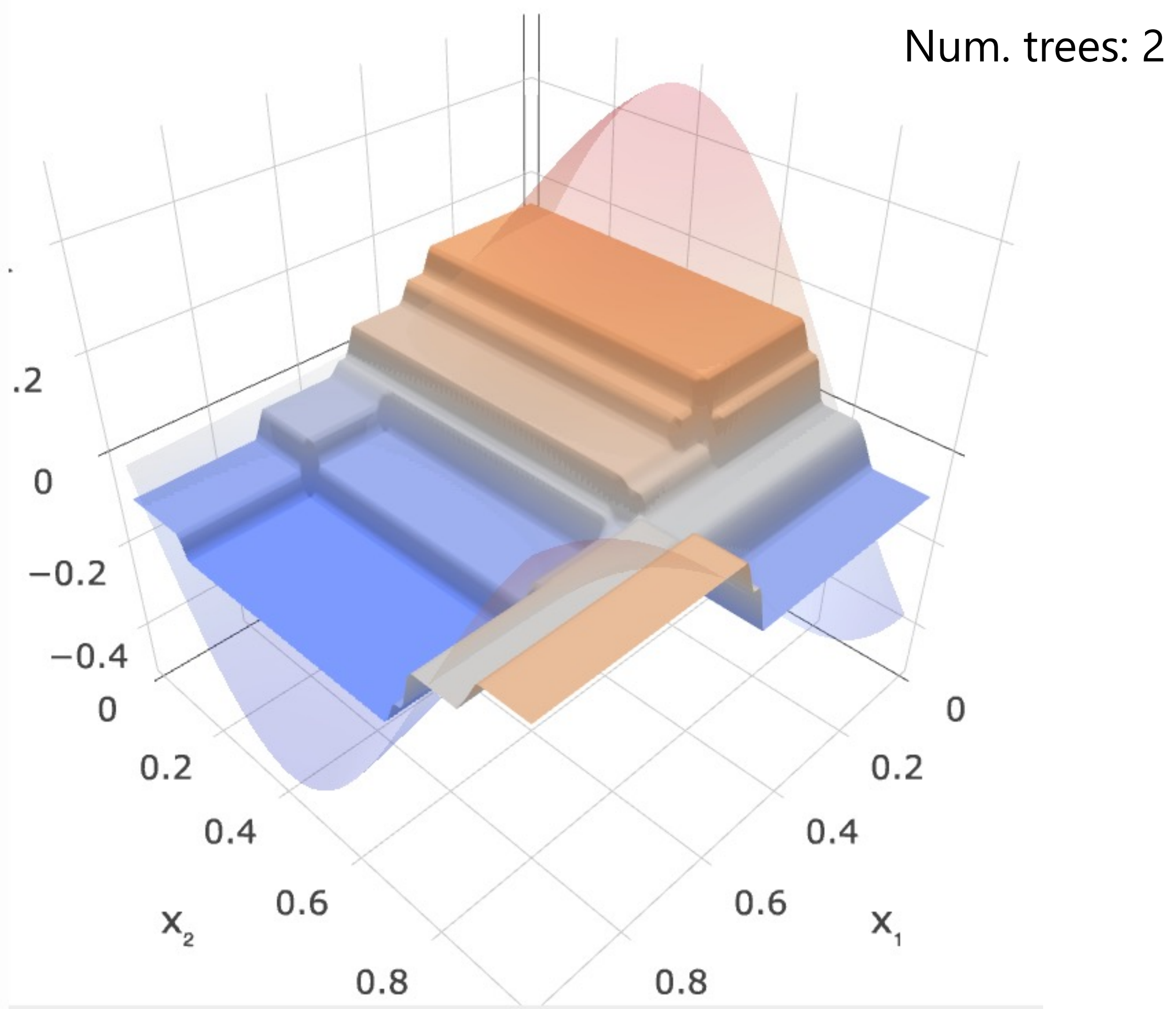
Nice demo: http://arogozhnikov.github.io/2016/06/24/gradient_boosting_explained.htm

Example



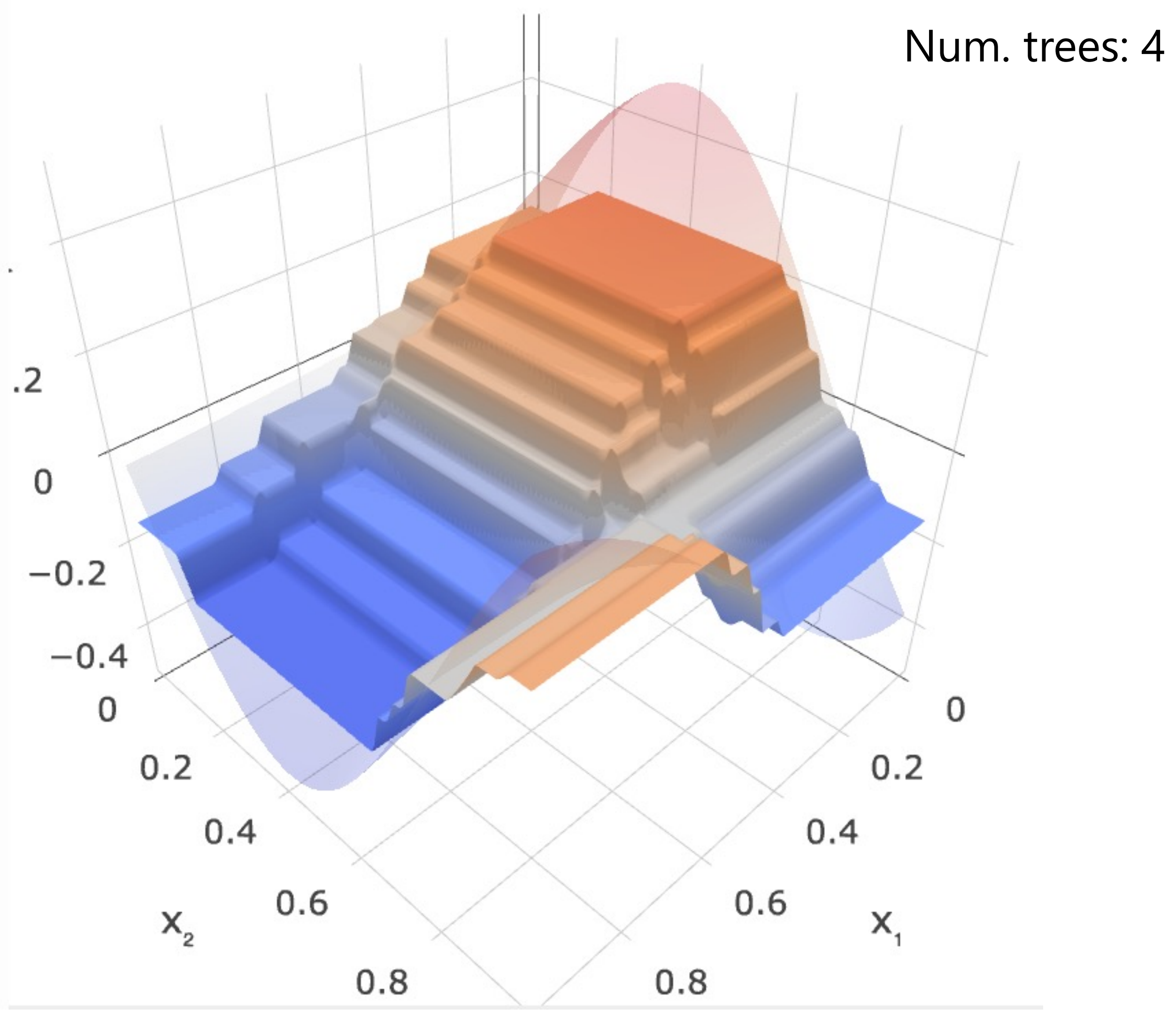
Nice demo: http://arogozhnikov.github.io/2016/06/24/gradient_boosting_explained.htm

Example



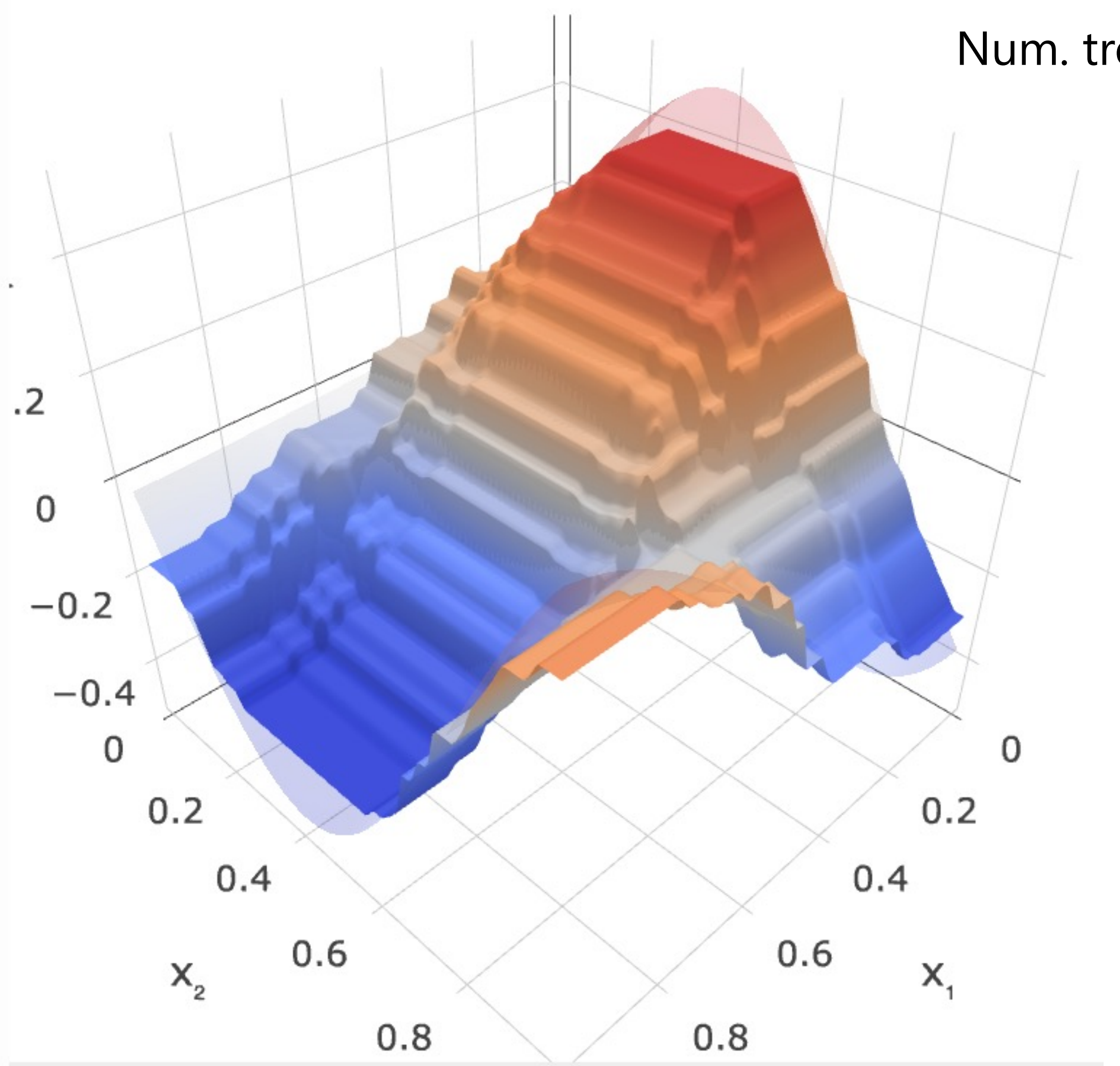
Nice demo: http://arogozhnikov.github.io/2016/06/24/gradient_boosting_explained.htm

Example



Nice demo: http://arogozhnikov.github.io/2016/06/24/gradient_boosting_explained.htm

Example



Nice demo: http://arogozhnikov.github.io/2016/06/24/gradient_boosting_explained.htm

Quadratic approximation

$$L(F(x)+f(x),y) \approx L(F(x),y) + \frac{\partial L(F,y)}{\partial F} f(x) + \frac{1}{2}\frac{\partial^2 L(F,y)}{\partial F^2}\left(f(x)\right)^2$$

$$= \underbrace{\frac{1}{2}\frac{\partial^2 L(F,y)}{\partial F^2}}_{\textbf{sample weights}} \left(f(x) + \underbrace{\frac{\frac{\partial L(F,y)}{\partial F}}{\frac{\partial^2 L(F,y)}{\partial F^2}}}_{\textbf{negative of the fitting targets}} \right)^2 + const\left(f(x)\right)$$

See also: very nice explanation in the xgboost documentation:

- https://xgboost.readthedocs.io/en/latest/tutorials/model.html

Summary

Ensembling may allow to reduce variance and/or bias of the base learners

Averaging the prediction of **independent** high-variance models reduces the variance

With bagging, the base learners are made (quasi-) independent using bootstrapping

- Can be done in parallel

With boosting, the base learners are built in sequence, each next one trying to improve upon the mistakes of the previous steps

- One of the most powerful classes of models

Question to you: does it make sense to boost linear regression?

Thank you!

Majid Sohrabi

msohrabi@hse.ru