



CAMBRIDGE

# Lecture 3: Inductive bias, regularization, and ensemble methods

Stefan Bucher

MACHINE LEARNING IN ECONOMICS  
UNIVERSITY OF CAMBRIDGE

Stefan Bucher



Prince (2023, chaps. 5, 9); Murphy (2022, chap. 18).<sup>1</sup>

- . Figures taken or adapted from Prince (2023). All rights belong to the original author and publisher. These materials are intended solely for educational purposes.

# Generalization and inductive bias

- **Generalization:** performance on unseen (test) data
- **Inductive bias:** assumptions that favour some solutions over others (e.g. smoothness, sparsity); determines model behavior in underdetermined regimes and hence generalization.
- Today: concrete tools to improve generalization.

# Outline

- . **Regularization** — constrain or perturb parametric models (L2, L1, dropout, transfer, etc.).
- : **Nonparametric methods** — Classification and Regression Trees (CART).
- ): **Ensemble methods** — Bagging, random forests, boosting, XGBoost.
- |: **Interpretability** — Shapley values and SHAP for feature attribution.
- |: **Prediction vs. inference**

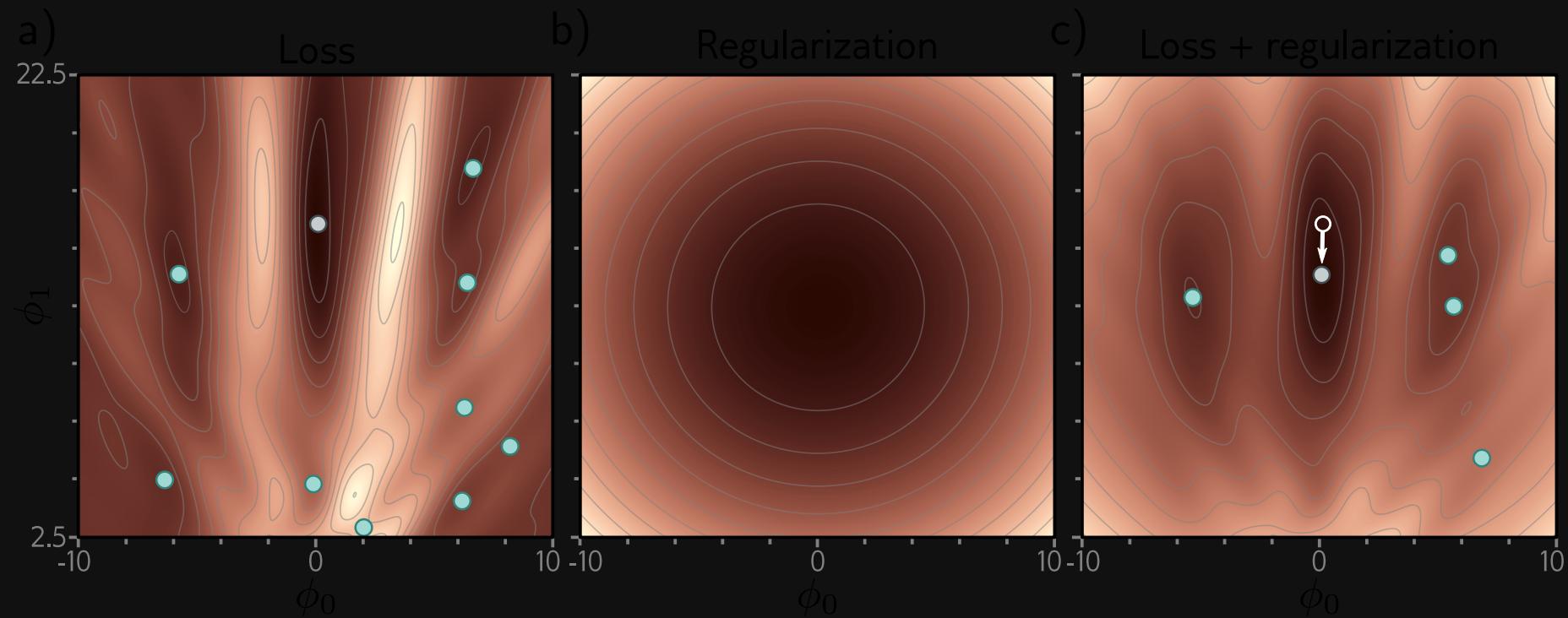
# Regularization

Goal: Reduce generalization gap between training and test performance.

Prince (2023, chap. 9)

# Explicit Regularization

$$\hat{\phi} = \arg \min_{\phi} \sum_{i=1}^I l_i[x_i, y_i] + \lambda g[\phi]$$



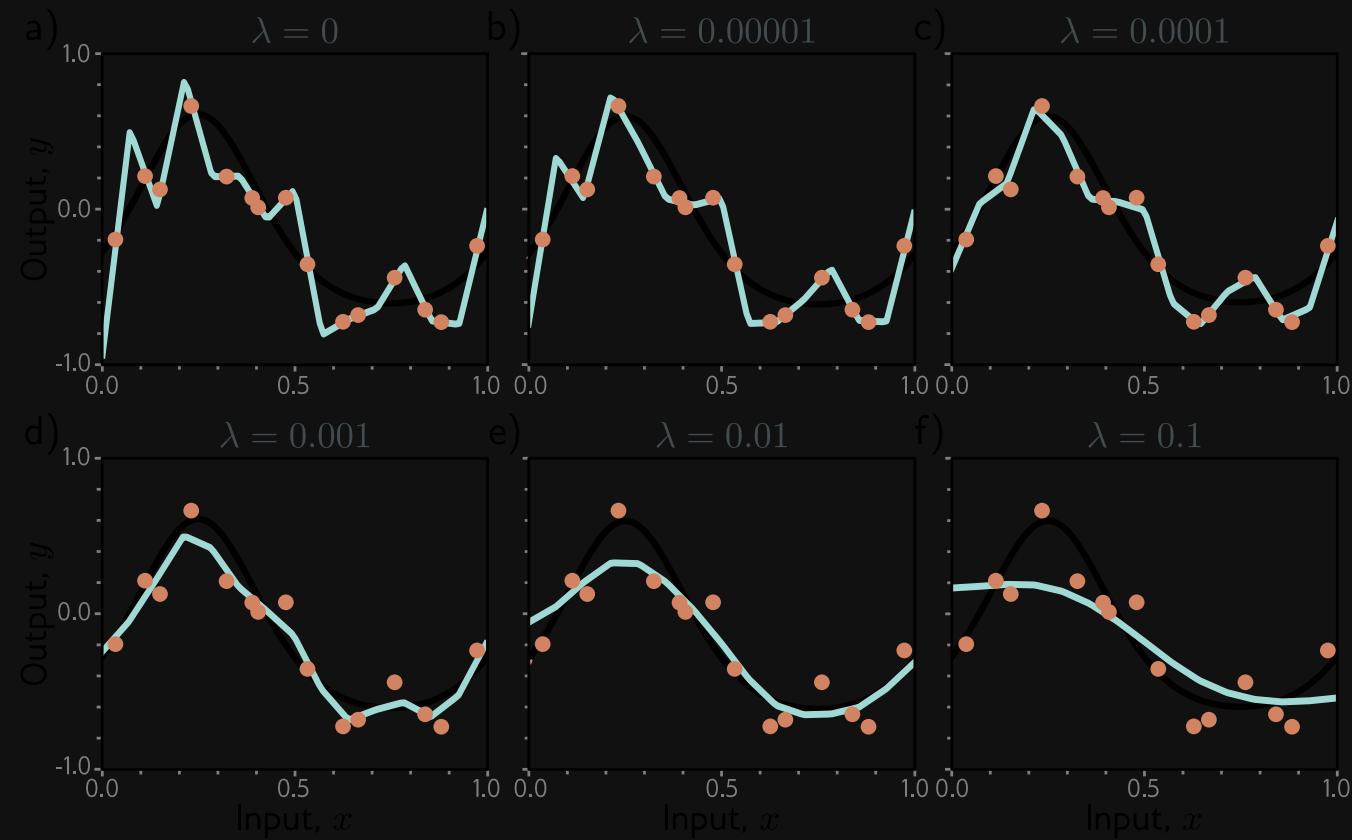
# Interpretation as Bayesian Prior

Maximum a posteriori (MAP) instead of maximum LLH

$$\begin{aligned}
 \hat{\phi} &= \arg \max_{\phi} \prod_{i=1}^I q(y_i | \theta_i = f[x_i, \phi]) P(\phi) \\
 &= \arg \max_{\phi} \log \prod_{i=1}^I q(y_i | \theta_i = f[x_i, \phi]) P(\phi) \\
 &= \arg \min_{\phi} -\log P(\phi) - \sum_{i=1}^I \log q(y_i | \theta_i = f[x_i, \phi]) \\
 &= \arg \min_{\phi} \lambda g[\phi] + \sum_{i=1}^I l_i[x_i, y_i]
 \end{aligned}$$

# L2 regularization: Ridge regression

$$g[\phi] = \sum_j \phi_j^2$$



# L1 regularization: Lasso regression

$$g[\phi] = \sum_j |\phi_j|$$

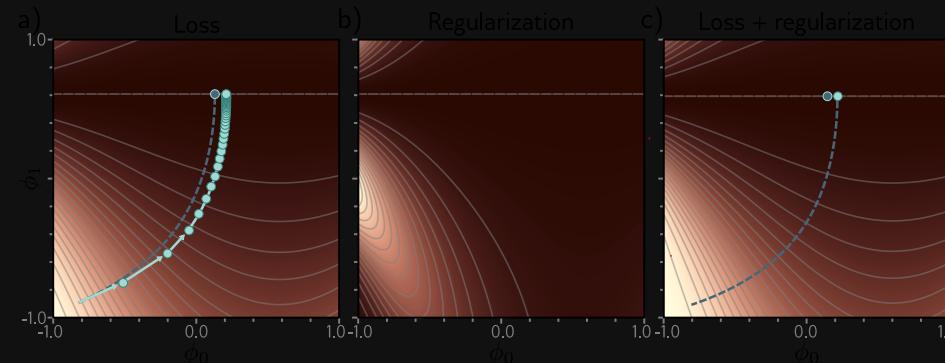
# Implicit Regularization

Discrete stochastic gradient descent on  $L$  arrives at same place as continuous descent  $\frac{d\phi}{dt} = -\frac{\partial \tilde{L}}{\partial \phi}$  on

$$\tilde{L}[\phi] = L[\phi] + \frac{\alpha}{4} \left| \left| \frac{\partial L}{\partial \phi} \right| \right|^2 + \frac{\alpha}{4B} \sum_{b=1}^B \left| \left| \frac{\partial L_b}{\partial \phi} - \frac{\partial L}{\partial \phi} \right| \right|^2$$

so is naturally biased

- away from steep gradients
- to stable gradients (low variance across batches), suggesting that “all data fits well” and promising better generalization

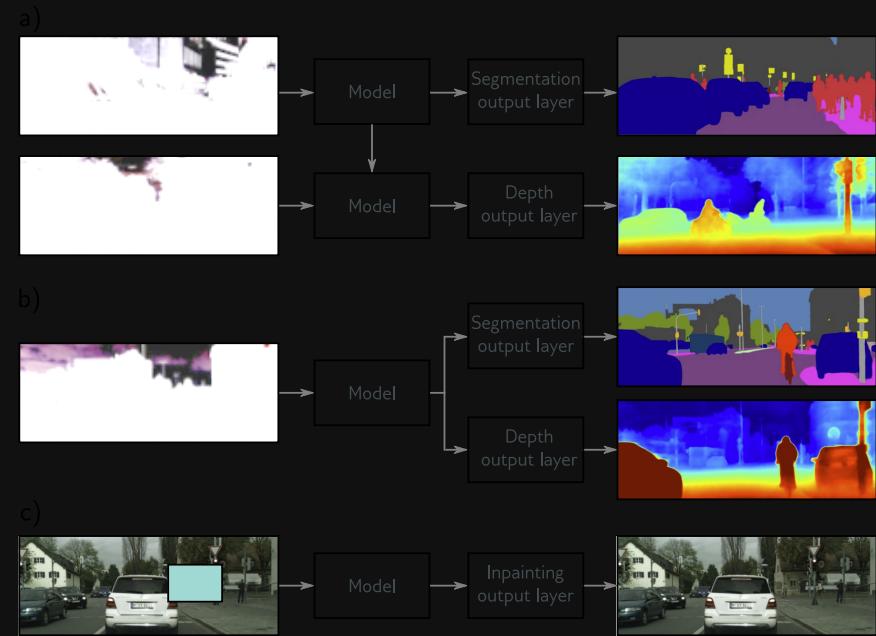


# Performance Heuristics

- **Early stopping:** halt training when validation loss starts rising; avoids overfitting by limiting effective model complexity w/o changing architecture.
- **Noise injection:** add noise to inputs, weights, or labels during training; acts as a smoothing penalty and improves robustness
  - adversarial training is a targeted variant
- **Normalization** (batch, layer, group): re-center and re-scale activations within each layer; stabilizes and accelerates training by reducing internal covariate shift
- **Bayesian inference:** instead of point estimates of the parameters, consider posterior  $P(\phi|\{x_i, y_i\}) \propto \prod_{i=1}^I q(y_i|x_i, \phi)P(\phi)$ 
  - principled but often intractable

# Transfer & Self-supervised Learning

- Transfer learning (pretrain on related training data)
  - train final layers
  - fine-tune whole model
- Multi-task learning
- Self-supervised learning
  - generative (fill in masked data)
  - contrastive (compare pairs of examples for relatedness)



# Data Augmentation

a) Original



b) Flip



c) Rotate and crop



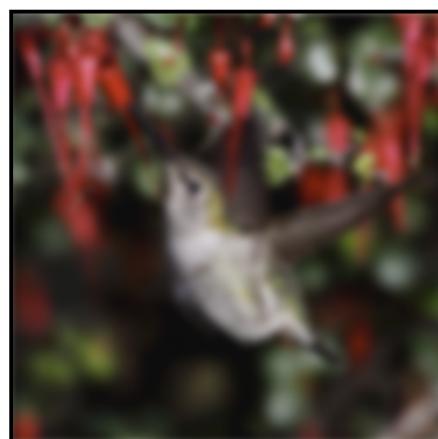
d) Vertical stretch



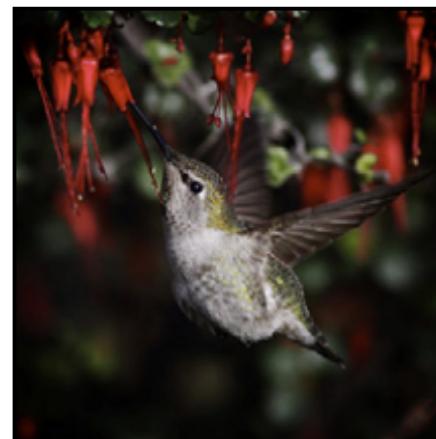
e) Color balance



f) Blur



g) Vignette

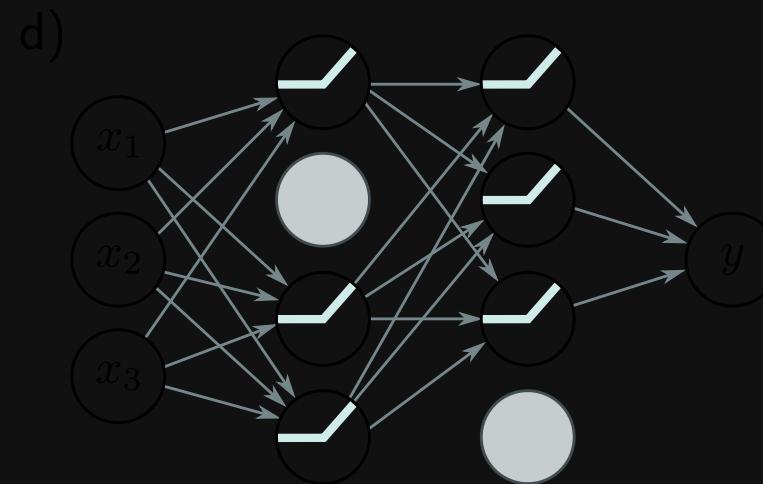
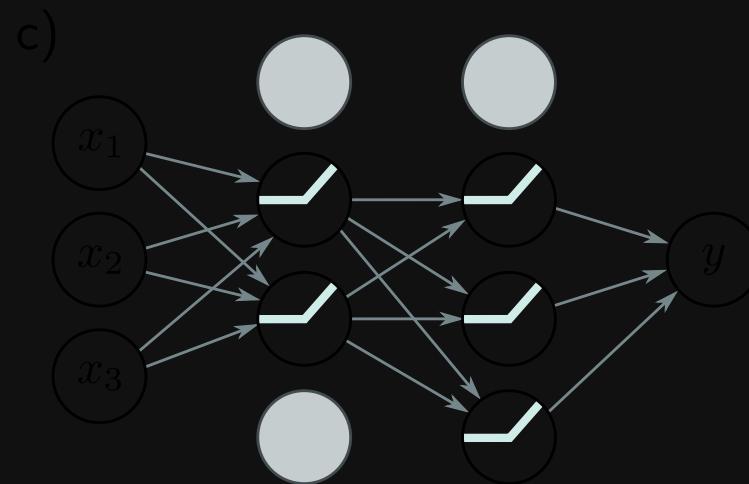
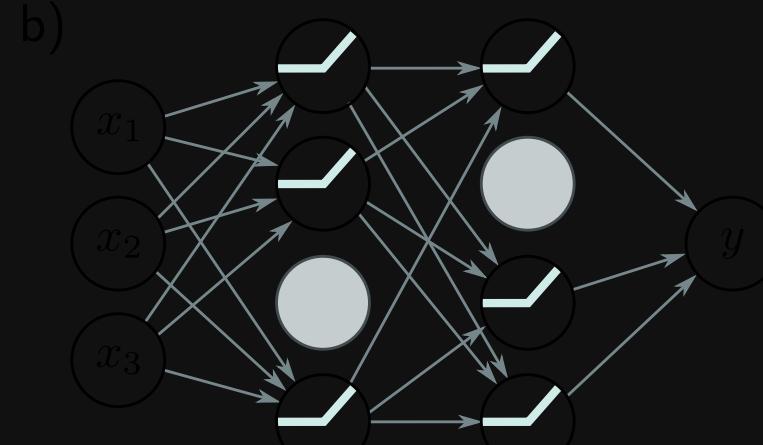
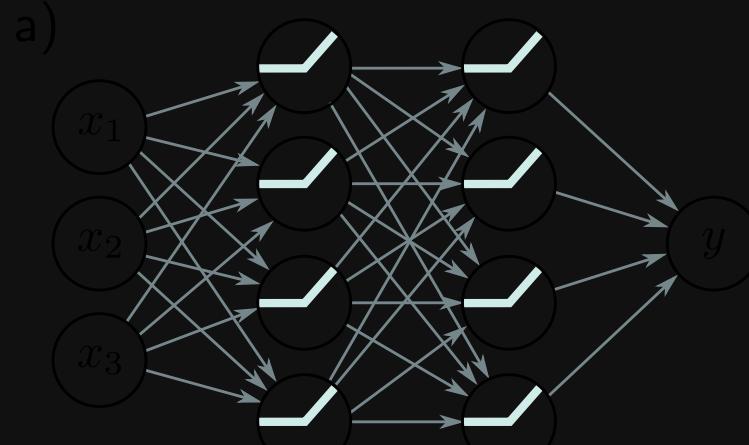


h) Pincushion

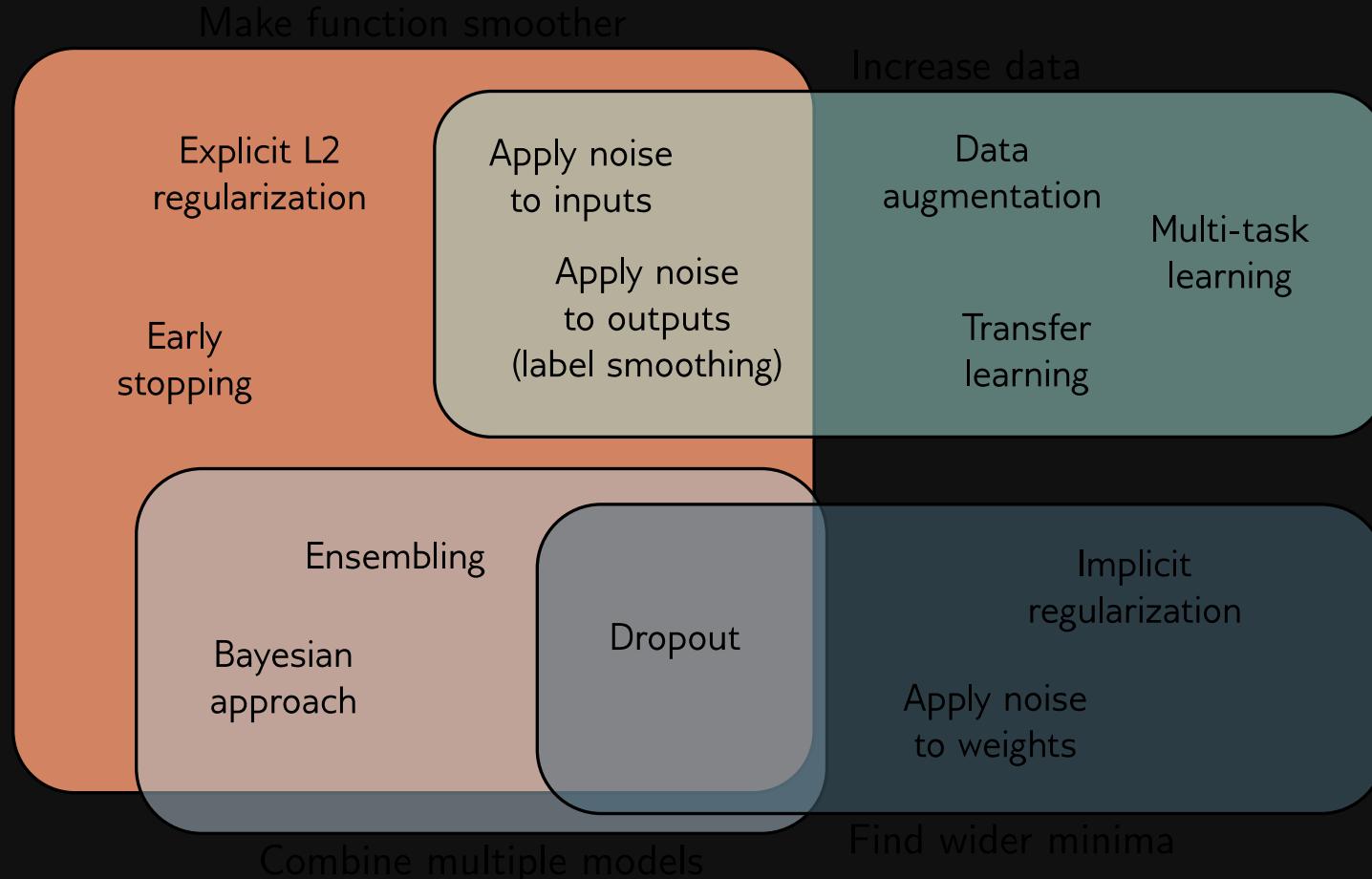


# Dropout

Clamping subset of hidden units to zero.



# Regularization Overview



# Ensemble methods

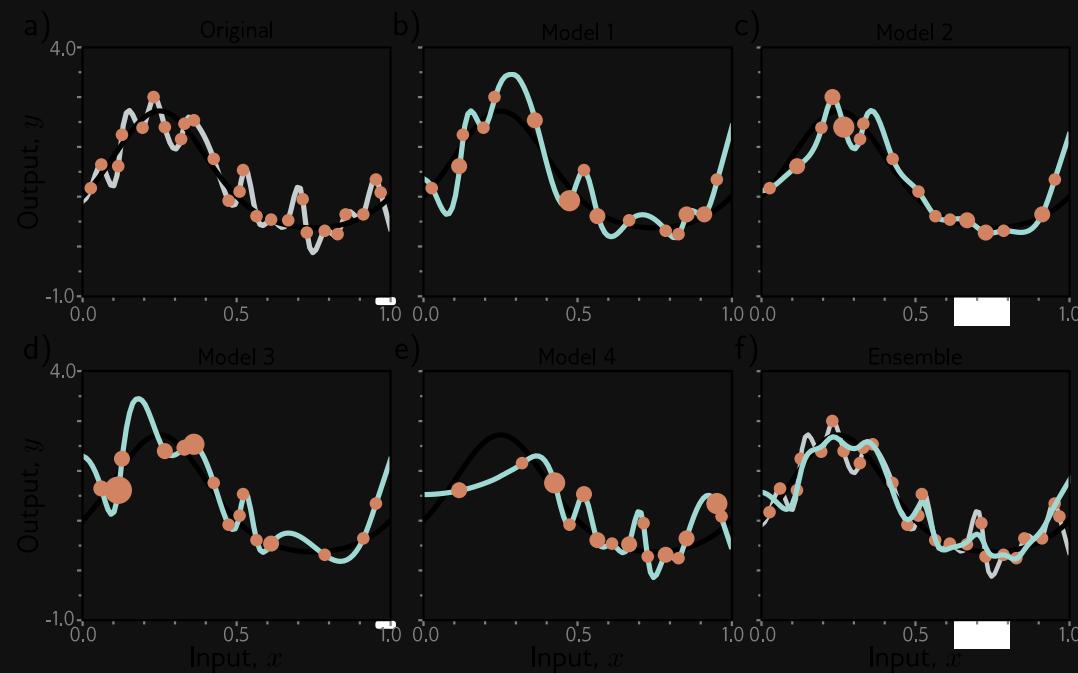
Murphy (2022, chaps. 18.2–18.5)

Explained through decision trees; ideas apply more generally.

# Ensemble Methods

Average predictions of multiple models, e.g.

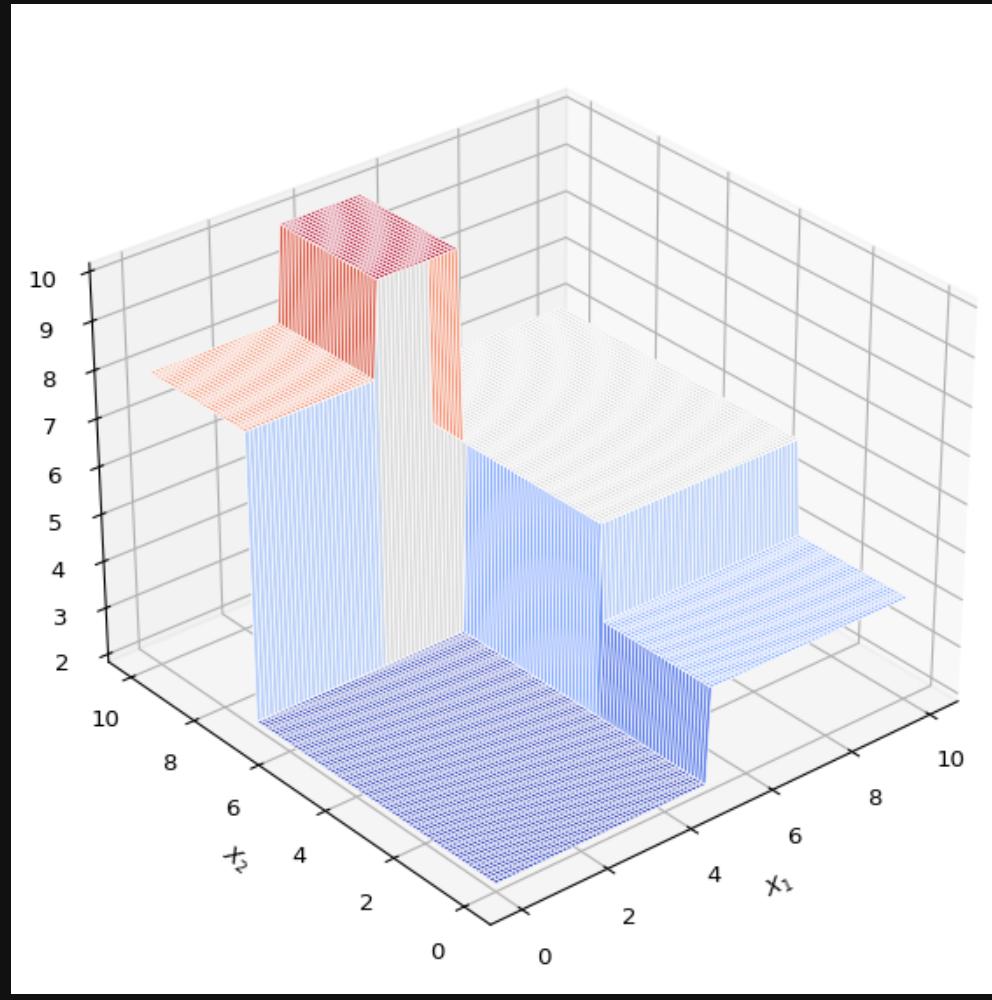
- different initializations
- Bagging (bootstrap aggregating)
  - i.e. resampling training data



# CART: Classification and Regression Trees

Cf. D300 & Murphy (2022, chap. 18.1).

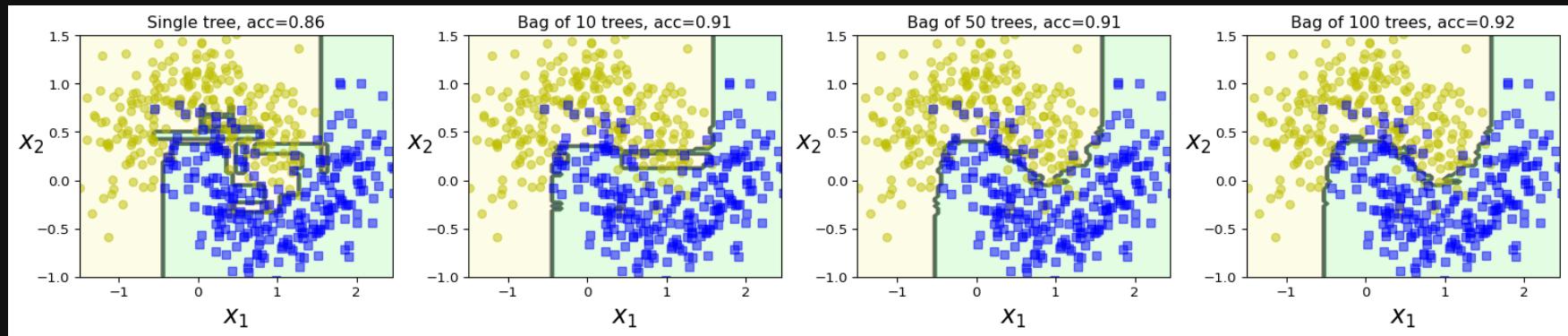
- Recursive axis-parallel splits: choose feature + threshold to minimize impurity
  - regression: MSE
  - classification: entropy or Gini index
- Partitions input space into (axis-aligned) regions of constant prediction.



Adapted from Murphy (2022, fig. 18.1).

# Bagging: Bootstrap aggregating

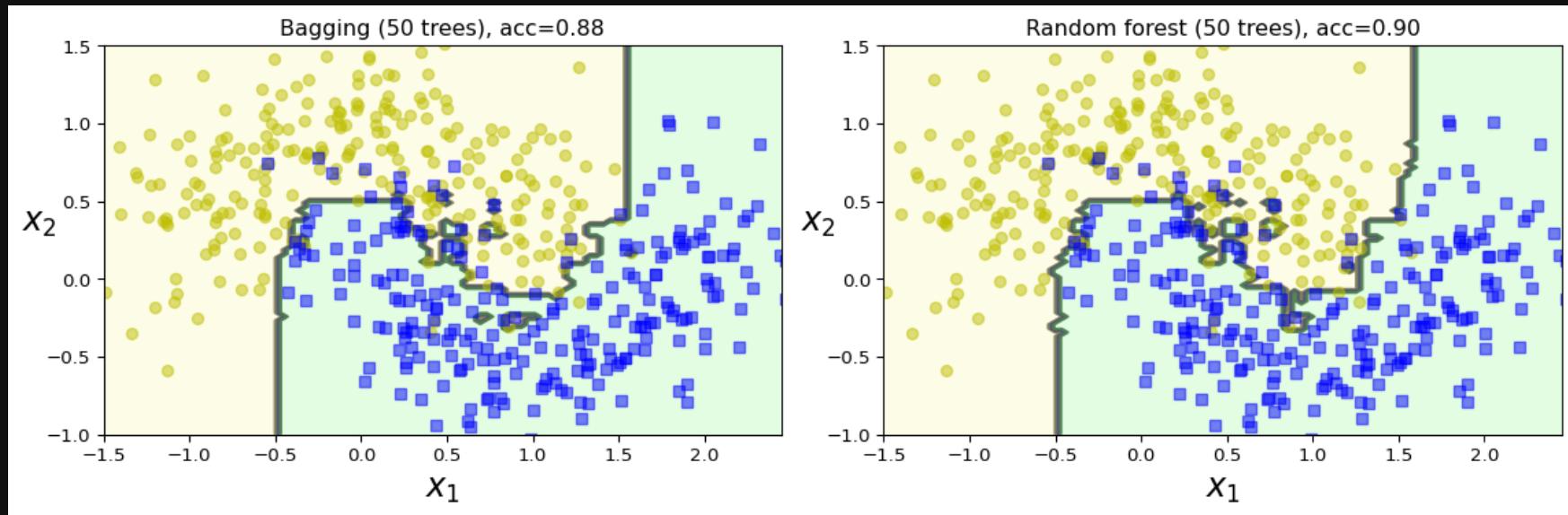
- Trees interpretable, but high variance (sensitive to data).
- **Bootstrap aggregating:** fit many models in parallel on bootstrap samples of the training data; average their predictions.
- Averaging (or voting) reduces variance.



Adapted from Murphy (2022, fig. 18.4).

# Random forests

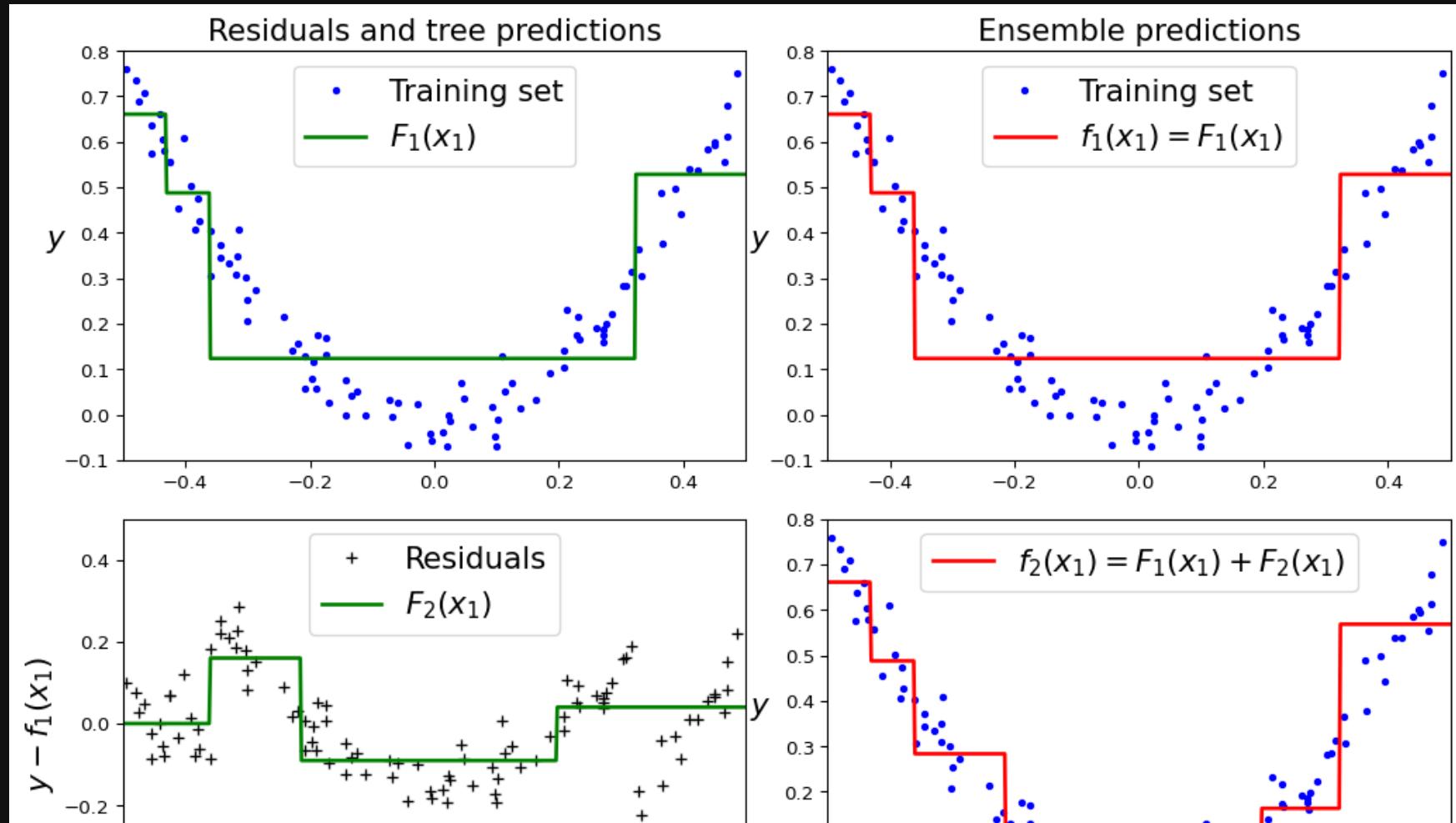
- Bagging but at each split use a random subset of features.
- Further decorrelates trees.

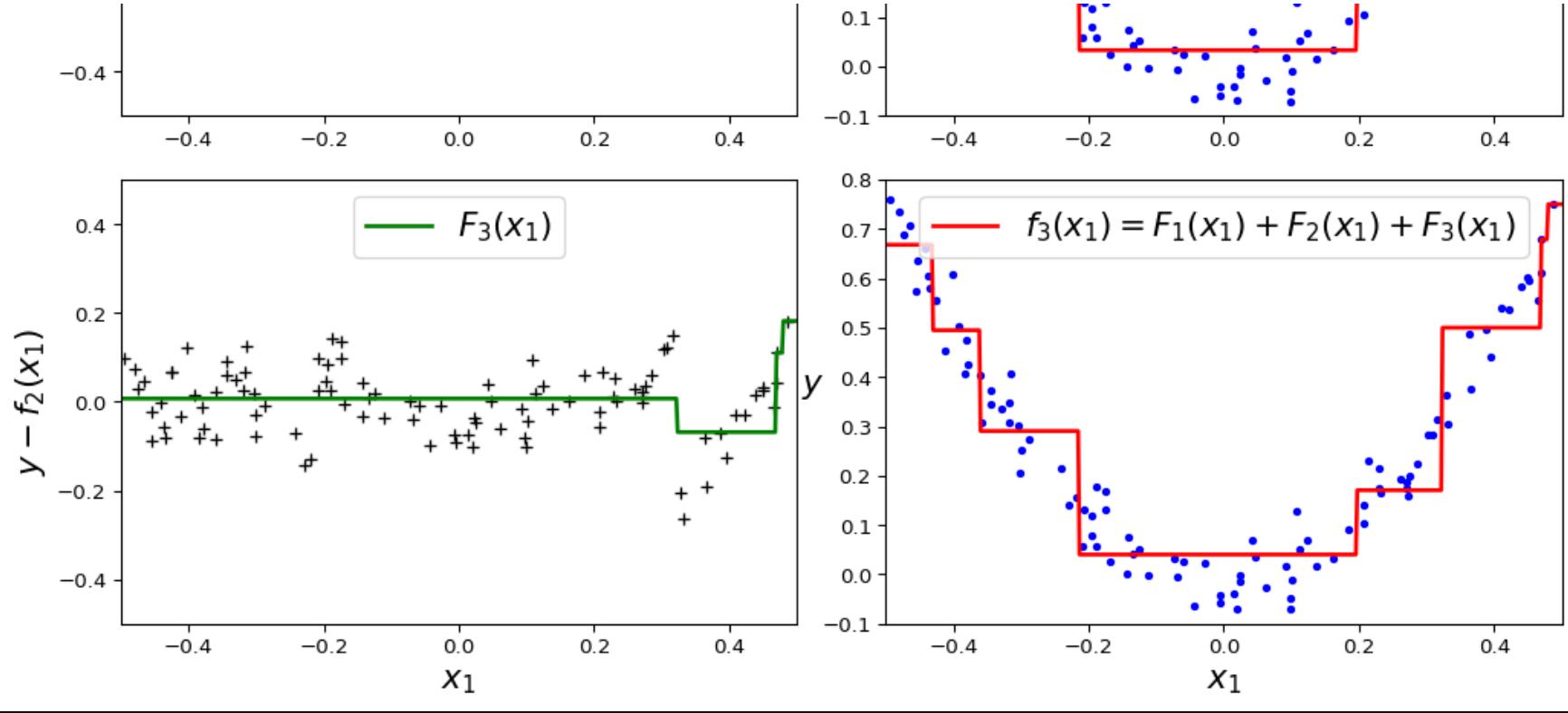


Adapted from Murphy (2022, fig. 18.4).

# Boosting

Sequential: each new model focuses on errors (residuals) of the previous ones to iteratively improve weak learners.





Adapted from Murphy (2022, fig. 18.6).

# Gradient Boosting and XGBoost

- **Gradient boosting:** sequentially fit shallow trees to the negative gradient (pseudo-residuals) of the loss.
  - Gradient descent in function space: each step adds a small tree pointing in the direction that most reduces prediction error.
- **XGBoost:** efficient implementation that adds L2 regularization on leaf weights, second-order (Newton) splits, column subsampling, and efficient handling of sparse data.
  - State-of-the-art for tabular data.

# Model interpretability

# Feature attribution: Shapley values

*Which input features contribute to model prediction?*

- Write prediction = baseline + sum of **additive feature contributions**:

$$\hat{y}(x) = \mathbb{E}[\hat{y}(X)] + \sum_{f=1}^D \sum_{S \subseteq F \setminus \{f\}} \frac{|S|! (D - |S| - 1)!}{D!} \left( f_x(S \cup \{f\}) - f_x(S) \right)$$

- **Shapley value:**  $f$ 's average marginal contribution over all subsets  $S$  of other features.
  - **Fair attribution** (cooperative game theory): unique rule satisfying *efficiency, symmetry, and consistency*
  - features = players, prediction = payoff
- **SHAP:** practical method to approximate these values.

# Shapley values

