Hierarchical Shrinkage: Improving the Accuracy and Interpretability of Tree-based Methods

Abhineet Agarwal

Yan Shuo Tan

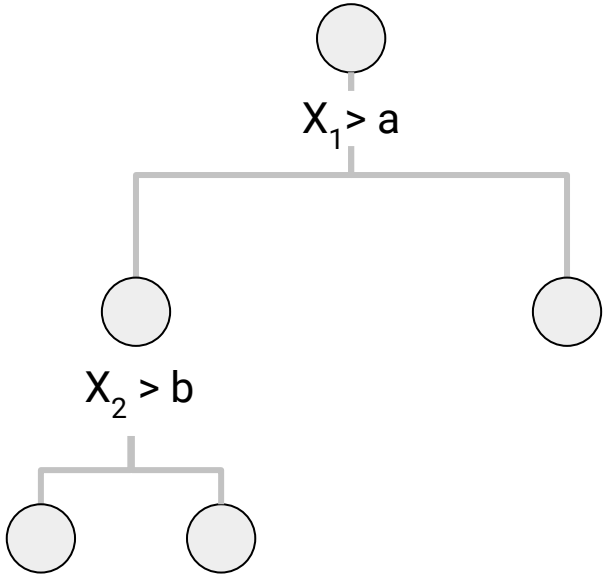
Omer Ronen

Chandan Singh

Bin Yu

Decision tree algorithms comprise two steps

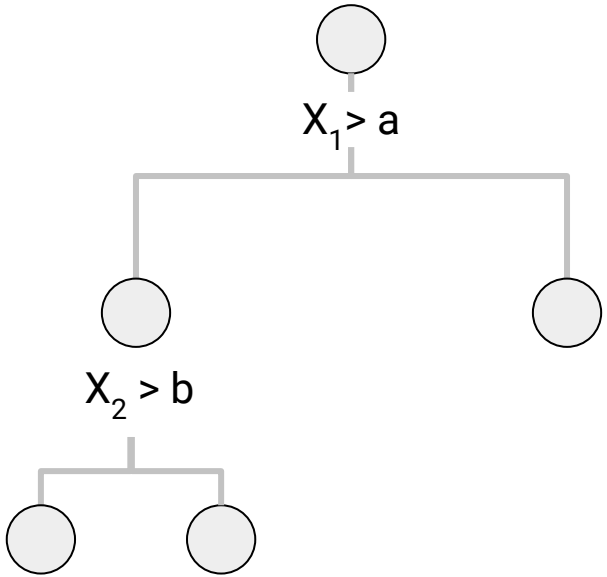
Step 1: Grow the tree



Decision tree algorithms comprise two steps

Step 1: Grow the tree

Step 2: Impute node values using the mean response of the training data within each node

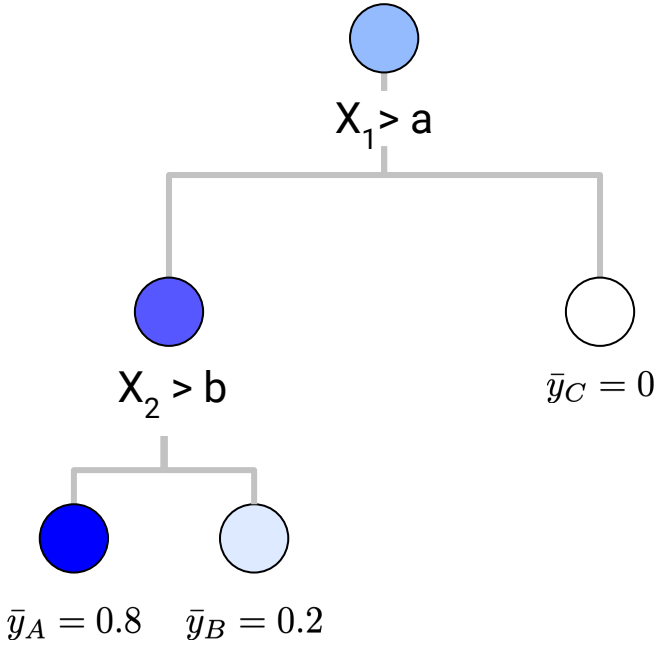


Decision tree algorithms comprise two steps

Step 1: Grow the tree

Different algorithms vary in this step

Step 2: Impute node values using the mean response of the training data within each node



How to grow the tree: two strategies

Greedy algorithms
(using local criterion)

- Most popular: CART [Breiman, Friedman, Olshen, Stone (1984)]
- Many others: C4.5 [Quinlan (1993)], ID3 [Quinlan (1986)], GUIDE [Loh (2009)],...

Global optimization
(of a loss function on space of trees)

- Dynamic programming: GOSDT [Lin, Hu, Rudin, Seltzer (2020)],...
- Mixed integer optimization: [Bertsimas, Dunn (2017)],...

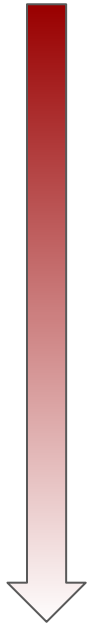
Fundamental problem:
Trees can easily overfit to the training data

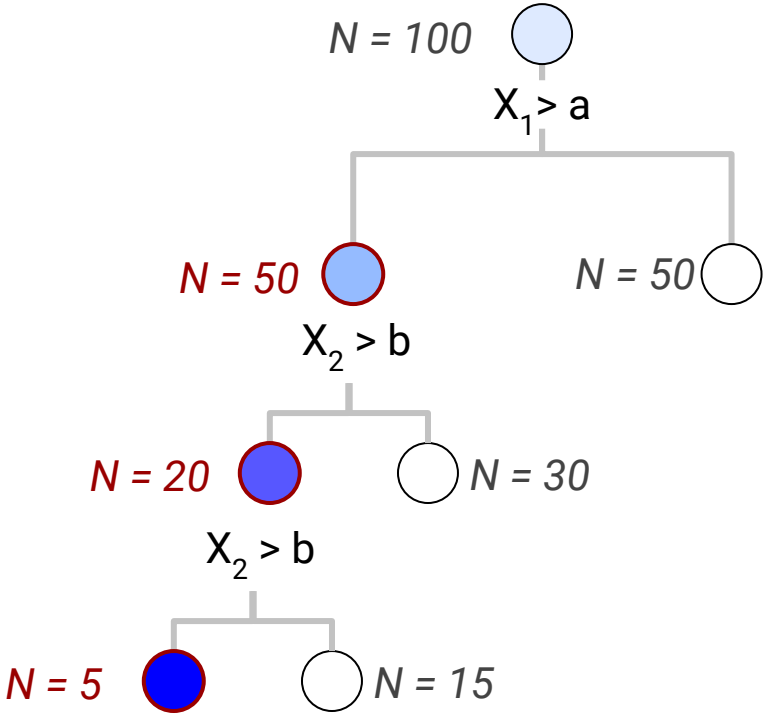
Leaf size decreases

Bias decreases ☺️

Sample size decreases

Variance increases ☹️



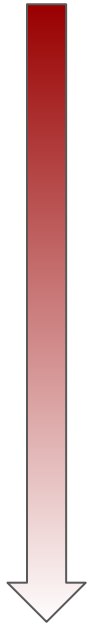


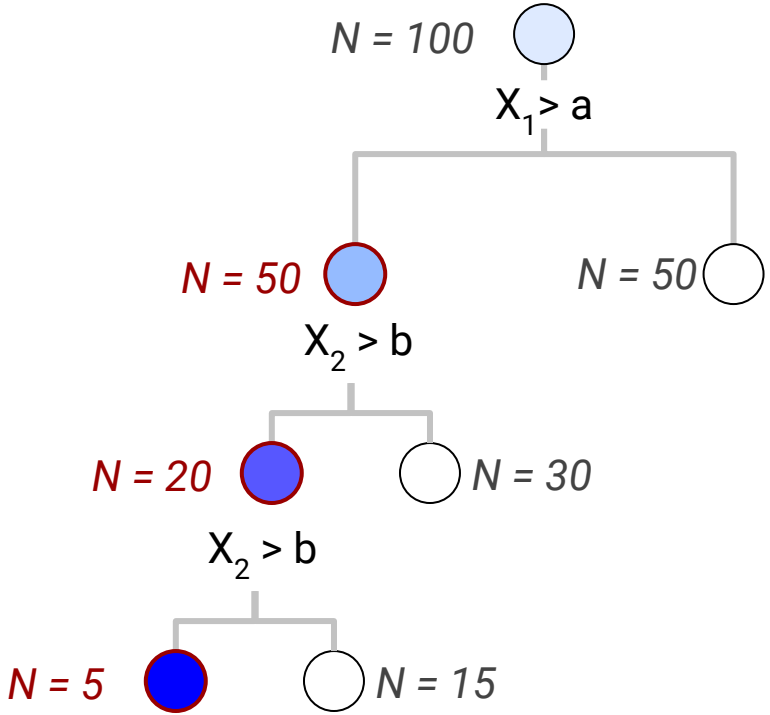
Leaf size decreases

Bias decreases ☺️

Sample size decreases

Variance increases ☹️





Fundamental problem:
Trees can easily overfit to the training data

Current methods for preventing overfitting

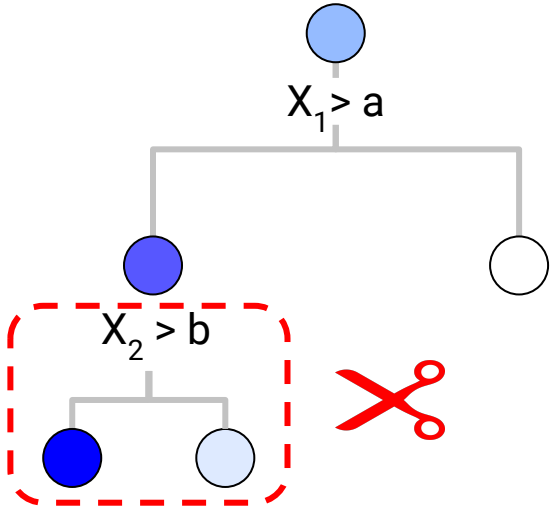
Greedy algorithms

- Use early stopping condition
- Prune the tree after growing

Global optimization

- Add a complexity penalty term to the loss function

All methods regularize the tree **structure** (step 1)



Current methods for preventing overfitting

Greedy algorithms

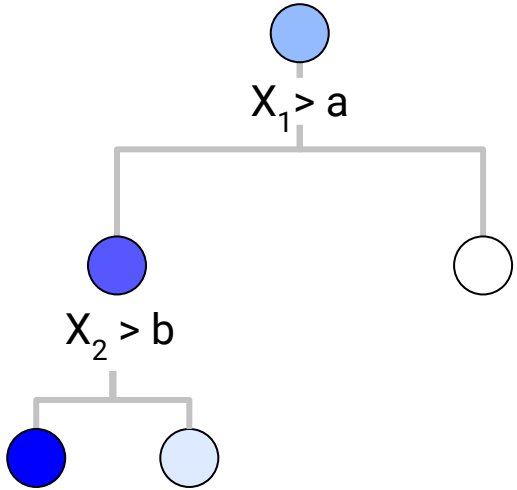
- Use early stopping condition
- Prune the tree after growing

Global optimization

- Add a complexity penalty term to the loss function

All methods regularize the tree **structure** (step 1)

We propose regularizing the tree **values** (step 2)



Current methods for preventing overfitting

Greedy algorithms

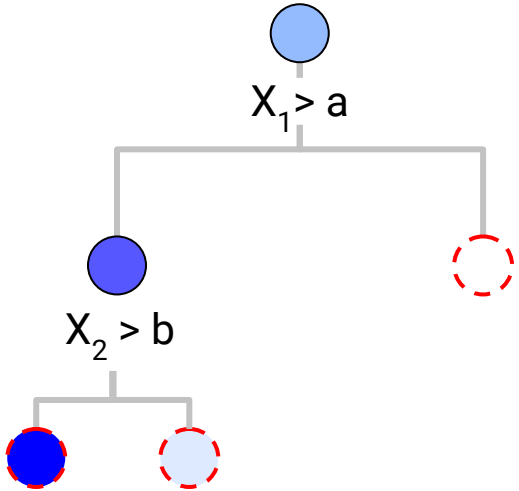
- Use early stopping condition
- Prune the tree after growing

Global optimization

- Add a complexity penalty term to the loss function

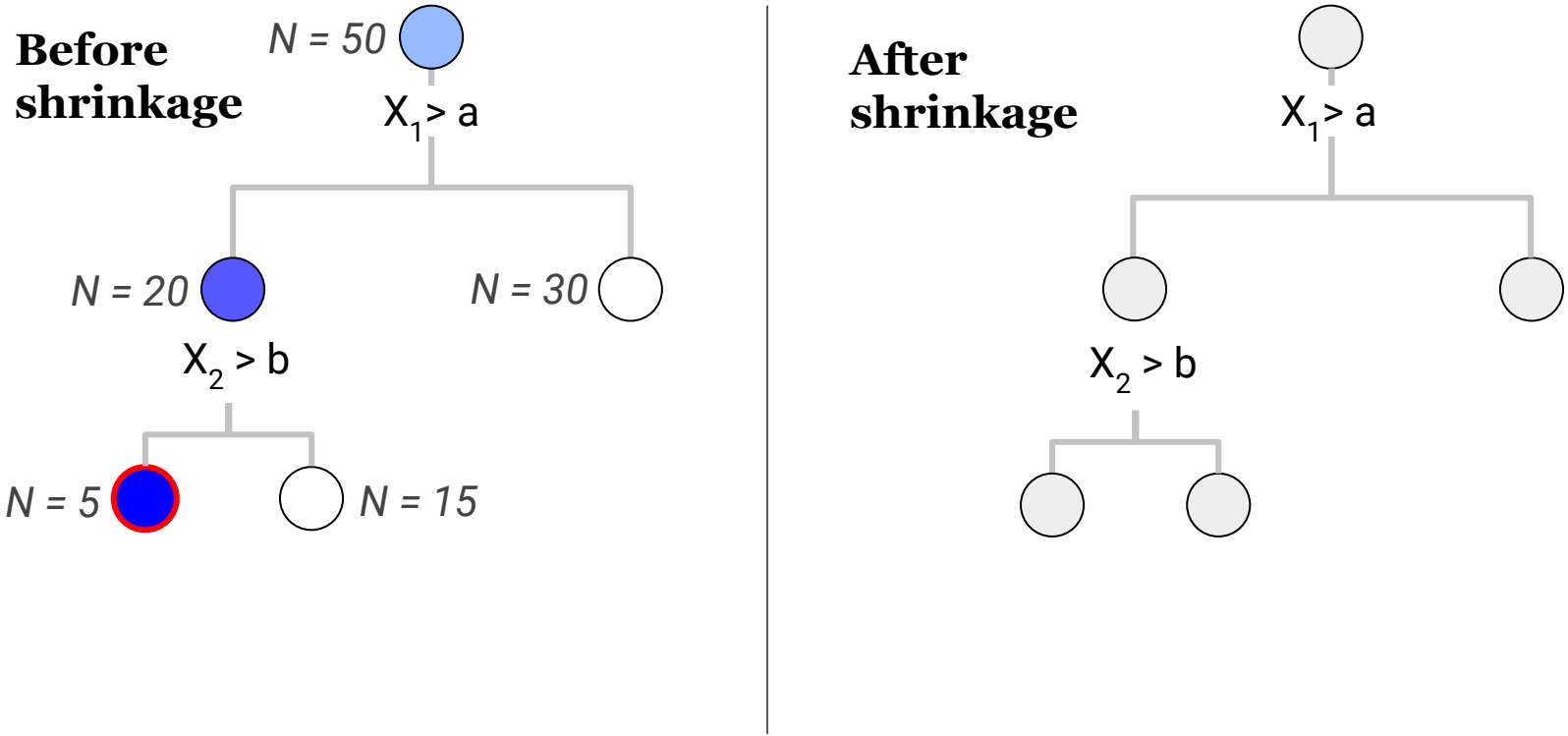
All methods regularize the tree **structure** (step 1)

We propose regularizing the tree **values** (step 2)



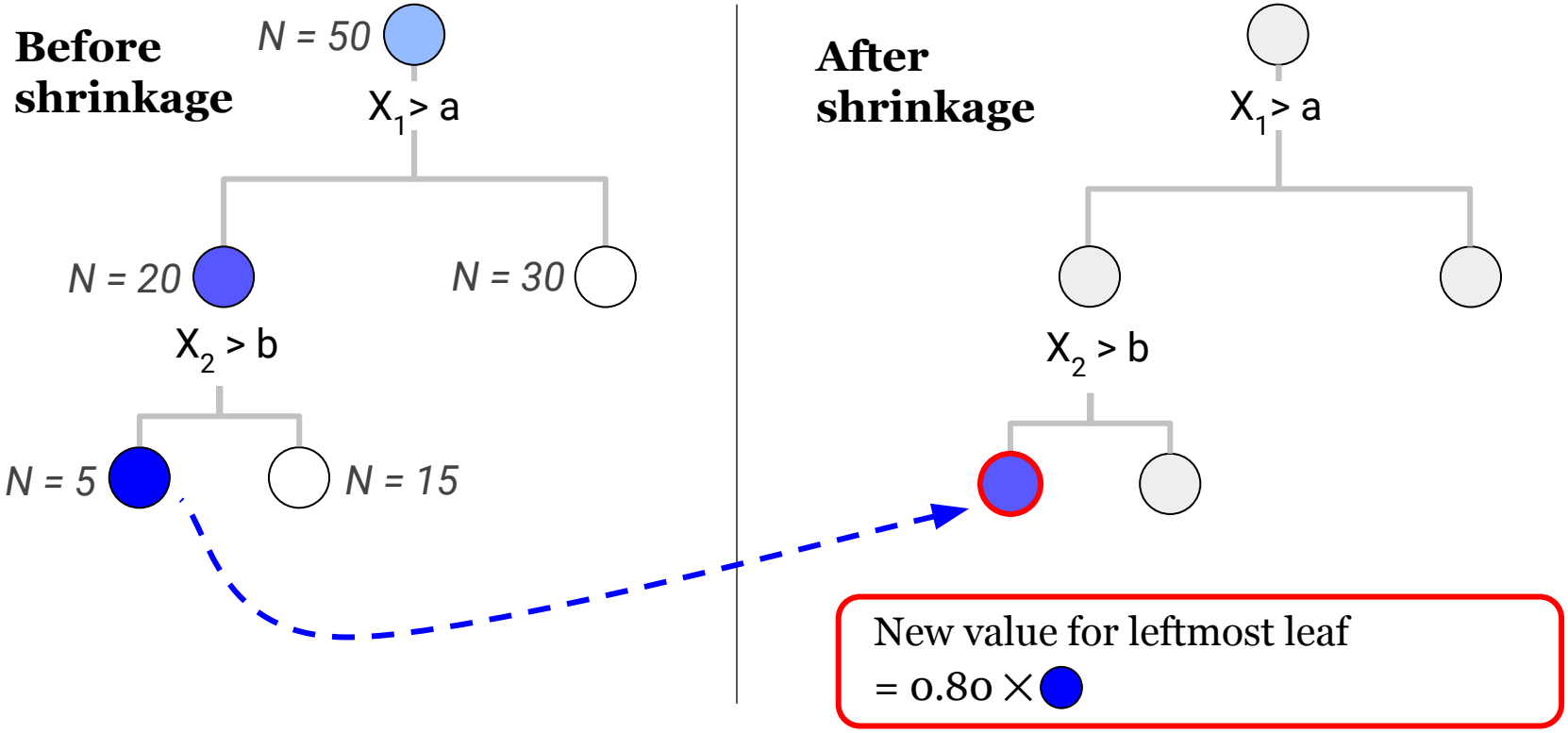
Introducing: Hierarchical Shrinkage (HS)

Shrink the value of each node to those of its ancestors



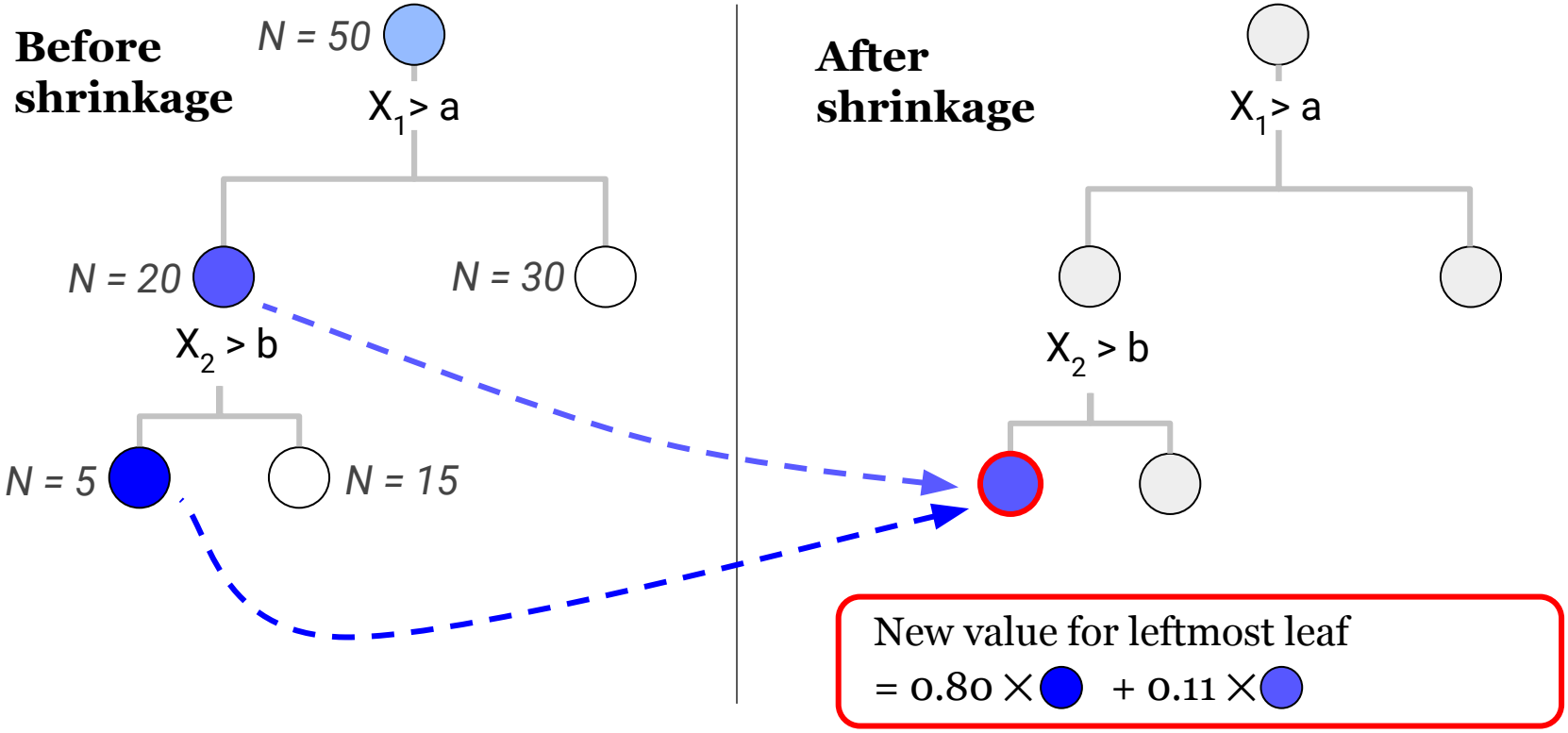
Introducing: Hierarchical Shrinkage (HS)

Shrink the value of each node to those of its ancestors



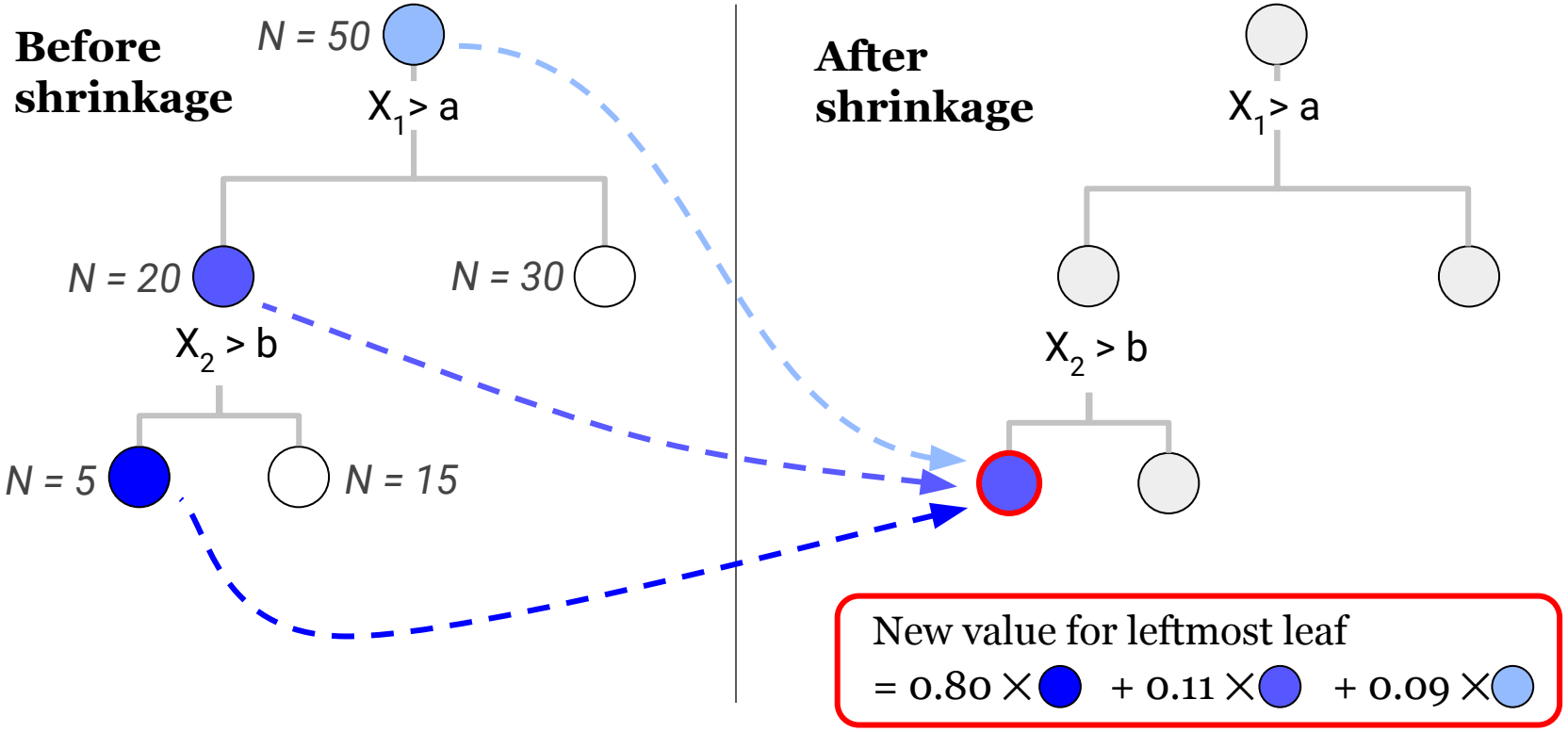
Introducing: Hierarchical Shrinkage (HS)

Shrink the value of each node to those of its ancestors



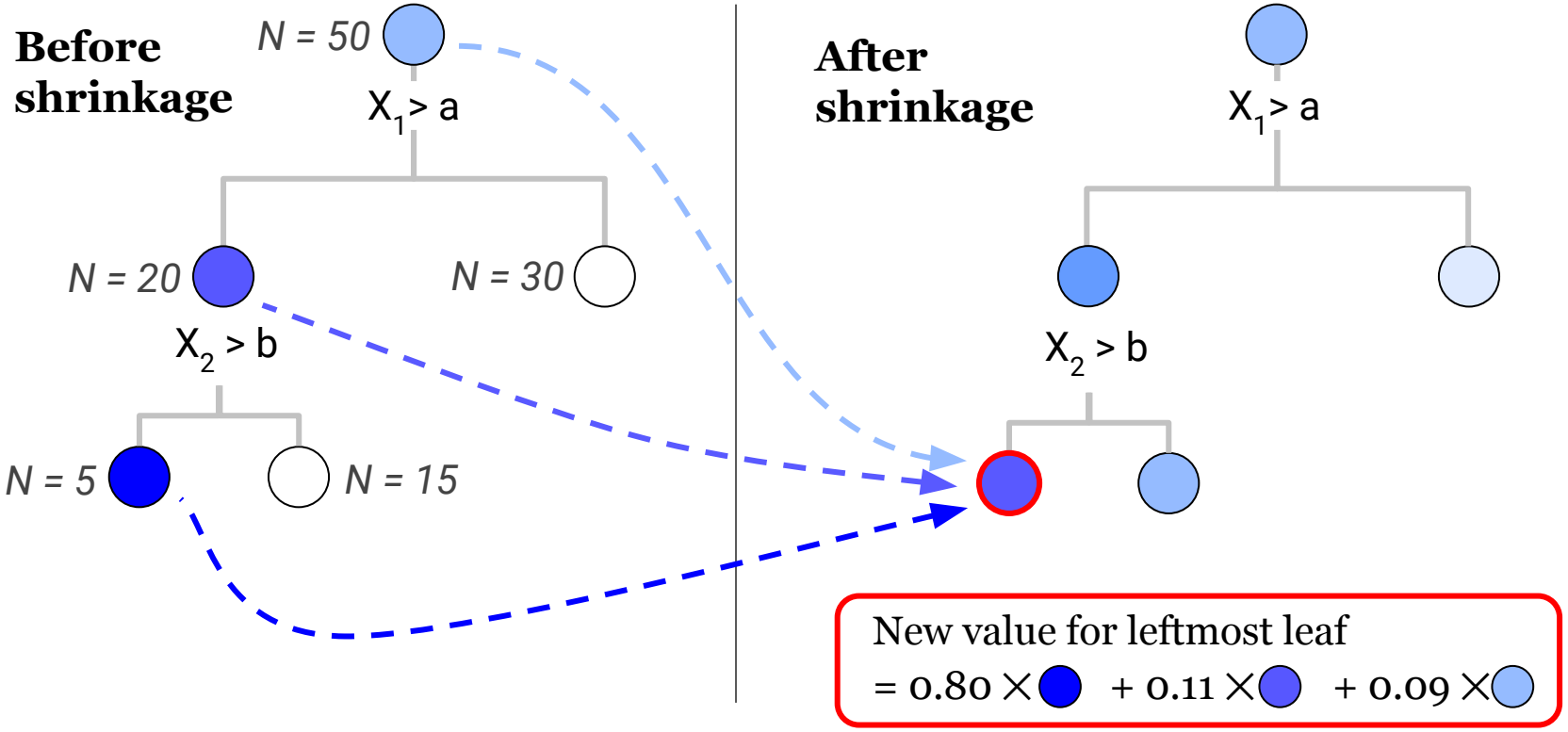
Introducing: Hierarchical Shrinkage (HS)

Shrink the value of each node to those of its ancestors

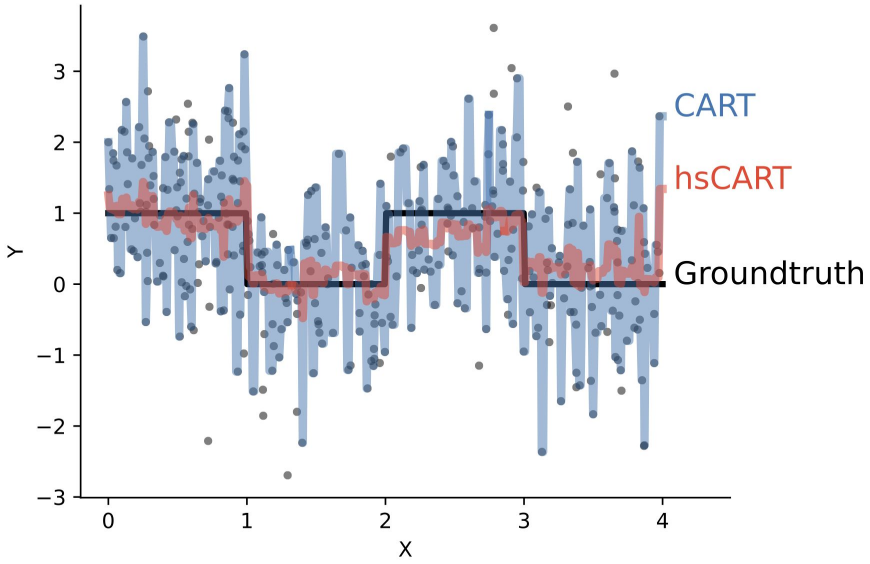


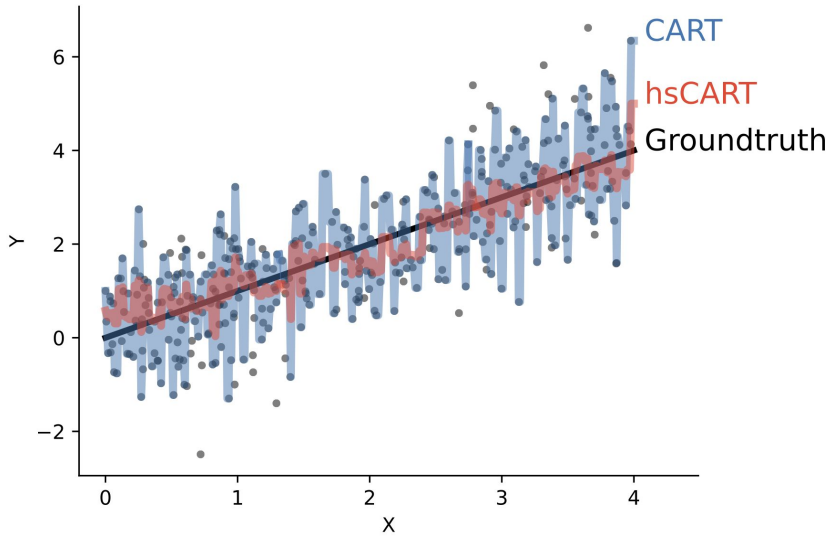
Introducing: Hierarchical Shrinkage (HS)

Shrink the value of each node to those of its ancestors



Effect on toy examples





How are weights determined?

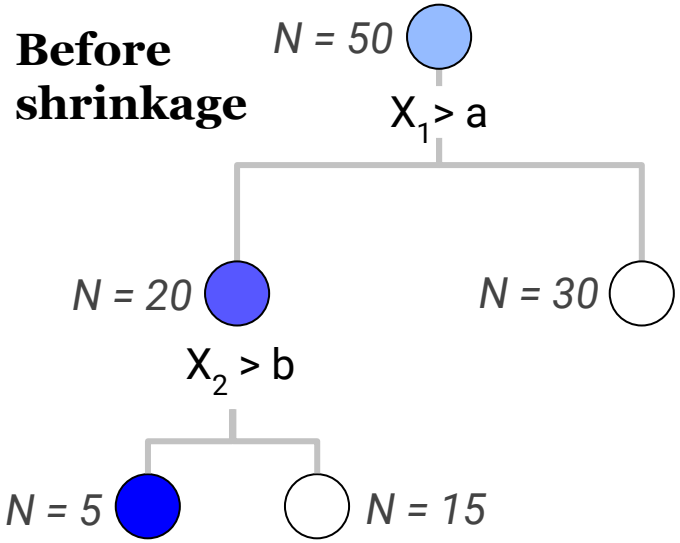
New value for leftmost leaf

= 0.80 ×  + 0.11 ×  + 0.09 × 

How are weights determined?

New value for leftmost leaf
= 0.80 × ● + 0.11 × ● + 0.09 × ●

Before shrinkage

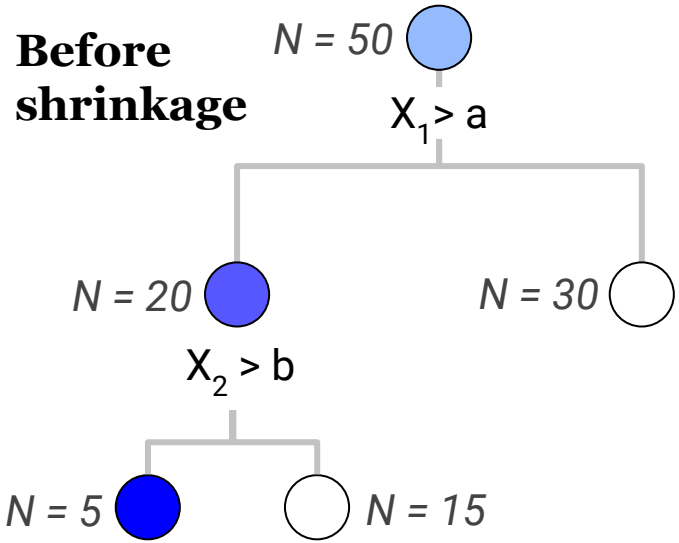


Old prediction ●

How are weights determined?

New value for leftmost leaf
= 0.80 × ● + 0.11 × ● + 0.09 × ●

Before shrinkage

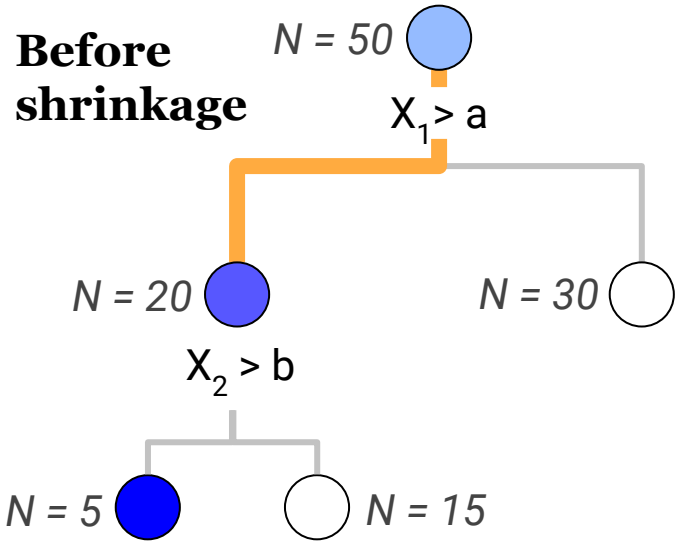


Old prediction ●
= ● +

How are weights determined?

New value for leftmost leaf
= 0.80 × ● + 0.11 × ● + 0.09 × ●

Before shrinkage

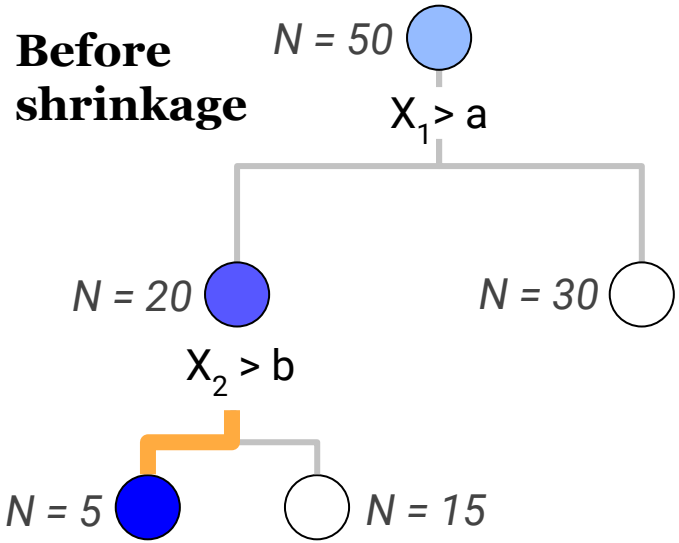


Old prediction ●
= ● + (● - ●) +

How are weights determined?

New value for leftmost leaf
= 0.80 × ● + 0.11 × ● + 0.09 × ●

Before shrinkage

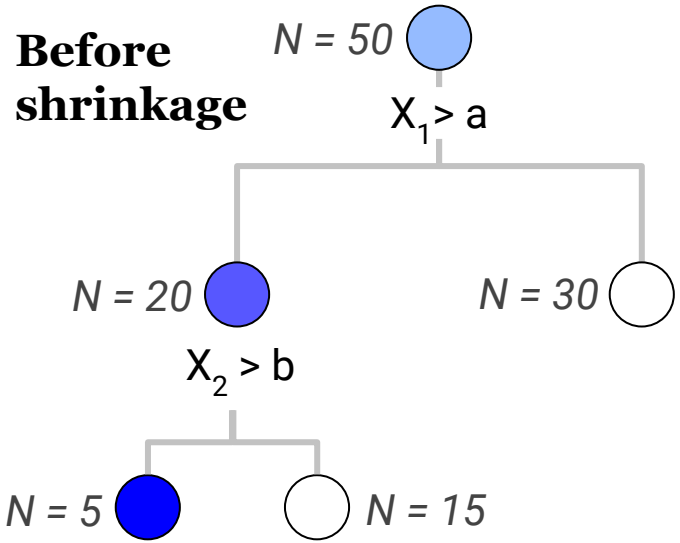


Old prediction ●
= ● + (● - ●) + (● - ●)

How are weights determined?

New value for leftmost leaf
= 0.80 × ● + 0.11 × ● + 0.09 × ●

Before shrinkage

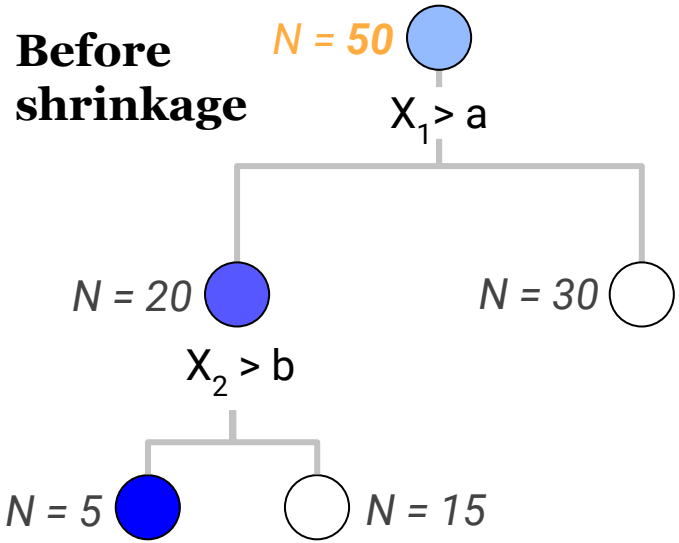


Old prediction ●
= ● + (● - ●) + (● - ●)

New prediction under HS
= ● + (● - ●) + (● - ●)

How are weights determined?

New value for leftmost leaf
= 0.80 × ● + 0.11 × ● + 0.09 × ●



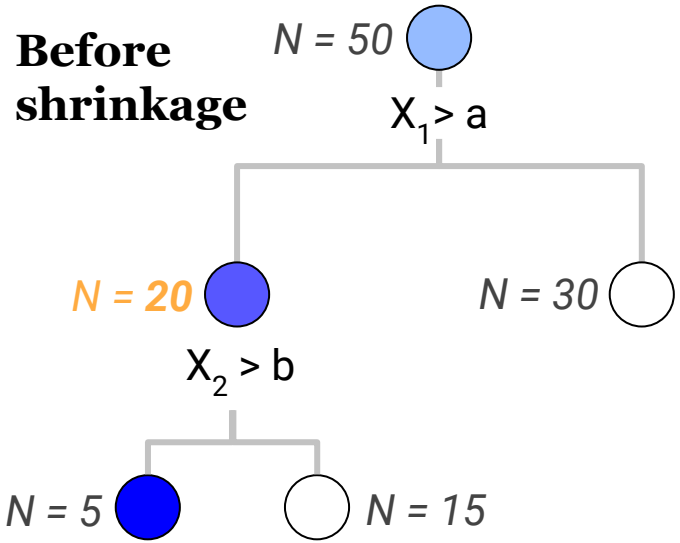
Old prediction ●
= ● + (● - ●) + (● - ●)

New prediction under HS

$$= \bigcirc + \frac{(\bullet - \bigcirc) + (\bullet - \bullet)}{1 + \lambda/50}$$

How are weights determined?

New value for leftmost leaf
= 0.80 × ● + 0.11 × ● + 0.09 × ●



Old prediction ●
= ● + (● - ●) + (● - ●)

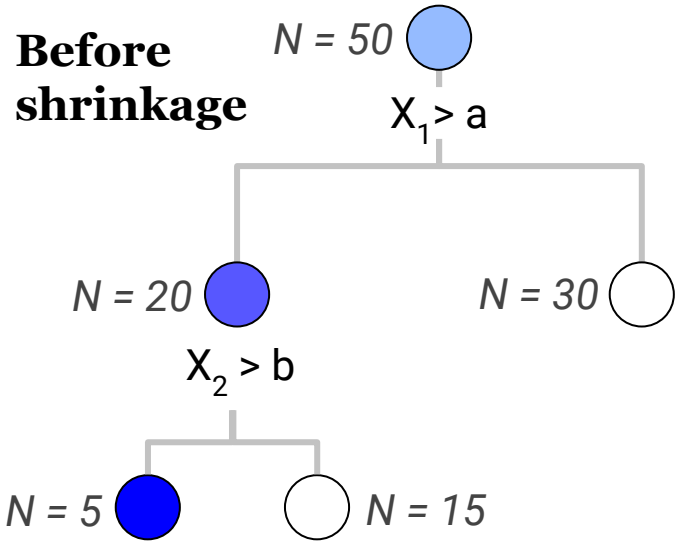
New prediction under HS

$$= \bigcirc + \frac{(\bullet - \bigcirc)}{1 + \lambda/50} + \frac{(\bullet - \bullet)}{1 + \lambda/20}$$

How are weights determined?

New value for leftmost leaf
= 0.80 × ● + 0.11 × ● + 0.09 × ●

Before shrinkage



Old prediction ●
= ● + (● - ●) + (● - ●)

New prediction under HS

$$= \bullet + \frac{(\bullet - \bullet)}{1 + \lambda/50} + \frac{(\bullet - \bullet)}{1 + \lambda/20}$$

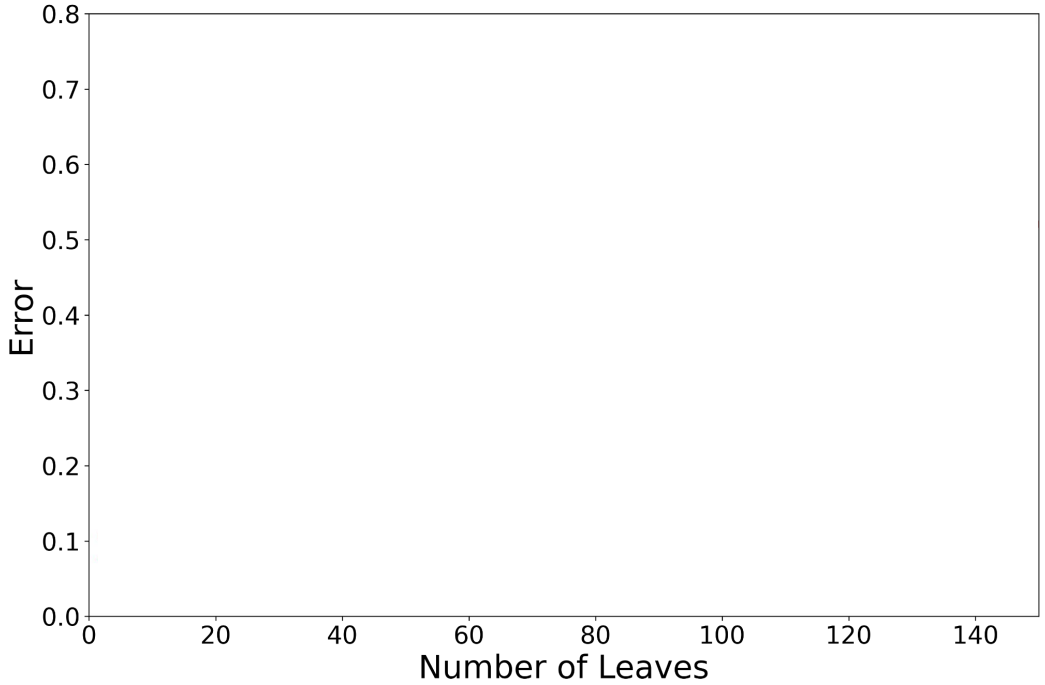
λ = hyperparameter to be tuned

Under the hood:
Connection to *ridge regression*

Because HS is applied after decision tree is grown…

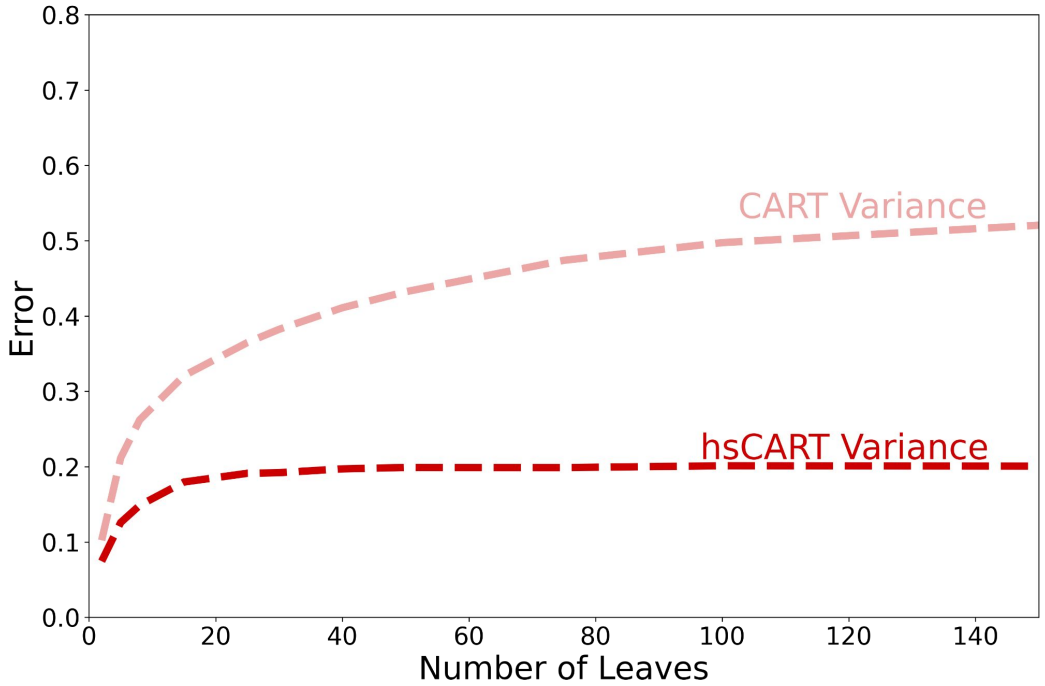
- Can be applied to any decision tree model
- Extremely fast and efficient
  - Does not need to refit the tree
  - Does not require access to training sample, only node values + sample sizes
  - Can be tuned using efficient leave-one-out-CV
- Can be used simultaneously with pruning or other types of regularization
- Can be applied to decision trees in an ensemble

Simulation*: Effect of HS at different no. of leaves



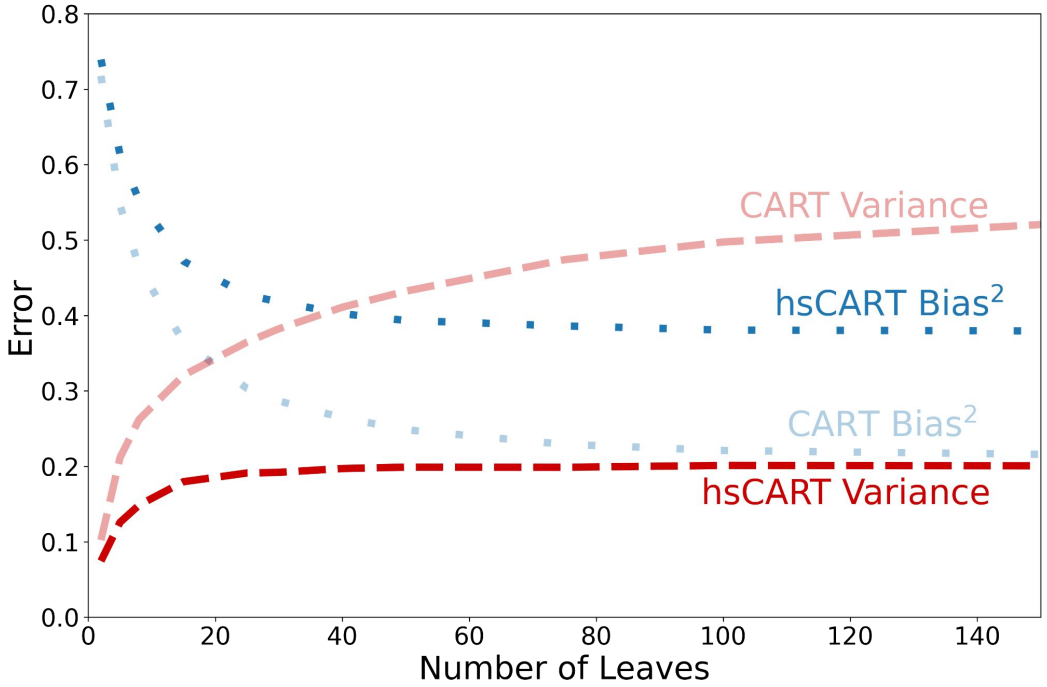
*with a sparse linear generative model

Simulation*: Effect of HS at different no. of leaves



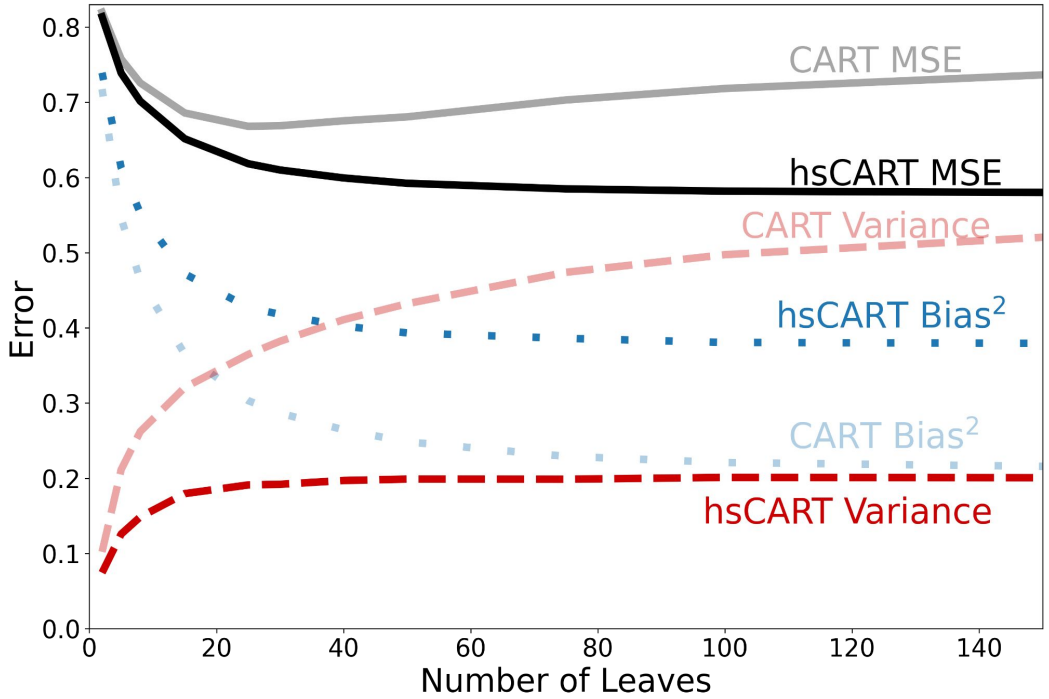
*with a sparse linear generative model

Simulation*: Effect of HS at different no. of leaves



*with a sparse linear generative model

Simulation*: Effect of HS at different no. of leaves



*with a sparse linear generative model

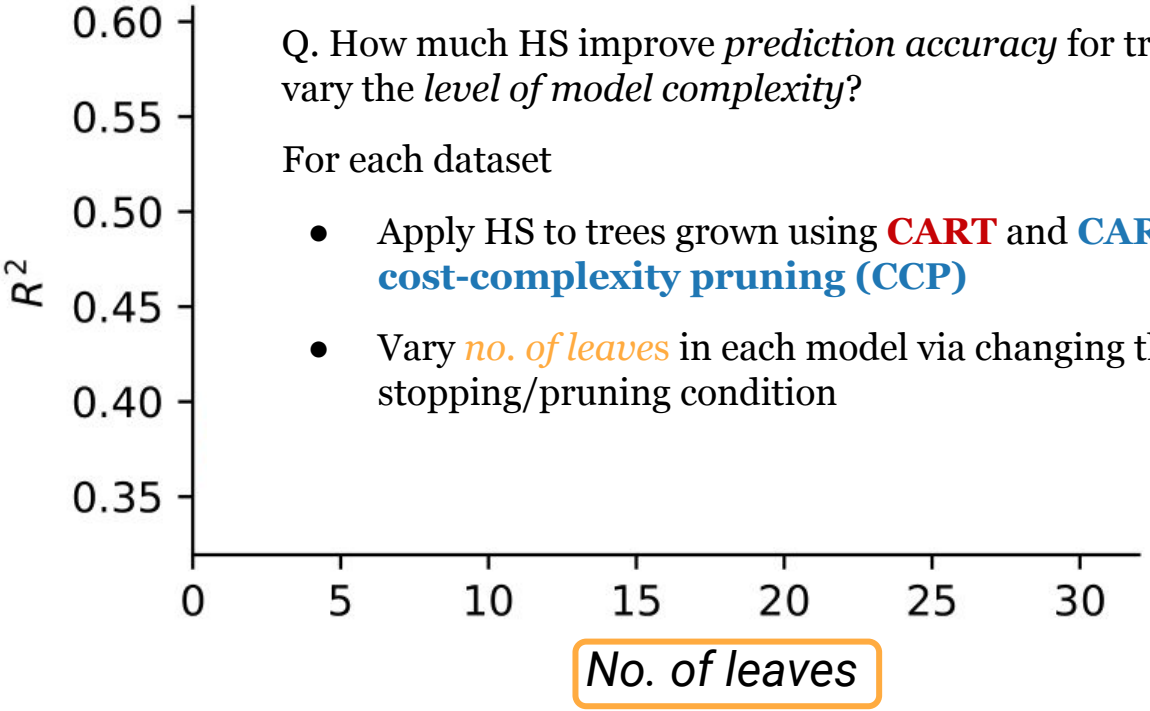

Results on real-world datasets

HS improves prediction accuracy of decision trees

Hierarchical shrinkage

1. Improves prediction accuracy (r^2) on regression datasets
2. Improves prediction accuracy (AUROC) on classification datasets
3. Performs better than alternate shrinkage schemes
4. Improves prediction accuracy of random forests

1. Regression results: r^2 vs *no. of leaves*

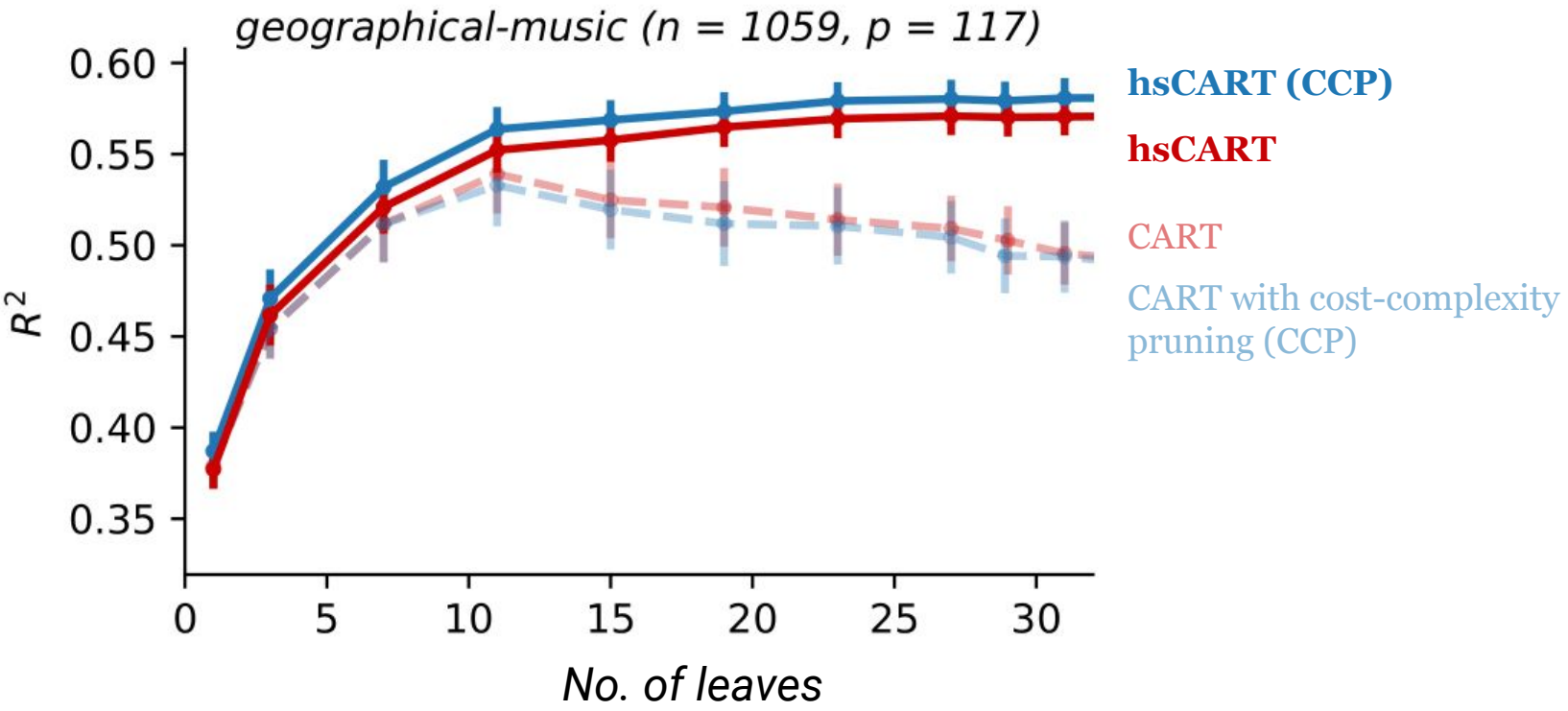


Q. How much HS improve *prediction accuracy* for trees as we vary the *level of model complexity*?

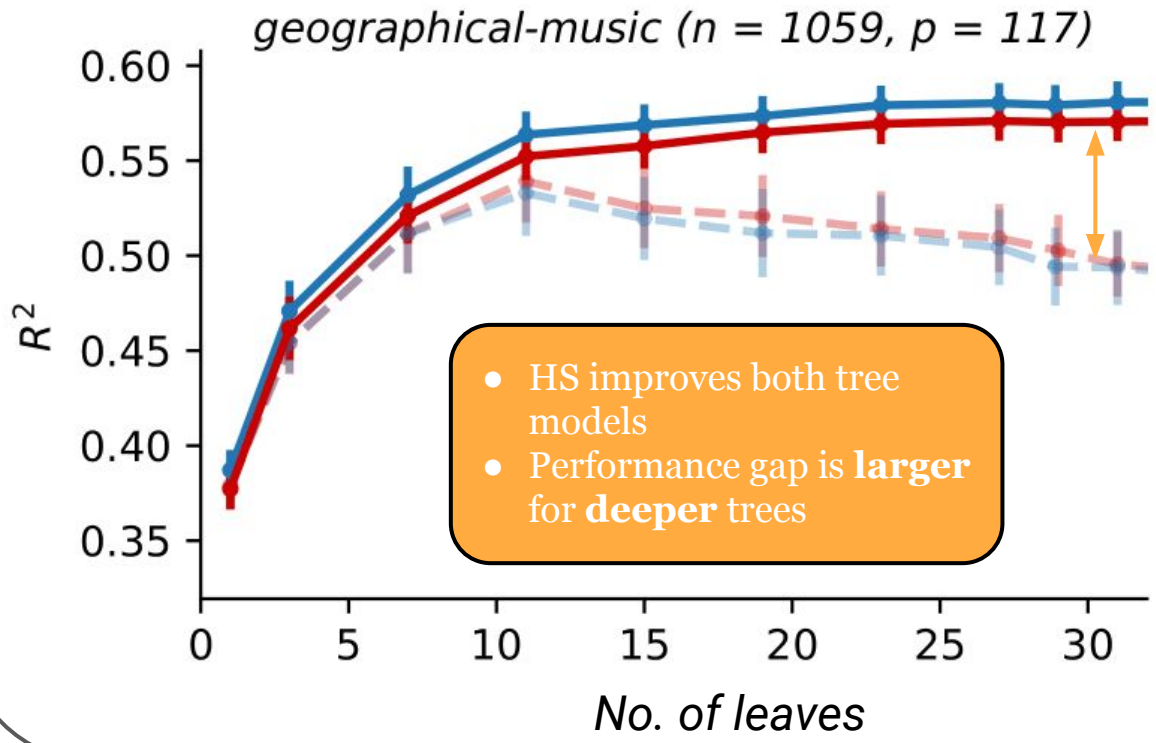
For each dataset

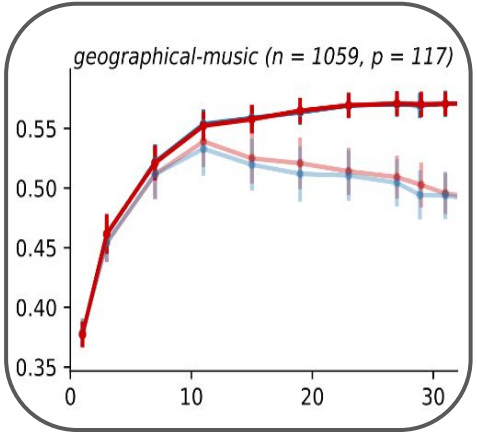
- Apply HS to trees grown using **CART** and **CART with cost-complexity pruning (CCP)**
- Vary *no. of leaves* in each model via changing the stopping/pruning condition

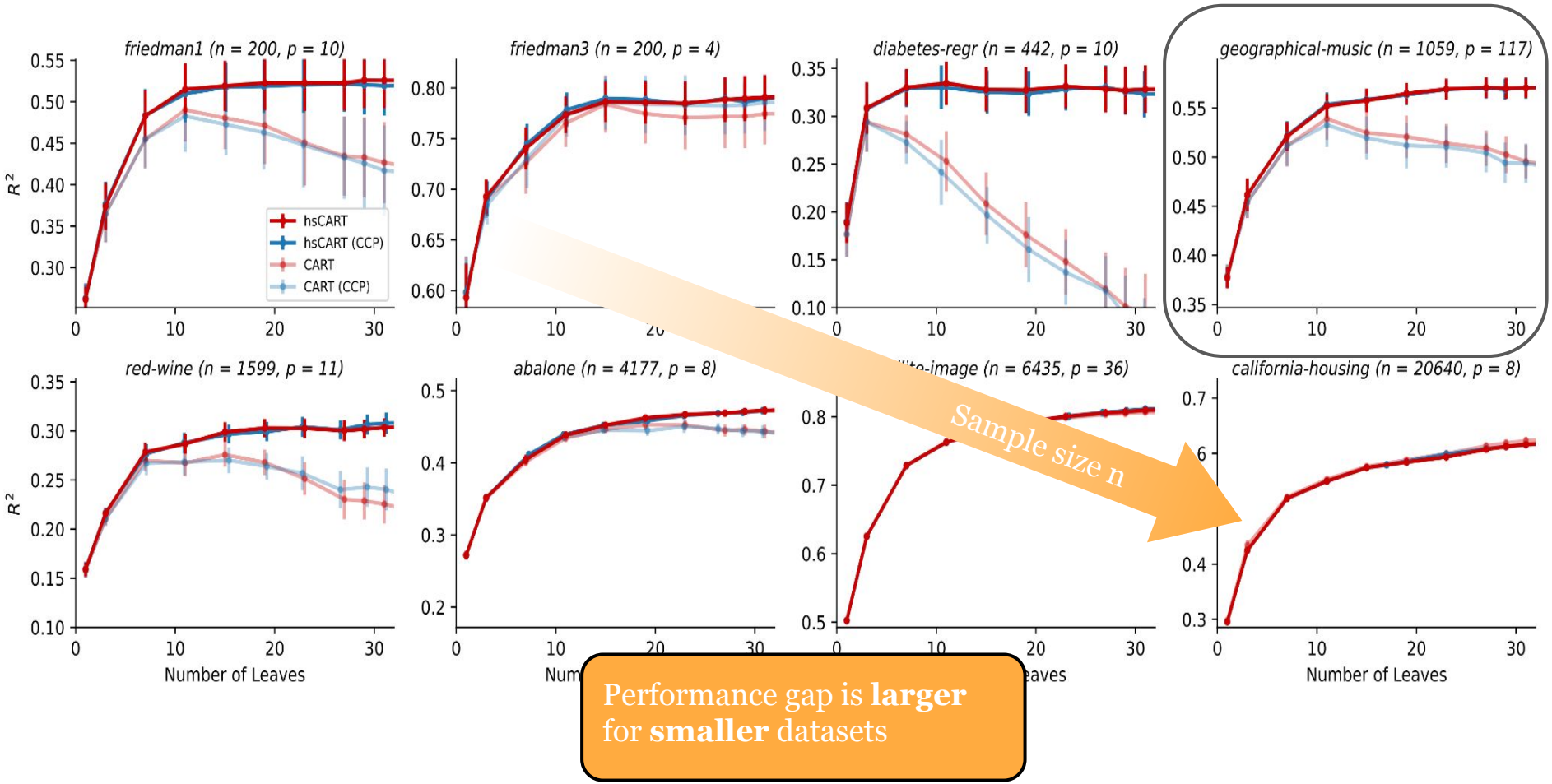
1. Regression results: r^2 vs *no. of leaves*



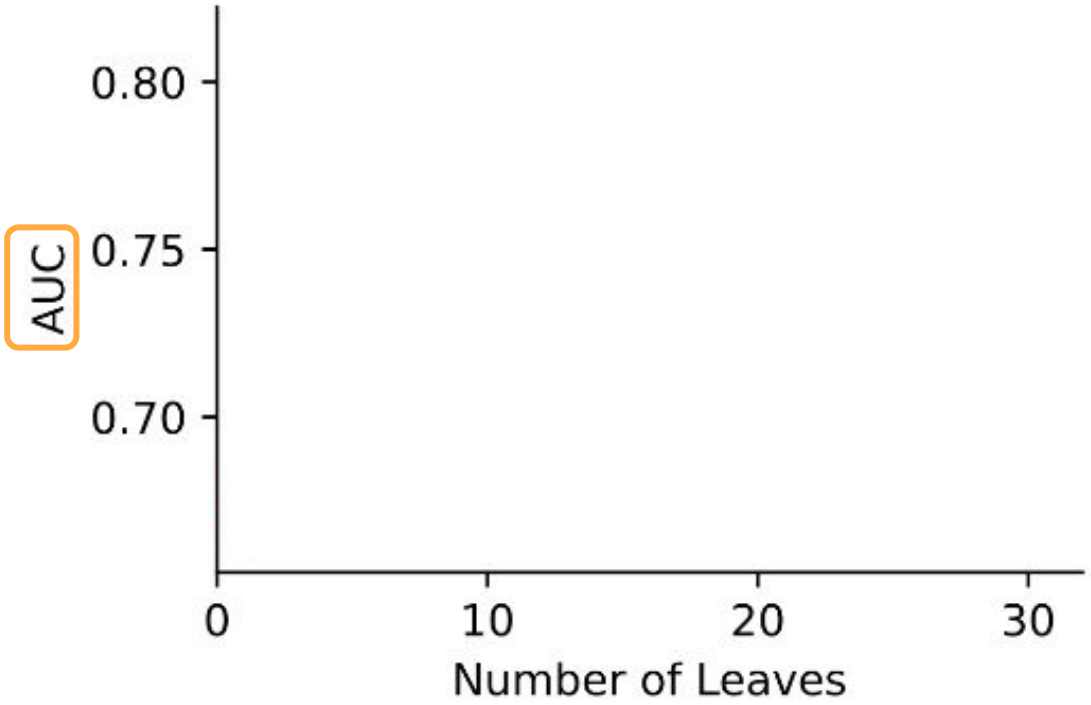
1. Regression results: r^2 vs *no. of leaves*



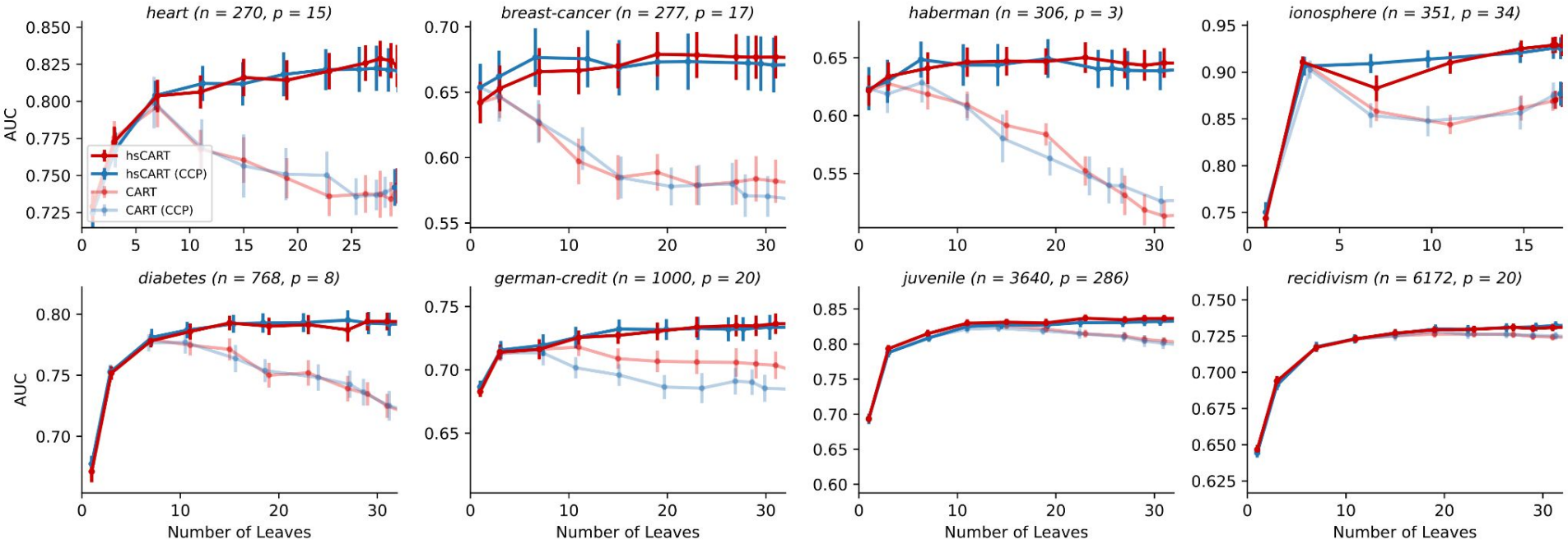

geographical-music (n = 1059, p = 117)
0.55
0.50
0.45
0.40
0.35
0
10
20
30


friedman1 (n = 200, p = 10)
friedman3 (n = 200, p = 4)
diabetes-regr (n = 442, p = 10)
geographical-music (n = 1059, p = 117)
red-wine (n = 1599, p = 11)
abalone (n = 4177, p = 8)
california-housing (n = 20640, p = 8)
hsCART
hsCART (CCP)
CART
CART (CCP)
R^2
Number of Leaves
Sample size n
Performance gap is **larger** for **smaller** datasets

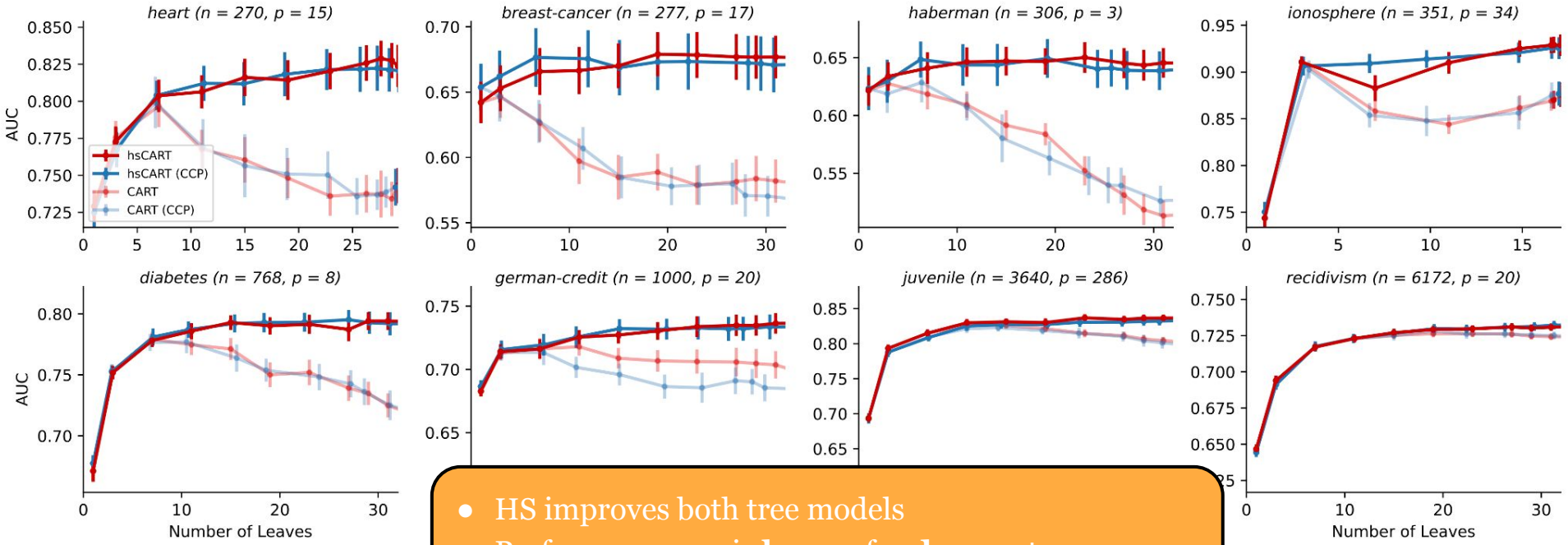
2. Classification results: *AUROC* vs *no. of leaves*



2. Classification results: *AUROC* vs *no. of leaves*

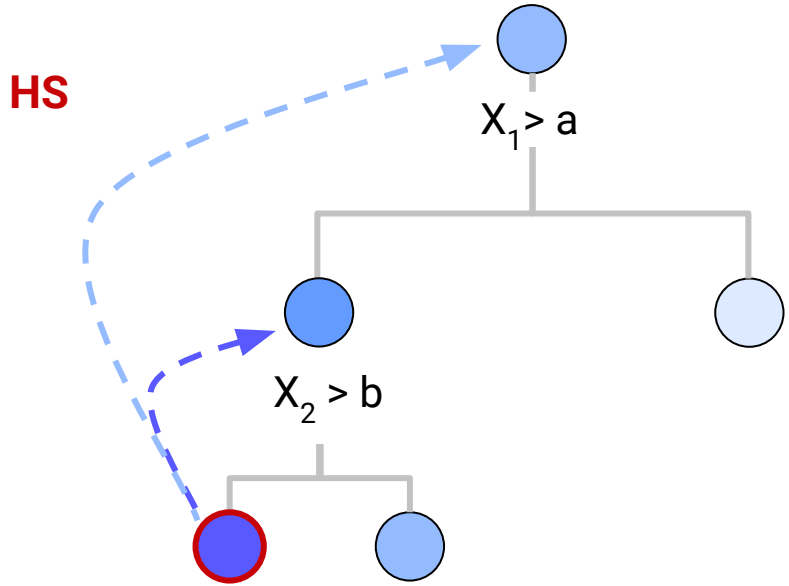


2. Classification results: *AUROC* vs *no. of leaves*

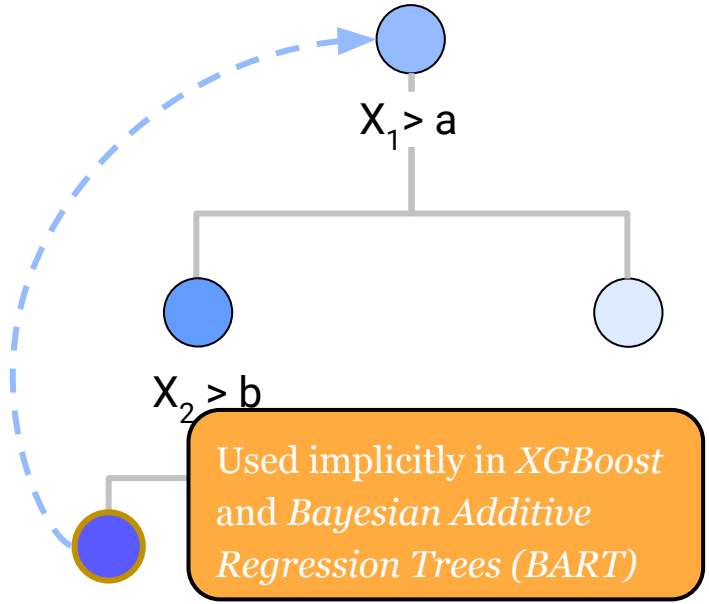


- HS improves both tree models
- Performance gap is **larger** for **deeper** trees
- Performance gap is **larger** for **smaller** datasets

3. An alternate shrinkage scheme: Leaf-based shrinkage

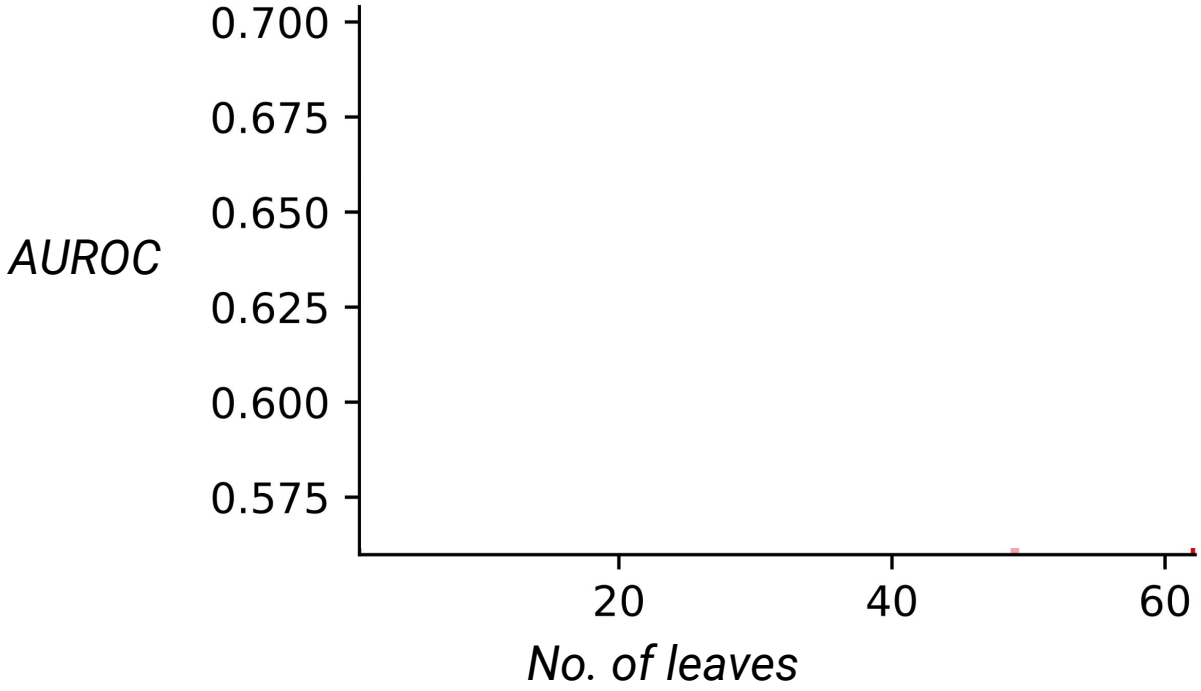


New prediction for leftmost leaf
= 0.80 × ● + 0.11 × ● + 0.09 × ●

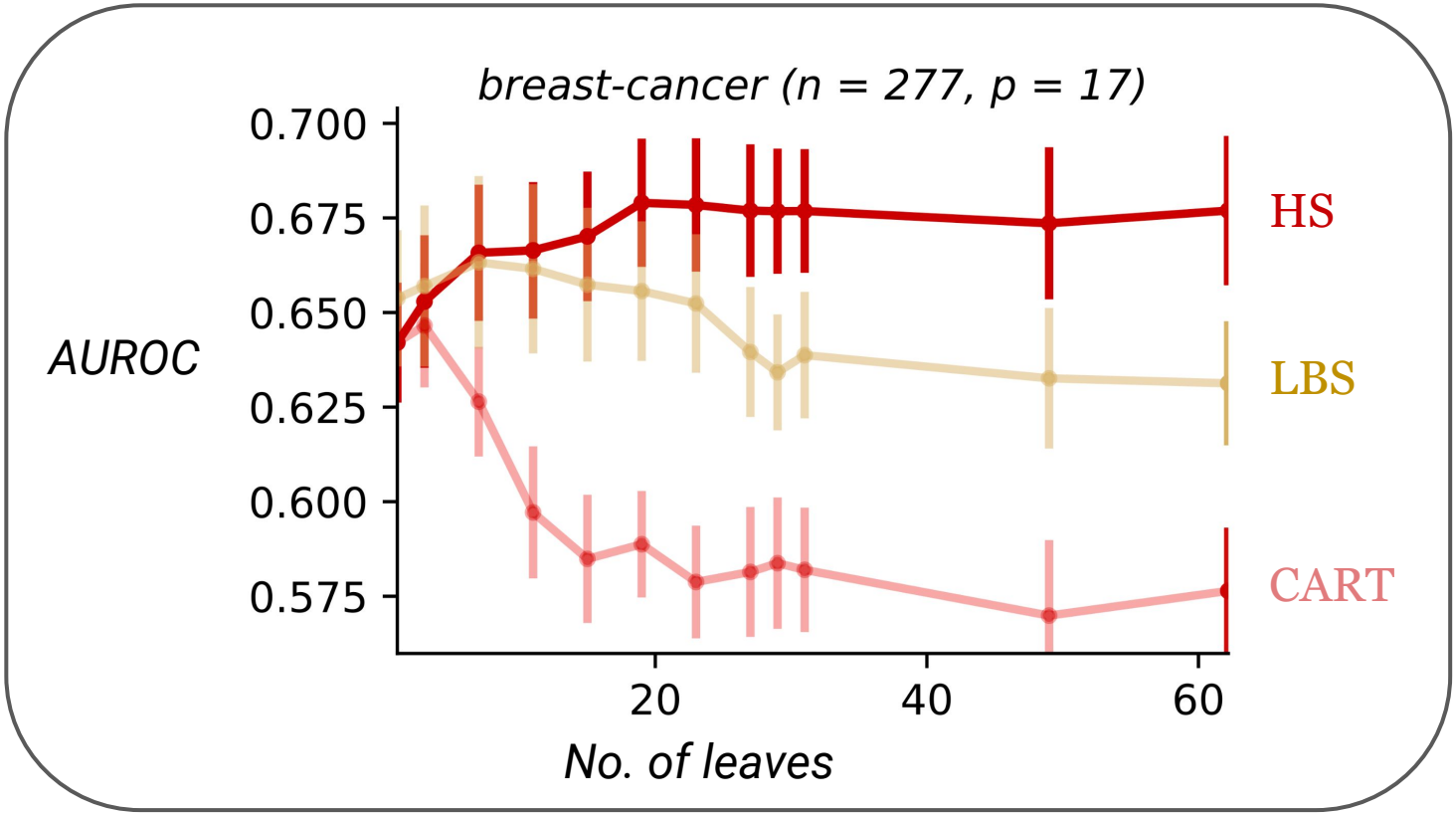


New prediction for leftmost leaf
= 0.80 × ● + 0.20 × ●

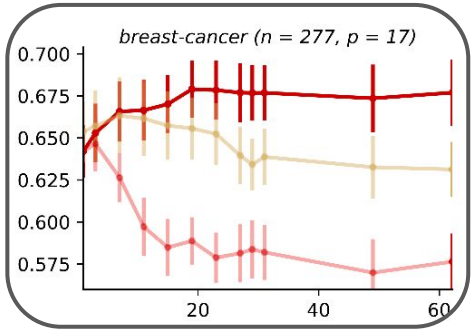
3. Hierarchical shrinkage vs. Leaf-based shrinkage



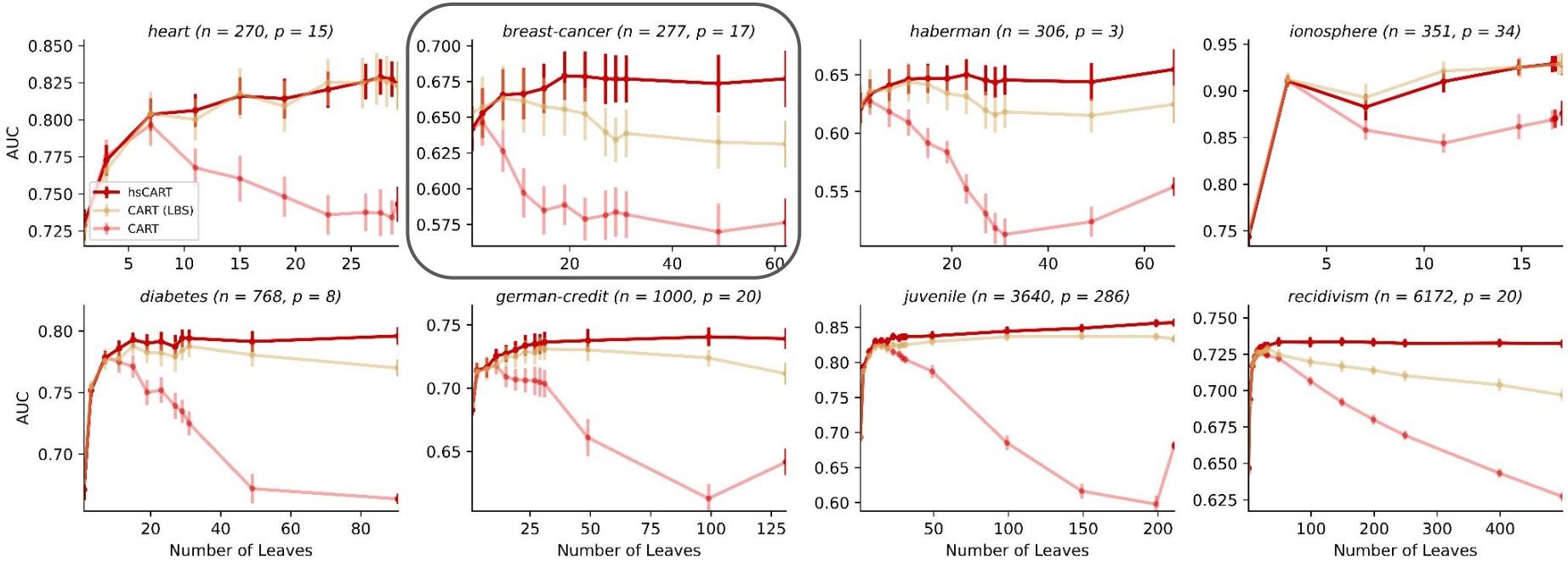
3. Hierarchical shrinkage vs. Leaf-based shrinkage



3. Hierarchical shrinkage vs. Leaf-based shrinkage



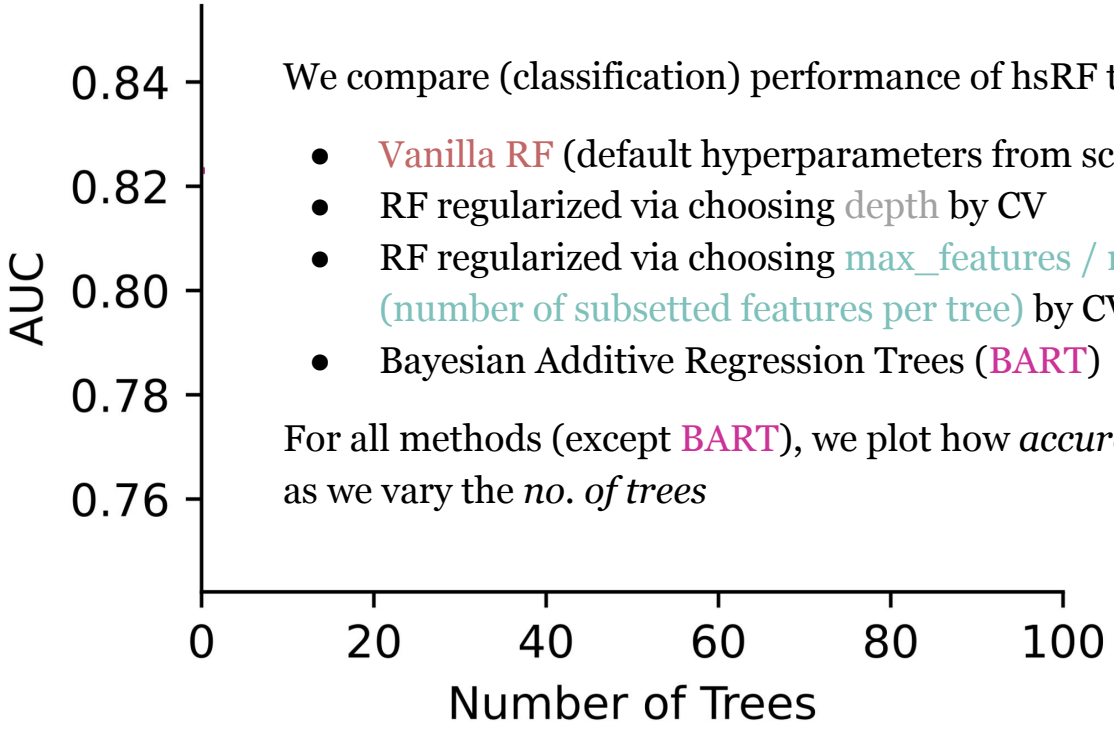
3. Hierarchical shrinkage vs. Leaf-based shrinkage



4. Applying HS to random forest (hsRF)

- HS can be applied to individual trees in an RF to regularize it
- How are RFs regularized?
  - Trees in RF typically not regularized
  - Breiman's insight: Randomness in RF acts as *implicit* regularization
- However, we show that HS improves RF performance significantly

4. Applying HS to random forest (hsRF)

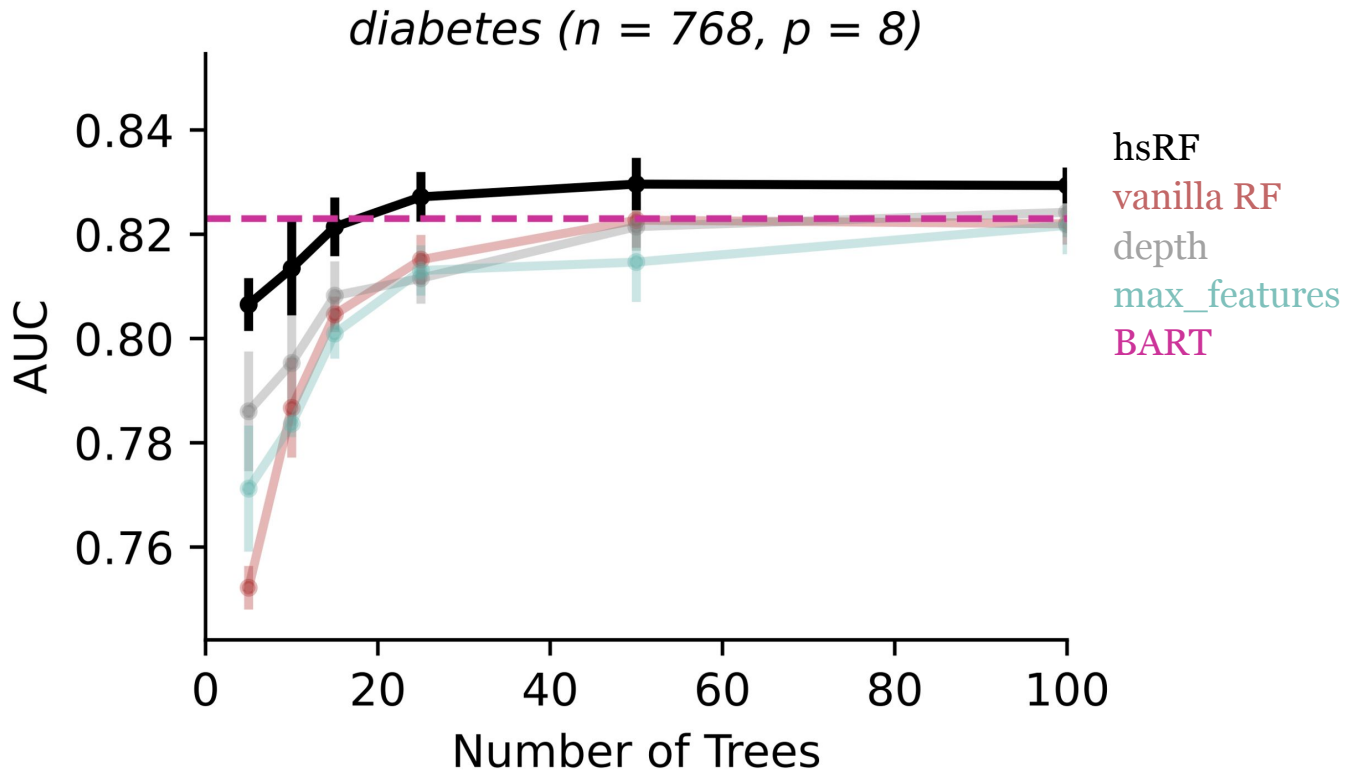


We compare (classification) performance of hsRF to

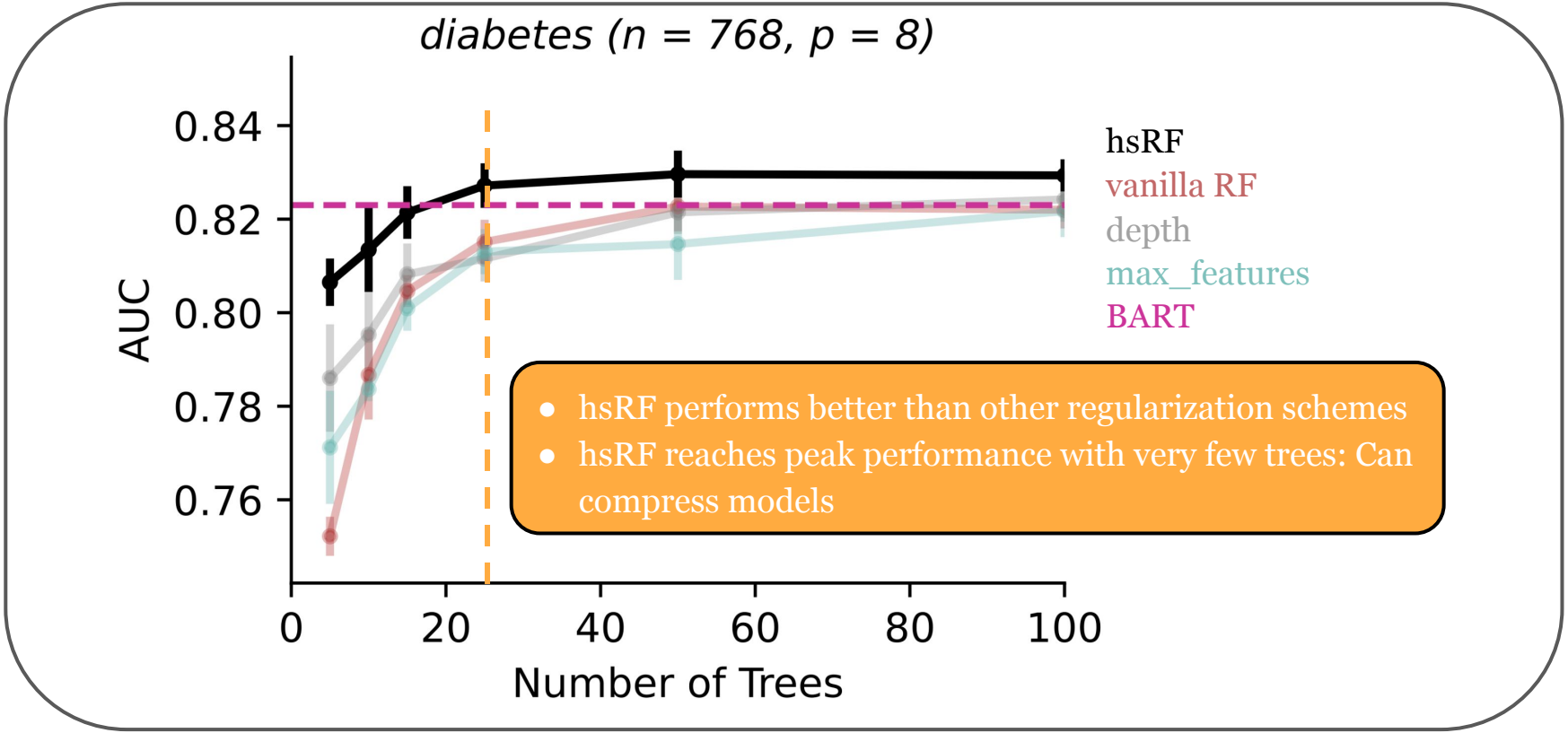
- Vanilla RF (default hyperparameters from scikit-learn)
- RF regularized via choosing depth by CV
- RF regularized via choosing max_features / m_try (number of subsetted features per tree) by CV
- Bayesian Additive Regression Trees (BART)

For all methods (except BART), we plot how *accuracy* changes as we vary the *no. of trees*

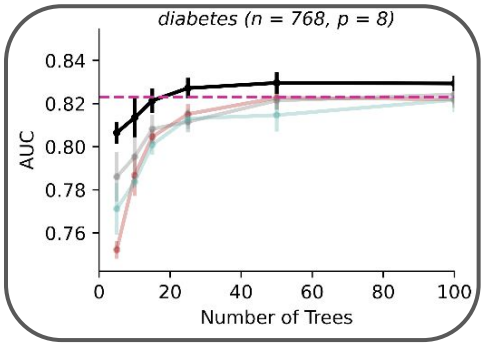
4. Applying HS to random forest (hsRF)



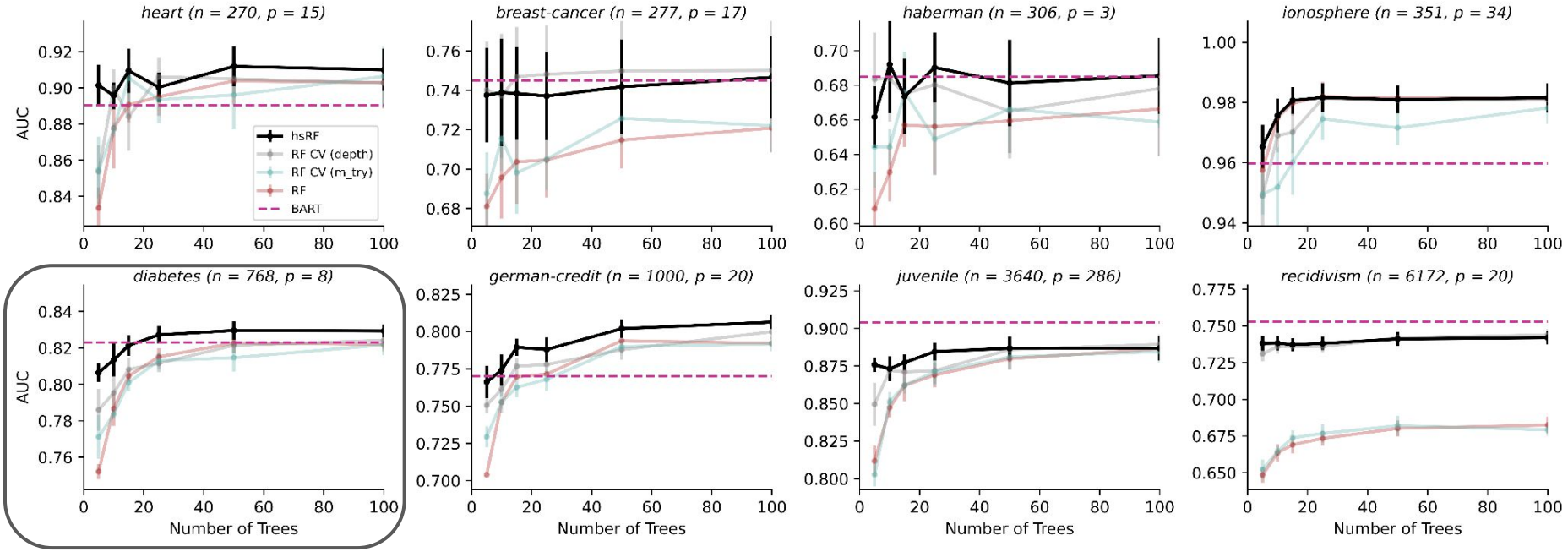
4. Applying HS to random forest (hsRF)



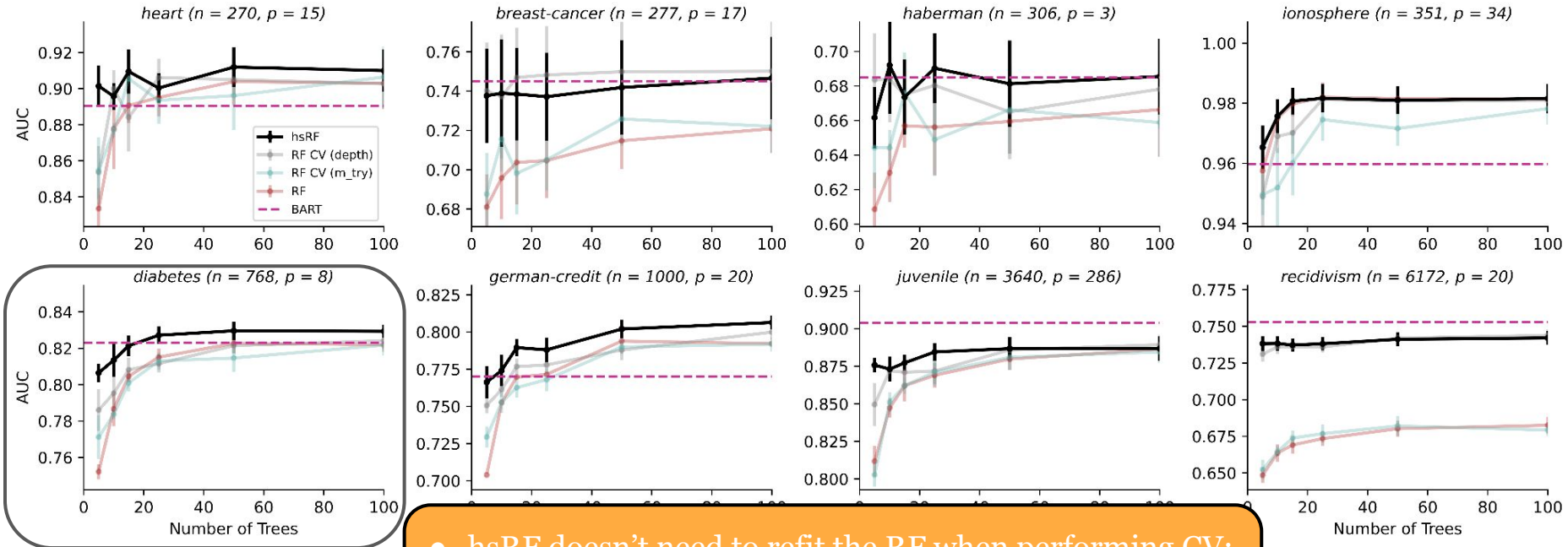
4. Applying HS to random forest (hsRF)



4. Applying HS to random forest (hsRF)



4. Applying HS to random forest (hsRF)



- hsRF doesn't need to refit the RF when performing CV: faster than other methods
- Performance gap still appears at 500 trees

Summary of results (prediction accuracy)

Hierarchical shrinkage

1. Improves prediction accuracy (r^2) on regression datasets
2. Improves prediction accuracy (AUROC) on classification datasets
3. Performs better than alternate shrinkage schemes
4. Improves prediction accuracy of random forests

HS improves interpretability of random forest (RF)

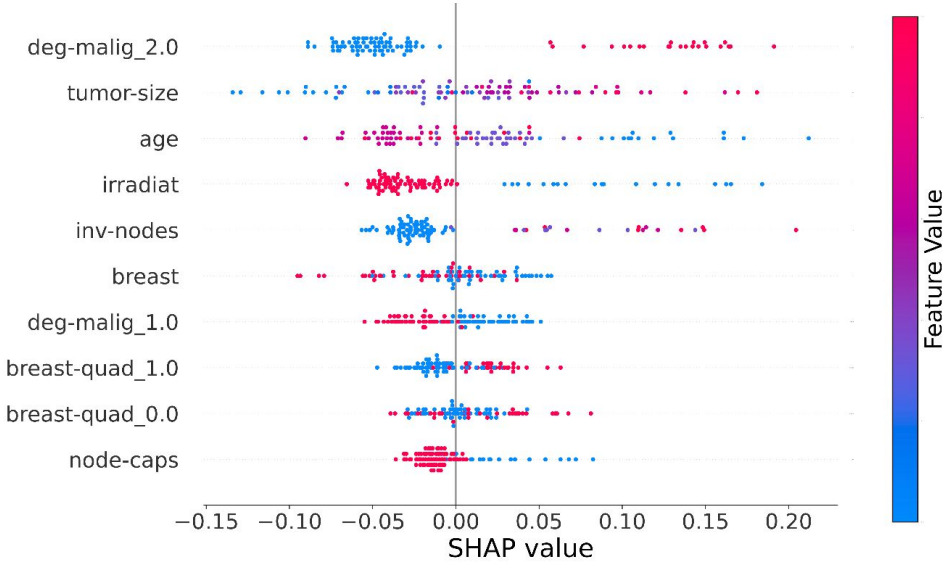
Hierarchical shrinkage

1. Simplifies decision boundaries
2. Makes SHAP values more clustered
3. Makes SHAP values more stable to dataset resampling

2. Refresher on SHAP [Lundberg, Lee (2017)]

- SHAP is a local feature importance score
- Usually summarized in a SHAP plot

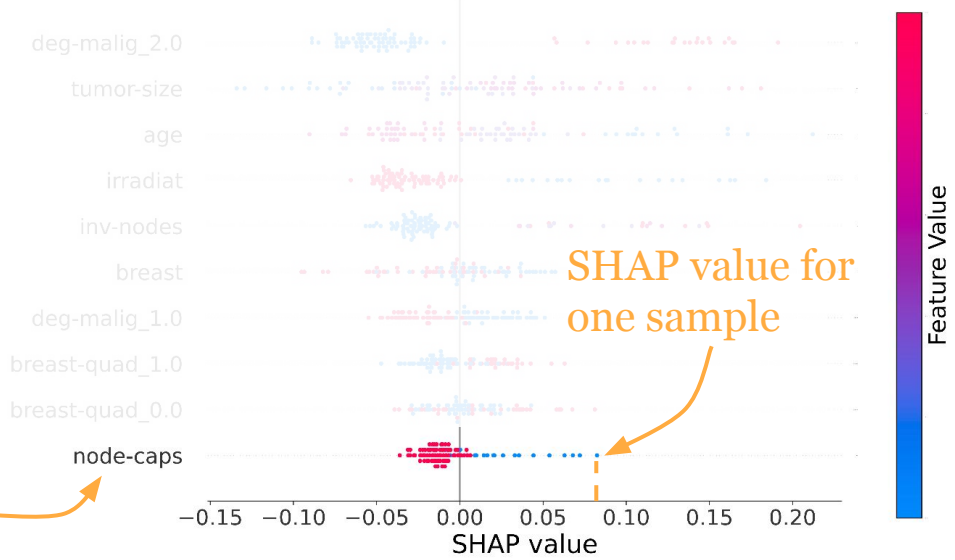
SHAP plot for
breast-cancer dataset



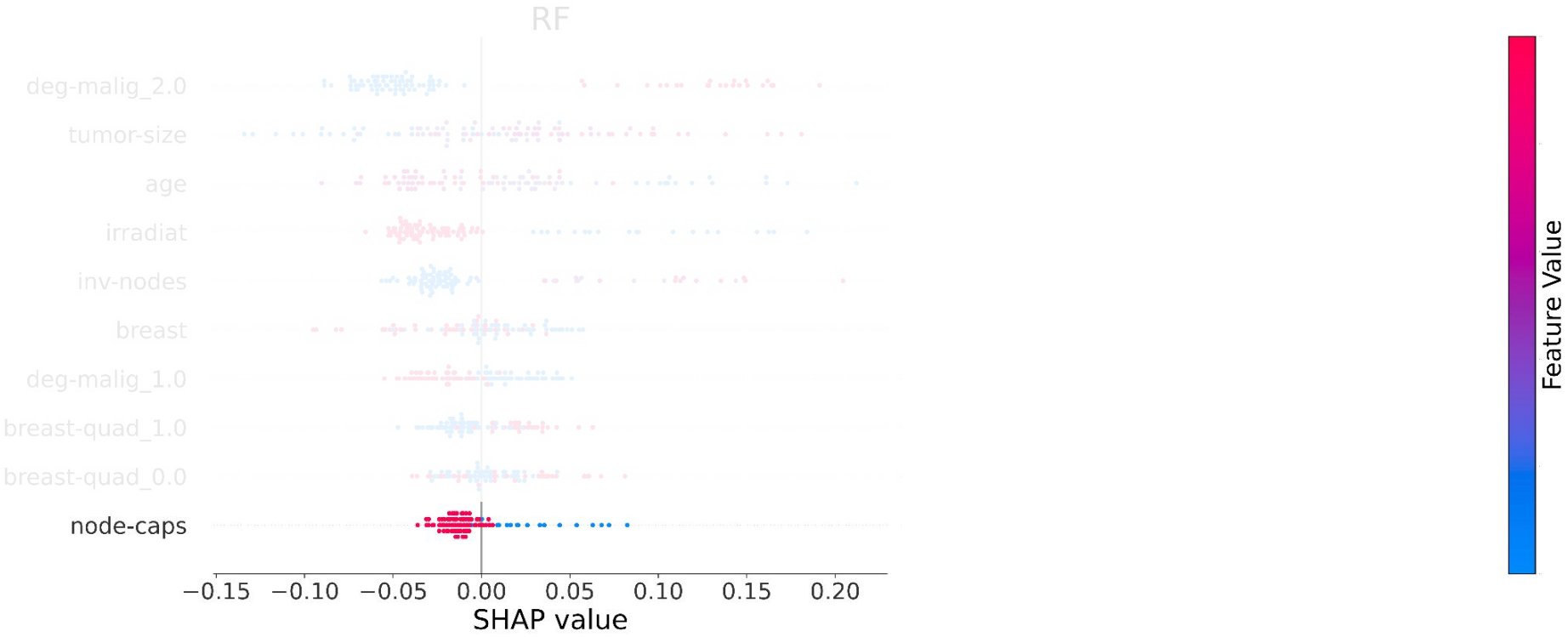
2. Refresher on SHAP [Lundberg, Lee (2017)]

- SHAP is a local feature importance score
- Usually summarized in a SHAP plot

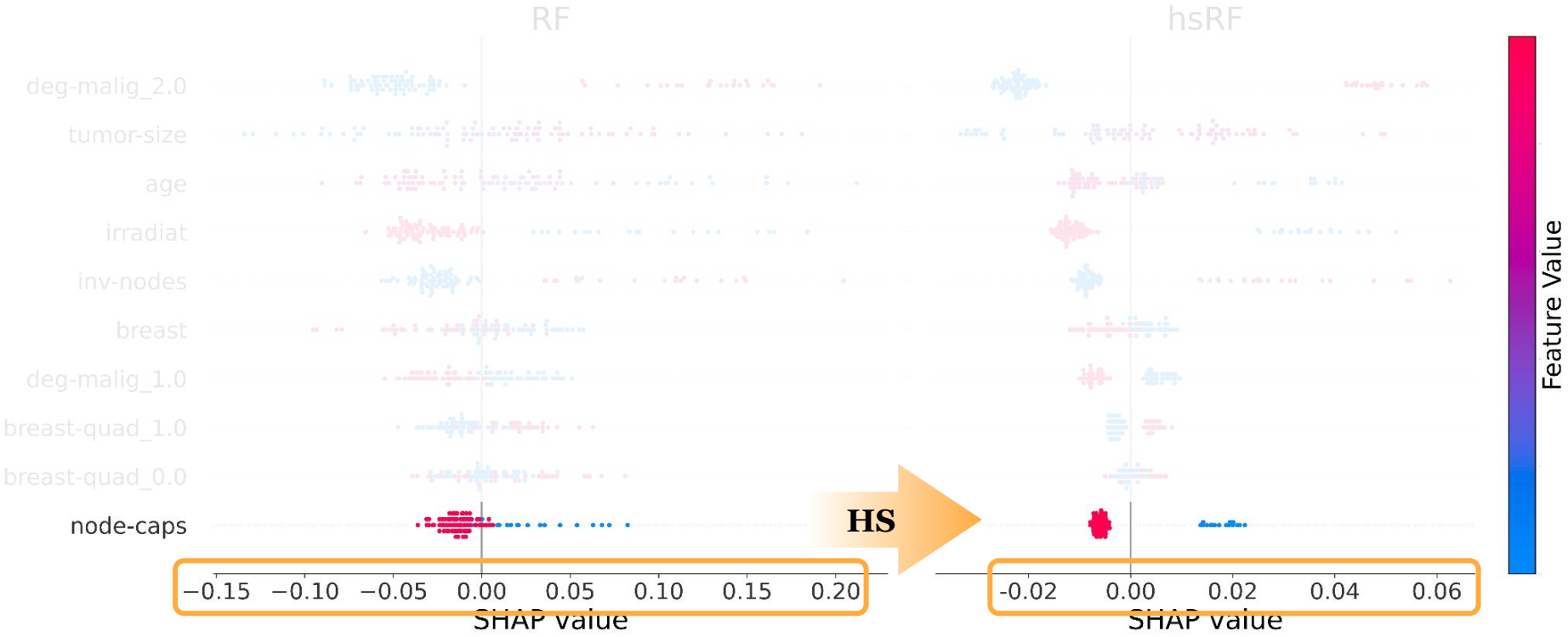
SHAP plot for
breast-cancer dataset



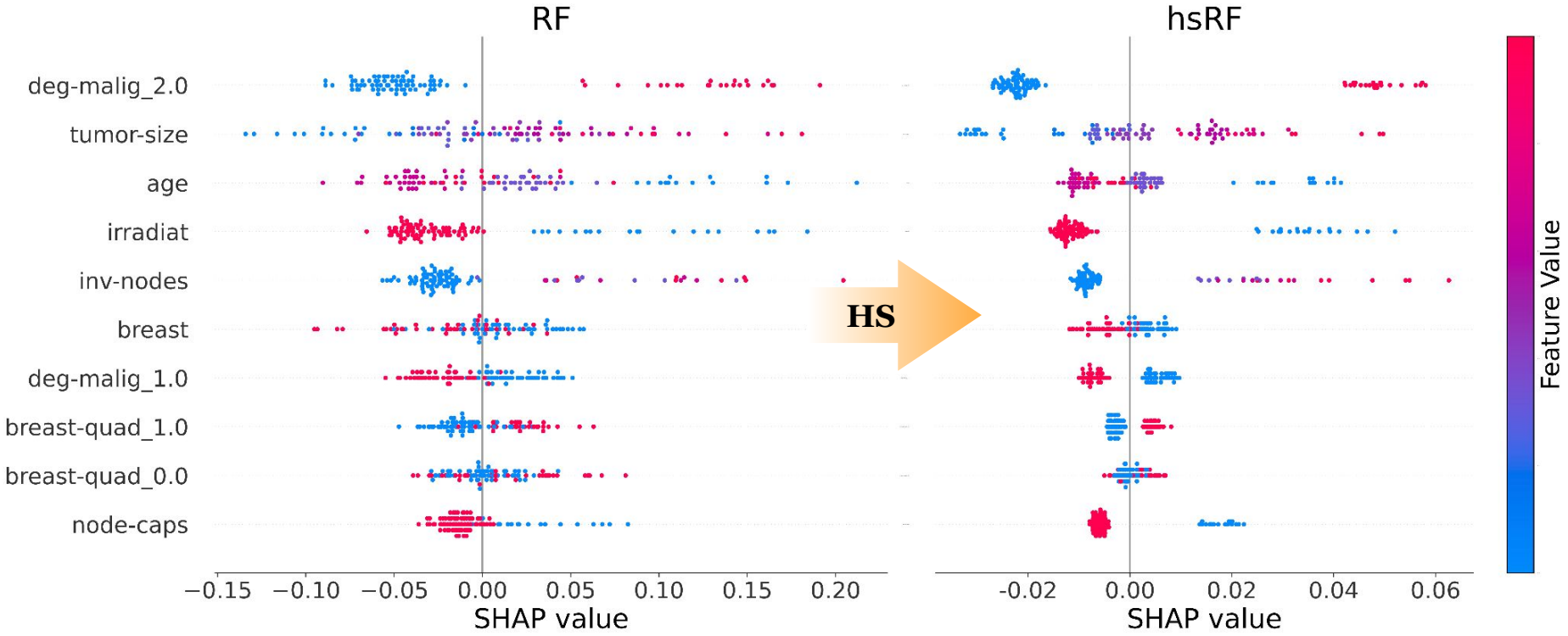
2. HS makes SHAP values more *clustered*



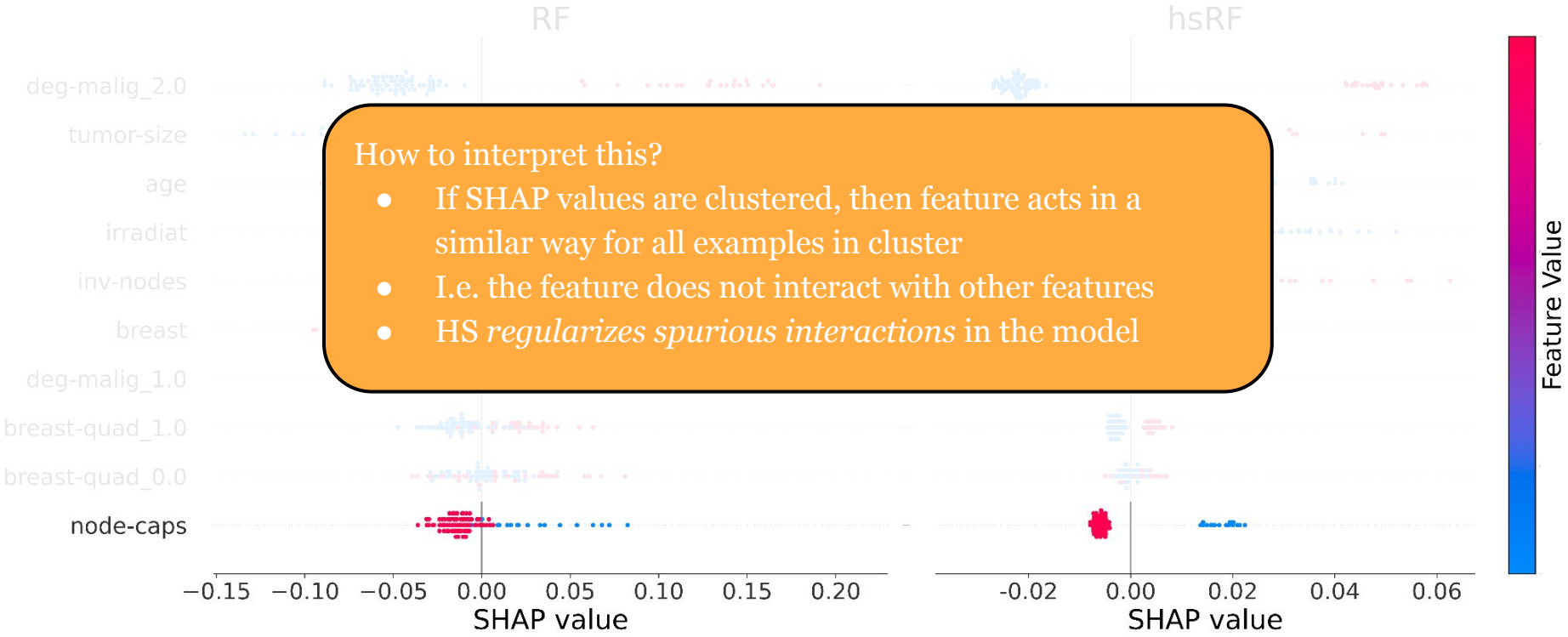
2. HS makes SHAP values more *clustered*



2. HS makes SHAP values more clustered



2. HS makes SHAP values more *clustered*



How to interpret this?

- If SHAP values are clustered, then feature acts in a similar way for all examples in cluster
- I.e. the feature does not interact with other features
- HS *regularizes spurious interactions* in the model

Conclusion

- Hierarchical shrinkage regularizes decision trees by shrinking the value of each node to those of its ancestors
- Is extremely fast and can be applied to any decision tree model or tree ensemble
- Improves prediction accuracy for decision tree and random forest models
- Improves interpretability of random forest models