Optimization of Optimal Sparse Decision Trees

Cynthia Rudin

Duke University

Can a typographical error lead to years of extra prison time?

Can a typographical error lead to years of extra prison time?

The New York Times

OP-ED CONTRIBUTOR

When a Computer Program Keeps You in Jail

By Rebecca Wexler

June 13, 2017

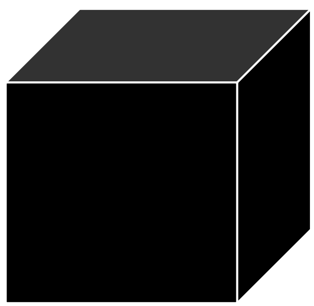
232

Glenn Rodriguez was denied parole because of a miscalculated “COMPAS” score.

How accurate is COMPAS?

COMPAS vs. CORELS

COMPAS: (Correctional Offender Management Profiling for Alternative Sanctions)

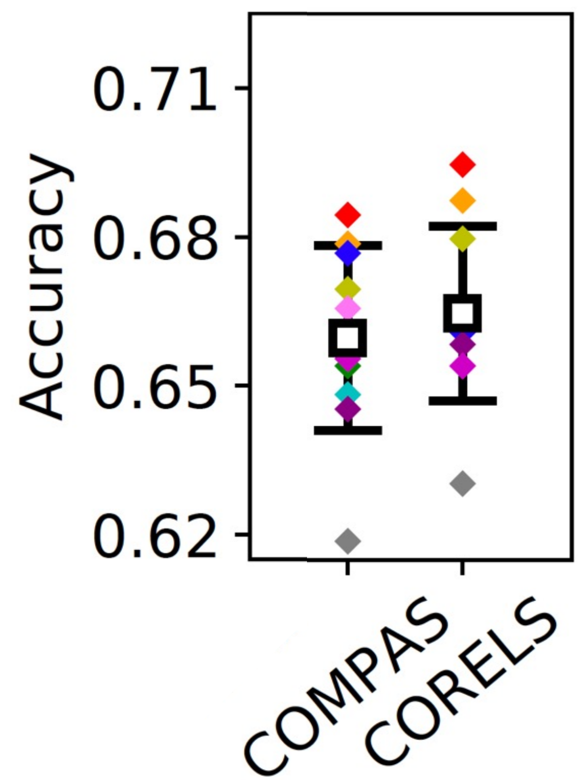


CORELS: (Certifiably Optimal RulE ListS, with Elaine Angelino, Nicholas Larus-Stone, Daniel Alabi, and Margo Seltzer, KDD 2017 & JMLR 2018)

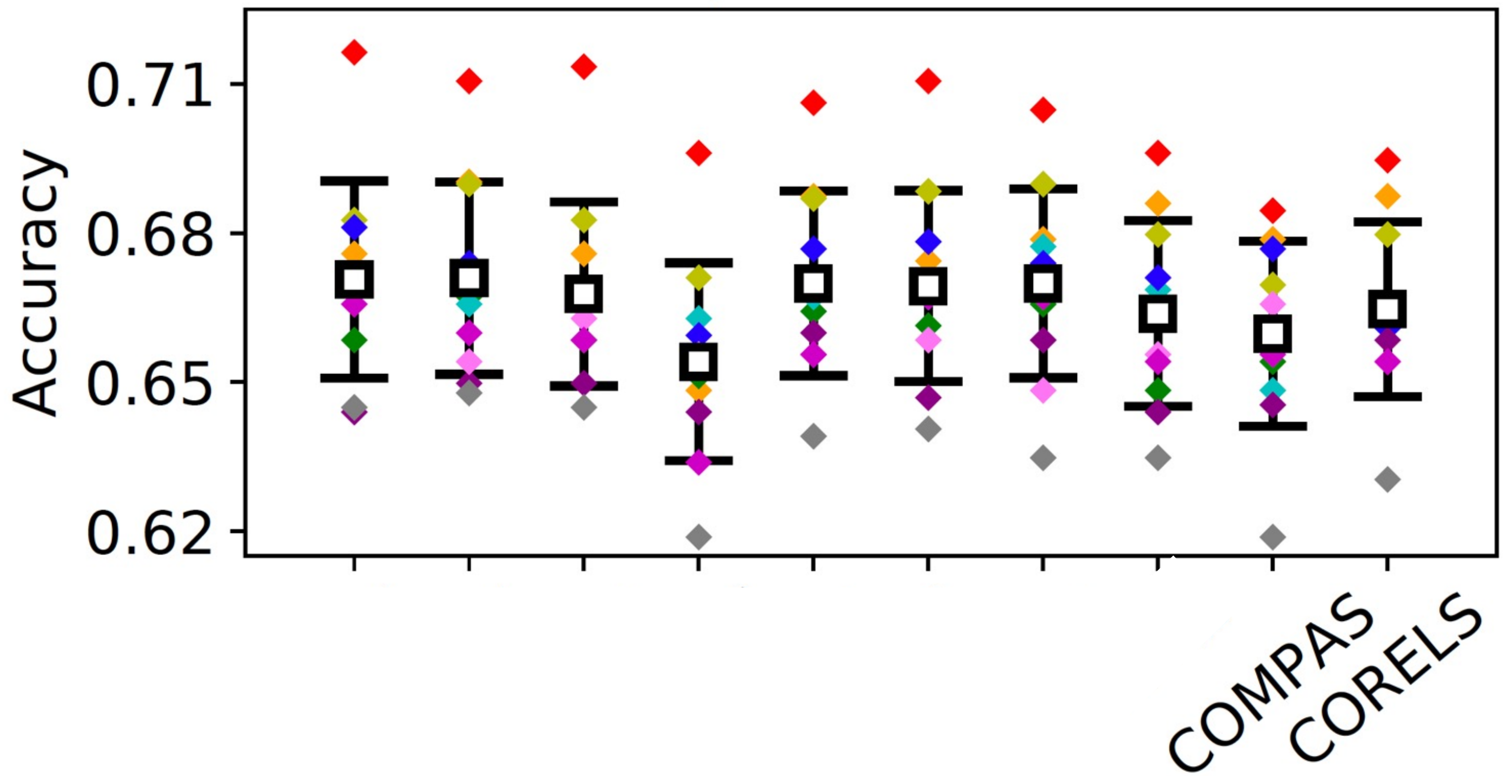
Here is the machine learning model:

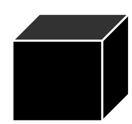
If age=19-20 and sex=male, then predict arrest
else if age=21-22 and priors=2-3 then predict arrest
else if priors >3 then predict arrest
else predict no arrest

Prediction of re-arrest within 2 years



Prediction of re-arrest within 2 years





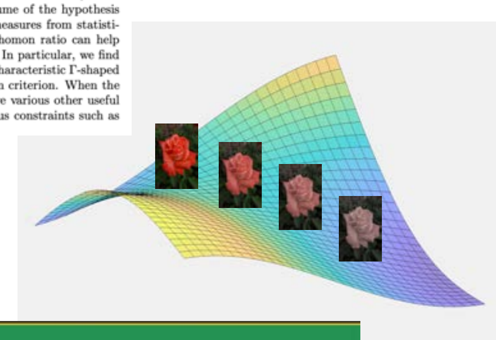
If age=19-20 and sex=male, then predict arrest
else if age=21-22 and priors=2-3 then predict arrest
else if priors >3 then predict arrest
else predict no arrest

* exemplifies a larger phenonenon

Problem spectrum

age 45
congestive heart failure? yes
takes aspirin
smoking? no
gender M
exercise? yes
allergies? no
number of past strokes 2
diabetes? yes

The *Rashomon effect* occurs when many different explanations exist for the same phenomenon. In machine learning, Leo Breiman used this term to characterize problems where many accurate-but-different models exist to describe the same data. In this work, we study how the Rashomon effect can be useful for understanding the relationship between training and test performance, and the possibility that simple-yet-accurate models exist for many problems. We consider the *Rashomon set*—the set of almost-equally-accurate models for a given problem—and study its properties and the types of models it could contain. We present the *Rashomon ratio* as a new measure related to simplicity of model classes, which is the ratio of the volume of the set of accurate models to the volume of the hypothesis space; the Rashomon ratio is different from standard complexity measures from statistical learning theory. For a hierarchy of hypothesis spaces, the Rashomon ratio can help modelers to navigate the trade-off between simplicity and accuracy. In particular, we find empirically that a plot of empirical risk vs. Rashomon ratio forms a characteristic Γ-shaped *Rashomon curve*, whose elbow seems to be a reliable model selection criterion. When the Rashomon set is large, models that are accurate—but that also have various other useful properties—can often be obtained. These models might obey various constraints such as interpretability, fairness, or monotonicity.



Tabular: All features are interpretable

- many problems in criminal justice, healthcare, social sciences, equipment reliability & maintenance, etc.
- features include counts, categorical data

Raw: Features are individually uninterpretable

- pixels/voxels, words, a bit of a sound wave

Problem spectrum

Very sparse models (trees, scoring systems)

With minor pre-processing, all
methods have similar performance

Neural networks

Tabular: All features are interpretable

- many problems in criminal justice, healthcare, social sciences, equipment reliability & maintenance, etc.
- features include counts, categorical data

Raw: Features are individually uninterpretable

- pixels/voxels, words, a bit of a sound wave

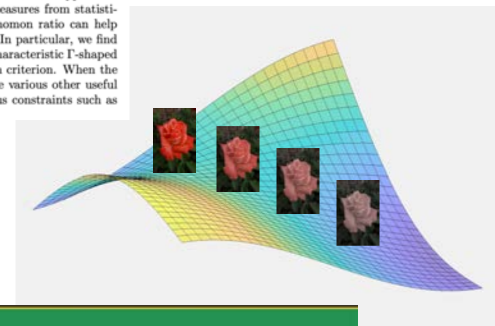
Problem spectrum

age 45
congestive heart failure? yes
takes aspirin
smoking? no
gender M
exercise? yes
allergies? no
number of past strokes 2
diabetes? yes

Tabular: All features are interpretable

- many problems in criminal justice, healthcare, social sciences, equipment reliability & maintenance, etc.
- features include counts, categorical data

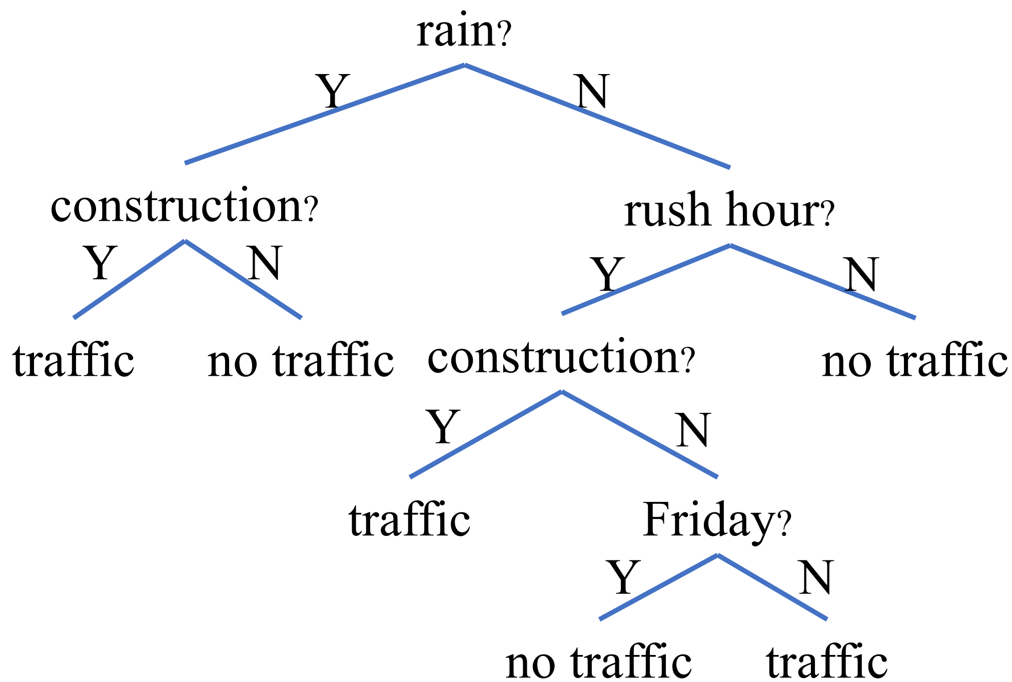
The *Rashomon effect* occurs when many different explanations exist for the same phenomenon. In machine learning, Leo Breiman used this term to characterize problems where many accurate-but-different models exist to describe the same data. In this work, we study how the Rashomon effect can be useful for understanding the relationship between training and test performance, and the possibility that simple-yet-accurate models exist for many problems. We consider the *Rashomon set*—the set of almost-equally-accurate models for a given problem—and study its properties and the types of models it could contain. We present the *Rashomon ratio* as a new measure related to simplicity of model classes, which is the ratio of the volume of the set of accurate models to the volume of the hypothesis space; the Rashomon ratio is different from standard complexity measures from statistical learning theory. For a hierarchy of hypothesis spaces, the Rashomon ratio can help modelers to navigate the trade-off between simplicity and accuracy. In particular, we find empirically that a plot of empirical risk vs. Rashomon ratio forms a characteristic Γ-shaped *Rashomon curve*, whose elbow seems to be a reliable model selection criterion. When the Rashomon set is large, models that are accurate—but that also have various other useful properties—can often be obtained. These models might obey various constraints such as interpretability, fairness, or monotonicity.



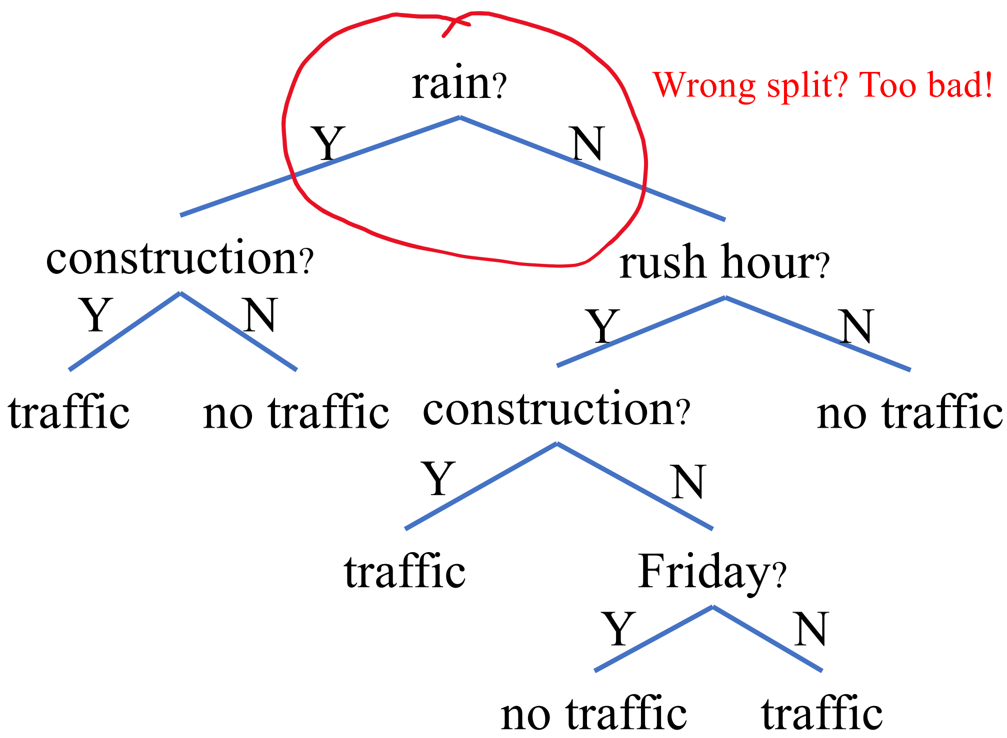
Raw: Features are individually uninterpretable

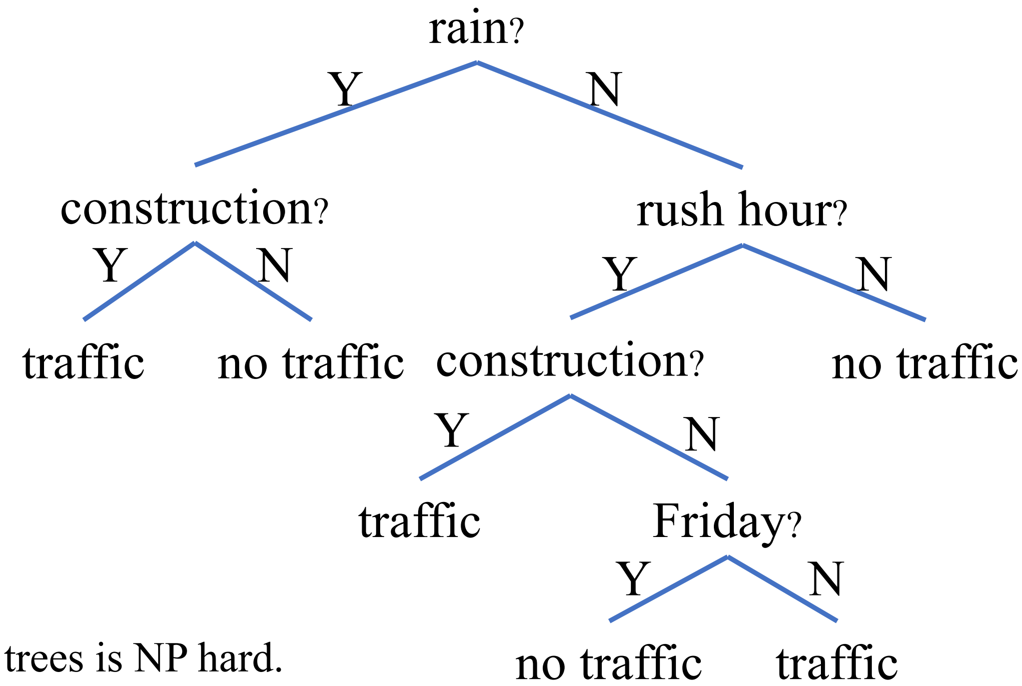
- pixels/voxels, words, a bit of a sound wave

Optimal Sparse Decision Trees



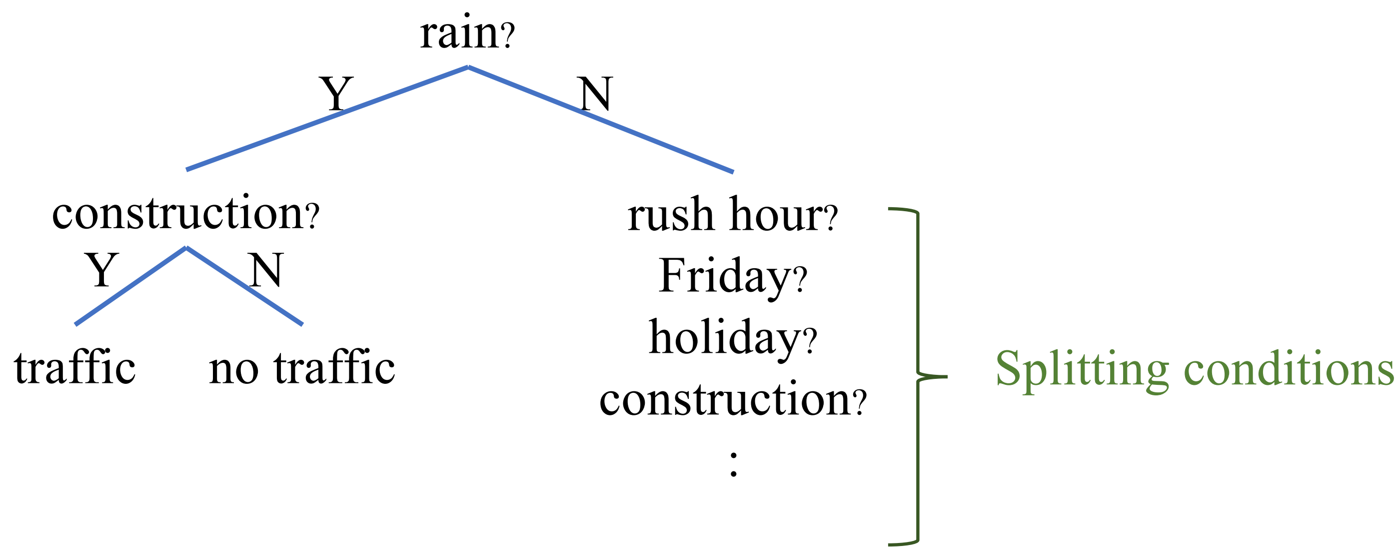
Optimal Sparse Decision Trees





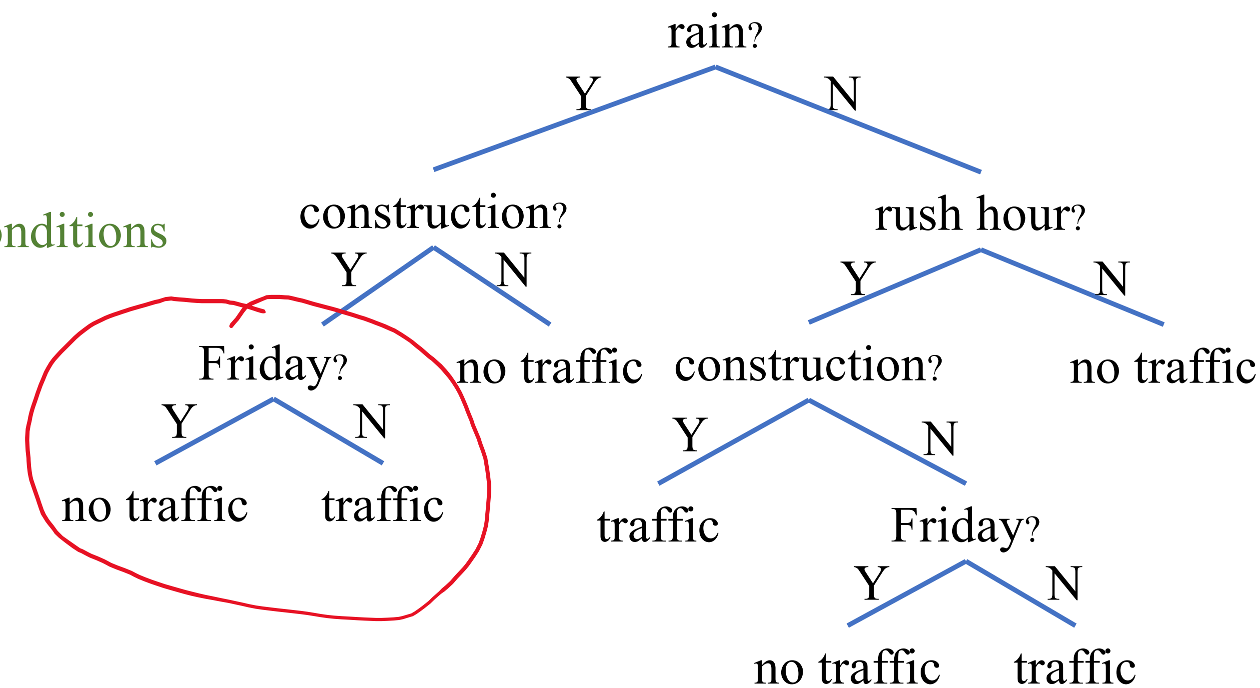
Optimal sparse decision trees is NP hard.
Factorial in the number of variables.

Greedy construction: both the splitting and pruning conditions are based on statistical testing.



Greedy construction: both the splitting and pruning conditions are based on statistical testing.

Pruning conditions



Automatic Interaction Detection (AID) (Morgan & Sonquist, 1963) regression trees

↓

THeta Automatic Interaction Detection (THAID) (Messenger & Mandell, 1972), classification trees

↓

CHi-squared Automatic Interaction Detector (CHAID) (Kass, 1980)

↓

Classification And Regression Trees (CART) (Breiman et al., 1984)

↓

ID3 (Quinlan, 1986), C4.5 (Quinlan, 1993)

Ensemble methods: Random Forest,
Boosted Decision Trees, BART

Global tree optimization, mid-1990's
Bennett, Street, Mangasarian

Global Tree Optimization:
A Non-greedy Decision Tree Algorithm

1994

Kristin P. Bennett
Email bennek@rpi.edu
Department of Mathematical Sciences
Rensselaer Polytechnic Institute
Troy, NY 12180 *

Improvements in splitting criteria for classification and regression
Hypothesis tests, de-biasing (Strobl), missing variables

Tutorials (Murthy 1998, Loh 2014, L. Rokach & O. Maimon 2004 - beware)

Other problems:
longitudinal data, survival curves:
Segal (1992), Simonoff (several papers)

Ensemble methods: Random Forest,
Boosted Decision Trees, BART

Global tree optimization, mid-1990’s
Bennett, Street, Mangasarian

Improvements in splitting criteria for classification and regression
Hypothesis tests, de-biasing (Strobl), missing variables

Other problems:
longitudinal data, survival curves:
Segal (1992), Simonoff (several papers)

Ensemble methods: Random Forest,
Boosted Decision Trees, BART

Global tree optimization, mid-1990's
Bennett, Street, Mangasarian

Improvements in splitting criteria for classification and regression
Hypothesis tests, de-biasing (Strobl), missing variables

Other problems:
longitudinal data, survival curves:
Segal (1992), Simonoff (several papers)

What I hope:

Fully optimal decision trees. User picks objective:

classification accuracy, weighted accuracy, F-score, AUC, partial AUC, precision, recall, etc.

Regularize with sparsity for interpretability.

"Trees sometimes choose irrelevant variables."
"Trees are sometimes 10% worse than ensembles."
"We can't tell how close to optimality our trees are."
"We need new splitting criteria for each objective."

Adapt to handle missing data / biases, etc.

Adapt to other problems

Fully optimal decision trees. User picks objective:

classification accuracy, weighted accuracy, F-score, AUC, partial AUC, precision, recall, etc.

Regularize with sparsity for interpretability.

Fully optimal decision trees. User picks objective:

classification accuracy, weighted accuracy, F-score, AUC, partial AUC, precision, recall, etc.

Regularize with sparsity for interpretability.

(Blanquero et al, 2020, Zantedeschi et al, 2020, S. Aghaei et al, 2020, G. Aglin et al., 2020, E. Demirovic et al. 2020)

Approaches:

- Genetic programming (e.g., Fan & Gray, 2005, Janikow & Malatkar, 2011), or neural networks
    - no optimality gap
- For classification data that is able to be perfectly separated: SAT solvers (Narodytska et al., 2018, Janota 2020)

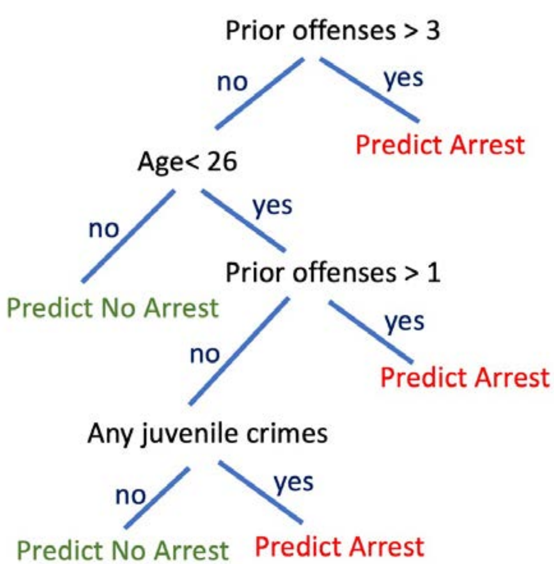
- Mathematical programming solvers (Bennett mid-1990's, Blanquero et al., 2018, Menickelly et al., 2018; Vilas Boas et al., 2019, Verwer & Zhang, BinOCT 2019)

- Dynamic programming / Branch and Bound
    - Garofalakis et al., DTC, 2003 (less relevant since it just finds subtrees of greedy-grown trees)
    - Nijssen & Fromont, DL8, 2007, Nijssen et al., DL8.5, 2020
    - Angelino et al, CORELS, 2018, Hu et al., OSDT 2019, Lin et al., GOSDT, 2020

with Jimmy Lin, Chudi Zhong, Diane Hu, Margo Seltzer

$$\min_{\text{tree}} \; \mathrm{R}\big(\text{tree}, \{(\mathbf{x}_i, y_i)\}_i\big) \text{ where}$$

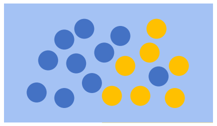
$$\mathrm{R}\big(\text{tree}, \{(\mathbf{x}_i, y_i)\}_i\big) = \underbrace{\frac{1}{n}\sum_{i=1}^{n} 1_{[\text{tree}(\mathbf{x}_i) \neq y_i]}}_{\text{Misclassification error}} + \underbrace{\mathrm{C}\ (\#\text{ leaves in tree})}_{\text{Sparsity}}$$



An example of an optimal tree on the Broward County Florida re-arrest data

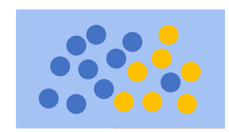
Dynamic programming / Branch and Bound

Start with the full dataset and a naive label

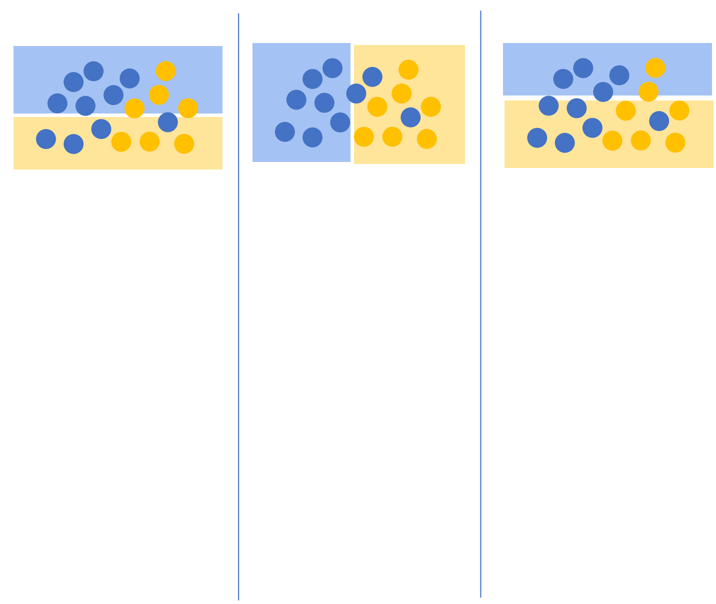


Dynamic programming / Branch and Bound

Start with the full dataset and a naive label



Split it into subsets using each feature

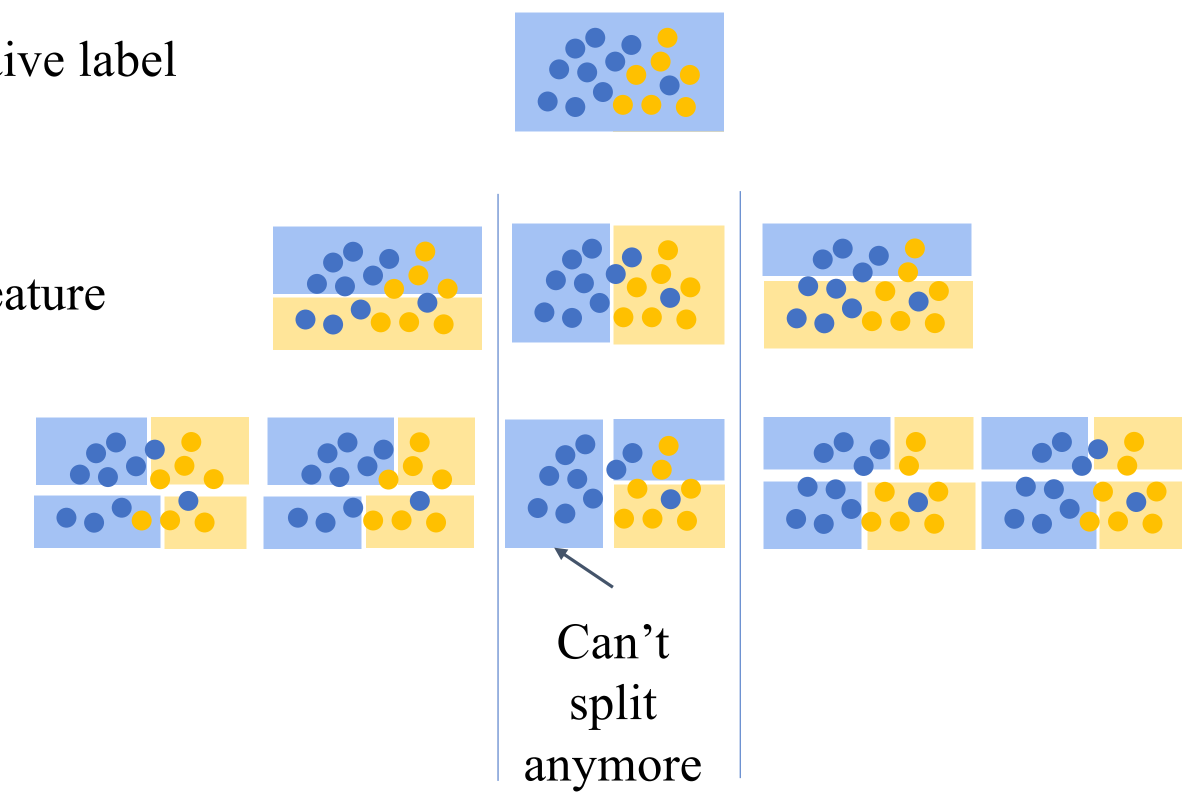


Dynamic programming / Branch and Bound

Start with the full dataset and a naive label

Split it into subsets using each feature

Keep splitting (if permitted)



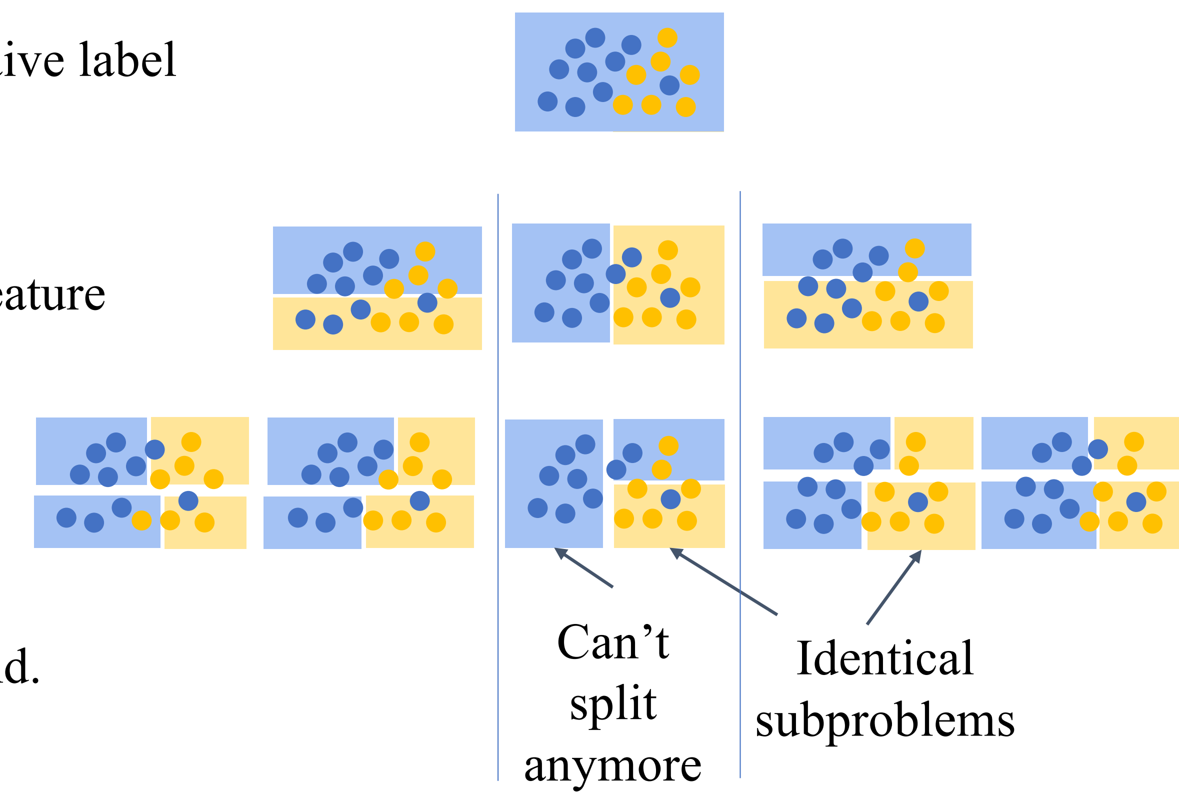
Dynamic programming / Branch and Bound

Start with the full dataset and a naive label

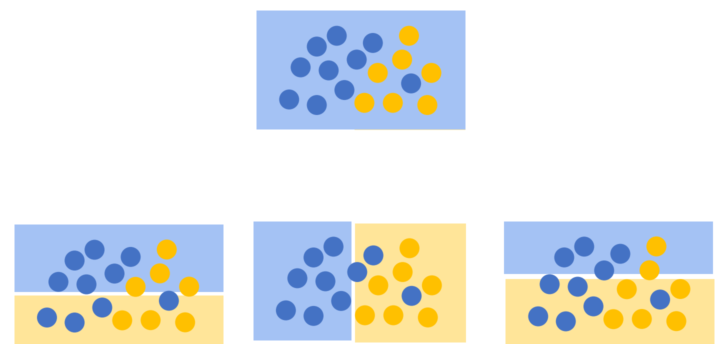
Split it into subsets using each feature

Keep splitting (if permitted)

Consolidate any duplication found.

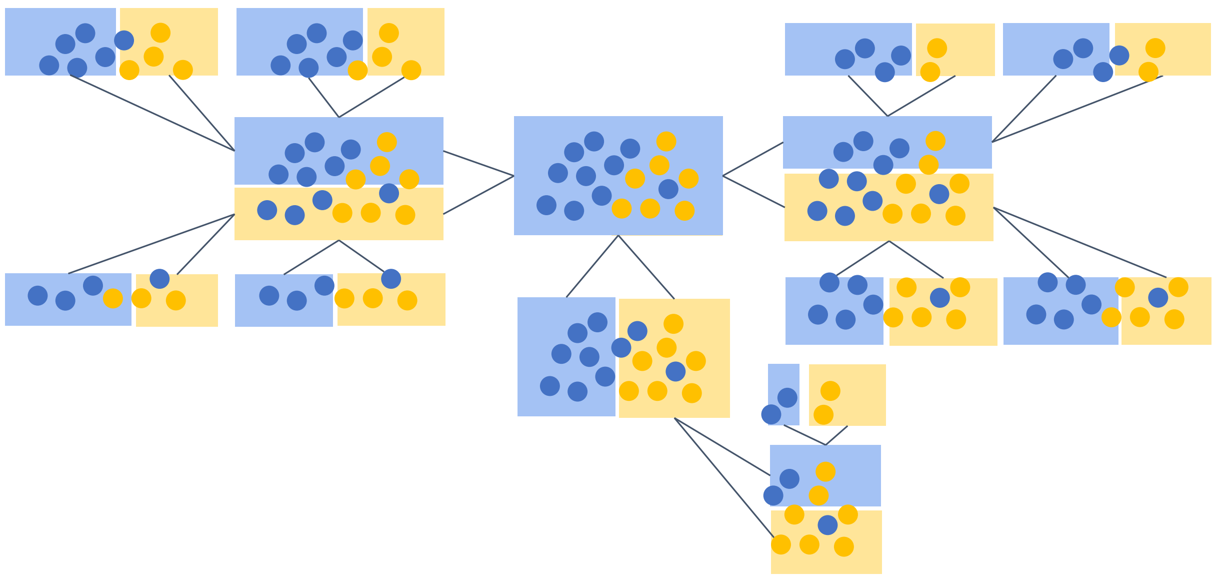


Dynamic programming / Branch and Bound



Dynamic programming / Branch and Bound

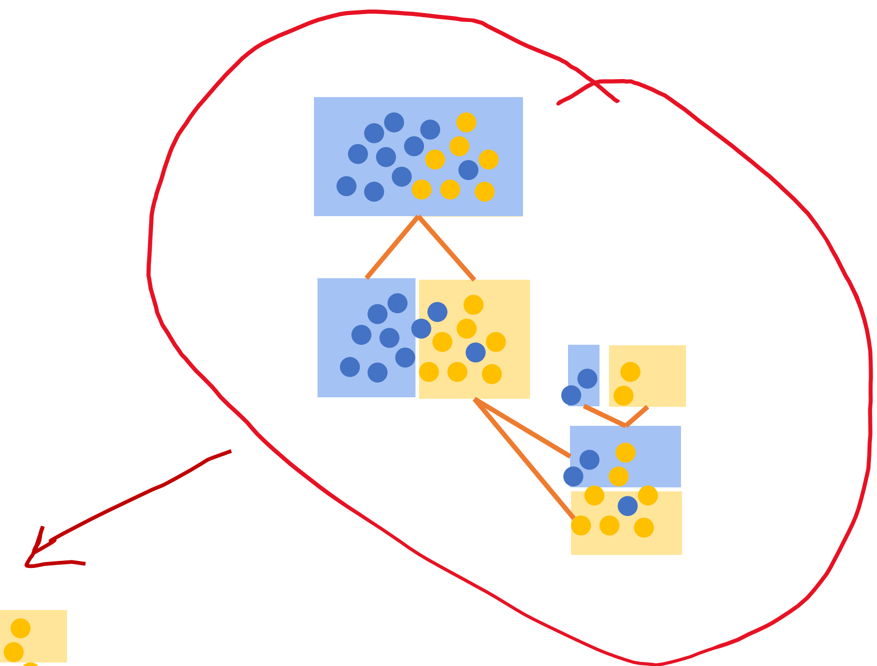
The solution to each subproblem yields the best feature to split on.



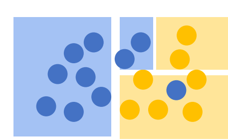
Dynamic programming / Branch and Bound

The solution to each subproblem yields the best feature to split on.

The optimal solution is found after all subproblems are "completed"



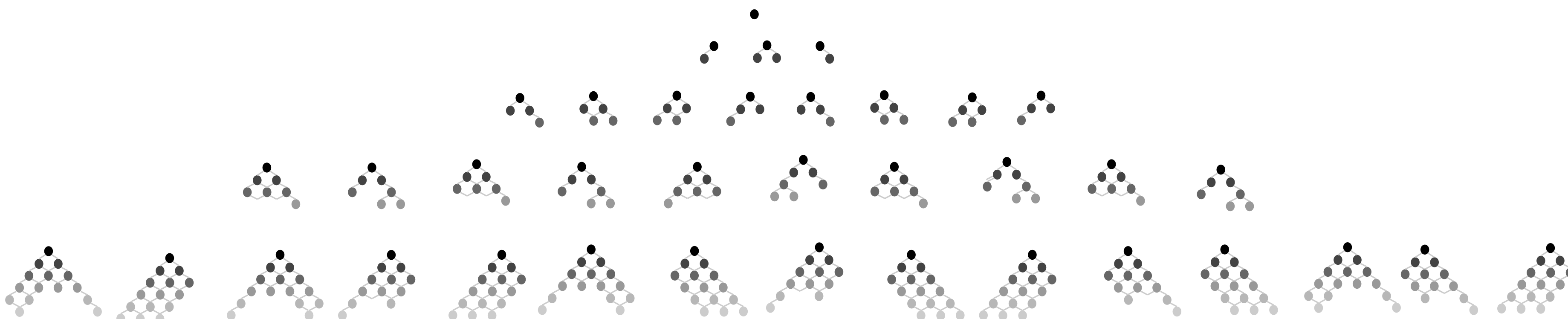
Some subproblems can be proven to yield non-optimal solutions



Dynamic programming / Branch and Bound

Analytical Bounds Reduce the Search Space

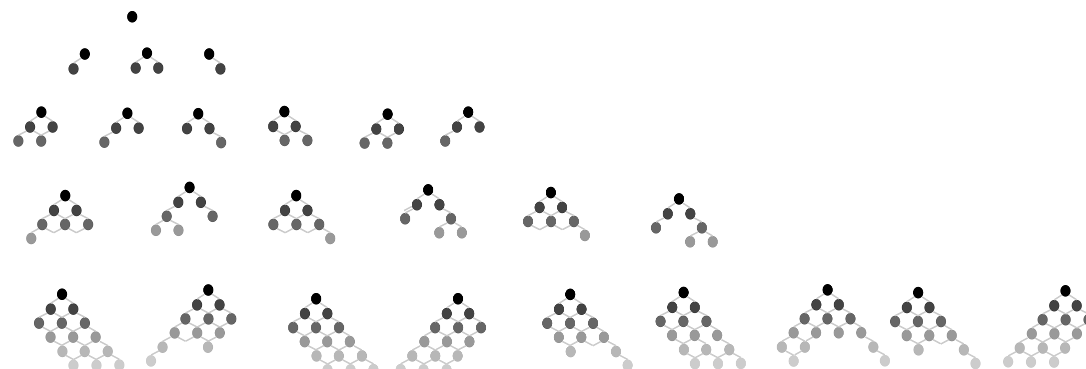
Theorems show that some partial trees can never be extended to form optimal trees.



Dynamic programming / Branch and Bound

Analytical Bounds Reduce the Search Space

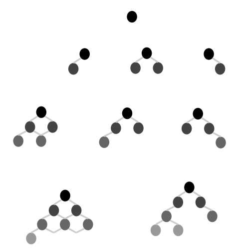
Theorems show that some partial trees can never be extended to form optimal trees.

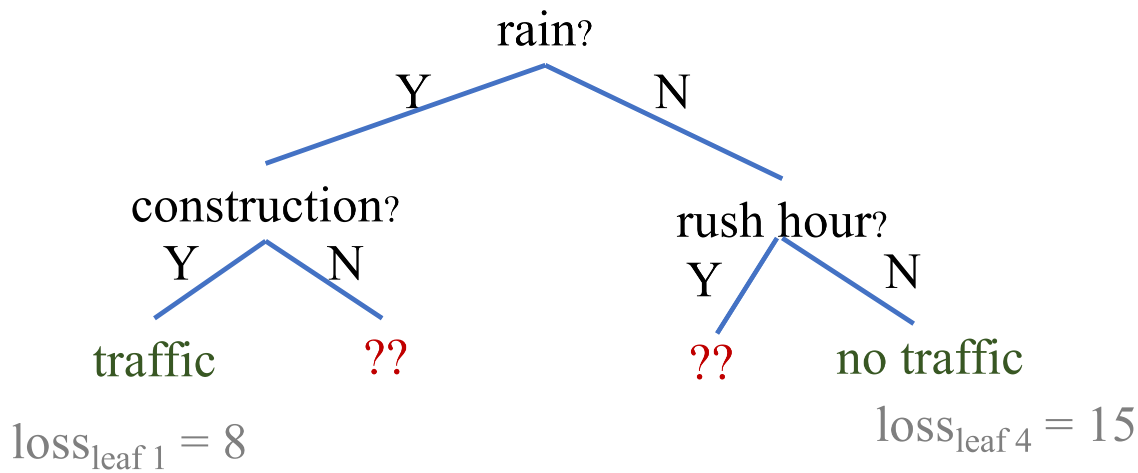


Dynamic programming / Branch and Bound

Analytical Bounds Reduce the Search Space

Theorems show that some partial trees can never be extended to form optimal trees.

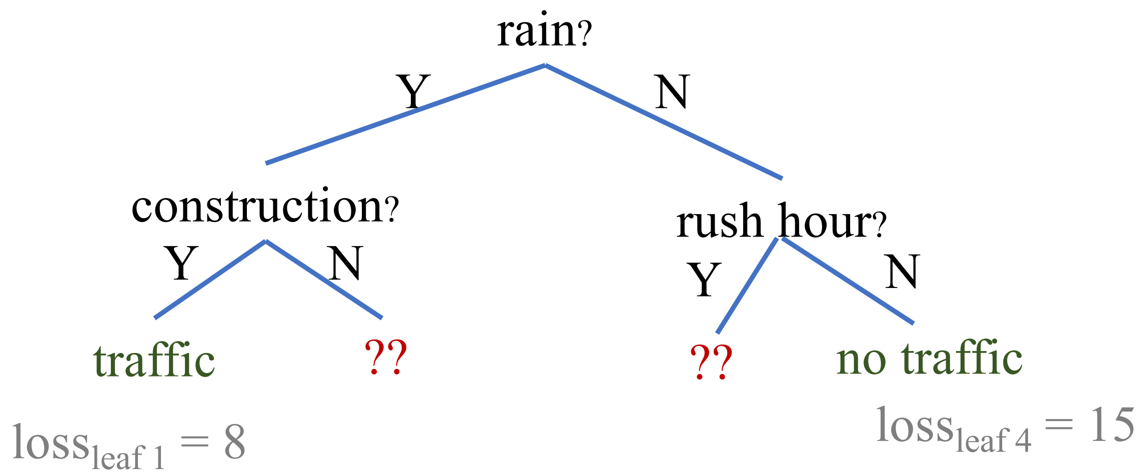




$$R(\text{tree}) = \frac{1}{n}\sum_{\text{leaf}} \text{loss(leaf)} + C\,(\#\text{ leaves})$$

$$= \frac{1}{n}\sum_{\text{fixed leaves}} \text{loss(leaf)} + \frac{1}{n}\sum_{\text{unfixed leaves}} \text{loss(leaf)} + C\,(\#\text{ leaves})$$

$$\geq \frac{1}{n}\sum_{\text{fixed leaves}} \text{loss(leaf)} + 0 + C\,(\#\text{ leaves}) =: b(\text{tree}_{\text{fixed}})$$

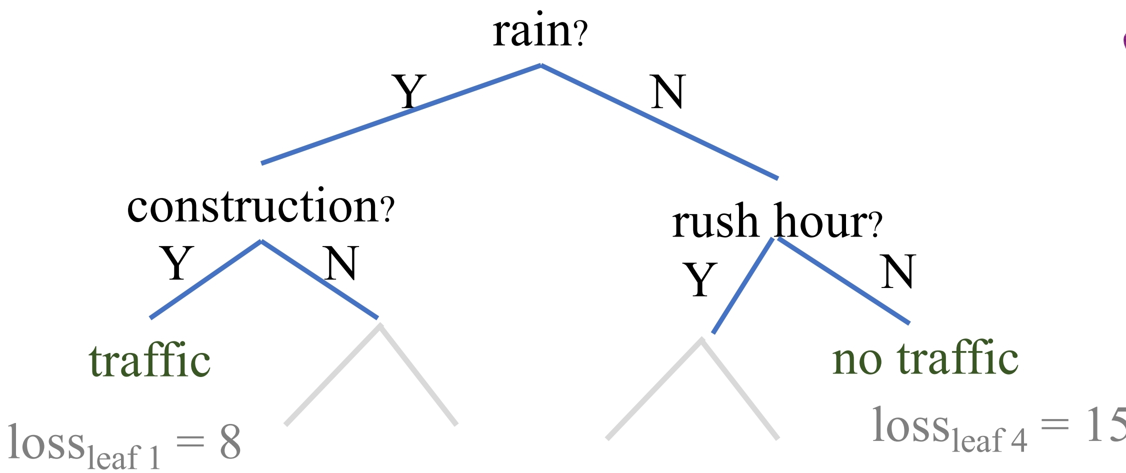


R(tree)

$\geq$

$b(\text{tree}_{\text{fixed}})$

Hierarchical Objective Lower Bound

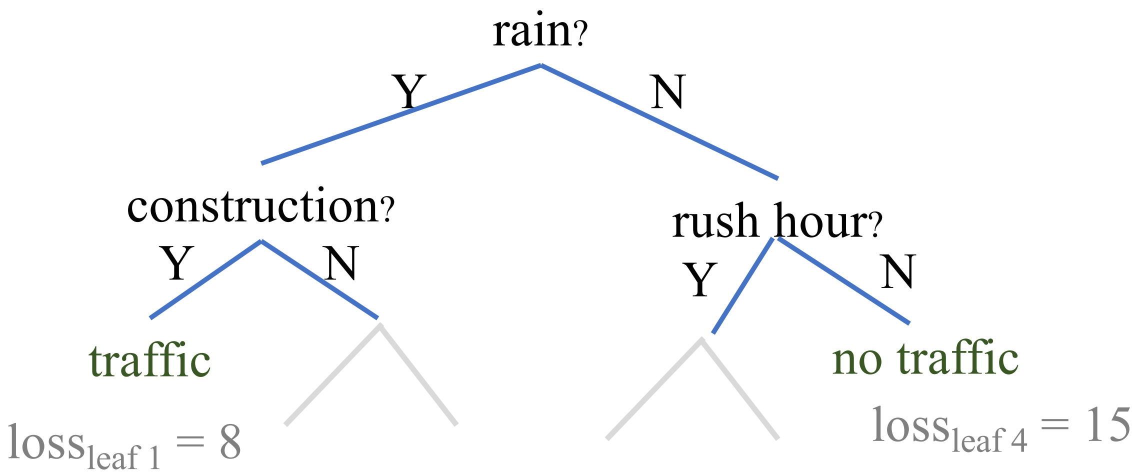


$$R(\text{tree}) \geq b(\text{tree}_{\text{fixed}})$$

Say my current best is a tree with loss $R_{\text{bestsofar}} = \frac{1}{n} \cdot 12 + C(3)$

$$< b(\text{tree}_{\text{fixed}}) \leq R(\text{tree})$$

This tree, and any of its children, will never be as good as current best.



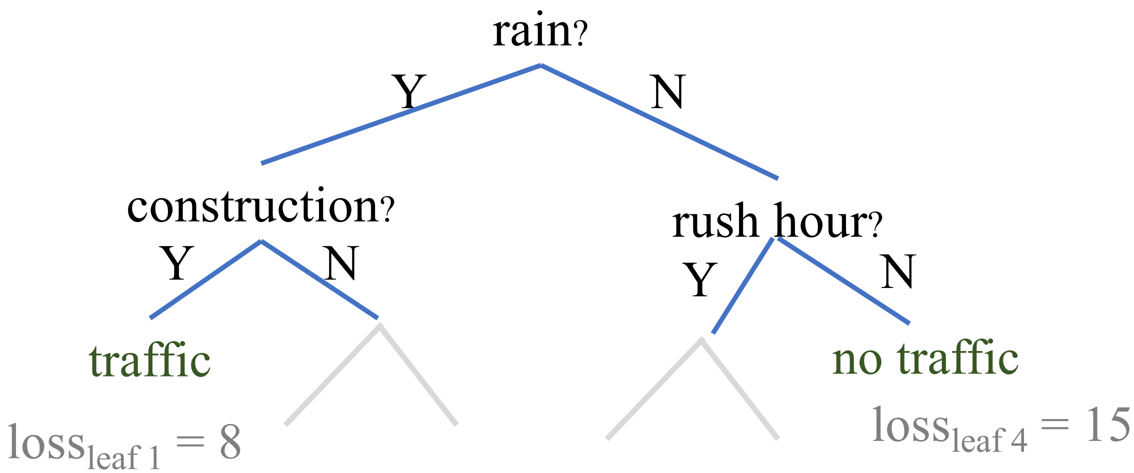
$R(\text{tree}) \geq b(\text{tree}_{\text{fixed}})$

Hierarchical Objective Lower Bound

If $R_{\text{bestsofar}} < b(\text{tree}_{\text{fixed}}) \leq R(\text{tree})$

then this tree, and its children, are all suboptimal.

Hierarchical Objective Lower Bound with Lookahead

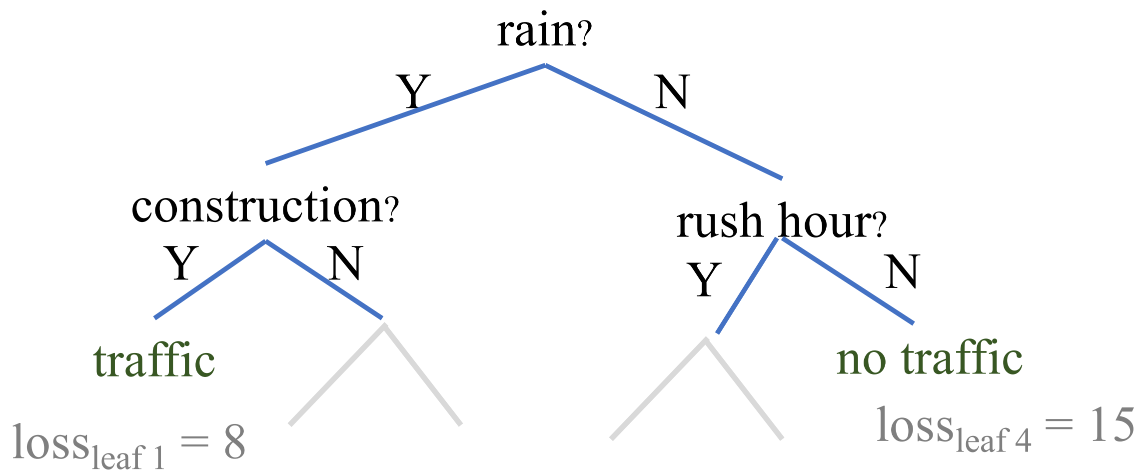


$$R(\text{tree}) \geq b(\text{tree}_{\text{fixed}})$$

ok… but if…

$$b(\text{tree}_{\text{fixed}}) < R_{\text{bestsofar}} < b(\text{tree}_{\text{fixed}}) + C \leq R \text{ (our tree with at least one child)}$$

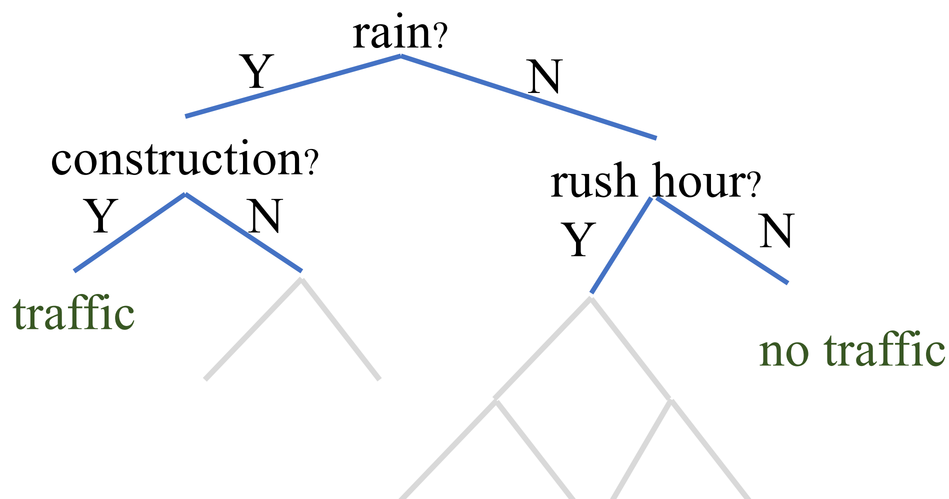
When we add even one child to our tree, it will be worse than current best.



Hierarchical Objective Lower Bound with Lookahead

If $R_{\text{bestsofar}} < b(\text{tree}_{\text{fixed}}) + C$ then

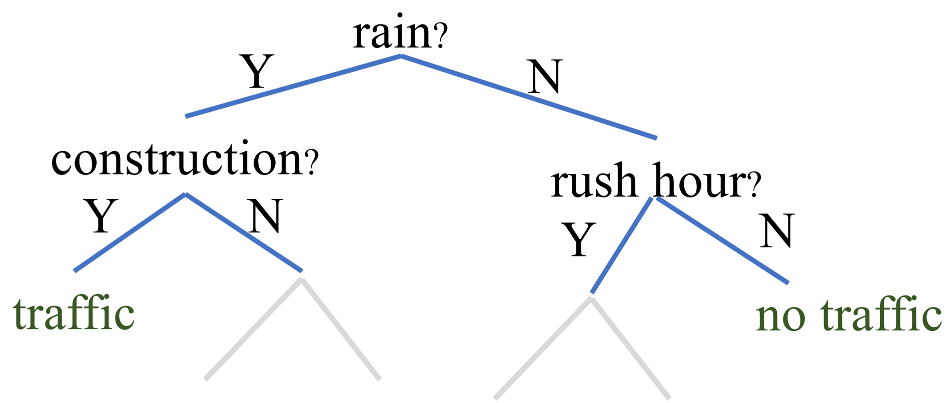
all its child trees are suboptimal.



Leaf Bound

Max # leaves of any optimal child tree

$$< \#\,\text{leaves(tree)} + \left\lfloor \frac{R_{\text{bestsofar}} - b(\text{tree})}{c} \right\rfloor.$$

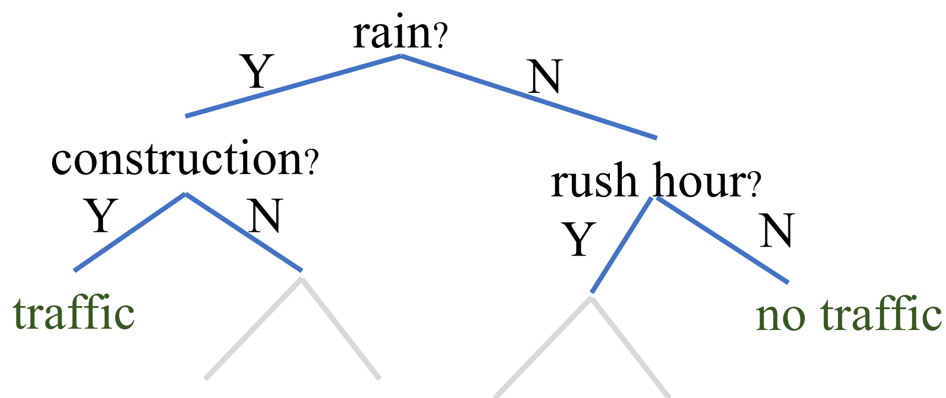


$$x_{16} = x_{73} = x_{83} = x_{71} = x_{23}$$

$$y_{16} = y_{73} = 1 \text{ and } y_{83} = y_{71} = y_{23} = -1$$

Equivalence Points Bound

Equivalent points with differing labels cannot all be classified correctly. The minority in each equivalence group must be misclassified.



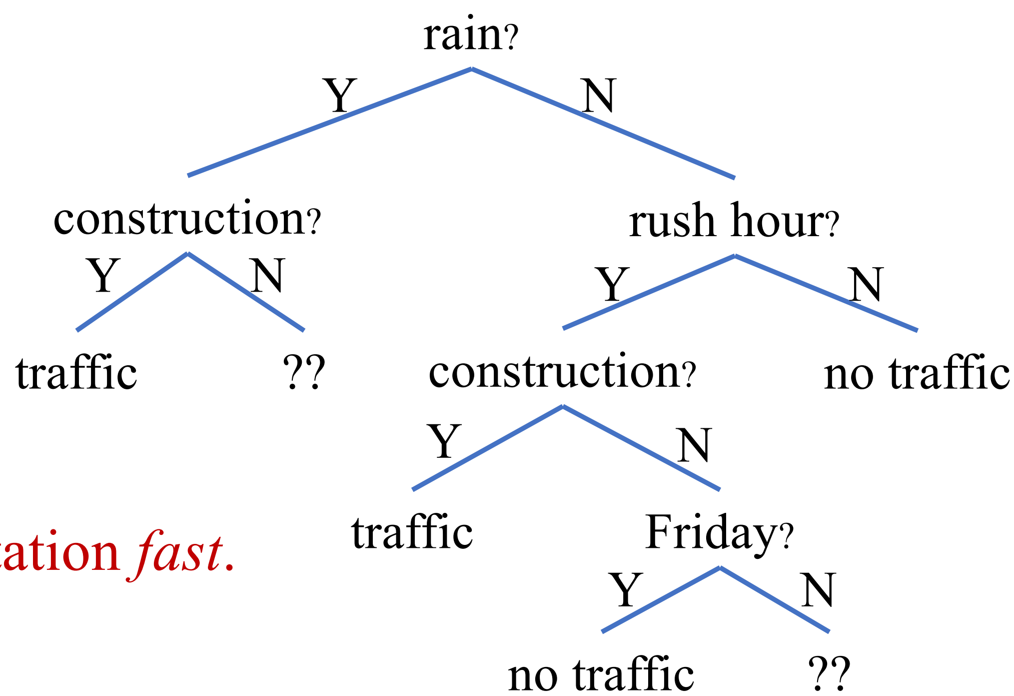
Incremental Progress Bound(s)

Each split must provide a reduction in loss of at least C.

Now for the computational speedups

Represent each subproblem by its contents.

rain & construction
[1000010001001110000..........................0]
rain & no construction
[0110001000000000110..........................1]
no rain & rush hour & construction
[0001000100000001000..........................0]
no rain & rush hour & no construction & Friday
[0000100000000000001..........................0]
no rain & rush hour & no construction & no Friday
[0000000010000000000..........................0]
no rain & no rush hour
[0000000000011000000..........................0]



Bitvector representation makes computation *fast*.

Permutation map: Discover identical trees already evaluated

rain & construction

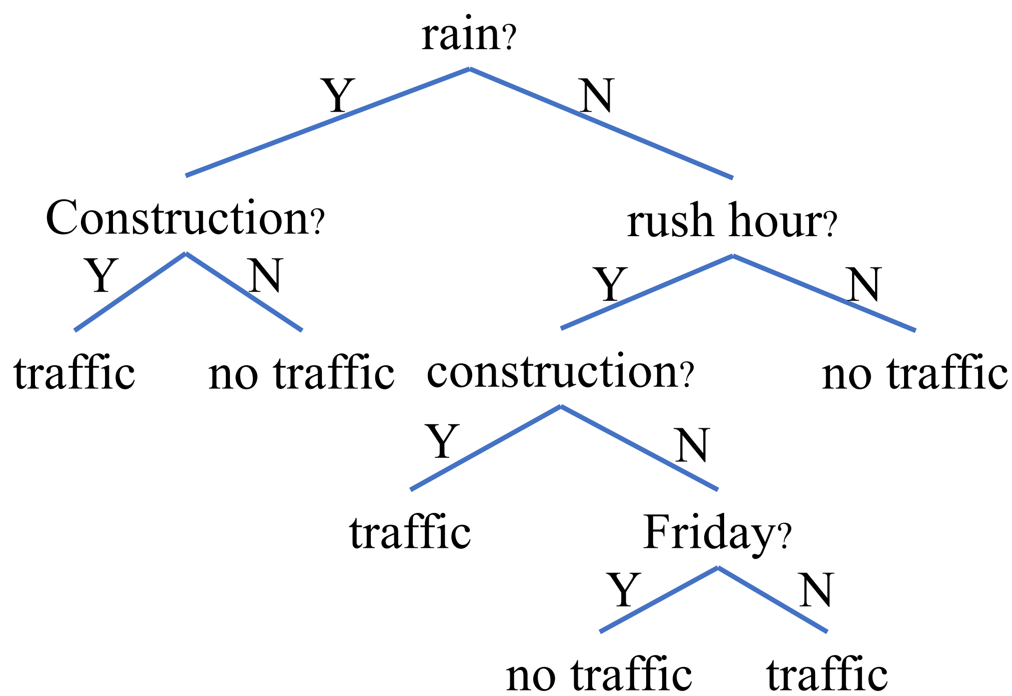
rain & no construction

no rain & rush hour & construction

no rain & rush hour & no construction & Friday

no rain & rush hour & no construction & no Friday

no rain & no rush hour



GOSDT - Generalized and Scalable Optimal Sparse Decision Trees
(Lin et al., ICML 2020)

Dynamic programming

Strong analytical bounds

Representation of each subproblem

Fast bit-vector computation = Speed

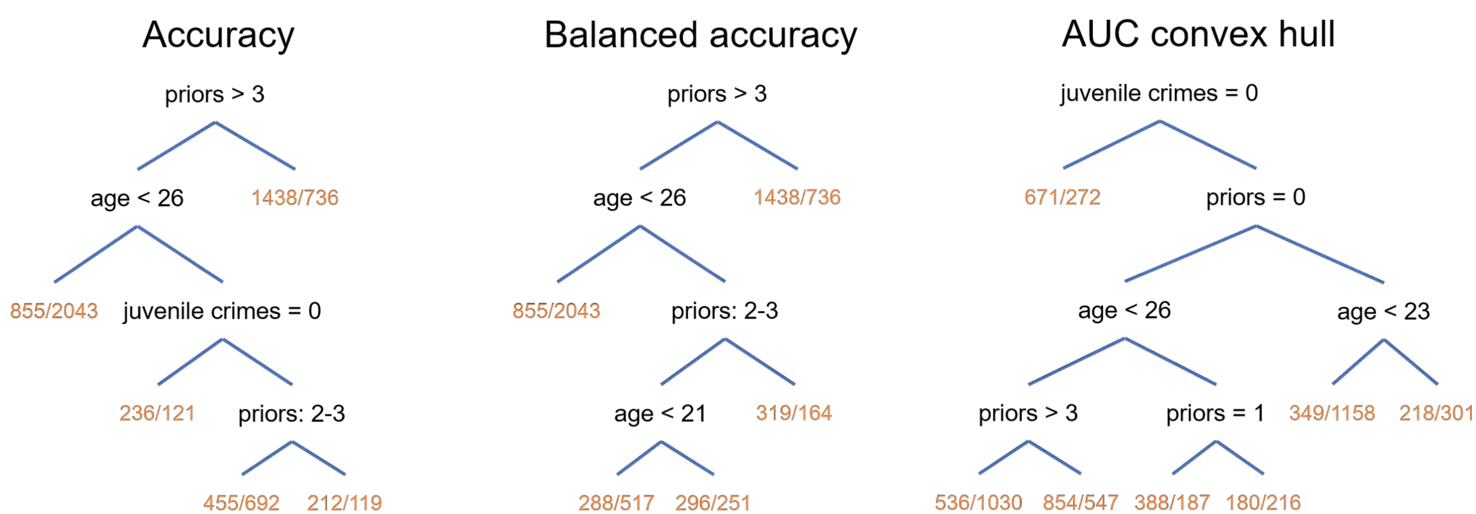
Consolidation of repeated subproblems

Permutation map

GOSDT - Generalized and Scalable Optimal Sparse Decision Trees (Lin et al., ICML 2020)

$$R(\text{tree, data}) = \text{loss(FP, FN)} + C\ (\#\ \text{leaves})$$

- Can optimize any loss function monotonically increasing in FP and FN (Balanced accuracy, weighted accuracy, F-1, precision, …)
- Can optimize rank statistics (AUC and partial area under the ROC convex hull)



GOSDT - Generalized and Scalable Optimal Sparse Decision Trees
(Lin et al., ICML 2020)

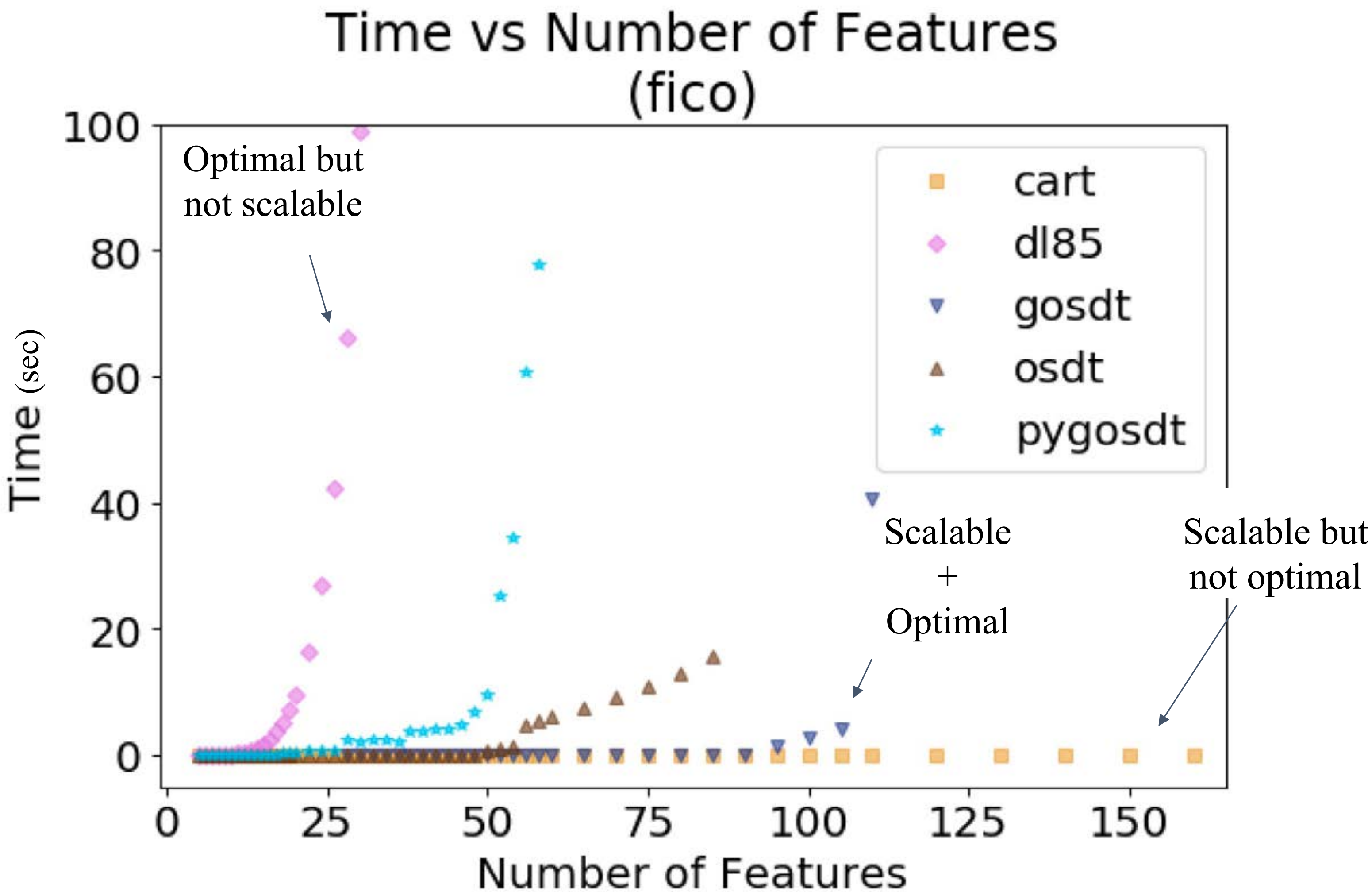
$$\text{R(tree, data)} = \text{loss(FP, FN)} + C \text{ (\# leaves)}$$

- Can optimize any loss function monotonically increasing in FP and FN (Balanced accuracy, weighted accuracy, F-1, precision, …)
- Can optimize rank statistics (AUC and partial area under the ROC convex hull)

Main experimental results:

- Similar classification error to black box methods.
- For custom losses, much better loss values than greedy decision trees.
- Sparser than all heuristic methods
- Orders of magnitude faster than the next best method.

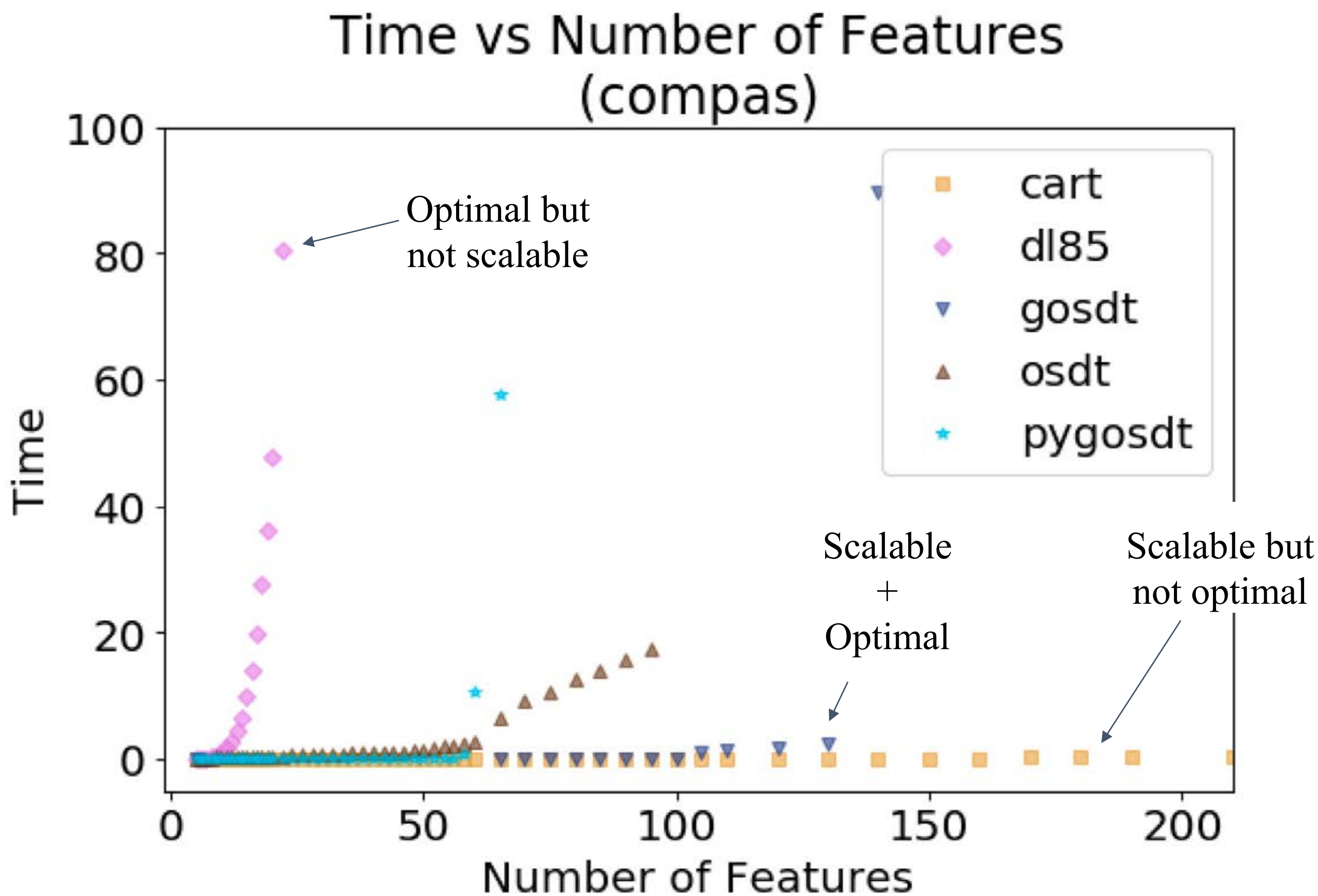
Scalability



Improvements in orders of magnitude

Note: BinOCT too slow to include.

Scalability

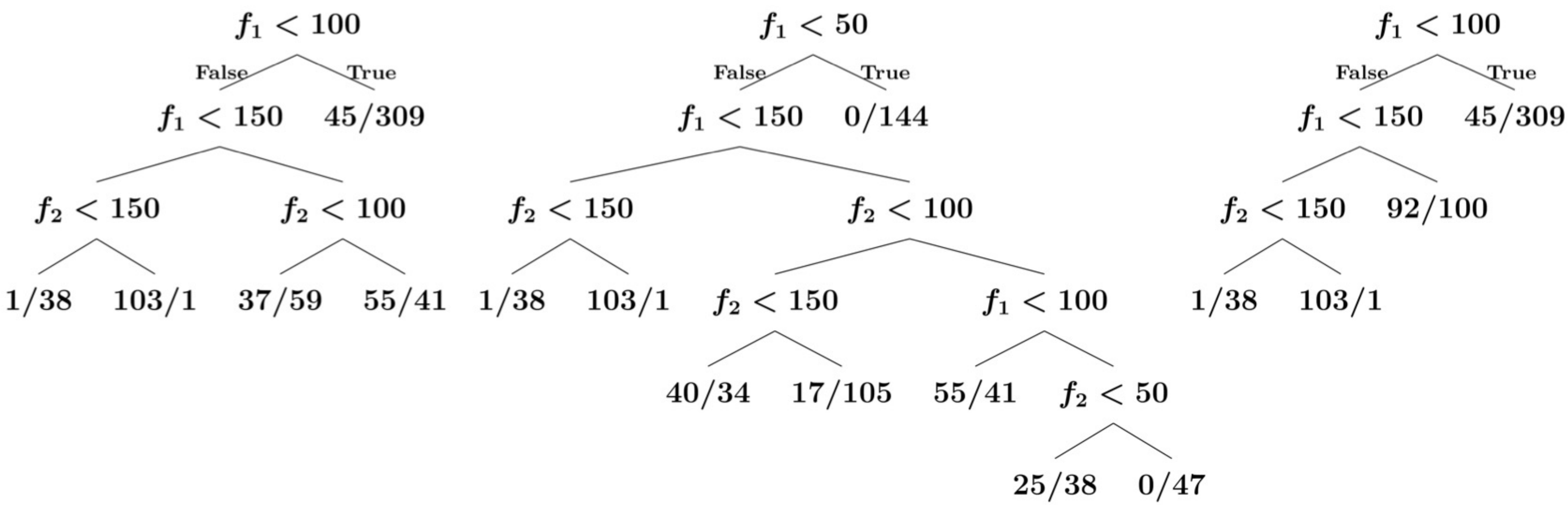


Improvements in orders of magnitude

Note: BinOCT too slow to include.

Flexibility to use different objectives

Some trees from FourClass



a: Accuracy b: AUC convex hull c: pAUC convex hull

Summary

Modern decision tree methods are not your old CART.

Jimmy Lin, Chudi Zhong, Diane Hu, Cynthia Rudin, Margo Seltzer
Generalized and Scalable Optimal Sparse Decision Trees. ICML, 2020.

Code: https://github.com/Jimmy-Lin/GeneralizedOptimalSparseDecisionTrees

CART

GOSDT

Thanks!