Trees and Rules and Optimization

M. Hakan Akyüz

Açıklanabilir Yapay Öğrenme - II

Outline

- Decision Trees (DTs)
  - CART
- Optimization Models
  - Optimal Classification Trees
  - State-of-the-art
- Rules and Rule Set Generation
  - Column Generation
  - Extensions
- Takeaways

Regression Trees

Distinct and nonoverlapping regions

$X_1, X_2, \ldots, X_p \longrightarrow R_1, R_2, \ldots, R_m$

Prediction: Mean of the response values for the training observations in R_m

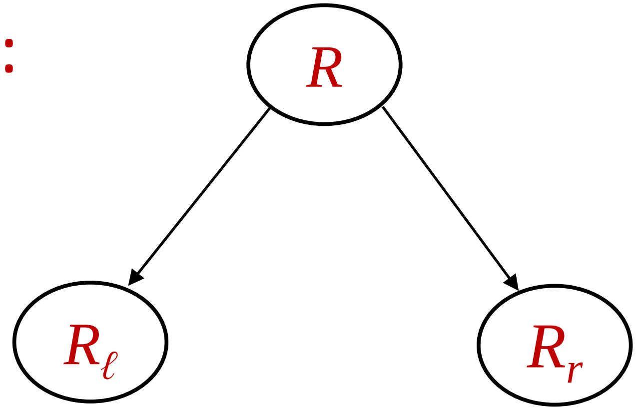
$$\hat{y}_m = \frac{1}{N_m} \sum_{\{x_i \in R_m\}} y_i$$

Goal: Finding the regions such that sum of squares is minimized

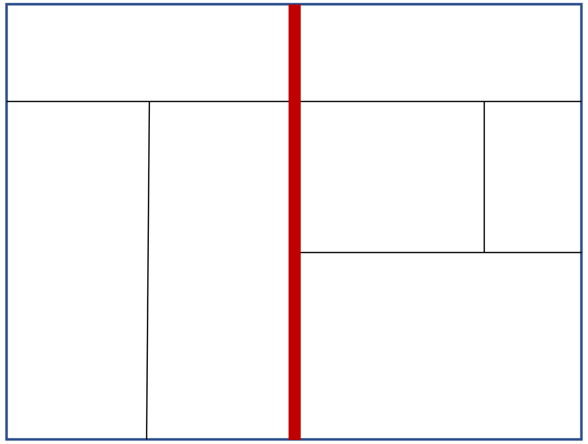
$$\sum_{j=1}^{m} \sum_{\{x_i \in R_j\}} (y_i - \hat{y}_j)^2$$

How to grow a Regression Tree?

Recursive Binary Splitting:



$$R_\ell(j,s) = \{X \mid X_j \le s\} \qquad R_r(j,s) = \{X \mid X_j > s\}$$



Find *j* and *s* minimizing the sum of squared error

$$\sum_{\{x_i \in R_\ell(j,s)\}} (y_i - \hat{y}_\ell)^2 + \sum_{\{x_i \in R_r(j,s)\}} (y_i - \hat{y}_r)^2$$

Until when?

Goal: Avoiding overfitting with a fully grown or large tree
(Selecting a subtree that leads to a lowest test error rate)

CART:
1) Grow a large tree until each leaf has a minimum sample size, e.g. 5.
2) Pruning large tree based on cost-complexity, $C_\alpha(T)$.

$$C_\alpha(T) = \sum_{m=1}^{|T|} \sum_{\{x_i \in R_m\}} (y_i - \hat{y}_m)^2 + \alpha|T|$$

Pruning: Successively collapse the internal node that produces the smallest per-node increase in sum of squared errors.
- Use k-fold cross validation to tune α.

Classification Trees

Similar to a regression tree but uses different error measures based on *impurity* of a region.

Prediction: Majority voting of classes among the training observations in R_m

$$\hat{p}_{mk} = \frac{1}{N_m} \times \sum_{x_i \in R_m} I(y_i = k)$$

$$\hat{y}_m = \arg\max_k \hat{p}_{mk}$$

Error Measures

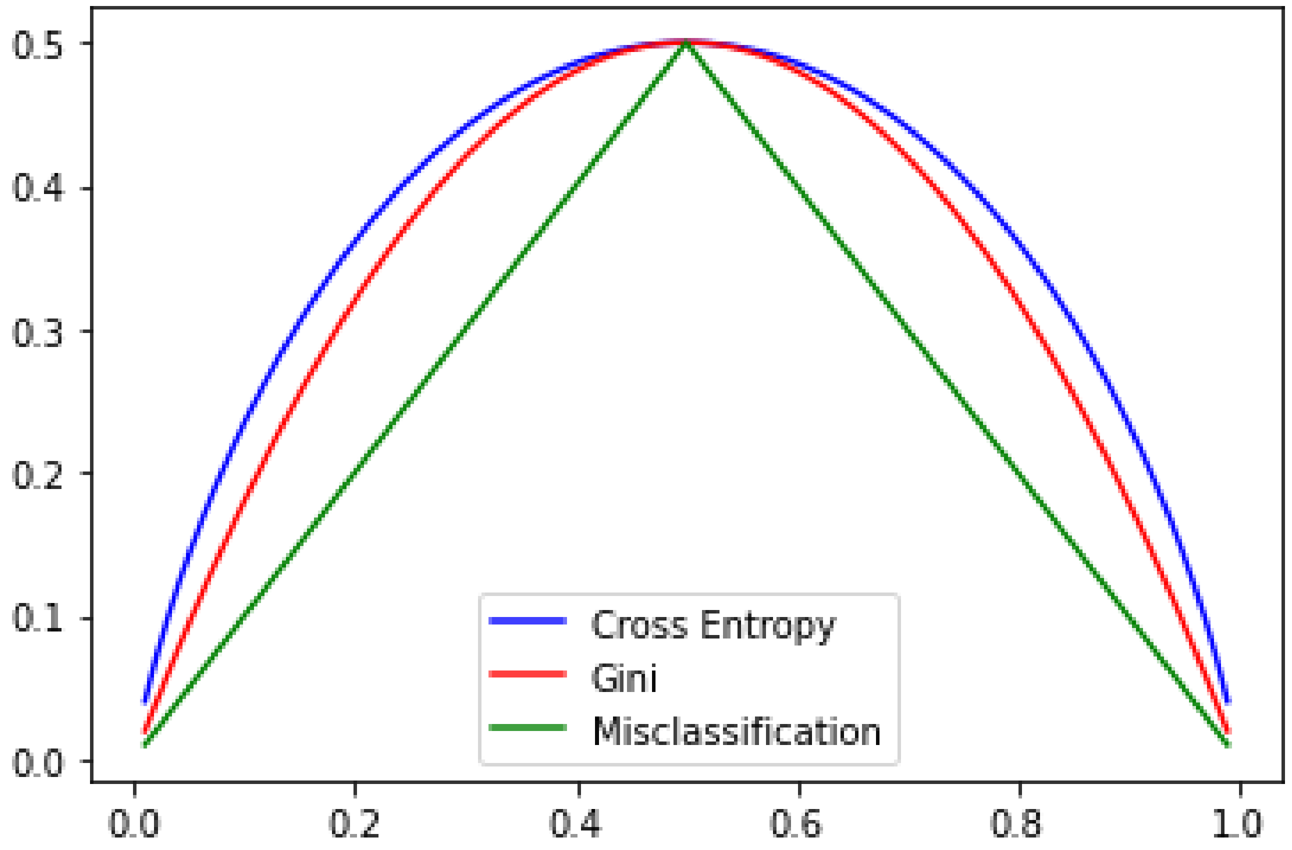
$$\frac{1}{N_m} \times \sum_{x_i \in R_m} I(y_i \neq \hat{y}_m) = 1 - \hat{p}_{m\hat{y}_m} \longrightarrow \text{Misclassification Error}$$

$$\sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk}) \longrightarrow \text{Gini Index}$$

$$-\sum_{k=1}^{K} \hat{p}_{mk} \log \hat{p}_{mk} \longrightarrow \text{Cross Entropy}$$

Which error measure?

- Use Gini or Cross Entropy to grow the tree.
- For pruning use misclassification error. Exercise: Why not the other way?
- Weight the node impurity measures by the number $N_{m\ell}$ and N_{mr} of observations in the two child nodes created by splitting node m.



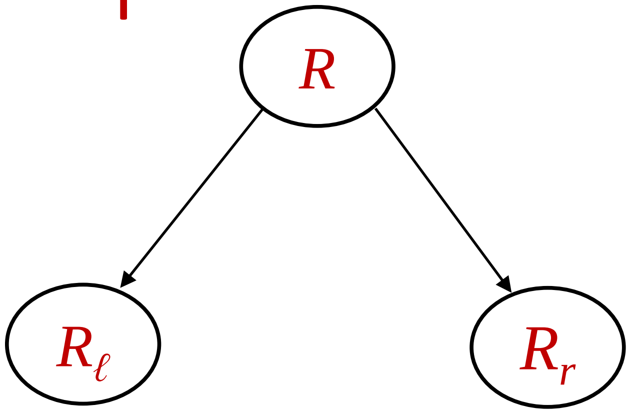
* Two class case. Cross Entropy is scaled to pass through (0.5,0.5).

$$\frac{1}{N_m} \times \sum_{x_i \in R_m} I(y_i \neq \hat{y}_m) = 1 - \hat{p}_{m\hat{y}_m}$$ → Misclassification Error

$$\sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk})$$ → Gini Index

$$-\sum_{k=1}^{K} \hat{p}_{mk} \log \hat{p}_{mk}$$ → Cross Entropy

How to select a split?



$$R_\ell(j,s) = \{X | X_j \leq s\} \quad R_r(j,s) = \{X | X_j > s\}$$

Find *j* and *s* minimizing the error (or maximizing the gain of split)

$$\sum_{\{x_i \in R_\ell(j,s)\}} (y_i - \hat{y}_\ell)^2 + \sum_{\{x_i \in R_r(j,s)\}} (y_i - \hat{y}_r)^2$$

Classification Trees

$$\hat{p}_{mk} = \frac{1}{N_m} \times \sum_{x_i \in R_m} I(y_i = k)$$

$$\hat{y}_m = \arg\max_k \hat{p}_{mk}$$

Gini Index:

$$\sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk}) \quad \text{<=>} \quad \boxed{1 - \sum_{k=1}^{K} \hat{p}_{mk}^2}$$

What is the gain of a split "s"?

Gain = (Error before the split) – (Error after the split)

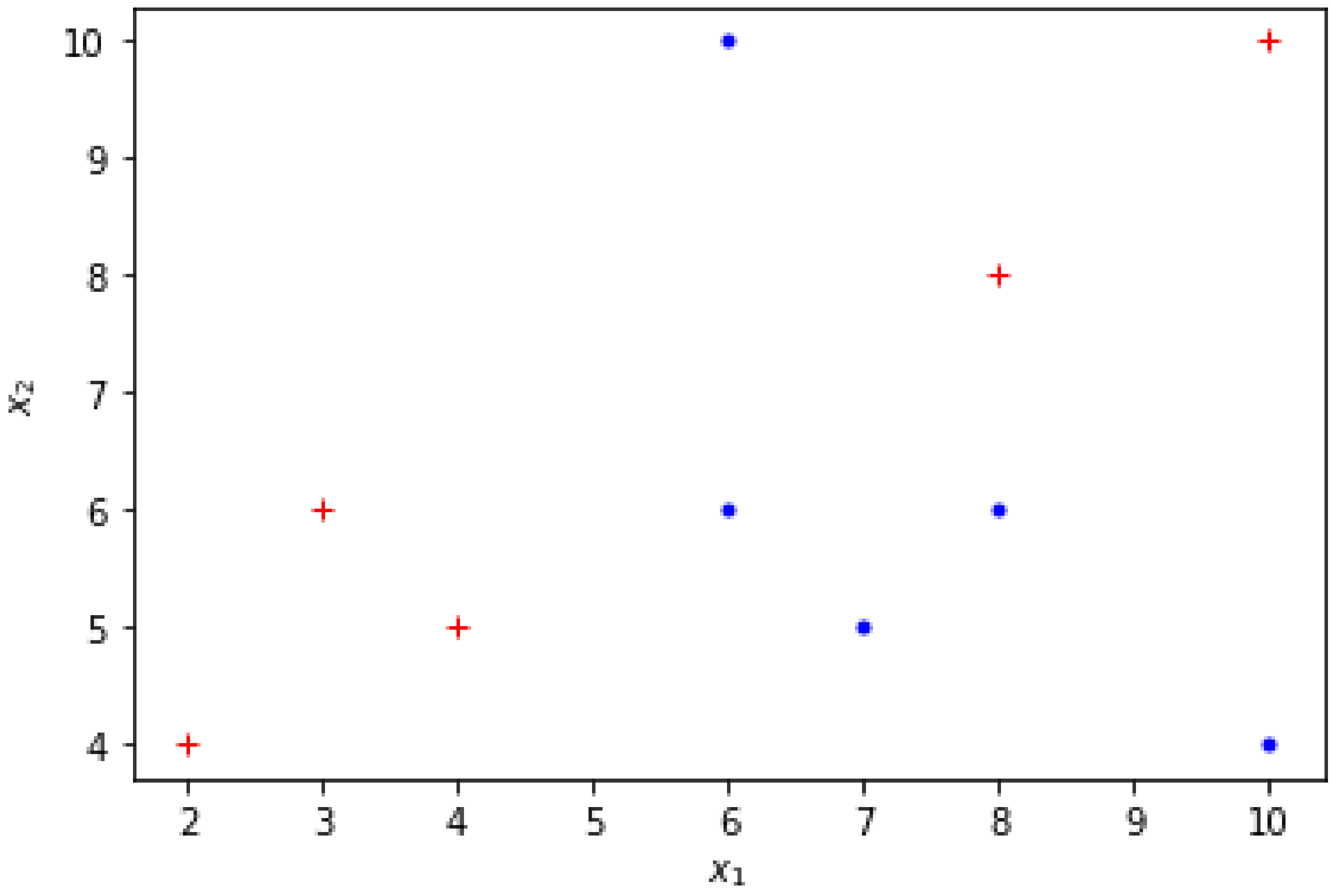
$$G(s) = \boxed{E(R)} - \boxed{E(s)}$$

$$\boxed{E(s) = \frac{N_{ml}}{N_m} E(R_l) + \frac{N_{mr}}{N_m} E(R_r)}$$

Error at node R

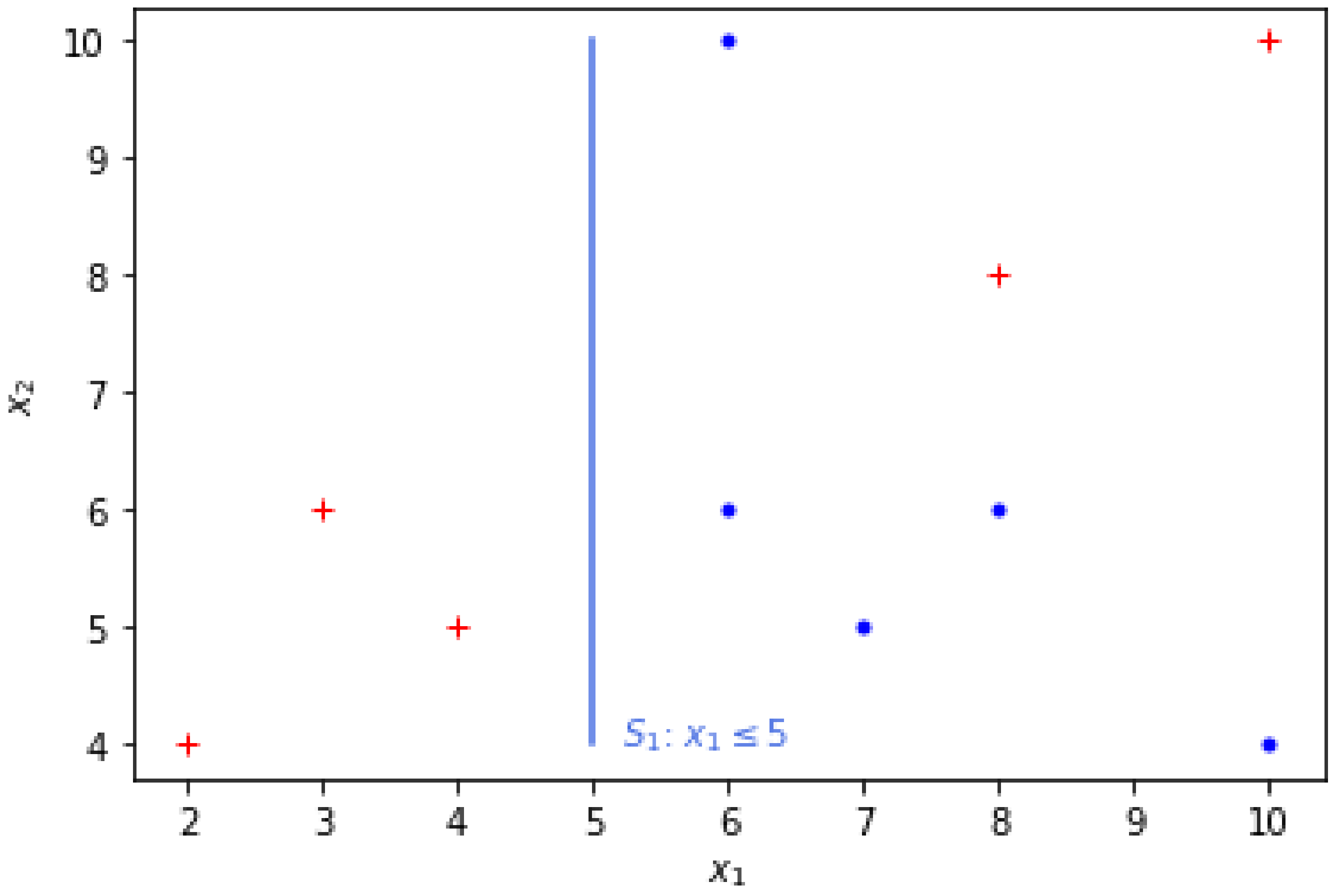
Error of split s

Classification – { +, • }



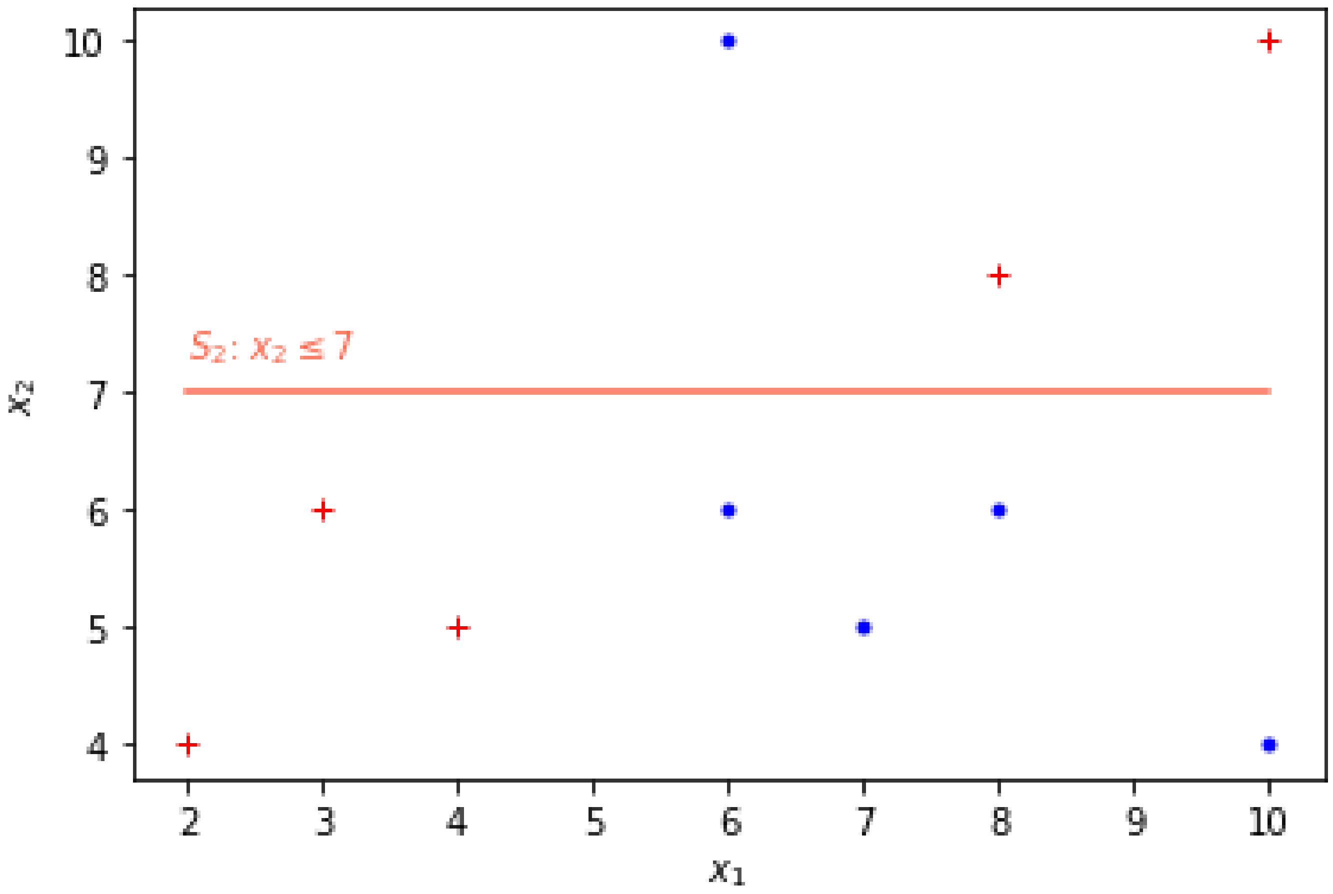
Classification – { +, • }

Split #1: $x_1 \leq 5$



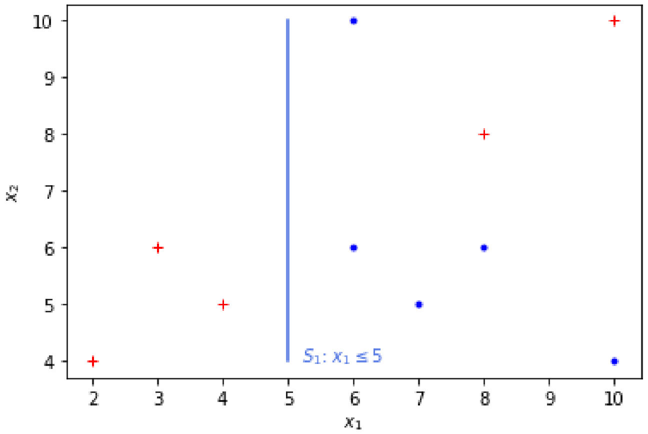
Classification – { +, • }

Split #2: $x_2 \leq 7$



Classification – { +, • }

Gini Index: $1 - \sum_{k=1}^{K} \hat{p}_{mk}^2$



Split #1: $x_1 \leq 5$

S_1

$E(R) = 1\text{-}(0.5)^2\text{-}(0.5)^2 = 0.5$

+++++ ••••• R

$R_\ell(x_1, 5) = \{x | x_1 \leq 5\}$

$R_r(x_1, 5) = \{x | x_1 > 5\}$

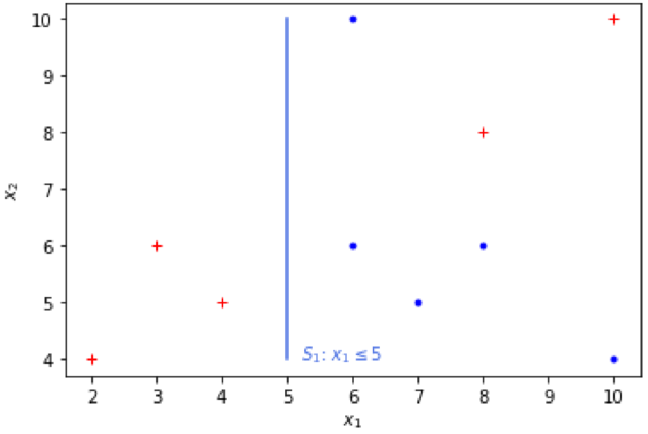
+++ R_ℓ

R_r ••••• ++

Classification – { +, • }

Gini Index: $1 - \sum_{k=1}^{K} \hat{p}_{mk}^2$

Split #1: $x_1 \leq 5$



$$\boxed{E(R) = 1\text{-}(0.5)^2\text{-}(0.5)^2 = 0.5}$$

+++++
•••••
R

$R_\ell(x_1, 5) = \{x | x_1 \leq 5\}$

$R_r(x_1, 5) = \{x | x_1 > 5\}$

+++ R_ℓ

R_r •••••
++

$$E(R_\ell) = 1\text{-}(1.0)^2\text{-}(0.0)^2 = 0.0$$

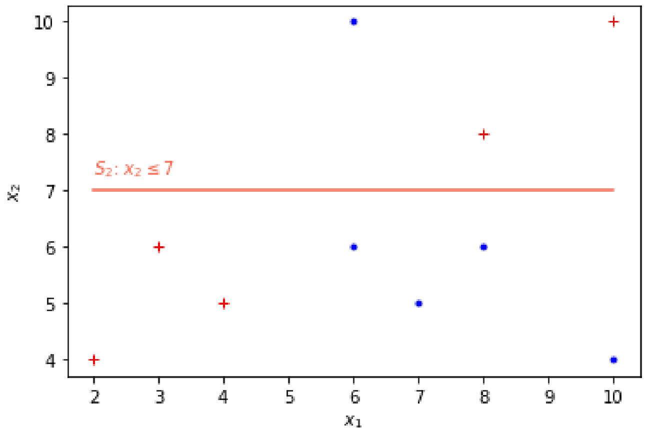
$$E(R_r) = 1\text{-}(2/7)^2\text{-}(5/7)^2 = 0.41$$

$$\begin{aligned} E(S_1) &= (3/10)\cdot E(R_\ell) + (7/10)\cdot E(R_r) \\ &= 0.3 \cdot 0.0 + 0.7 \cdot 0.41 \\ &= 0.29 \end{aligned}$$

$$\boxed{\begin{aligned} G(S_1) &= E(R) - E(S_1) \\ &= 0.5 - 0.29 \\ &= 0.21 \end{aligned}}$$

Classification – { +, • }

Gini Index: $1 - \sum_{k=1}^{K} \hat{p}_{mk}^2$



Split #2: $x_2 \le 7$

S_2

$E(R) = 1-(0.5)^2-(0.5)^2 = 0.5$

+++++
•••••

R

$R_\ell(x_2, 7) = \{x | x_2 \le 7\}$

$R_r(x_2, 7) = \{x | x_2 > 7\}$

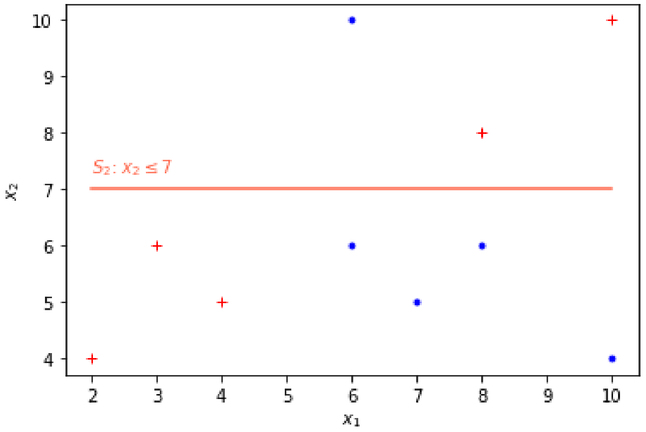
••••
+++ R_ℓ

R_r •
++

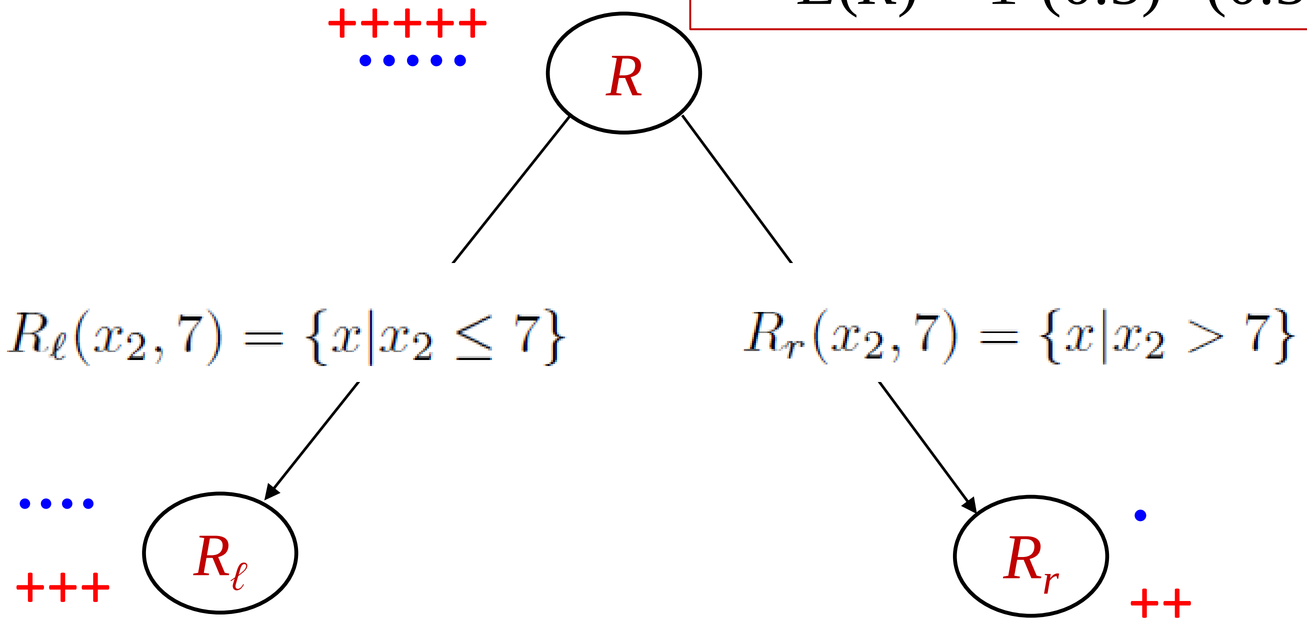
Classification – { +, • }

Gini Index: $1 - \sum_{k=1}^{K} \hat{p}_{mk}^2$

Split #2: $x_2 \leq 7$



$$E(R) = 1-(0.5)^2-(0.5)^2 = 0.5$$



$$E(R_\ell) = 1-(4/7)^2-(3/7)^2 = 0.49$$

$$E(R_r) = 1-(1/3)^2-(2/3)^2 = 0.44$$

$$\begin{aligned} E(S_2) &= (7/10) \cdot E(R_\ell) + (3/10) \cdot E(R_r) \\ &= 0.7 \cdot 0.49 + 0.3 \cdot 0.44 \\ &= 0.343 + 0.132 \\ &= 0.48 \end{aligned}$$

$$\begin{aligned} G(S_2) &= E(R) - E(S_2) \\ &= 0.5 - 0.48 \\ &= 0.02 \end{aligned}$$

Optimization Models?

Generic Model

$$\begin{array}{ll} \text{minimize} & f(\boldsymbol{x}) \\ \text{subject to} & g(\boldsymbol{x}) \geq \mathbf{0}. \end{array}$$

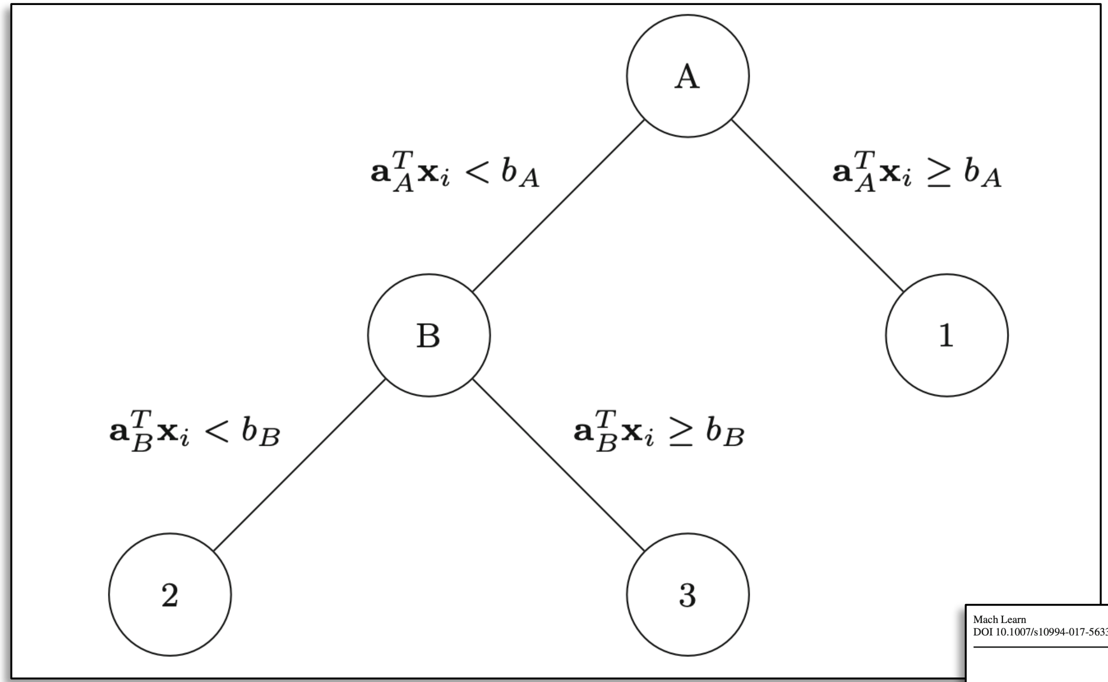
Integer Programming Model

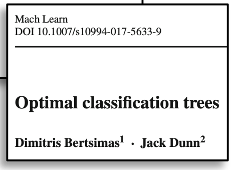
$$\begin{array}{ll} \text{minimize} & \boldsymbol{c}^\intercal \boldsymbol{x} \\ \text{subject to} & \boldsymbol{A}\boldsymbol{x} \geq \boldsymbol{b}, \\ & \boldsymbol{x} \in \{\mathbf{0}, \mathbf{1}\}. \end{array}$$

Linear Programming Model

$$\begin{array}{ll} \text{minimize} & \boldsymbol{c}^\intercal \boldsymbol{x} \\ \text{subject to} & \boldsymbol{A}\boldsymbol{x} \geq \boldsymbol{b}, \\ & \boldsymbol{x} \geq \mathbf{0}. \end{array}$$

Optimal Classification Trees (OCT)





Mach Learn
DOI 10.1007/s10994-017-5633-9

Optimal classification trees

Dimitris Bertsimas[1] · Jack Dunn[2]

$$\boldsymbol{x}_i \in \mathbb{R}^p, y_i \in \{1, ..., K\},\ i = 1, \ldots, n$$

classification error

number of branch nodes in T

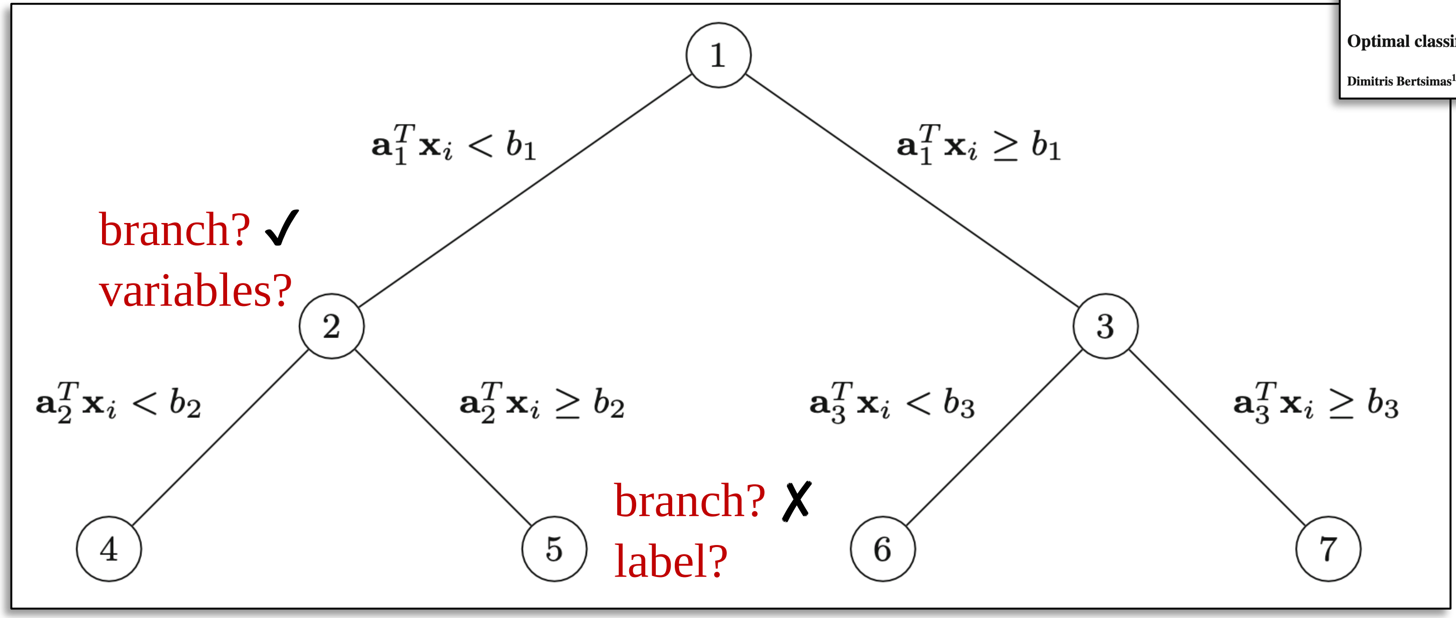
tree

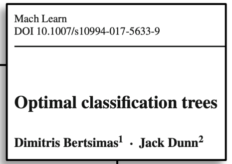
$$\begin{array}{ll} \text{minimize} & R_{xy}(T) + \alpha|T| \\ \text{subject to} & N_x(l) \geq N_{\min}, \quad l \in \text{leaves of } T \end{array}$$

number points in leaf l

minimum number of samples in any leaf

OCT - Setup



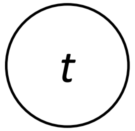

Mach Learn
DOI 10.1007/s10994-017-5633-9

Optimal classification trees

Dimitris Bertsimas[1] · Jack Dunn[2]

$\mathcal{T}_B$: branch nodes

$\mathcal{T}_l$: leaf nodes



$p(t)$: parent node of t

$A(t)$: set of ancestors of node t

$A_L(t)$: "left" ancestors of node t

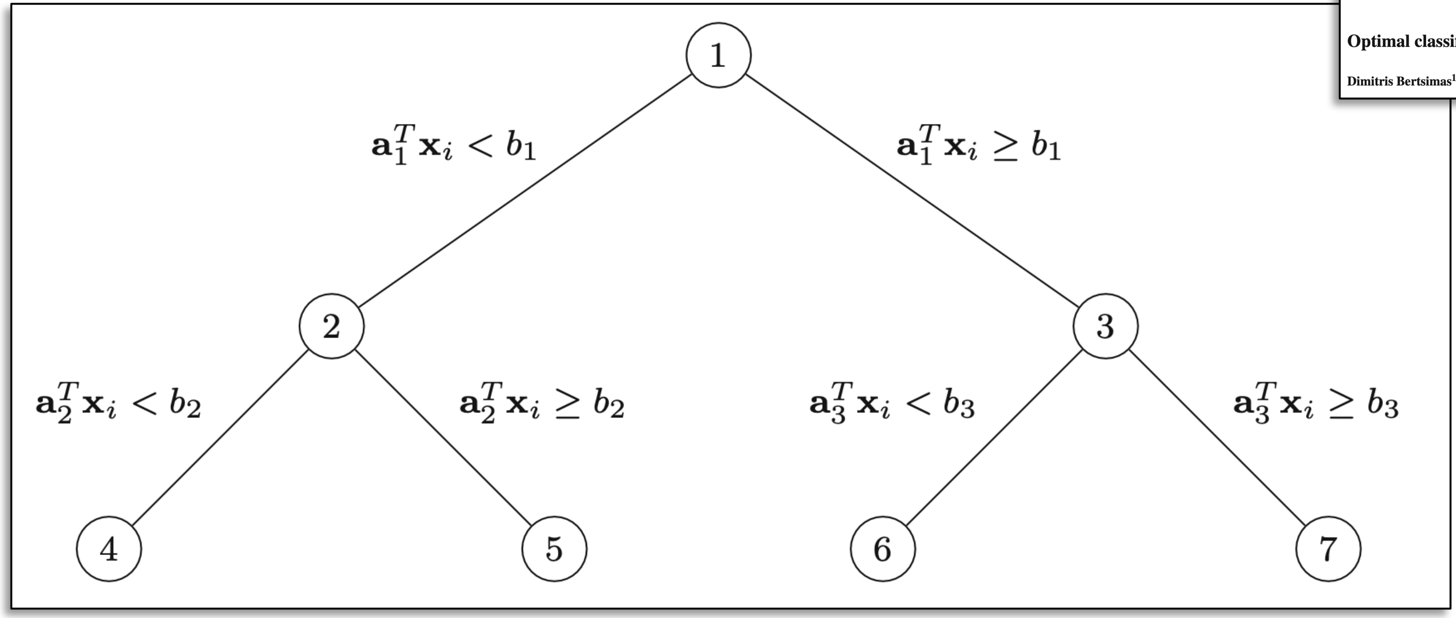
$A_R(t)$: "right" ancestors of node t

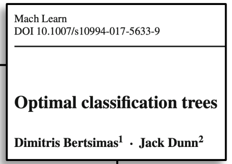
$$A_L(5) = \{1\},\ A_R(5) = \{2\}$$

$$A(t) = A_L(t) \cup A_R(t)$$

OCT – Splitting Constraints

Univariate Trees





Mach Learn
DOI 10.1007/s10994-017-5633-9

Optimal classification trees

Dimitris Bertsimas[1] · Jack Dunn[2]

$\boldsymbol{a}_t$: unit vector

$d_t \in \{0,1\}$: split or not

$$\sum_{j=1}^{p} a_{jt} = d_t, \quad t \in \mathcal{T}_B$$

$$0 \leq b_t \leq d_t, \quad t \in \mathcal{T}_B$$

$$a_{jt} \in \{0,1\}, \quad j = 1,\ldots,p;\ t \in \mathcal{T}_B$$

$$d_t \leq d_{p(t)},\ t \in \mathcal{T}_B \backslash \{1\}$$

OCT – Sample Tracking Constraints

Univariate Trees

z_{it} : $\boldsymbol{x}_i$ is in node t or not

l_t : leaf t contains points or not

$$
\begin{aligned}
& z_{it} \leq l_t, && t \in \mathcal{T}_L \\
& \textstyle\sum_{i=1}^{n} z_{it} \geq N_{\min} l_t, && t \in \mathcal{T}_L \\
& \textstyle\sum_{t \in \mathcal{T}_L} z_{it} = 1, && i = 1, \ldots, n
\end{aligned}
$$

$$
\begin{aligned}
& \boldsymbol{a}_m^\intercal \boldsymbol{x}_i + \varepsilon \leq b_m + M_1(1 - z_{it}) && i = 1, \ldots, n; t \in \mathcal{T}_L \ ; \ m \in A_L(t) \\
& \boldsymbol{a}_m^\intercal \boldsymbol{x}_i \geq b_m - M_2(1 - z_{it}) && i = 1, \ldots, n; t \in \mathcal{T}_L \ ; \ m \in A_R(t)
\end{aligned}
$$

OCT – Objective Function

Univariate Trees

$$Y_{ik} = \begin{cases} +1, & \text{if } y_i = k; \\ -1, & \text{otherwise} \end{cases}$$

$$N_{kt} = \frac{1}{2}\sum_{i=1}^{n}(1+Y_{ik})z_{it}, \;\; k=1,\dots,K; \;\; t \in \mathcal{T}_L$$

$$c_t = \arg\max_{k=1,\dots,K}\{N_{kt}\} \quad \text{and} \quad c_{kt} = \begin{cases} 1, & \text{if } c_t = k; \\ 0, & \text{otherwise} \end{cases}$$

$$N_t = \sum_{i=1}^{n} z_{it}, \;\; t \in \mathcal{T}_L$$

$$L_t = N_t - \max_{1,\dots,K}\{N_{kt}\} = \min_{k=1,\dots,K}\{N_t - N_{kt}\}$$

$$\begin{aligned} L_t &\geq N_t - N_{kt} - M(1 - c_{kt}), && k = 1,\dots,K; t \in \mathcal{T}_L, \\ L_t &\leq N_t - N_{kt} + M c_{kt}, && k = 1,\dots,K; t \in \mathcal{T}_L, \\ L_t &\geq 0, && t \in \mathcal{T}_L \end{aligned}$$

$$(M = n)$$

OCT – Overall Model

Univariate Trees

$$\begin{aligned}
\text{minimize} \quad & \frac{1}{\hat{L}} \sum_{t \in \mathcal{T}_L} L_t + \alpha \sum_{t \in \mathcal{T}_B} d_t \\
\text{subject to} \quad & L_t \geq N_t - N_{kt} - M(1 - c_{kt}), \quad k = 1, \dots, K; t \in \mathcal{T}_L, \\
& L_t \leq N_t - N_{kt} + M c_{kt}, \quad k = 1, \dots, K; t \in \mathcal{T}_L, \\
& L_t \geq 0, \quad t \in \mathcal{T}_L \\
& N_{kt} = \frac{1}{2} \sum_{i=1}^{n} (1 + Y_{ik}) z_{it}, \ k = 1, \dots, K; \ t \in \mathcal{T}_L \\
& N_t = \sum_{i=1}^{n} z_{it}, \ t \in \mathcal{T}_L \\
& \boldsymbol{a}_m^\intercal \boldsymbol{x}_i + \varepsilon \leq b_m + M_1(1 - z_{it}) \quad i = 1, \dots, n; \ t \in \mathcal{T}_B; \ m \in A_L(t) \\
& \boldsymbol{a}_m^\intercal \boldsymbol{x}_i \geq b_m - M_2(1 - z_{it}) \quad i = 1, \dots, n; \ t \in \mathcal{T}_B; \ m \in A_R(t) \\
& z_{it} \leq l_t, \quad t \in \mathcal{T}_L \\
& \textstyle\sum_{i=1}^{n} z_{it} \geq N_{\min} l_t, \quad t \in \mathcal{T}_L \\
& \textstyle\sum_{i=1}^{n} z_{it} = 1, \quad i = 1, \dots, n \\
& \textstyle\sum_{j=1}^{p} a_{jt} = d_t, \quad t \in \mathcal{T}_B \\
& 0 \leq b_t \leq d_t, \quad t \in \mathcal{T}_B \\
& d_t \leq d_{p(t)}, \ t \in \mathcal{T}_B \backslash \{1\} \\
& z_{it}, l_t \in \{0, 1\}, i = 1, \dots, n; t \in \mathcal{T}_L, \\
& a_{jt}, d_t \in \{0, 1\}, i = 1, \dots, n; t \in \mathcal{T}_B
\end{aligned}$$

$$\begin{aligned}
\text{minimize} \quad & R_{xy}(T) + \alpha |T| \\
\text{subject to} \quad & N_x(l) \geq N_{\min}, \quad l \in \text{leaves of } T
\end{aligned}$$

Number of Binary Variables

$$\mathcal{O}(n 2^{D})$$

tree depth

OCT – Take Away Messages

- A large-scale mixed integer optimization model
- Relies heavily on the speed of state-of-the-art solvers (*e.g.*, Gurobi)
- Can be easily extended to splits with hyperplanes (multivariate trees)
- Empirically shown that overfitting does not occur
- The solution time increases drastically with the number of samples and the tree depth
- Random Forest still performs better when it comes to accuracy

Mach Learn
DOI 10.1007/s10994-017-5633-9

Optimal classification trees

Dimitris Bertsimas[1] · Jack Dunn[2]

State-of-the-art

Mach Learn
DOI 10.1007/s10994-017-5633-9

Optimal classification trees

Dimitris Bertsimas[1] · Jack Dunn[2]

n = 5456 ; p = 2 ; K = 4 ; (2017)

Strong Optimal Classification Trees

Sina Aghaei
Center for Artificial Intelligence in Society, University of Southern California, Los Angeles, CA 90089, USA, saghaei@usc.edu
Andrés Gómez
Department of Industrial and Systems Engineering, Los Angeles, CA 90089, USA, gomezand@usc.edu
Phebe Vayanos
Center for Artificial Intelligence in Society, University of Southern California, Los Angeles, CA 90089, USA, phebe.vayanos@usc.edu

n = 3196 ; p = 38 ; K = 2 ; (2021)

The Thirty-Third AAAI Conference on Artificial Intelligence (AAAI-19)

Learning Optimal Classification Trees Using a Binary Linear Program Formulation

Sicco Verwer   Yingqian Zhang

n = 4601 ; p = 57 ; K = 2 ; (2019)

arXiv version 1 (2020) 1-48   Submitted 4/00; Published 10/00

MurTree: Optimal Classification Trees via Dynamic Programming and Search

Emir Demirović   E.DEMIROVIC@TUDELFT.NL
Delft University of Technology
Delft, The Netherlands
Anna Lukina   ANNA.LUKINA@IST.AC.AT
Institute of Science and Technology Austria
Klosterneuburg, Austria
Emmanuel Hebrard   HEBRARD@LAAS.FR
LAAS CNRS
Toulouse, France
Jeffrey Chan   JEFFREY.CHAN@RMIT.EDU.AU
RMIT University
Melbourne, Australia

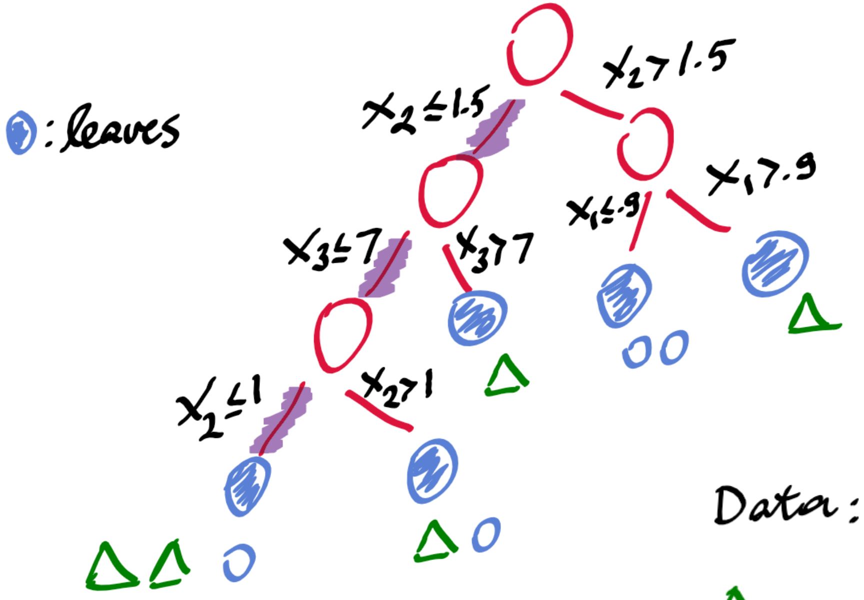
n = 43500 ; p = 181 ; K = 7 ; (2020)

Rules and Rule Set Generation?

RULE-BASED METHODS USING OPTIMIZATION

IF AGE > 25 AND PASTCREDIT = TRUE THEN ACCEPT

DECISION TREES

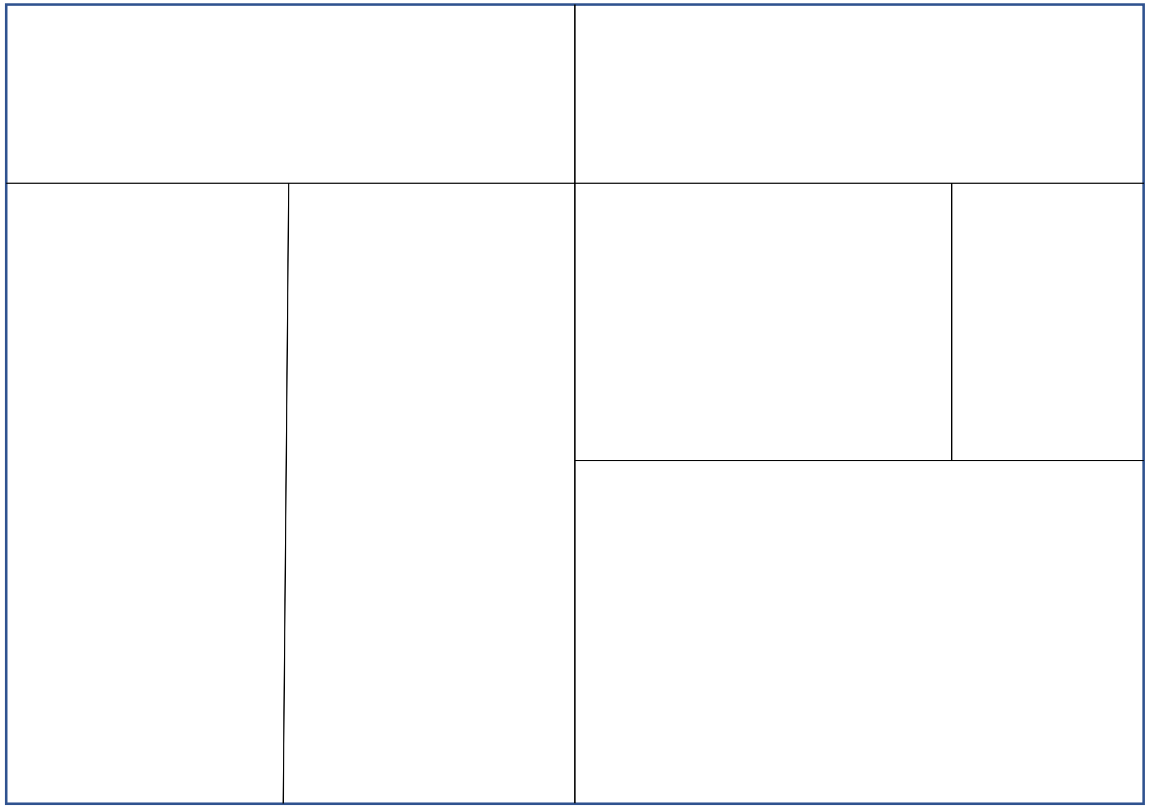


Rule j (leaf) : $x_2 \le 1$ & $x_3 \le 7$

Data: {△, △, ○, △, ○, ○}

△ ← labels
○ ↙

DECISION TREES vs RULES

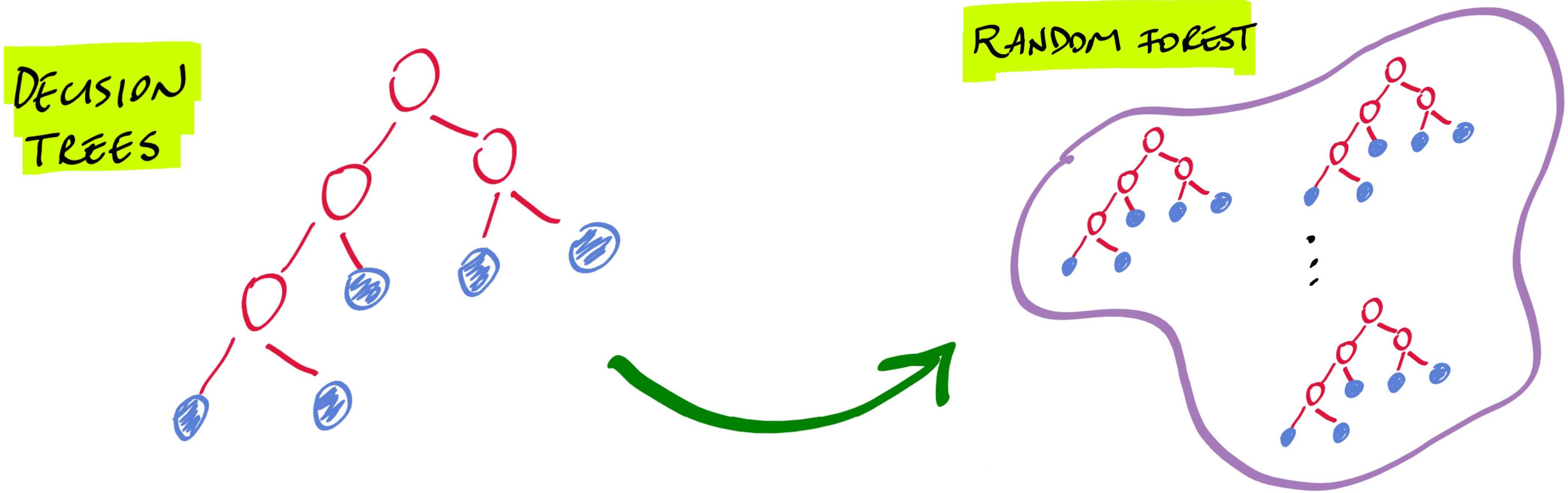


Rules are simple and easy to understand. In some cases they are easier to understand than the decision trees.

What are possible measures for interpretability?

- Set Size (number of rules)
- Rule length (number of clauses)
- Cover (number of samples covered)
- Overlap (between two rules)

Rule Generation



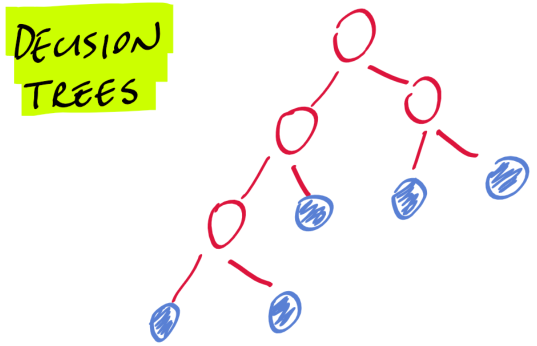
Variance ↓ Accuracy ↑

Interpretability ↓

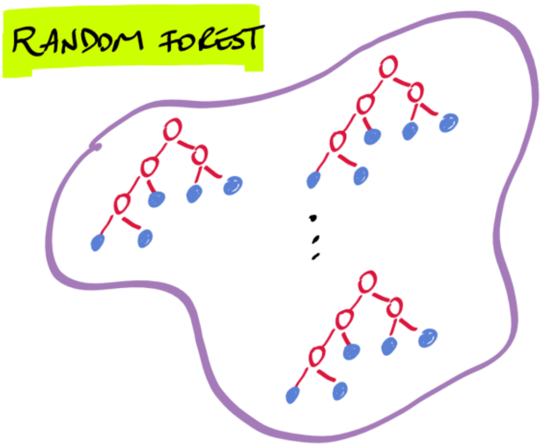
Rule Generation for Classification: Scalability, Interpretability, and Fairness

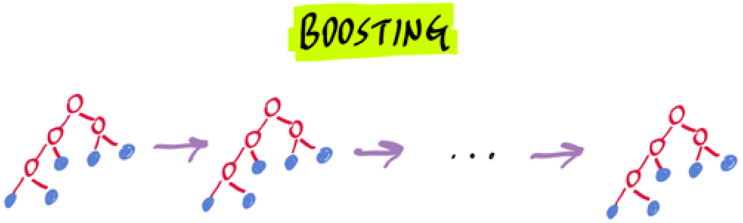
Adia Lumadjeng[*], Tabea Röber[**], M. Hakan Akyüz[*], Ş. İlker Birbil[**]
[*]Erasmus University Rotterdam, 3000 DR, Rotterdam, The Netherlands
[**]University of Amsterdam, 11018 TV, Amsterdam P.O. Box 15953, The Netherlands

Scalability



Rules
Rules
Rules
…





Optimization
with
Constraints

$$\cancel{\begin{array}{ll} \text{minimize} & f(\boldsymbol{x}) \\ \text{subject to} & g(\boldsymbol{x}) \geq \mathbf{0}. \end{array}}$$

$$\begin{array}{ll} \text{minimize} & \boldsymbol{c}^\top \boldsymbol{x} \\ \text{subject to} & \boldsymbol{A}\boldsymbol{x} \geq \boldsymbol{b}, \\ & \cancel{\boldsymbol{x} \in \{\mathbf{0}, \mathbf{1}\}}. \end{array}$$

$$\boxed{\begin{array}{ll} \text{minimize} & \boldsymbol{c}^\top \boldsymbol{x} \\ \text{subject to} & \boldsymbol{A}\boldsymbol{x} \geq \boldsymbol{b}, \\ & \boldsymbol{x} \geq \mathbf{0}. \end{array}}$$

LP!

Scalable Rule Generation for Learning

Modeling

$\mathcal{D} = \{(\boldsymbol{x}_i, y_i) : i \in \mathcal{I}\}$

$\boldsymbol{x}_i \in \mathbb{R}^p, y_i \in \{1, \ldots, K\}$

$$y_i = k \implies \boldsymbol{y}_i = (-\tfrac{1}{K-1}, -\tfrac{1}{K-1}, \ldots, \underset{k^{\text{th}}}{1}, \ldots, -\tfrac{1}{K-1})^\intercal \in \mathbb{R}^K$$

prediction of rule *j* for sample *i* → $\boldsymbol{R}_j(\boldsymbol{x}_i) \in \mathbb{R}^K$

prediction for sample *i* → $\hat{\boldsymbol{y}}_i(\boldsymbol{w}) = \sum_{j \in J} a_{ij} \boldsymbol{R}_j(\boldsymbol{x}_i) w_j$

classification loss for sample *i* → $\mathcal{L}(\hat{\boldsymbol{y}}_i(\boldsymbol{w}), \boldsymbol{y}_i) = \max\{1 - \kappa \hat{\boldsymbol{y}}_i(\boldsymbol{w})^\intercal \boldsymbol{y}_i, 0\}$

Total Loss → $\sum_{i \in \mathcal{I}} \underbrace{\mathcal{L}(\hat{\boldsymbol{y}}_i(\boldsymbol{w}), \boldsymbol{y}_i)}_{v_i}$

set of rules

rule *j* covers sample *i* or not

weight of rule *j*

$\kappa = (K-1)/K$

auxiliary variables

Master Problem

$$\boxed{v_i \geq \max\{1 - \kappa \hat{\boldsymbol{y}}_i(\boldsymbol{w})^\intercal \boldsymbol{y}_i, 0\}}$$

$$\begin{array}{lll} \text{minimize} & \overbrace{\lambda \sum_{j\in\mathcal{J}} c_j w_j}^{(i)} + \overbrace{\sum_{i\in\mathcal{I}} v_i}^{(ii)} & \\ \text{subject to} & \sum_{j\in\mathcal{J}} \hat{a}_{ij} w_j + v_i \geq 1, & i \in \mathcal{I}, \\ & v_i \geq 0, & i \in \mathcal{I}, \\ & w_j \geq 0, & j \in \mathcal{J} \end{array}$$

$$\hat{a}_{ij} = \kappa a_{ij} \boldsymbol{R}_j(\boldsymbol{x}_i)^\intercal \boldsymbol{y}_i$$

(i) total **weighted** cost of using rules

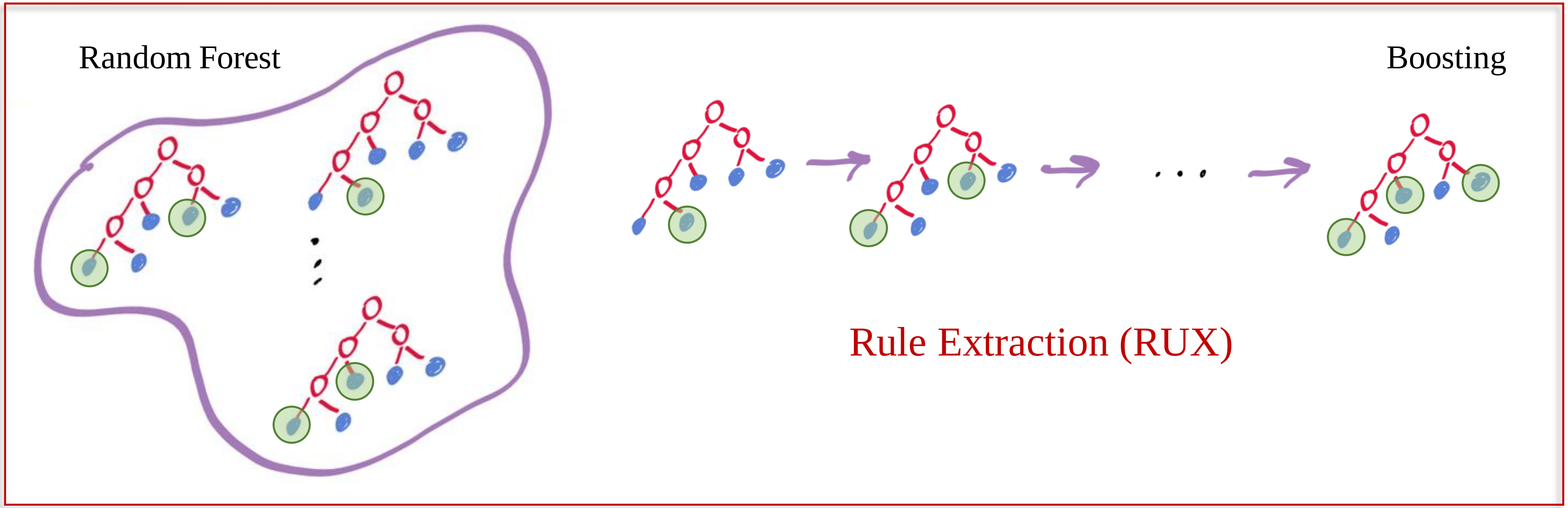
(ii) total loss from misprediction

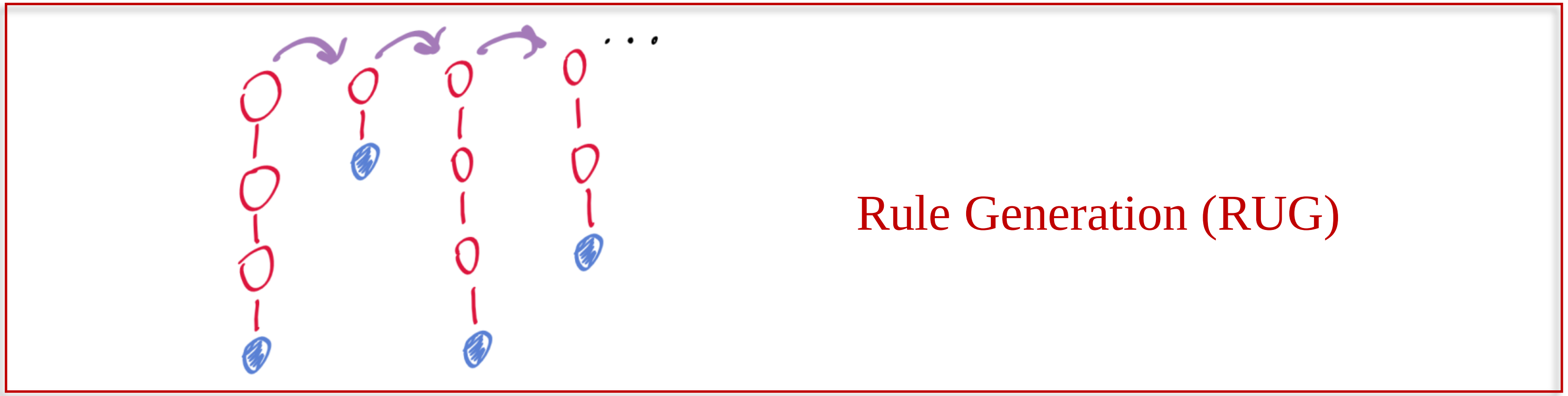
scaling hyperparameter

Set of rules (aka columns)?

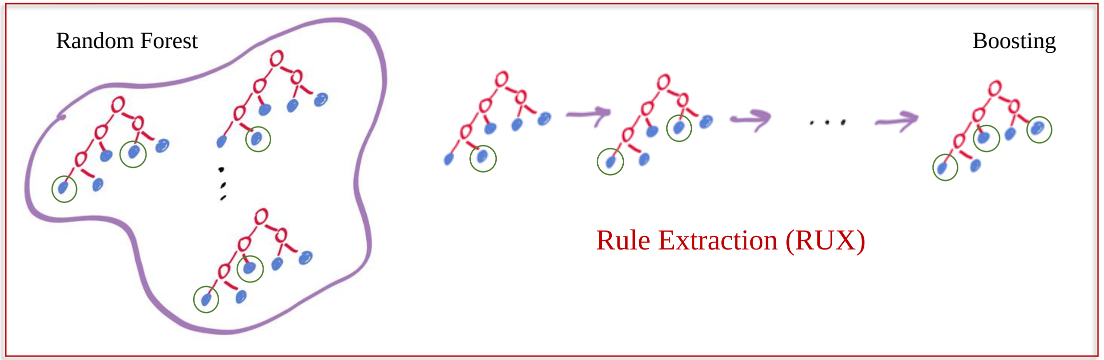
Rule Discovery

Set of rules (aka columns)?





RUX



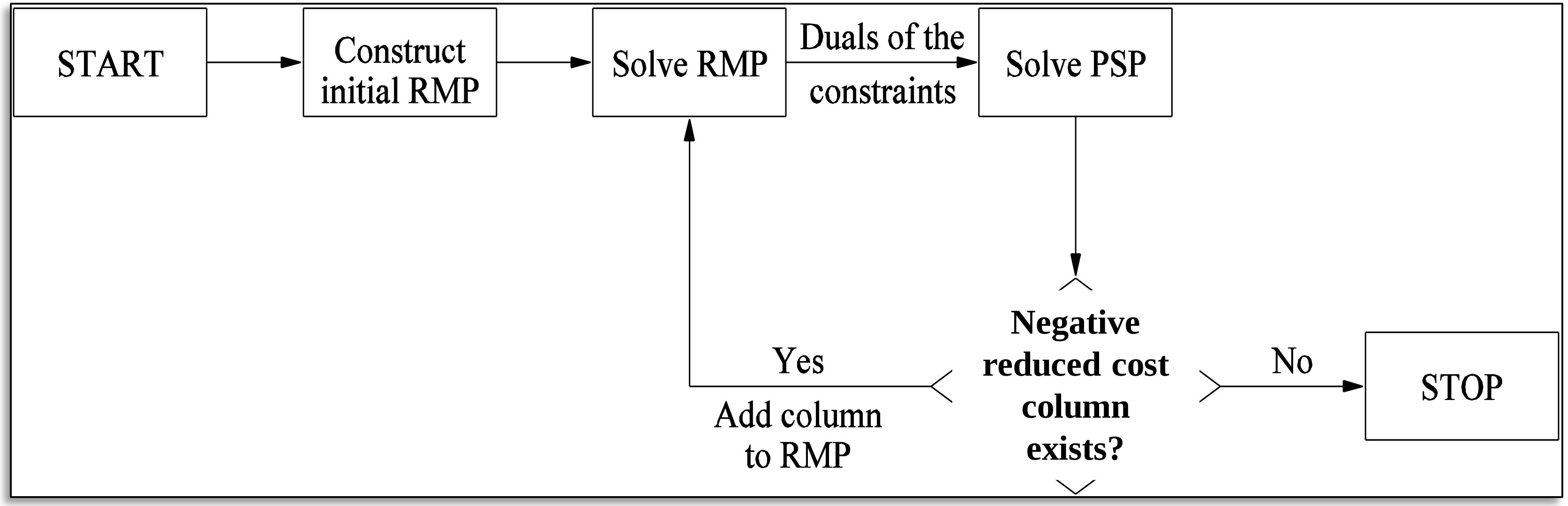
Master Problem

$$
\begin{aligned}
\text{minimize} \quad & \lambda \sum_{j \in \mathcal{J}} c_j w_j + \sum_{i \in \mathcal{I}} v_i & \\
\text{subject to} \quad & \sum_{j \in \mathcal{J}} \hat{a}_{ij} w_j + v_i \geq 1, & i \in \mathcal{I}, \\
& v_i \geq 0, & i \in \mathcal{I}, \\
& w_j \geq 0, & j \in \mathcal{J}
\end{aligned}
$$

Algorithm - Rule Extraction

1: **Input:** training data, $\mathcal{D} = \{(\boldsymbol{x}_i, y_i) : i \in \mathcal{I}\}$
2: Train a tree ensemble to construct rule pool $\mathcal{J}$
3: $\mathcal{J}^* \leftarrow$ Solve master problem using $\mathcal{J}$
4: **return** $\mathcal{J}^*$ and w_j, $j \in \mathcal{J}^*$

Column Generation

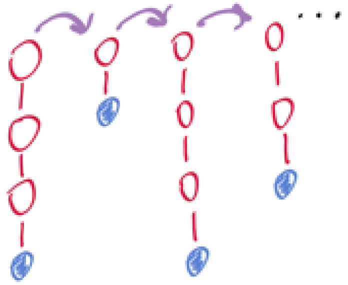


RMP*

$$
\begin{array}{lll}
\text{minimize} & \lambda \sum_{j \in \mathcal{J}_t} c_j w_j + \sum_{i \in \mathcal{I}} v_i & \\
\text{subject to} & \sum_{j \in \mathcal{J}_t} \hat{a}_{ij} w_j + v_i \geq 1, & i \in \mathcal{I}, \\
& v_i \geq 0, & i \in \mathcal{I}, \\
& w_j \geq 0, & j \in \mathcal{J}_t
\end{array}
$$

* restricted master problem

RUG



Rule Generation (RUG)

Dual Problem (J_t)

$$\begin{array}{lll} \text{maximize} & \sum_{i\in\mathcal{I}} \beta_i & \\ \text{subject to} & \sum_{i\in\mathcal{I}} \hat{a}_{ij}\beta_i \le \lambda c_j, & j \in \mathcal{J}_t, \\ & 0 \le \beta_i \le 1, & i \in \mathcal{I} \end{array}$$

PSP*

$$\min_{j\in\mathcal{J}/\mathcal{J}_t} \left\{ \lambda c_j - \sum_{i\in\mathcal{I}} \hat{a}_{ij}\beta_i^{(t)} \right\}$$

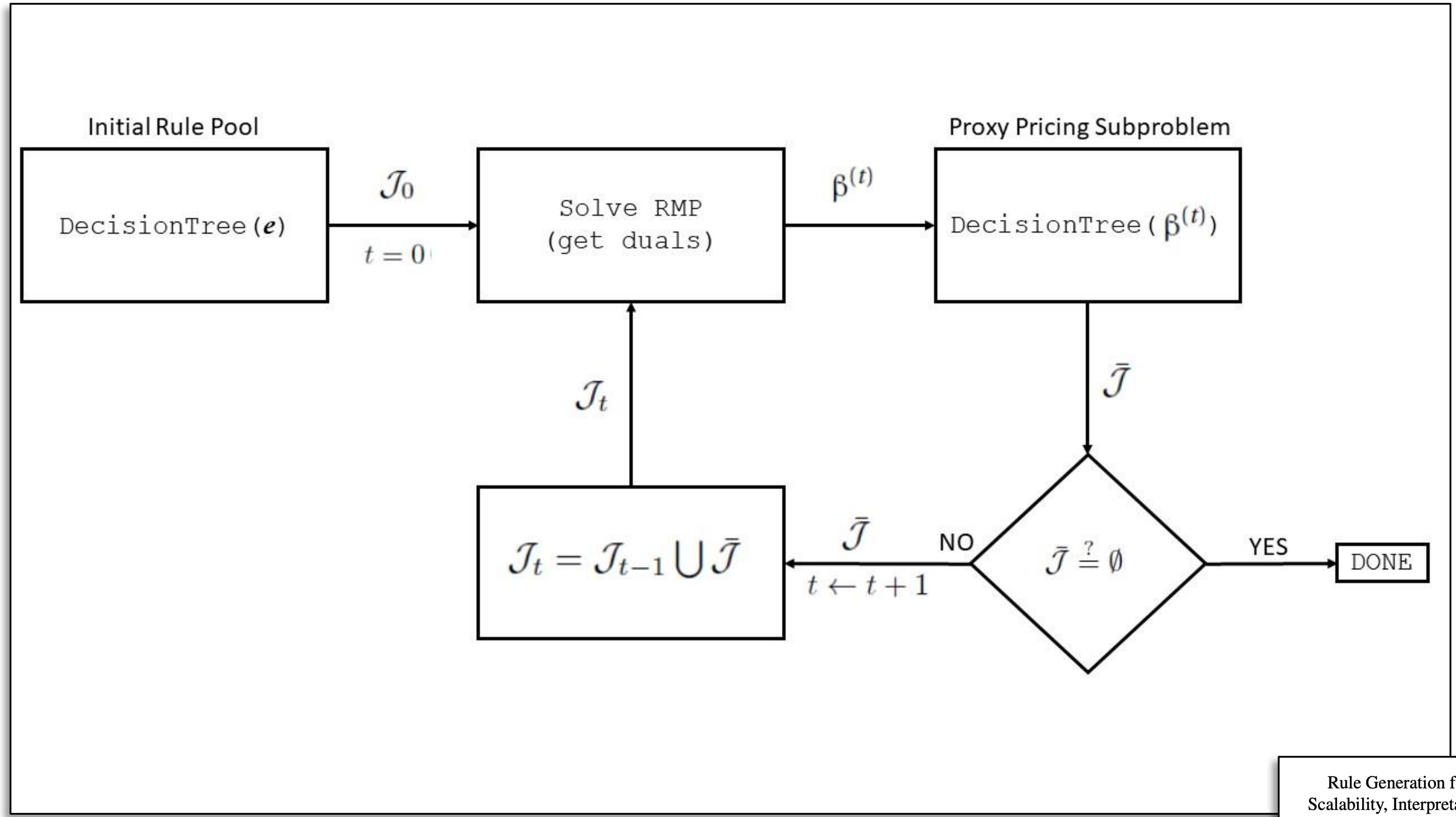
Proposition. *PSP is NP-hard*

* pricing subproblem

Proxy PSP

PSP

$$\min_{j \in \mathcal{J}/\mathcal{J}_t} \left\{ \lambda c_j - \sum_{i \in \mathcal{I}} \hat{a}_{ij} \beta_i^{(t)} \right\}$$



Rule Generation for Classification:
Scalability, Interpretability, and Fairness

Adia Lumadjeng*, Tabea Röber**, M. Hakan Akyüz*, Ş. İlker Birbil**
*Erasmus University Rotterdam, 3000 DR, Rotterdam, The Netherlands
**University of Amsterdam, 1018 TV, Amsterdam P.O. Box 15953, The Netherlands

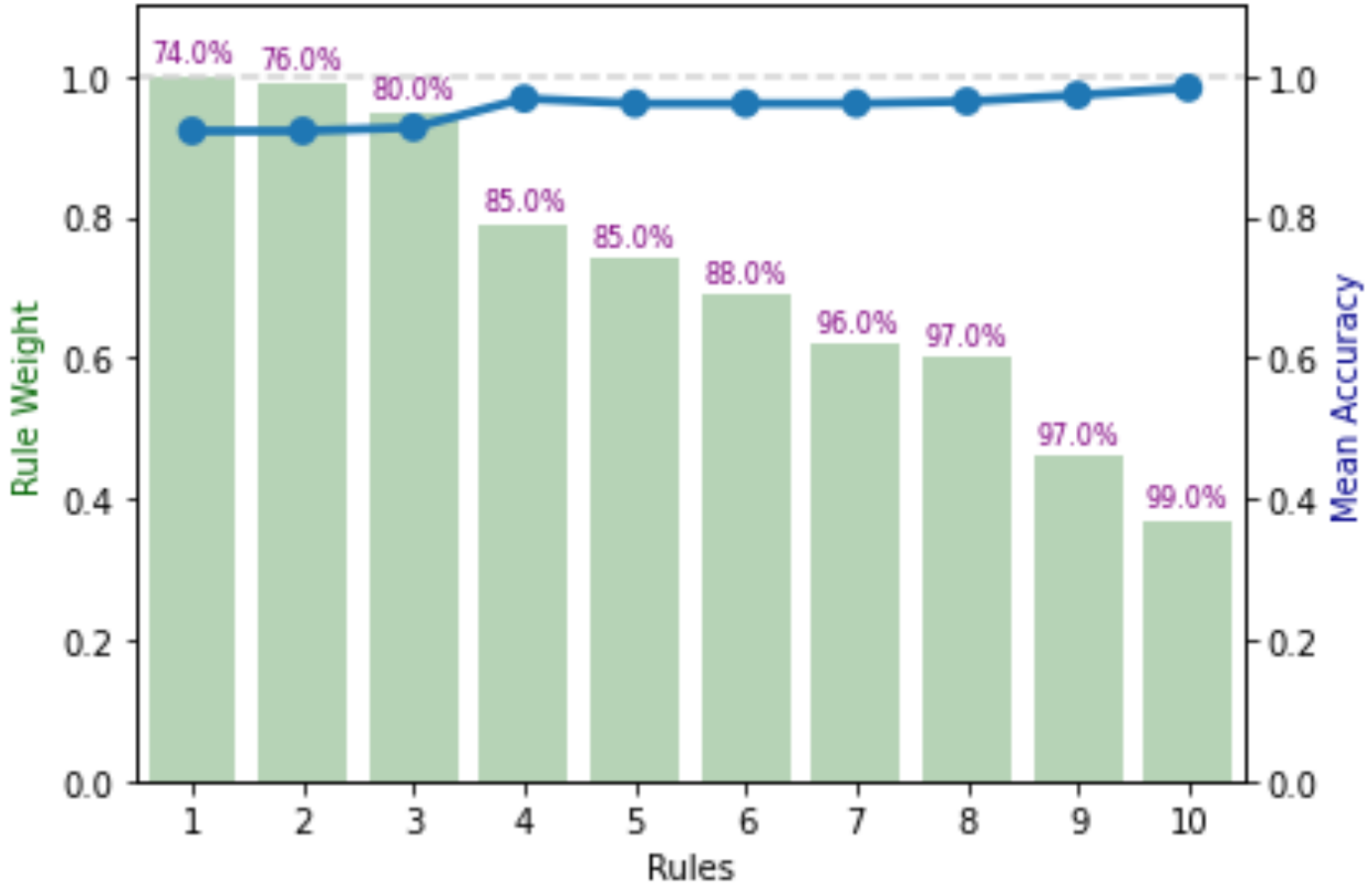
“Banknote” Dataset

Accuracy

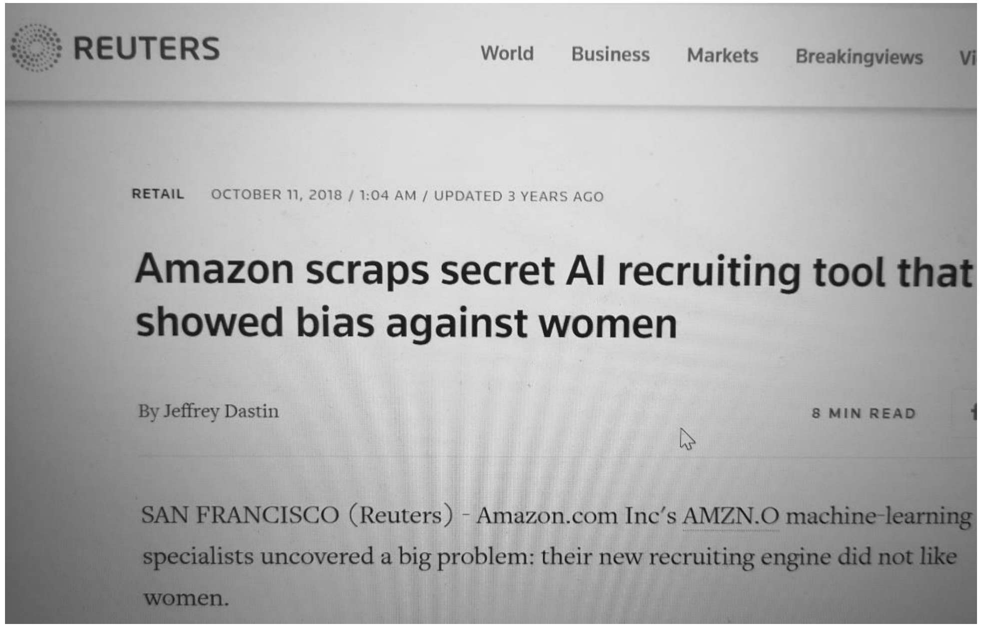
Random Forest: 0.9660194174757282
AdaBoost: 0.9975728155339806
RUXRF: 0.987864077669029
RUXADA: 0.9975728155339806
RUG: 1.0

Number of Rules

Random Forest: 712
AdaBoost: 776
RUXRF: 32
RUXADA: 39
RUG: 40



Extensions: Fairness


REUTERS World Business Markets Breakingviews

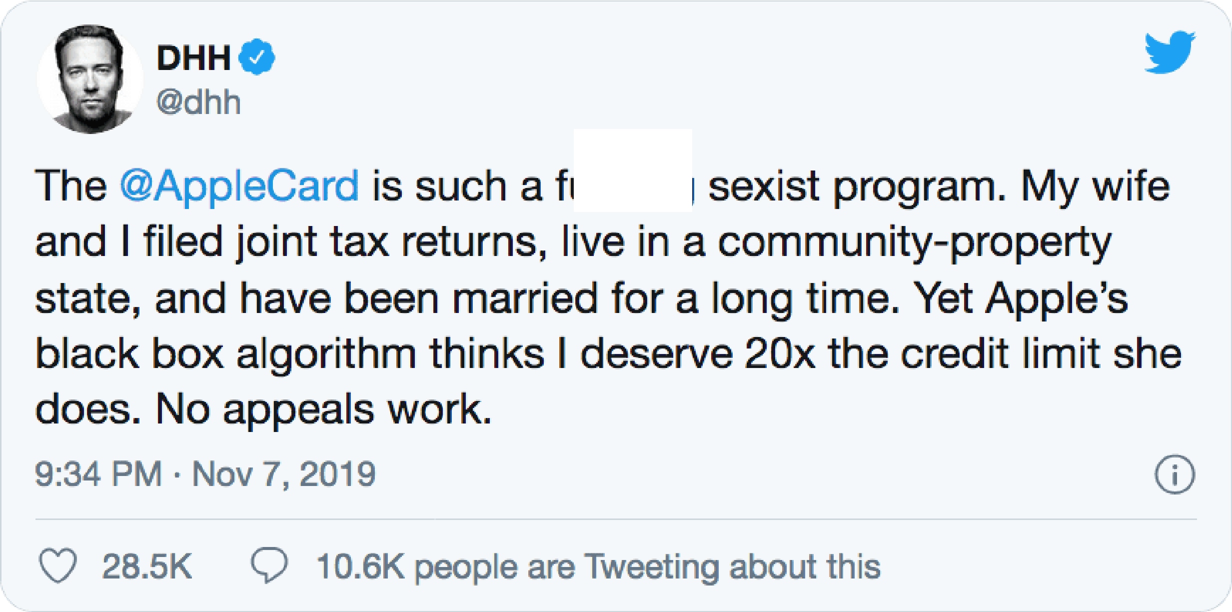
RETAIL OCTOBER 11, 2018 / 1:04 AM / UPDATED 3 YEARS AGO

Amazon scraps secret AI recruiting tool that showed bias against women

By Jeffrey Dastin 8 MIN READ

SAN FRANCISCO (Reuters) - Amazon.com Inc's AMZN.O machine-learning specialists uncovered a big problem: their new recruiting engine did not like women.

“Everyone wanted this holy grail,” one of the people said. “They literally wanted it to be an engine where I’m going to give you 100 resumes, it will spit out the top five, and we’ll hire those.”


DHH
@dhh

The @AppleCard is such a f[redacted] sexist program. My wife and I filed joint tax returns, live in a community-property state, and have been married for a long time. Yet Apple’s black box algorithm thinks I deserve 20x the credit limit she does. No appeals work.

9:34 PM · Nov 7, 2019

28.5K 10.6K people are Tweeting about this

https://thenextweb.com/news/ai-models-need-to-be-interpretable-rather-than-just-explainable

Fairness

- Sensitive Attribute (female/male, race, age etc., G)

- Additional constraints are needed.

- There are more than 20 fairness definitions (Verma and Rubin, 2018).

  - Group fairness (Statistical parity):
    $$P(\hat{Y}=1 \mid G=m) = P(\hat{Y}=1 \mid G=f).$$
  - Predictive parity:
    $$P(Y=1 \mid \hat{Y}=1, G=m) = P(Y=1 \mid \hat{Y}=1, G=f).$$
  - Equal opportunity (False negative rate):
    $$P(\hat{Y}=0 \mid Y=1, G=m) = P(\hat{Y}=0 \mid Y=1, G=f).$$
  - Equalized Odds (Disparate Mistreatment):
    $$P(\hat{Y}=1 \mid Y=i, G=m) = P(\hat{Y}=1 \mid Y=i, G=f),\ i \in 0, 1.$$
  - Overall accuracy equality:
    $$P(\hat{Y}=Y, G=m) = P(\hat{Y}=Y, G=f).$$

Binary classification

What happens when there are multiple classes?

Equalized Odds:

$$P(\hat{y}_i \neq y_i | y_i = k, G = g) = P(\hat{y}_j \neq y_j | y_j = k, G = g'), \qquad \forall i, j \in \mathcal{I}, \forall k \in \mathcal{K} \text{ and } \forall g, g' \in \mathcal{G}$$

$$\begin{array}{lll} \text{minimize} & \lambda \sum_{j \in \mathcal{J}} c_j w_j + \sum_{i \in \mathcal{I}} v_i & \\ \text{subject to} & \sum_{j \in \mathcal{J}} \hat{a}_{ij} w_j + v_i \geq 1, & i \in \mathcal{I}, \\ & v_i \geq 0, & i \in \mathcal{I}, \\ & w_j \geq 0, & j \in \mathcal{J} \end{array}$$

Fairness Constraints

$$\sum_{i \in \mathcal{P}_{k,1}} v_i - \sum_{i \in \mathcal{P}_{k,2}} v_i \leq \epsilon \qquad \forall k \in \mathcal{K} \text{ and } \forall g, g' \in \mathcal{G}$$

$$\sum_{i \in \mathcal{P}_{k,2}} v_i - \sum_{i \in \mathcal{P}_{k,1}} v_i \leq \epsilon \qquad \forall k \in \mathcal{K} \text{ and } \forall g, g' \in \mathcal{G}$$

FairRUX FairRUG

$$\boxed{\mathcal{P}_{k,g} = \{i \in \mathcal{I} : y_i = k, G = g\}}$$

Equal Overall Mistreatment:

$$\sum_{k\in\mathcal{K}} P(\hat{y}_i \neq y_i = k | G = g) = \sum_{k\in\mathcal{K}} P(\hat{y}_i \neq y_i = k | G = g'), \qquad \forall g, g' \in \mathcal{G}$$

$$\begin{array}{lll} \text{minimize} & \lambda \sum_{j\in\mathcal{J}} c_j w_j + \sum_{i\in\mathcal{I}} v_i & \\ \text{subject to} & \sum_{j\in\mathcal{J}} \hat{a}_{ij} w_j + v_i \geq 1, & i \in \mathcal{I}, \\ & v_i \geq 0, & i \in \mathcal{I}, \\ & w_j \geq 0, & j \in \mathcal{J} \end{array}$$

Fairness Constraints

$$\left\{ \begin{array}{ll} \sum_{i\in\mathcal{I}_g} v_i - \sum_{i\in\mathcal{I}_{g'}} v_i \leq \epsilon & \forall g, g' \in \mathcal{G} \\ \sum_{i\in\mathcal{I}_{g'}} v_i - \sum_{i\in\mathcal{I}_g} v_i \leq \epsilon & \forall g, g' \in \mathcal{G} \end{array} \right.$$

FairRUX

FairRUG

$$\mathcal{I}_g = \{i \in \mathcal{I} : G = g\}$$

Rule Generation – Take Aways

- Linear programming brings in scalability
- The generated rules come with their associated weights
- Proxy PSP can be solved very fast
- Fairness constraints can be easily added
- There could still be too many rules leading to a less interpretable model
- Accuracy can be inferior compared against ensemble methods

Rule Generation for Classification: Scalability, Interpretability, and Fairness

Adia Lumadjeng*, Tabea Röber**, M. Hakan Akyüz*, Ş. İlker Birbil**
*Erasmus University Rotterdam, 3000 DR, Rotterdam, The Netherlands
**University of Amsterdam, 11018 TV, Amsterdam P.O. Box 15953, The Netherlands

$n = 245057$; $p = 3$; $K = 2$;
$n = 58509$; $p = 48$; $K = 11$; (2022)

Optional Readings

Interpretable and Fair Boolean Rule Sets via Column Generation

Connor Lawless[1] Sanjeeb Dash[2] Oktay Günlük[1] Dennis Wei[2]

Mach Learn
DOI 10.1007/s10994-017-5633-9

Optimal classification trees

Dimitris Bertsimas[1] · Jack Dunn[2]

Rule Generation for Classification: Scalability, Interpretability, and Fairness

Adia Lumadjeng*, Tabea Röber**, M. Hakan Akyüz*, Ş. İlker Birbil**
*Erasmus University Rotterdam, 3000 DR, Rotterdam, The Netherlands
**University of Amsterdam, 11018 TV, Amsterdam P.O. Box 15953, The Netherlands