A Scalable Two Stage Approach to Computing Optimal Decision Sets

Alexey Ignatiev[1], Edward Lam[1,2], Peter J. Stuckey[1], Joao Marques-Silva[3]

[1] Monash University, Australia [2] CSIRO Data61, Australia [3] ANITI, IRIT, CNRS, France

Problem definition

Assume standard classification scenario with *training data* $\mathcal{E} = \{e_1, \dots, e_M\}$. A data instance $e_i \in \mathcal{E}$ is a pair $(\mathbf{v}_i, c_i)$ where $\mathbf{v}_i \in \mathbb{F}$ is a vector of feature values and $c_i \in \mathcal{C}$ is a class. An instance e_i *associates* a vector of feature values $\mathbf{v}_i$ with a class $c_i \in \mathcal{C}$.

A decision set is an *unordered set* of *rules*. Each rule π is from the set $\mathcal{R} = \prod_{r=1}^{K}\{f_r, \neg f_r, \mathfrak{u}\}$, where $\mathfrak{u}$ represents a *don't care* value. For each instance $e \in \mathcal{E}$, a rule of the form $\pi \Rightarrow c$, $\pi \in \mathcal{R}$, $c \in \mathcal{C}$ is interpreted as "if the feature values of example e agree with π then the rule predicts that example e has class c".

#	Day	Venue	Weather	TV Show	*Date?*
e_1	Weekday	Dinner	Warm	Bad	No
e_2	Weekend	Club	Warm	Bad	Yes
e_3	Weekend	Club	Warm	Bad	Yes
e_4	Weekend	Club	Cold	Good	No

IF TV Show = Good THEN Date = No
IF Day = Weekday THEN Date = No
IF TV Show = Bad ∧ Day = Weekend THEN Date = Yes

Typical approach to the problem

input : training data E
output: *smallest* decision set ϕ

$N \leftarrow LB$ — *# N equals a lower bound on $|\phi|$, which is often set to 1*

while True:
  $F \leftarrow \texttt{Encode}(E, N)$ — *# encode problem "is there a decision set ϕ of size N for data E?"*
  $(st, \mu) \leftarrow \texttt{Oracle}(F)$ — *# call a reasoning oracle to answer the question*
  if st **is** True:
    break
  $N \leftarrow N + 1$
$\phi \leftarrow \texttt{ExtractRules}(\mu)$ — *# extract decision set ϕ from satisfying assignment μ*
return ϕ

encoding is too large!
(does not scale)

Our approach

divide the process into two stages:

1. enumerate individual rules
- compute all possible rules
- MaxSAT-based
- incremental!
- breaking symmetric rules

2. compute smallest rule cover
- reduced to set cover
- solved with ILP/MaxSAT

+

each class is computed *independently*

the idea is to scale better

Stage 1 — learning rules

each rule is a solution to MaxSAT formula

$$\psi \triangleq H \wedge S$$

***H* — hard clauses**

1. **coverage constraints:**
   - **rule must cover ≥ 1 right instances**
2. **discrimination constraints:**
   - **rule must not cover any wrong instances**

***S* — soft clauses**

- **minimize the number of used literals**

$O(K + M)$ variables and $O(K \times M)$ clauses

(K — number of features, M — number of training instances)

Stage 2 — computing rule cover

#	Day	Venue	Weather	TV Show	*Date?*
e_1	Weekday	Dinner	Warm	Bad	No
e_2	Weekend	Club	Warm	Bad	Yes
e_3	Weekend	Club	Warm	Bad	Yes
e_4	Weekend	Club	Cold	Good	No

$\pi_1 = [$ IF Day = Weekday THEN Date = No $]$
$\pi_2 = [$ IF Venue = Dinner THEN Date = No $]$
$\pi_3 = [$ IF Weather = Cold THEN Date = No $]$
$\pi_4 = [$ IF TV Show = Good THEN Date = No $]$

$b_j \in \{0, 1\}$ and $s_j = |\pi_j|$ for each π_j **$A = (a_{ij})$, $a_{ij} = 1$ iff π_j covers e_i**

	π_1	π_2	π_3	π_4
a_{ij}	1	1	0	0
	0	0	1	1
s_j	1	1	1	1

$$\textbf{minimize} \sum_j s_j \cdot b_j$$

$$\textbf{subject to} \sum_j a_{ij} \cdot b_j \geq 1, \forall i$$

Breaking symmetric rules

#	Day	Venue	Weather	TV Show	*Date?*
e_1	Weekday	Dinner	Warm	Bad	No
e_2	Weekend	Club	Warm	Bad	Yes
e_3	Weekend	Club	Warm	Bad	Yes
e_4	Weekend	Club	Cold	Good	No

IF TV Show = Good THEN Date = No vs. IF Weather = Cold THEN Date = No

rules covering same instances are symmetric
no point in computing both!

for each rule, add one clause enforcing
all following rules to cover ≥ 1 other instance

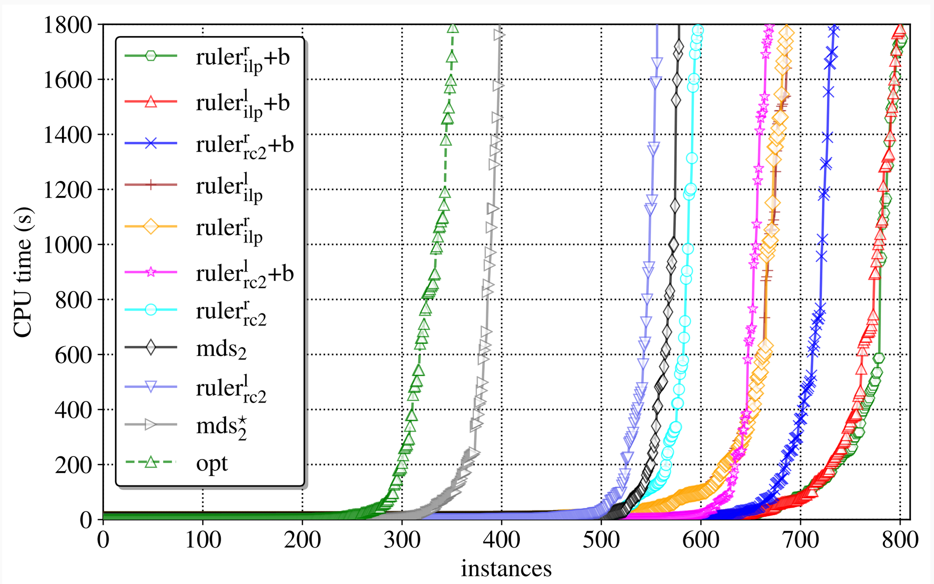
Experimental setup:

- **machine configuration:**
  - Intel Xeon Silver-4110 2.10GHz with 64GByte RAM, running Debian Linux, 1800s timeout + 8GB memout
- **UCI Machine Learning Repository + Penn Machine Learning Benchmarks**
  - **1065 benchmarks** in total (71 datasets × 5-cross validation × 3 quantized families)
  - **3–384 features** (one-hot encoded), **14–67557 training** instances

- **competition tested:**
  - **mds_2** – minimization of number of rules
  - **$\text{mds}_2^\star$** – *lexicographic* minimization of number of rules + literals
  - **opt** – minimization of number of literals

same code base and SAT solver – Glucose 3!

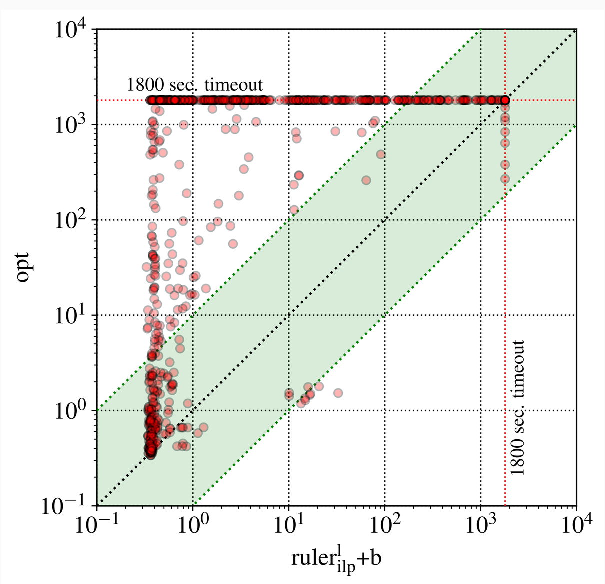
- **ruler_*°**
  - $\circ \in \{l, r\}$ – optimization criterion
  - **stage 1** – **incremental calls** to RC2 MaxSAT solver
  - **stage 2** – $* \in \{rc2, ilp\}$ – either *RC2 MaxSAT* or *Gurobi ILP*
  - **ruler_*°+b** – symmetry breaking *enabled*

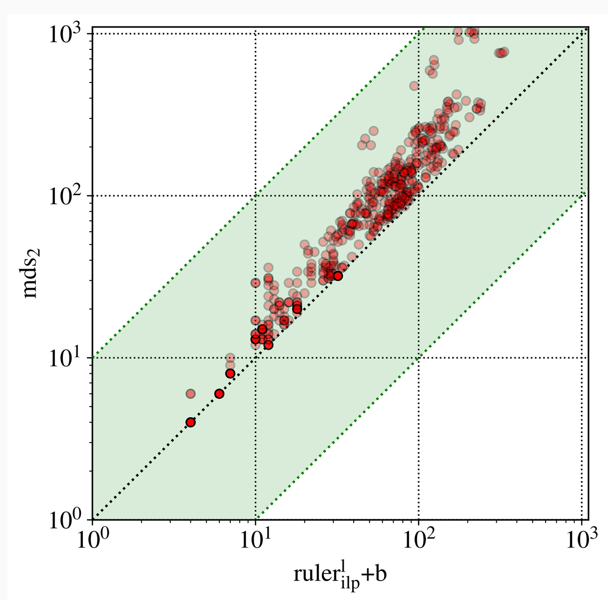


Performance comparison:

$\text{ruler}_{\text{ilp}}^{\text{l}}$+b vs. **opt**:
up to 4 orders of magnitude improvement!

breaking symmetric rules:
avg. # of rules goes down **from 19604.4 to 563.7**





Model size comparison:

$\text{ruler}_{\text{ilp}}^{\text{l}}$+b vs. **$\text{mds}_2$**:
halves avg. size (**62.2** vs. **116.2**)

$\text{mds}_2^\star$ vs. **mds_2**:
lexicographic optimization pays off
(but slower!)