A Scalable Two Stage Approach to Computing Optimal Decision Sets

Alexey Ignatiev¹, Edward Lam¹,², Peter J. Stuckey³, Joao Marques-Silva³
¹ Monash University, Australia ² CSIRO Data61, Australia ³ ANITI, IRIT, CNRS, France

Problem definition
Assume standard classification scenario with training data $E = \{e_1, \ldots, e_N\}$. A data instance $e_i \in E$ is a pair $(v_i, c_i)$ where $v_i \in \Xi$ 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 $v_i$ with a class $c_i \in \mathcal{C}$. A decision set is an unordered set of rules. Each rule $\pi$ is from the set $\mathcal{R} = \prod_{i=1}^{N}(f_i, -f_i, u_i)$, where $u_i$ represents a don't care value. For each instance $e_i \in E$, a rule of the form $\pi \Rightarrow c_i \in \mathcal{C}$ is interpreted as "if the feature values of example $e_i$ agree with $\pi$ then the rule predicts that example $e_i$ has class $c_i$.

Typical approach to the problem

- **input**: training data $E$
- **output**: smallest decision set $\phi$

\[ N \leftarrow LB \]
\[ \text{while True:} \]
\[ F \leftarrow \text{Encode}(E, N) \]
\[ (st, \mu) \leftarrow \text{Oracle}(F) \]
\[ \text{if it is True:} \]
\[ \text{break} \]
\[ N \leftarrow N + 1 \]
\[ \phi \leftarrow \text{ExtractRules}(\mu) \]
\[ \text{return } \phi \]

**Stage 1 — learning rules**

- **each rule is a solution to MaxSAT formula**
  \[ \psi \models H \land S \]

  - **$H$** = hard clauses
  - **$S$** = soft clauses
  - 1. coverage constraints:
    - minimize the number of used literals
    - rule must cover $\geq 1$ right instances
  - 2. discrimination constraints:
    - rule must not cover any wrong instances

  \[ O(K + M) \text{ variables and } O(K \times M) \text{ clauses} \]

  - $K$ = number of features, $M$ = number of training instances

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

**Stage 2 — computing rule cover**

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

\[ a_{ij} \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix} \]

\[ s_j \begin{bmatrix} 1 & 1 & 1 \end{bmatrix} \]

\[ \text{minimize } \sum s_j \cdot b_j \]

\[ \text{subject to } \sum a_{ij} \cdot b_j \geq 1, \forall i \]

**Experimental setup**

- **competition tested:**
  - $\text{mds}_1$ – minimization of number of rules
  - $\text{mds}_2$ – lexicographic minimization of number of rules + literals
  - $\text{opt}$ – minimization of number of literals

- **ruler**
  - $\alpha \in \{1, 2\}$ = optimization criterion
  - stage 1 = incremental calls to RC2-MaxSAT solver
  - stage 2 = $\alpha \in \{rc2, ilp\}$ = either RC2-MaxSAT or GaroBI LP
  - $\text{ruler}_1$, $\text{ruler}_2$ = symmetry breaking enabled

**Performance comparison**

- $\text{rules}_{1/2}^1$, $\text{rules}_{1/2}^2$ vs opt:
  - up to 4 orders of magnitude improvement!

- breaking symmetric rules:
  - avg. # of rules goes down from 18609.4 to 563.7

**Model size comparison**

- $\text{rules}_{1/2}^1$, $\text{mds}_2$ halve avg. size (62 vs 116.2)
- $\text{mds}_2$ vs $\text{mds}_1$:
  - lexicographic optimization pays off (but slower?)

**Our approach**

- divide the process into two stages:
  1. enumerate individual rules
  2. compute smallest rule cover

- compute all possible rules
- MaxSAT-based
- reduced to set cover
- incremental!
- breaking symmetric rules
- solved with ILP/MaxSAT
- each class is computed independently
- the idea is to scale better