A General Framework for Learning an Ensemble of Decision Rules

Krzysztof Dembczyński[1] Wojciech Kotłowski[1]
Roman Słowiński[1,2]

Institute of Computing Science, Poznań University of Technology,
60-965 Poznań, Poland
{kdembczynski, wkotlowski, rslowinski}@cs.put.poznan.pl

Systems Research Institute, Polish Academy of Sciences, 01-447 Warsaw, Poland

ECML/PKDD Workshop – LeGo 2008

Motivation

- **Decision rule** is a simple logical pattern in the form:

  "if $condition$ then $decision$".

- A **simple classifier** voting for some class when the condition is satisfied and **abstaining** from vote otherwise.
- Example:

  if duration $>= 36$
  and savings status ≥ 1000
  and employment $\neq unemployed$
  and purpose $= furniture/equipment$,
  $then$ risk level is low

- Main advantage of decision rules is their **simplicity** and **human-interpretable** form handling **interactions** between attributes.

Motivation

- The most popular rule induction algorithms are based on **sequential covering**: AQ, CN2, Ripper.
- **Forward stagewise additive modeling** or **boosting** that treats rules as **base classifiers** in the **ensemble** can be seen as a generalization of sequential covering.
- Algorithms such as RuleFit, SLIPPER, LRI or MLRules follow boosting approach and are quite **similar** with the difference in the chosen **loss function** and **minimization technique**.
- We investigated a **general rule ensemble algorithm** using variety of loss functions and minimization techniques, and taking into account other issues, such as **regularization** by **shrinking** and **sampling**.

Main Contribution

- We showed theoretically and confirmed empirically that the choice of **minimization technique** implicitly controls the **rule coverage** – one of techniques (**constant-step minimization**) is characterized by the parameter that directly influences the rule coverage.
- It follows from a large experiment that the choice of loss function and minimization technique does **not significantly** improves the accuracy.
- Proper regularization specific for decision rules has **significant** impact on the accuracy.

Rule Ensembles and LeGo

- Local patterns such as rules can be **combined** into the global model by boosting.
- In general, the construction of patterns should be guided by a **global criterion**, and **only** in specific domains one can consider such phases as single rule generation, rule selection and global model construction as **independent**.
- Local pattern should be a **sort of knowledge** extracted from the data by which we are capable of giving **accurate predictions** – therefore, patterns should be discovered having prediction accuracy in mind being globally defined criterion.
- One can consider a **trade-off** between **interpretability** and **accuracy** of such patterns.

Classification Problem

- The aim is to **predict** an **unknown value** of an attribute $y \in \{-1, 1\}$ of an object using **known** joint **values** of other **attributes** $\mathbf{x} = (x_1, x_2, \ldots, x_n) \in \mathcal{X}$.
- The task is to **learn** a function $f(\mathbf{x})$ that predicts **accurately** the value of y by using a **training set** $\{y_i, \mathbf{x}_i\}_1^N$.
- The accuracy of function f is measured in terms of the **risk**:

$$R(f) = \mathbb{E}[L(y, f(\mathbf{x}))],$$

where **loss function** $L(y, f(\mathbf{x}))$ is a penalty for predicting $f(\mathbf{x})$ if the actual class label is y, and the expectation is over joint distribution $P(y, \mathbf{x})$.

Decision Rule

- Decision rule can be treated as function returning constant **response** $\alpha \in \mathbb{R}$ in some axis-parallel (rectangular) **region** S in attribute space $\mathcal{X}$ and zero outside S.
- Value of $\mathrm{sgn}(\alpha)$ indicates **decision** (class) and $|\alpha|$ expresses the **confidence** of predicting the class.
- Function $\Phi(\mathbf{x})$ indicates whether an object $\mathbf{x}$ satisfies the **condition part** of the rule: $\Phi(\mathbf{x}) = 1$, if $\mathbf{x} \in S$, otherwise $\Phi(\mathbf{x}) = 0$.
- Decision rule can be written as:

$$r(\mathbf{x}) = \alpha\Phi(\mathbf{x}).$$

Ensemble of Decision Rules

- Ensemble of decision rules is a **linear combination** of M decision rules:

  $$f_M(\mathbf{x}) = \alpha_0 + \sum_{m=1}^{M} \alpha_m \Phi_m(\mathbf{x}),$$

  where α_0 is a constant value, which can be interpreted as a **default rule**, covering the whole attribute space $\mathcal{X}$.
- Construction of an optimal combination of rules minimizing the risk on training set:

  $$f_M^*(\mathbf{x}) = \arg\min_{f_M} \sum_{i=1}^{N} L(y_i, \alpha_0 + \sum_{m=1}^{M} \alpha_m \Phi_m(\mathbf{x}))$$

  is a **hard optimization problem**.

Learning an Ensemble of Decision Rules (ENDER)

- One starts with the default rule:

$$\alpha_0 = \arg\min_{\alpha} \sum_{i=1}^{N} L(y_i, \alpha).$$

- In each subsequent iteration m, one generates a rule:

$$r_m(\mathbf{x}) = \arg\min_{\Phi,\alpha} \sum_{i=1}^{N} L(y_i, f_{m-1}(\mathbf{x}_i) + \alpha\Phi(\mathbf{x}_i)),$$

  where $f_{m-1}(\mathbf{x})$ is a classification function after $m-1$ iterations.

  Since the exact solution of this problem is still computationally hard, it is proceeded in **two steps**.

Step 1: Constructing Condition Part of the Rule

- Find Φ_m as a **greedy** solution of the problem:

$$\Phi_m = \arg\min_{\Phi} \mathcal{L}_m(\Phi) \simeq \arg\min_{\Phi} \sum_{i=1}^{N} L(y_i, f_{m-1}(\mathbf{x}_i) + \alpha\Phi(\mathbf{x}_i)).$$

- Four **minimization techniques** are considered:
  - **Simultaneous minimization** is applied to loss functions for which a closed-form solution for α_m can be given.
  - **Gradient descent** is applied to any differentiable loss function and relies on approximating $L(y_i, f_{m-1}(\mathbf{x}_i) + \alpha\Phi(\mathbf{x}_i))$ up to the first order.
  - **Gradient boosting** minimizes the squared-error between rule outputs and the negative gradient of any differentiable loss function.
  - **Constant-step minimization** restricts $\alpha \in \{-\beta, \beta\}$, with β being a fixed parameter.

Step 1: Constructing Condition Part of the Rule

- Greedy procedure for finding Φ_m works in the way resembling **generation of decision trees** – an algorithm constructs only one path from the root to the leaf.
- This procedure ends if $\mathcal{L}_m(\Phi)$ cannot be decreased – there is a **trade-off** between **covered** and **uncovered examples**.
- Contrary to the generation of decision trees, a minimal value of $\mathcal{L}_m(\Phi)$ is a **natural** stop criterion.
- Rules do **adapt** to the problem; no additional stop criteria are needed.

Step 2: Computing Rule Response

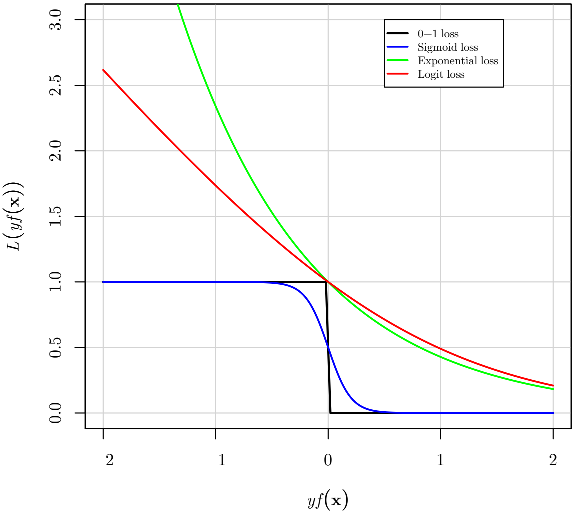
- Find α_m, the solution to the following **line-search** problem with Φ_m found in the previous step:

$$\alpha_m = \arg\min_{\alpha} \sum_{i=1}^{N} L(y_i, f_{m-1}(\mathbf{x}_i) + \alpha\Phi_m(\mathbf{x}_i)).$$

- Depending on the loss function, **analytical** or **approximate** solution exists.

Loss Functions

- Three loss functions are considered: **exponential**, **logit** and **sigmoid** loss being margin-sensitive surrogates of 0-1 loss.



Rule Response and Loss Functions

- For the exponential loss, a **closed-form** solution for α_m exists (simultaneous minimization can be performed in case of this function).
- For the logit loss there is no analytical solution for optimal rule response α_m and the solution is obtained by single **Newton-Raphson** step.
- Because of non-convexity of the sigmoid loss, α_m is chosen to be a small **constant step** along the direction of the negative gradient (constant-step minimization tailored for this loss function).

Minimization Techniques and Rule Coverage

- Denote examples **correctly classified** by the rule by

$$R_+ = \{i\colon\ y_i \alpha \Phi(\mathbf{x}_i) > 0\}.$$

- Denote examples **misclassified** by the rule by

$$R_- = \{i\colon\ y_i \alpha \Phi(\mathbf{x}_i) < 0\}.$$

- Let $w_i^{(m)}$ be **weights** of training examples in m-th iteration:

$$w_i^{(m)} = -\frac{\partial L(y_i f_{m-1}(\mathbf{x}_i))}{\partial (y_i f_{m-1}(\mathbf{x}_i))}.$$

  In the case of the exponential loss, $w_i^{(m)}$ is exactly a value of loss for $\mathbf{x}_i$ after $m-1$ iterations.

Minimization Techniques and Rule Coverage

- Simultaneous minimization

$$\mathcal{L}_m(\Phi) = -\sqrt{\sum_{i \in R_+} w_i^{(m)}} + \sqrt{\sum_{i \in R_-} w_i^{(m)}}.$$

- Gradient descent

$$\mathcal{L}_m(\Phi) = -\sum_{i \in R_+} w_i^{(m)} + \sum_{i \in R_-} w_i^{(m)}.$$

- Gradient boosting

$$\mathcal{L}_m(\Phi) = \frac{-\sum_{i \in R_+} w_i^{(m)} + \sum_{i \in R_-} w_i^{(m)}}{\sqrt{\sum_{i=1}^{N} \Phi(\mathbf{x}_i)}}.$$

- Gradient descent produces **the most general** rules.

Minimization Techniques and Rule Coverage

- Gradient descent can be defined alternatively by:

$$\mathcal{L}_m(\Phi) = \sum_{i \in R_-} w_i^{(m)} + \frac{1}{2} \sum_{\Phi(\mathbf{x}_i)=0} w_i^{(m)}.$$

- Constant-step minimization (exponential loss) **generalizes** gradient descent:

$$\mathcal{L}_m(\Phi) = \sum_{i \in R_-} w_i^{(m)} + \ell \sum_{\Phi(\mathbf{x}_i)=0} w_i^{(m)},$$

  where

$$\ell = \frac{1 - e^{-\beta}}{e^{\beta} - e^{-\beta}} \in [0, 0.5), \qquad \beta = \log \frac{1-\ell}{\ell}.$$

- **Increasing** ℓ (or **decreasing** β) results in **more** general rules ($\beta \rightarrow 0$ corresponds to gradient descent).

Minimization Techniques and Rule Coverage

- Constant-step minimization for any twice-differentiable loss:

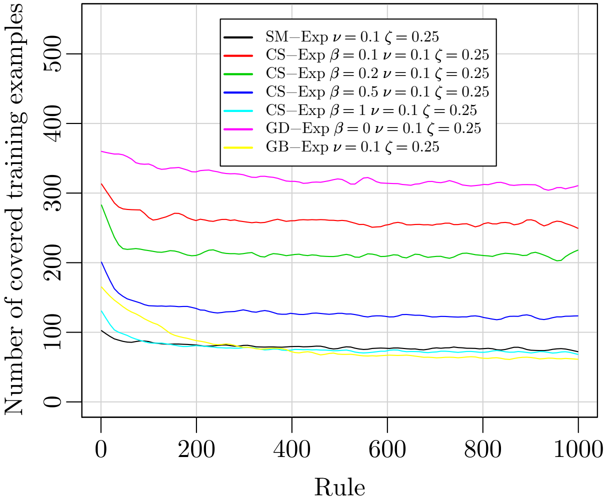
$$\mathcal{L}_m(\Phi) = \sum_{i \in R_-} w_i^{(m)} + \frac{1}{2} \sum_{\Phi(\mathbf{x}_i)=0} \left(w_i^{(m)} - \beta v_i^{(m)} \right)$$

where

$$v_i^{(m)} = \frac{1}{2} \frac{\partial^2 L(y_i f_{m-1}(\mathbf{x}_i) + y_i \gamma)}{\partial (y_i f_{m-1}(\mathbf{x}_i) + y_i \gamma)^2}, \quad \text{for some } \gamma \in [0, \beta].$$

- For convex loss functions **increasing** β **decreases** the penalty for abstaining from classification.
- For sigmoid loss, as β **increases**, uncovered correctly classified examples ($y_i f_{m-1}(\mathbf{x}_i) > 0$) are penalized **less**, while the penalty for uncovered misclassified examples ($y_i f_{m-1}(\mathbf{x}_i) < 0$) **increases**.

Rule Coverage (artificial data)



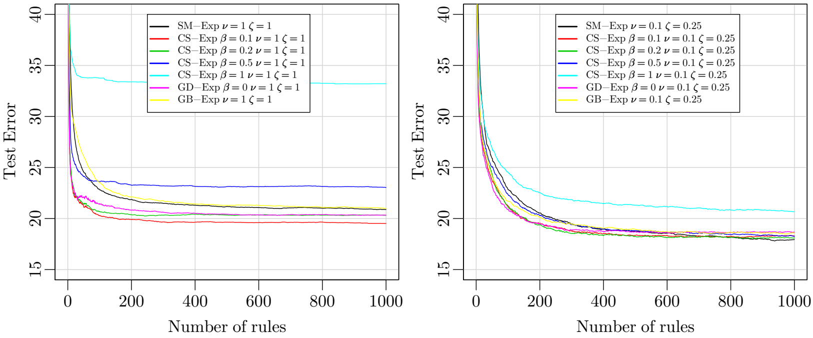
Performance

- Decision rule has the form of n-dimensional rectangle with **VC dimension** equal to $2n$ (VC dimension does not depend on the number of cuts).
- Theoretical results (Schapire et al. 1998) suggest that an ensemble of base classifiers with low VC dimension and high prediction confidence (margin) on the dataset **generalizes** well, regardless of the size of the ensemble.
- Sigmoid loss has **tighter** upper bound of the **misclassification error** than bound obtained for general ensemble (Mason et at. 1999), but minimization of this loss does **not** result in a **booster** (Duffy and Helmbold, 2000).
- Minimization of the exponential and logit loss on training set can be treated as estimation of **conditional probabilities**, while the sigmoid loss being a continuous approximation of 0-1 loss estimates the **dominant** class.

Performance

- **Regularization** of the classifier usually improves performance.
- The rule is **shrinked** (multiplied) by the amount $\nu \in (0, 1]$ towards rules already present in the ensemble – for small ν, such an approach gives similar results as **penalized learning problem** with L_1 regularization over all possible decision rules (Efron et al. 2004).
- Procedure for finding Φ_m works on a **subsample** of original data, drawn without replacement – such an approach produces **more diversified** and **less correlated** rules, and also **decreases computing time**.
- Value of α_m is calculated on **all** training examples – this usually decreases $|\alpha_m|$ and plays the role of **regularization**.
- These three elements (shrinking, sampling, and calculating α_m on the entire training set) constitute a competitive technique to **pruning**.

Unregularized vs. Regularized Solution (artificial data)

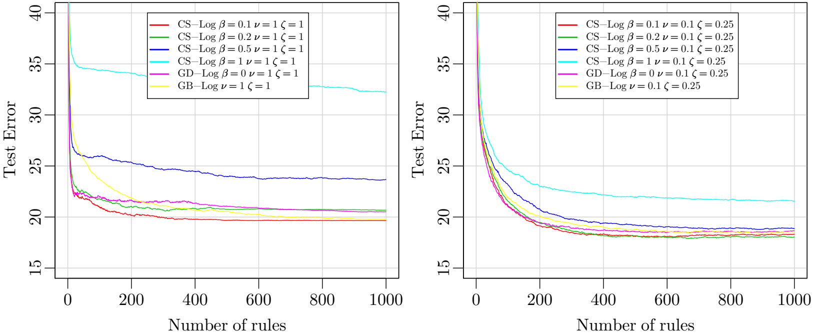


Plots for the exponential loss (Bayes rate 10%); regularized solution with shrinkage $\nu = 0.1$ and sampling $\zeta = 0.25$ (fraction of training examples).

Unregularized vs. Regularized Solution (artificial data)

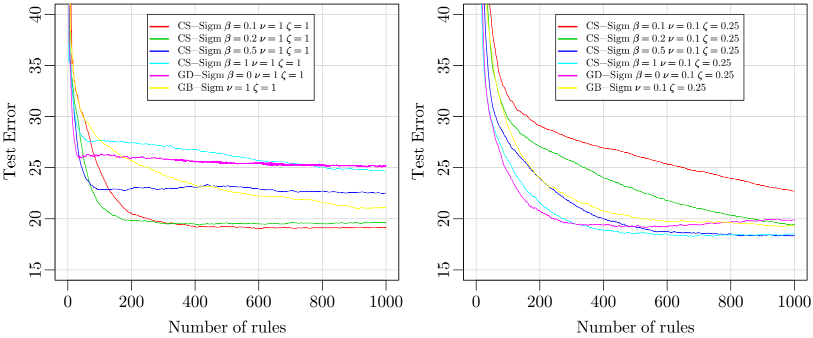
ENDER	Unregularized		Regularized	
	Test error [%]	Time [s]	Test error [%]	Time [s]
Sm-Exp	20.877±0.255	4.625	17.940±0.229	1.969
CS-Exp $\beta = 0.1$	19.513±0.286	8.063	18.300±0.235	5.399
CS-Exp $\beta = 0.2$	20.320±0.234	5.296	18.110±0.212	4.735
CS-Exp $\beta = 0.5$	23.040±0.306	3.703	18.240±0.239	2.890
CS-Exp $\beta = 1.0$	33.203±0.687	3.047	20.683±0.267	1.813
GD-Exp $\beta = 0.0$	20.333±0.290	15.515	18.670±0.282	6.062
GB-Exp	20.993±0.240	5.937	18.573±0.227	3.063

Unregularized vs. Regularized Solution (artificial data)



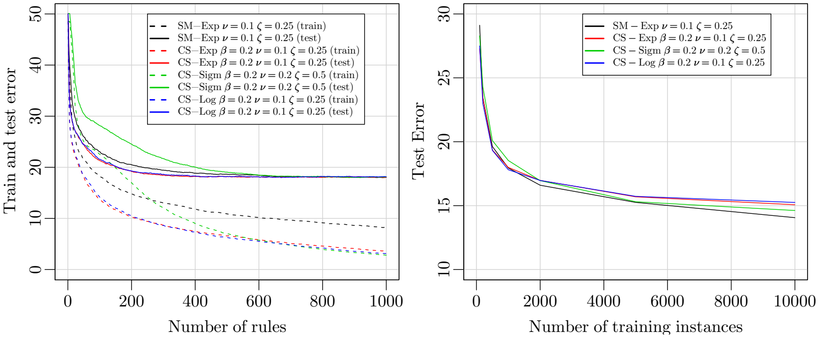
Plots for the logit loss (Bayes rate 10%); regularized solution with shrinkage $\nu = 0.1$ and sampling $\zeta = 0.25$ (fraction of training examples).

Unregularized vs. Regularized Solution (artificial data)



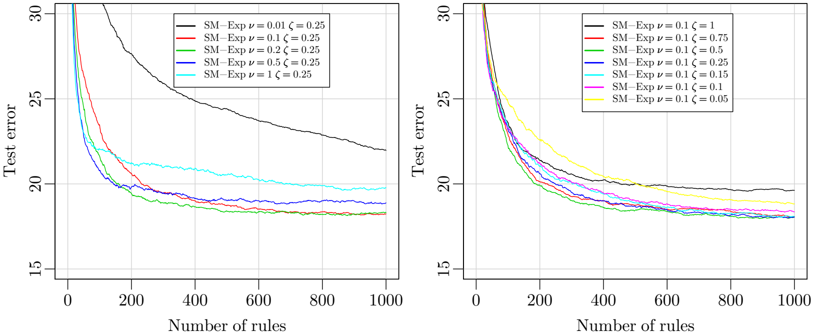
Plots for the sigmoid loss (Bayes rate 10%); regularized solution with shrinkage $\nu = 0.1$ and sampling $\zeta = 0.25$ (fraction of training examples).

Best Classifiers (artificial data)



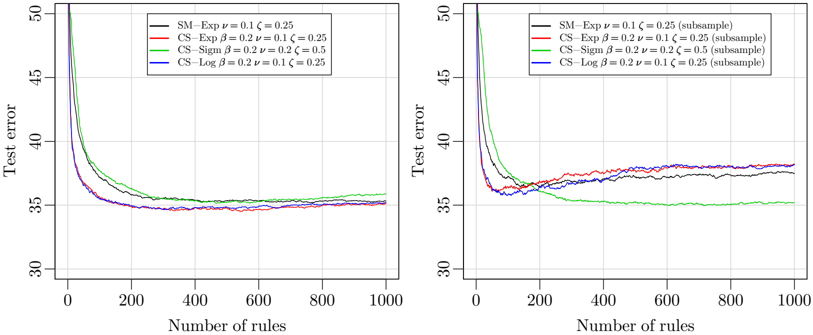
For each loss function the best minimization technique and the best values of the parameters are chosen, with exception of the exponential loss where the simultaneous minimization was treated separately from the other techniques (Bayes rate 10%).

Shrinkage and Sampling



Varying values of ν and ζ for rule ensemble based on simultaneous minimization of the exponential loss (Bayes rate 10%).

Computing Rule Response on All Training Examples



Computation of rule response over all and a subsample of training examples (Bayes rate 30%).

Related Works

- SLIPPER (Cohen and Singer, 1999)
  - Uses AdaBoost scheme with confidence-rated predictions (simultaneous minimization with the exponential loss).
  - Performs pruning by dividing training set into "growing" and "pruning" part.
- LRI (Weiss and Indurkhya, 2000)
  - Generates rules in the form of DNF formulas.
  - Uses specific re-weighting scheme based on cumulative error that corresponds to minimization of the polynomial loss by gradient descent technique.
- MLRules (Dembczyński et al., 2008)
  - Derived from the maximum likelihood principle (corresponds to minimization of logit loss by gradient descent).
  - Natural generalization to multi-class problems.

Related Works

- RuleFit (Friedman and Popescu, 2005)
  - First tree ensemble is learned and then rules are produced from the generated trees.
  - Rule ensemble is then fitted with L_1 regularization.
- Ensemble of Decision Trees
  - Natural stop criterion for building single rules; no additional parameters needed.
  - Each rule is built optimally with respect to previously generated rules.
  - Rules can discover regions that are hardly obtained by trees.
- Sequential covering
  - Using 0-1 loss in the boosting framework corresponds to sequential covering – loss decreases down to 0 for all correctly covered examples what resembles removing such objects from training set.

Computational Experiment

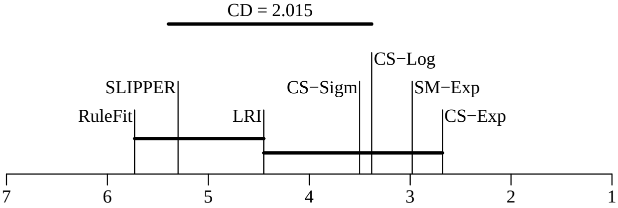
- Comparison with SLIPPER, LRI and RuleFit on 20 binary problems taken from UCI Repository:
  - SLIPPER: 500 iterations, rest of parameters default.
  - LRI: 200 rules per class, 2 disjunctions of length 5 per rule, features frozen after 50 rounds.
  - RuleFit: 500 trees, average tree size 4, rule-linear mode.
  - ENDER: the best four classifiers from the artificial data experiment, 500 rules.
- Experiment settings:
  - Accuracy estimated using 10-fold cross-validation.
  - Following Demšar (2006), Friedman test based on **average ranks** is applied.

Results

Dataset	CS-Log	SM-Exp	CS-Exp	CS-Sigm	SLIPPER	LRI	RuleFit
haberman	26.8(4.5)	25.5(1.0)	26.2(3.0)	25.8(2.0)	26.8(4.5)	27.5(7.0)	27.2(6.0)
breast-c	28.3(5.0)	27.9(3.0)	27.2(1.0)	27.3(2.0)	27.9(4.0)	29.3(6.0)	29.7(7.0)
diabetes	24.5(2.0)	24.6(3.5)	24.6(3.5)	23.6(1.0)	25.4(6.0)	25.4(5.0)	26.2(7.0)
credit-g	23.3(2.0)	23.5(3.0)	22.8(1.0)	24.2(5.0)	27.7(7.0)	23.9(4.0)	25.9(6.0)
credit-a	13.5(4.5)	13.5(4.5)	12.3(2.0)	13.8(6.0)	17.0(7.0)	12.2(1.0)	13.2(3.0)
ionosphere	6.3(3.0)	6.0(2.0)	5.7(1.0)	6.5(4.5)	6.5(4.5)	6.8(6.0)	8.5(7.0)
colic	15.0(5.0)	14.7(3.5)	14.4(2.0)	12.8(1.0)	15.1(6.0)	16.1(7.0)	14.7(3.5)
hepatitis	19.5(7.0)	18.2(4.0)	18.8(5.0)	16.2(1.0)	16.7(2.0)	18.0(3.0)	19.4(6.0)
sonar	16.8(5.0)	15.4(3.0)	16.4(4.0)	14.5(1.0)	26.4(7.0)	14.9(2.0)	19.7(6.0)
heart-statlog	16.7(1.0)	17.0(2.0)	17.4(3.5)	17.4(3.5)	23.3(7.0)	19.6(6.0)	18.5(5.0)
liver-disorders	26.4(4.0)	25.8(3.0)	24.9(1.0)	24.9(2.0)	30.7(7.0)	26.6(5.0)	30.7(6.0)
vote	3.2(1.0)	3.4(2.5)	3.4(2.5)	4.6(5.0)	5.0(6.0)	3.9(4.0)	5.1(7.0)
heart-c-2	16.9(4.0)	15.5(3.0)	15.2(1.0)	15.5(2.0)	19.5(7.0)	18.5(5.0)	18.9(6.0)
heart-h-2	17.0(1.0)	17.6(3.0)	17.3(2.0)	19.3(6.0)	20.0(7.0)	18.3(4.0)	18.3(5.0)
breast-w	3.9(4.5)	3.9(4.5)	3.6(3.0)	3.1(1.0)	4.3(7.0)	3.3(2.0)	4.1(6.0)
sick	1.5(1.0)	1.6(3.0)	1.8(4.0)	6.1(7.0)	1.6(2.0)	1.8(5.0)	1.9(6.0)
tic-tac-toe	0.9(1.0)	4.2(3.0)	8.1(5.0)	19.0(7.0)	2.4(2.0)	12.2(6.0)	5.3(4.0)
spambase	5.2(4.0)	4.6(2.0)	4.6(1.0)	5.2(5.0)	5.9(7.0)	4.9(3.0)	5.9(6.0)
cylinder-bands	21.9(6.0)	18.7(3.0)	19.4(4.0)	15.4(1.0)	21.7(5.0)	16.5(2.0)	38.1(7.0)
kr-vs-kp	0.9(2.0)	0.9(3.0)	1.0(4.0)	3.5(7.0)	0.6(1.0)	3.1(6.0)	2.9(5.0)
avg. rank	3.38	2.98	2.68	3.5	5.3	4.45	5.73

Results

- Friedman test states that classifiers are **not** equally good.
- Post-hoc analysis: calculating the **critical difference** (CD) according to the Nemenyi statistics.
- $CD = 2.015$; algorithms with **difference** in **average ranks** more than 2.015 are **significantly** different.



Summary

- **ENDER** – a general framework for rule induction based on boosting with strong prediction power maintaining interpretability.
- Rule coverage can be implicitly controlled by minimization technique.
- Loss function and minimization technique does not significantly influences the accuracy.
- Proper regularization improves results significantly.
- Rule ensemble interpretation – still to do . . .