Knowledge-Based Systems 194 (2020) 105480

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

Original software publication

RuleKit: A comprehensive suite for rule-based learning☆

Adam Gudyś *, Marek Sikora, Łukasz Wróbel

Faculty of Automatic Control, Electronics and Computer Science, Silesian University of Technology, 44-100 Gliwice, Poland

ARTICLE INFO

Article history:
Received 20 September 2019
Received in revised form 3 January 2020
Accepted 4 January 2020
Available online 7 January 2020

Keywords:
Rule learning
Classification
Regression
Survival analysis
User-guided induction
Knowledge discovery

ABSTRACT

Rule-based models are often used for data analysis as they combine interpretability with predictive power. We present RuleKit, a versatile tool for rule learning. Based on a sequential covering induction algorithm, it is suitable for classification, regression, and survival problems. The presence of a user-guided induction facilitates verifying hypotheses concerning data dependencies which are expected or of interest. The powerful and flexible experimental environment allows straightforward investigation of different induction schemes. The analysis can be performed in batch mode, through RapidMiner plug-in, or R package. The software is available at GitHub (https://github.com/adaa-polsl/RuleKit) under GNU AGPL-3.0 license.

1. Introduction

Thanks to the combination of predictive and descriptive capabilities, rules have been applied in machine learning for decades. Amongst many rule induction strategies, sequential covering is one of the most popular [1]. It consists in iterative addition of rules explaining a part of the training set as long as all the examples are covered. As this approach leads to different models than those obtained by extracting rules from trees induced with a divide-and-conquer strategy [2], it is a valuable knowledge discovery tool.

2. Problems and background

In the previous research we confirmed the effectiveness of our variant of sequential covering strategy on dozens of data sets in classification, regression, and survival analysis [3,4]. We also showed a usefulness of user-guided induction, which allows introducing user's preferences or domain knowledge to the model [5]—feature particularly valuable in medical applications.

In spite of numerous advantages, relatively few sequential covering rule induction algorithms are available as ready-to-use software. The examples are CN2 [6] included in the Orange suite [7], AQ [8] implemented in Rseslib 3 [9], or RIPPER [10] and M5Rules [11] contained in Weka [12]. Other existing data mining packages, like SPMF [13], focus on inducing different kind of rules, e.g., association or sequential rules.

We present RuleKit, a comprehensive suite for training and evaluating rule-based data models. Equipped with multiple useful features like user-guided induction, it is the first tool suitable for classification, regression, and survival analysis problems. It additionally stands out from the competitors with handiness—beside batch experimental environment it can be integrated with RapidMiner and R.

3. Software framework

3.1. Software architecture

RuleKit consists of three modules: command line application, RapidMiner plug-in, and R package.

The first variant comes with an XML-based experimental environment. It facilitates automated investigation of various algorithm configurations over multiple data sets. Different experimental schemes are supported and tens of performance metrics are provided for model assessment.

The RapidMiner plug-in provides user with two operators: *RuleKit Generator* and *RuleKit Performance*. The former is a RapidMiner *learner* that induces various types of rule models. The latter extends the standard RM *Performance* operator and allows calculation of performance metrics as well as gathering model characteristics.

The R package offers a single `learn_rules` functions for training and applying a model. It returns a named list containing

☆ No author associated with this paper has disclosed any potential or pertinent conflicts which may be perceived to have impending conflict with this work. For full disclosure statements refer to https://doi.org/10.1016/j.knosys.2020.105480.

* Corresponding author.
E-mail address: adam.gudys@polsl.pl (A. Gudyś).

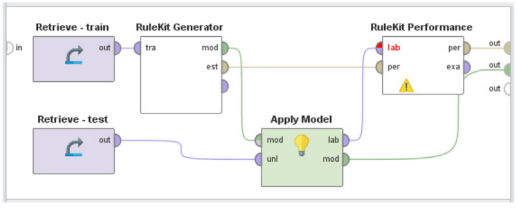
https://doi.org/10.1016/j.knosys.2020.105480

Table 1
Results of RuleKit and competing software averaged over selected data sets (each run in 10-fold CV): number of rules (#Rules), accuracy (Acc), balanced accuracy (BAcc), root relative squared error (RRSE), label vs prediction correlation (Corr). Standard deviations of indicators are also given. All algorithms were run with default parameters.

Classification (50 data sets)				Regression (47 data sets)			
Package	#Rules	Acc [%]	BAcc [%]	Package	#Rules	RRSE [%]	Corr [%]
RuleKit	45 ± 50	83.5 ± 11.3	77.9 ± 15.8	RuleKit	26 ± 20	79.5 ± 25.7	65.6 ± 24.7
RIPPER	10 ± 12	82.1 ± 12.3	76.6 ± 16.8	M5Rules	4 ± 4	42.0 ± 30.3	74.6 ± 29.2
CN2	79 ± 71	81.3 ± 12.9	75.1 ± 17.1				

```
<experiment>
<parameter_sets>
<parameter_set name="mincov=8, Entropy_User_C2">
   <param name="min_rule_covered">8</param>
   <param name="induction_measure">BinaryEntropy</param>
   <param name="pruning_measure">UserDefined</param>
   <param name="user_pruning_equation">2 * p / n</param>
   <param name="voting_measure">C2</param>
</parameter_set>
</parameter_sets>

<datasets>
<dataset>
   <label>Future Customer</label>
   <out_directory>./results-deals</out_directory>
   <training>
      <report_file>training.log</report_file>
      <train>
         <in_file>deals-train.arff</in_file>
         <model_file>deals.mdl</model_file>
      </train>
      ...
```

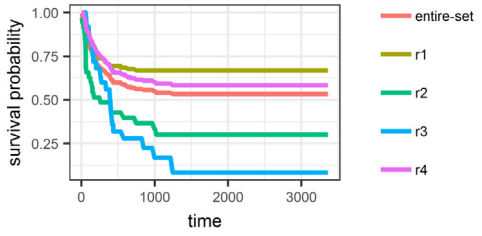
(a)



(b)

```
library(ggplot2); library(reshape2); library(rulekit)

formula <- survival::Surv(survival_time,survival_status) ~ .
control <- list(min_rule_covered = 5)
results <- rulekit::learn_rules(formula,control,bone_marrow)

# extract outputs:
rules <- results[["rules"]] # list of rules
cov <- results[["train-coverage"]] # coverage information
surv <- results[["estimator"]] # survival function estimates
perf <- results[["test-performance"]] # test performance

# melt data set for automatic plotting of multiple series
melted_surv <- reshape2::melt(surv, id.var="time")

# plot survival functions estimates
ggplot(melted_surv, aes(x=time, y=value, color=variable)) +
geom_line(size=1.0) +
xlab("time") + ylab("survival probability") +
theme_bw() + theme(legend.title=element_blank())
```

(c)



(d)

Fig. 1. RuleKit usage examples. (a) XML experiment description, (b) RapidMiner process, (c) R package code for analyzing *BMT-Ch* survival data, (d) visualization of survival estimates of generated rules.

induced rules, survival function estimates, test set performance metrics, etc.

RuleKit comes with a collection of data sets in ARFF format that represent classification, regression, and survival analysis problems.

3.2. Software functionalities

The following features make RuleKit a powerful data analysis tool:

(i) Suitability to classification, regression, and survival analysis. The processing of the latter follows our log rank-based LR-Rules algorithm [4].
(ii) Multiplicity of algorithm parameters. E.g., over 40 rule quality measures are available with an additional possibility to define own formulas.
(iii) User-guided induction according to GuideR algorithm [5]. The possibility to specify the initial set of rules, preferred and forbidden conditions/attributes, together with the multiplicity of options and modes allow suiting the model to user's requirements. This may be useful, e.g., in verifying hypotheses concerning data dependencies which are expected or of interest.

4. Implementation and empirical results

The suite is implemented in Java, therefore it is multiplatform. Independent steps of induction algorithms are distributed over multiple threads allowing RuleKit to take advantage of multi-core architectures. Bit-level parallelism is also employed for maximum computational performance. The documented API facilitates integration of the library with other projects and/or extending its functionality. All depending projects (RapidMiner, Renjin, MigLayout, JUnit, Gradle) are bundled in the package.

In Table 1 we present a comparison of RuleKit with competing software on selected classification and regression data sets (see GitHub site for complete results). Survival analysis problems as well as user-guided induction were investigated in details in our previous work [4,5].

In classification problems, RuleKit rule sets were most accurate, while RIPPER [10] produced most comprehensible models. This was due to the fact that RIPPER generates a decision list rather then an unordered set of rules. Note however, that the interpretability of the former is limited, as the rules cannot be considered independently. CN2 [6] was the last in terms of both indicators. AQ [8] failed to analyze all the data sets, thus it was not included in the table. For regression problems, RuleKit was inferior to M5Rules in terms of number of rules and the predictive power. This was expected, though, as M5Rules produces a decision list with linear models in rule consequences.

Table 2
Software metadata.

Nr.	Software metadata description	
S1	Current software version	1.1.0
S2	Permanent link to executables of this version	https://github.com/adaa-polsl/RuleKit/releases/tag/v1.1.0
S3	Legal Software License	GNU AGPL-3.0
S4	Computing platform/Operating System	Microsoft Windows, Linux, OS X
S5	Installation requirements & dependencies	Java Development Kit 11
S6	Link to user manual	Tutorial: https://github.com/adaa-polsl/RuleKit Full manual: https://github.com/adaa-polsl/RuleKit/wiki
S7	Support email for questions	adam.gudys@polsl.pl

Table 3
Code metadata.

Nr.	Code metadata description	
C1	Current code version	1.1.0
C2	Permanent link to code/repository used of this code version	https://github.com/adaa-polsl/RuleKit/releases/tag/v1.1.0
C3	Legal Code License	GNU AGPL-3.0
C4	Code versioning system used	git
C5	Software code languages, tools, and services used	Java, R, Gradle
C6	Compilation requirements, operating environments & dependencies	Java Development Kit 11
C7	Link to developer documentation/manual	Manual: https://github.com/adaa-polsl/RuleKit/wiki JavaDoc: https://adaa-polsl.github.io/RuleKit/
C8	Support email for questions	adam.gudys@polsl.pl

5. Illustrative examples

Batch mode. This example demonstrates running a RuleKit batch analysis on *deals* classification data set (prediction whether a person making a purchase will be a future customer). The batch mode is run with `java -jar RuleKit experiments.xml` command, where XML file describes parameter sets and data sets to be investigated (Fig. 1a).

As a result of the training, a text report is produced. It contains a list of generated rules (with corresponding confusion matrices and statistical significance), information about examples coverage, model characteristics (no. of rules/conditions, average rule precision/coverage, etc.), and performance metrics calculated on the training set (accuracy, error, etc.). Depending on the problem, the significance of rules is established with different tests (Fisher's exact, χ^2, or log-rank). The training may be followed by applying the model on unlabeled data. In this stage, a comma-separated table with values of performance metrics evaluated on the test set is produced.

RapidMiner plug-in. An alternative way of performing an experiment is integrating RuleKit with RapidMiner. In the Fig. 1b we present an example RM process which performs analysis on the *deals* data set.

R package. As a last test case, we present the application of RuleKit R package for analyzing factors contributing to the patients' survival following bone marrow transplants. The corresponding data set (*BMT-Ch*) is integrated with the package in the form of the standard R data frame. In Fig. 1c–d we provide an example R code for training a model and a visualization of corresponding survival functions estimates.

6. Conclusions

We demonstrated that RuleKit can be successfully applied for training and evaluation of rule-based models in classification, regression, and survival tasks. The multiplicity of options and modes together with the powerful and flexible experimental environment makes presented suite a useful tool for data analysis and knowledge discovery. Possible extensions of RuleKit include improved handling of multi-class imbalanced problems [14] or induction of action rules [15] and oblique rules [16]. The applicability of the suite could be additionally enhanced by providing Python wrapper or standalone graphical interface.

Acknowledgments

This work was supported by The National Centre for Research and Development, Poland within the programme Prevention Practices and Treatment of Civilization Diseases—STRATEGMED III grant no. STRATEGMED3/304586/5/NCBR/2017 and Faculty of Automatic Control, Electronics and Computer Science at Silesian University of Technology within the statutory research projects BK-204/RAU2/2019 and BKM-576/RAU2/2019.

Appendix. Required metadata

Current executable software version
See Table 2.

Current code version
See Table 3.

References

[1] J. Fürnkranz, D. Gamberger, N. Lavrač, Foundations of Rule Learning, Springer-Verlag, Berlin, Heidelberg, 2012.
[2] L. Breiman, J.H. Friedman, R.A. Olshen, et al., Classification and Regression Trees, Chapman & Hall/CRC, Boca Raton, London, 1984.
[3] Ł. Wróbel, M. Sikora, M. Michalak, Rule quality measures settings in classification, regression and survival rule induction—an empirical approach, Fund. Inform. 149 (4) (2016) 419–449.
[4] Ł. Wróbel, A. Gudyś, M. Sikora, Learning rule sets from survival data, BMC Bioinformatics 18 (1) (2017) 285.
[5] M. Sikora, Ł. Wróbel, A. Gudyś, GuideR: A guided separate-and-conquer rule learning in classification, regression, and survival settings, Knowl.-Based Syst. 173 (2019) 1–14.
[6] P. Clark, T. Niblett, The CN2 induction algorithm, Mach. Learn. 3 (4) (1989) 261–283.
[7] J. Demšar, T. Curk, A. Erjavec, et al., Orange: Data mining toolbox in python, J. Mach. Learn. Res. 14 (1) (2013) 2349–2353.
[8] R.S. Michalski, On the quasi-minimal solution of the general covering problem, in: V International Symposium on Information Processing, Vol. A3, 1969, pp. 125–128.
[9] A. Wojna, R. Latkowski, Rseslib 3: library of rough set and machine learning methods with extensible architecture, in: Transactions on Rough Sets XXI, Springer, Berlin, Heidelberg, 2019, pp. 301–323.
[10] W.W. Cohen, Fast effective rule induction, in: 12th International Conference on Machine Learning, Morgan Kaufmann, 1995, pp. 115–123.

[11] G. Holmes, M. Hall, E. Frank, Generating rule sets from model trees, in: Advanced Topics in Artificial Intelligence, Springer, Berlin, Heidelberg, 1999, pp. 1–12.
[12] I.H. Witten, E. Frank, M.A. Hall, et al., Data Mining: Practical Machine Learning Tools and Techniques, fourth ed., Morgan Kaufmann, San Francisco, 2016.
[13] P. Fournier-Viger, J.C.-W. Lin, A. Gomariz, et al., The SPMF open-source data mining library version 2, in: Machine Learning and Knowledge Discovery in Databases, Springer, Cham, 2016, pp. 36–40.
[14] C. Zhang, J. Bi, S. Xu, et al., Multi-Imbalance: An open-source software for multi-class imbalance learning, Knowl.-Based Syst. 174 (2019) 137–143.
[15] A. Hajja, Z.W. Ras, A. Wieczorkowska, Hierarchical object-driven action rules, J. Intell. Inf. Syst. 42 (2) (2014) 207–232.
[16] M. Sikora, A. Gudyś, CHIRA—Convex hull based iterative algorithm of rules aggregation, Fund. Inform. 123 (2) (2013) 143–170.