RuleMiner: An Interactive Web Tool for Rule-Based Data Analysis

Marek Sikora[1,2,*], Dawid Macha[1,2,*], Joanna Badura[1,*], Artur Kozłowski[1] and Łukasz Wróbel[1,2]

[1] Łukasiewicz Research Network – Institute of Innovative Technologies EMAG, ul. Leopolda 31, 40-189 Katowice, Poland
[2] Department of Computer Networks and Systems, Silesian University of Technology, ul. Akademicka 16, 44-100 Gliwice, Poland

Abstract

Rule induction is a powerful tool for data mining. It can be used to create understandable prediction models that are presented as a set of rules. Because of their interpretability, they can be used and understood by a wide range of users – from data science experts to professionals in other fields who want to analyse their data. For this reason, it is beneficial to offer systems for rule induction with a graphical user interface (GUI) that would be usable also for users without programming skills. This article proposes a RuleMiner – a new web service for rule induction and rule-based data analysis. RuleMiner application is a development of the original RuleKit library co-developed by the authors. The platform allows users to run RuleKit algorithms for classification, regression, and survival rules at the GUI level without requiring programming knowledge. It also offers the functionalities to work with sets of rules and datasets, visualise the results, perform user-driven rule induction, analyse the created models, explore the coverage of individual rules and examples, analyse models' predictive capabilities, and perform predictions for new examples. The RuleMiner platform is publicly available. The article presents the related work, shows RuleKit comparison with selected algorithms, highlights the most important RuleMiner functionalities, and provides an illustrative example of the proposed system's usage.

Keywords

rule induction, data mining application, RuleKit

1. Introduction

The article presents a RuleMiner web service for rule induction and rule-based data analysis as the development of the RuleKit library [1]. The RuleKit includes classification, regression, and survival rule induction algorithms developed by our team. Based on this library, we have also developed algorithms for action rule induction [2], rule-based expalinability [3] and user-driven rule induction [4].

Until now, the RuleKit algorithms were available only to people who were proficient in programming, or it required our involvement in research, e.g. as in [5]. Thus, the motivation behind creating the RuleMiner application was to share a wide range of tools for knowledge discovery, data mining, and generating rule-based prediction systems accessible to a broad audience. We are particularly interested in making the RuleMiner application beneficial to people from various scientific fields who want to use rule systems to analyse their own datasets. RuleMiner allows users to run RuleKit algorithms from the GUI level (thus without programming knowledge) and analyse quantitative results in a user-friendly form. It also allows users to work with rulesets and datasets – the user can edit them, filter them, run user-driven induction, visualise results, analyse coverage of individual rules and examples, and more.

In this article, we show the effectiveness of our computation kernel (RuleKit) and compare it with other methods. The article also presents the most important functionalities and shows an illustrative example of system usage. All information about the proposed application can be found on the RuleMiner website at https://ruleminer.ai/ and on our GitHub page at https://github.com/ruleminer/ruleminer. The

RuleML+RR'24: Companion Proceedings of the 8th International Joint Conference on Rules and Reasoning, September 16–22, 2024, Bucharest, Romania

*Corresponding author.
marek.sikora@polsl.pl (M. Sikora); dawid.macha@emag.lukasiewicz.gov.pl (D. Macha); joanna.badura@emag.lukasiewicz.gov.pl (J. Badura)
0000-0002-2393-9761 (M. Sikora); 0000-0001-5673-3439 (D. Macha); 0000-0001-7821-1375 (J. Badura); 0000-0003-1195-5198 (A. Kozłowski); 0000-0002-5715-6239 (Ł. Wróbel)

RuleMiner application is available at https://app.ruleminer.ai/. Additional research results can be found in the report available also at our GitHub repository[1]. In accordance with good research practice, we have also made public the detailed numerical results of our research[2] and the datasets[3].

2. Related work

Rule induction is used for tasks of knowledge discovery and, because of its interpretability, for explainability (XAI) tasks [6]. There are multiple approaches to rule induction. A popular one is the separate-and-conquer strategy (SoC), also known as the sequential covering approach. This strategy was proposed by Michalski [7] and includes methods for generating a ruleset or a rules list. An extensive discussion of the covering approach and a review of SoC-based rule induction algorithms was presented by Funkranz [8] and can be found in book [9]. SoC is still used to define new rule induction algorithms, particularly in regression problems [1, 10], action rule induction [2], and survival rule induction [11]. Related approaches are methods that connect separate-and-conquer strategy with rough set theory (RST) [12]. There are also methods that look for locally optimal rules for a specific observation from a training dataset [13, 14]. The next group of algorithms are the methods that are based on ensemble learning [15]. They can use boosting [16, 17] or bagging [18] techniques. Rule induction can also involve optimising the entire ruleset based on a given loss function [19, 20]. The literature also describes simpler examples of rule inductions, e.g., the OneR algorithm [21], that generates a set of rules that test only one particular attribute. Some of the mentioned methods are suitable for solving both classification and, with some adjustments, regression problems.

2.1. Algorithms and implementations

There are numerous implementations of rule-based algorithms and data mining programming libraries with rule-based methods. The given section presents a selection of them to demonstrate the currently available tools.

Rule induction can involve different types of rules, but the most common ones are association rules and rules with conclusions created based on one attribute. In the latter case, we can generate classification rules (that is, decision rules) or regression rules. It is also possible to create survival rules that, in conclusion, have an estimator of the survival function.

The association rules can be generated in numerous software, e.g. in Orange library [22] or Weka package [23]. These libraries are general data mining purpose libraries, and the association rules are only one of many available algorithms. The association rules can also be generated by programming libraries e.g. Python apyori library (https://github.com/ymoch/apyori, last accessed 2024/06/07) or R package arules [24].

The second group of software includes tools that implement algorithms for generating classification, regression, or survival rules. Below, we present well-recognised algorithms and the methods that we used in experiments, which will be presented in further sections of the article. The exemplary algorithms for rule induction and their implementations are as follows:

- AQ15 classification algorithm that is available in Rseslib 3 library [25].
- BOOMER [26] algorithm for learning ensembles of gradient-boosted multi-label classification rules.
- BRCG [19] algorithm for classification rule induction for binary datasets, with implementation available in AIX360 package [27].
- CN2 algorithm [28] for generating classification rules. Its implementations can be found in Orange library [22].

[1] https://github.com/ruleminer/ruleminer/blob/main/reports/comparing_RuleKit_with_other_methods/classification_summary.md
[2] https://github.com/ruleminer/ruleminer/tree/main/reports/comparing_RuleKit_with_other_methods/results
[3] https://github.com/ruleminer/datasets/tree/main/classification

- Interpretable Decision Sets (IDS) algorithm [20]. The Python package pyIDS [29] offers the implementation of the algorithm.
- JRip algorithm [30], also called RIPPER. It is implemented in Weka package [23], and it is also available in Altair AI Studio (previously RapidMiner[4]) and AIX360 package [27].
- LORD [14]. It is an algorithm for learning locally optimal classification rules.
- M5Rules algorithm [31] implemented in Weka package [23] for regression problems; however, it generates a ruleset based on trees.
- MLRules [16] available in Weka package [23] for classification problems.
- OneR [21]. Its implementation can be found in the Weka package [23].
- RuleFit algorithm [15] for generating classification rules for data with binary labels. The implementation is available in Python imodels library [32]. It can also be used for regression datasets.
- RuleKit [1]. It offers the induction of classification, regression and survival rules. Its implementation is available in the GitHub repository at https://github.com/adaa-polsl/RuleKit.

There is also available software that integrates numerous rule-based algorithms. Rseslib 3 library [25] implements the AQ15 algorithm and numerous methods for generating decision rules from reducts. In the KEEL software library [33], there are available multiple rule learning algorithms for classification, e.g. AQ15, CN2, OneR, RIPPER and others. Several rule-based algorithms can also be found in the already mentioned libraries: Weka [23], imodels [32], and AIX360 [27]; however, these packages are not solely focused on rule induction. There are also available frameworks for creating rule learners. The example can be SeCo Framework (documentation can be found at https://ke-tud.github.io/resources/SeCo.html, last accessed 2024/06/07).

As mentioned before, this is only a selection of available algorithms and their implementations. However, there is a visible dominance of algorithms for the induction of classification rules. There are few algorithms for the induction of regression rules, and only RuleKit offers the algorithm for generating rulesets for censored datasets. We can conclude that the RuleKit tool is the most versatile in terms of the data for which it can perform rule induction (classification problems with binary labels and for problems with multiple classes, regression data and survival datasets).

2.2. Visual tools for rule induction

On the market, we can find programmes offering a graphical user interface (GUI) for performing data mining and predictions using rules. These programmes have the advantage of being accessible not only to users experienced in programming languages who can write their own programs.

Our RuleMiner system focuses on inducing classification, regression, and survival rules. Therefore, our overview of available GUI applications for creating rule-based models only covers these types of inductions.

Orange library [22] offers an open-source GUI application that can be used for many data mining tasks. In this application, multiple widgets are available, e.g. for loading data, visualisations, and modelling. They can be moved and connected with each other to create the data mining process. Orange uses the CN2 algorithm to induct classification rules. It offers a simple viewer for rules with basic statistics, such as the length and quality of each rule. Similar systems are already mentioned Altair AI Studio (formerly Altair RapidMiner), KEEL [33] and KNIME [34]. The Altair AI studio and KEEL give the widgets to generate classification rule-based models. The KNIME system has an implemented method for fuzzy rule learners, but these rules are outside this overview's scope. KNIME also allows the creation of user-defined rules that can be used to label new datasets.

Another system is Weka [23] – machine learning software that provides an interface for analysing datasets. As was mentioned in Section 2.1, Weka can generate classification and regression rules by multiple algorithms, which may be the advantage of this tool. After the induction process, Weka gives

[4] https://altair.com/altair-rapidminer, last accessed 2024/06/07

an overview of the generated rules and provides basic statistics of the ruleset. Weka also offers a KnowledgeFlow tool that can be used to create data mining processes.

A very simple tool is QMAK [35], which is a part of Rseslib 3, and it uses the Rseslib 3 library as the source of classification models. QMAK provides only simple functionality for generating and applying classification rules to a dataset.

The system similar in some aspects to RuleMiner is a service EasyMiner [36]. It can be used to create classification models based on association rules. The user can load the dataset and modify the attributes. Then, based on the provided association rule pattern, the system creates a ruleset. In the article [37], the authors also present the functionalities for the edition of rule-based models, which distinguishes the system from other tools described before. The system also provides measures of predictive model quality.

Most of the tools discussed (Orange, KNIME, KEEL, Weka, QMAK) are not solely focused on rule induction, and because of this, they do not provide advanced functionalities for manipulating rule-based models and results. These five tools are also all desktop applications. It means that the datasets and analysis are unavailable to the user when using different devices. Only the EasyMiner system, which is focused on applying rule induction to data, provides interesting functionalities specific to rule-based models, such as adding, editing, removing rules, attribute modification or creating new rules. When applying the changes, the user can also interactively validate the model's predictive performance. EasyMiner is an online tool, so the program does not need to be configured on the user's hardware. The disadvantage of the EasyMiner system is the support of only association and classification rule induction.

3. RuleMiner

Considering available programmes for rule induction, we propose RuleMiner – an online system for rule-based data mining. It is based on our original software for rule induction called RuleKit [1]. It is acknowledged in the research field, as evidenced by numerous publications in reputable journals [4, 3]. Additionally, it is constantly developed, and new software is created based on it.

RuleMiner is an analytical tool that allows users to work with data using rules. It can be used to work with classification, regression, and survival data. RuleMiner offers the possibility to predict data using rulesets, evaluate the quality of rules and rulesets, create rules based on the user's specific requirements (user-driven rule induction), visualise results, compare rulesets with syntactic and semantic rule similarity and generate reports.

RuleMiner is a web application that does not require programming knowledge. Thanks to this, it is available to many users. Projects can be accessed from many devices, and there is no need to configure the application on the user's hardware. Due to the application's web-based nature, the imported data are stored in a centralised database. Users are obligated not to enter personal data, unanonymised sensitive data, or legally protected information into the application. However, those who wish to work with sensitive, protected, or proprietary data can still do so by applying data anonymization or pseudonymization techniques before uploading their datasets. These processes help ensure that the data is altered in a way that prevents the identification of individuals or the exposure of confidential information. It is important to note that even with these precautions, the service provider is not liable for the stored data. The latest versions of the terms of service and details on each user's available resources under different plans can be accessed on the following website: https://ruleminer.ai/.

3.1. RuleKit algorithms and implementations

RuleKit is a computational kernel of the RuleMiner application. It is implemented in Java and is open-source, available at https://github.com/adaa-polsl/RuleKit. In RuleMiner, a RuleKit python wrapper was used (available at https://github.com/adaa-polsl/RuleKit-python). RuleKit is suitable for classification, regression, and survival problems. However, different software was also created based on it. RuleKit and RuleKit-related software enable rule induction, rule filtering, user-driven rule induction, contrast

set mining, action rule mining, induction of rules with complex and M-of-N conditions, and evaluation of elementary condition importance. The full list of software and publications related to RuleKit and RuleMiner can be found on our GitHub repository (https://github.com/ruleminer/ruleminer) and RuleMiner website (https://ruleminer.ai/publications/). The article discusses the first version of RuleMiner, which does not yet implement all the functionality available in RuleKit-related software. Our ambition, however, is that the RuleMiner application will provide more possibilities for using RuleKit-related algorithms in the future.

3.1.1. Comparing RuleKit with other algorithms

Most of the available data mining and rule induction software, described in section 2.1, offers rule induction for classification datasets. Because of this, we compared RuleKit with other methods in the context of classification rule induction.

For the RuleKit comparison, we have chosen well-known separate-and-conquer methods (AQ15 [25], CN2 [28], JRip [30]) but also methods representing different approaches to rule induction (MLRules [16], RuleFit [15], LORD [14], BRCG [19], IDS [20], OneR [21]), all with available implementations. The algorithms were mostly run with their default parameters. CN2 algorithm, which is similar to RuleKit, was run with two versions of parameters: default parameters (denoted below as *CN2_default*) and parameters that match the default RuleKit parameters (denoted as *CN2_v2*). RuleKit was launched with default parameters and with two measures – C2 (denoted as *RuleKit_C2*) and Correlation (*RuleKit_Corr*). The experiments were performed based on 10-fold cross-validations on 50 smaller datasets (most of them come from the UCI and OpenML repositories) and 9 datasets that we classified as bigger datasets (up to 300,000 rows in a dataset). The analyses were performed separately for binary and multi-label classification datasets and separately for smaller and bigger datasets. As the comparison metrics, we have chosen classification accuracy (denoted as ACC) on test and train datasets, balanced accuracy (BAcc) on test and train datasets, number of rules, number of conditions and computation time.

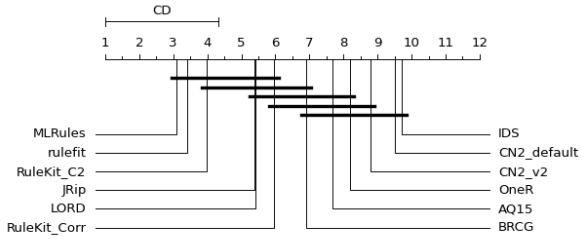
This article presents only the results of selected metrics due to the limited article space. However, the extensive experiments report is available on our GitHub page[5]. The report also provides detailed information about the implementations used, the parameters of each algorithm and information about available numerical results in CSV files and the datasets on which the analyses were performed. The results presented are from the current report as of June 7, 2024. However, we intend to update the report with additional results from other algorithms in the future. We are also open to conducting experiments using other implementations of rule-based methods, which can be submitted to us.

The obtained results are presented using Critical Difference (CD) diagrams [38]. In the article, we present CD diagrams (see Fig. 1) for ACC calculated on test datasets for smaller binary and multi-label datasets, BAcc calculated on test datasets for bigger binary and multi-label datasets, the computational time for smaller multi-label datasets and computational time for bigger binary datasets. The LORD algorithm was not included in the balanced accuracy rankings because the implementation did not output BAcc metric results.

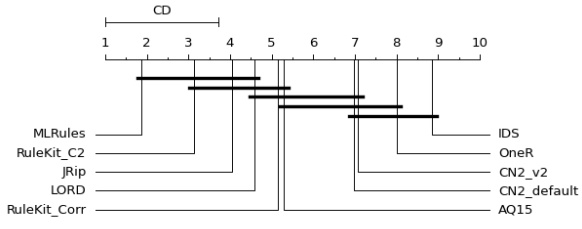
The experiments showed that RuleKit is a good base solution for our RuleMiner platform. It performs well, especially against other coverage algorithms. The basic form of the method can generate good rule classifiers, and the created models have good predictive abilities. The flaw of the RuleKit basic version is the computation time, which puts it in the middle of the compared algorithms but ahead of standard coverage algorithms. To reduce computational time, the number of generated rules could be limited – RuleKit and RuleMiner can adjust the number of generated rules (also presented in the extensive report). We also conclude that RuleKit is a good computation kernel of the RuleMiner platform because it is the only algorithm that has the ability to generate survival rules.

MLRules and LORD are RuleKit's biggest competitors – MLRules because of its prediction accuracy and LORD because of its computation time. Considering this, we plan to add to RuleMiner the possibility

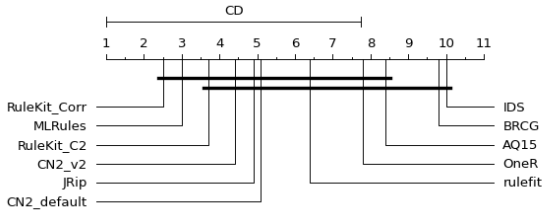
[5] https://github.com/ruleminer/ruleminer/blob/main/reports/comparing_RuleKit_with_other_methods/classification_summary.md



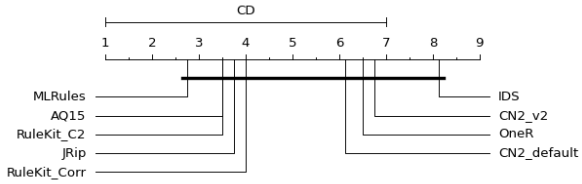
(a) ACC for smaller binary datasets.



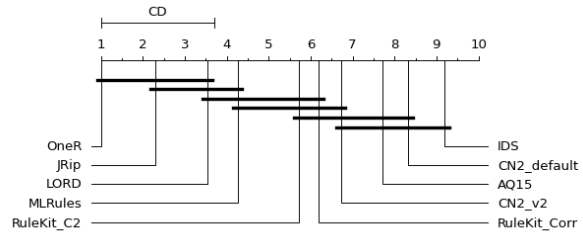
(b) ACC for smaller multi-label datasets.



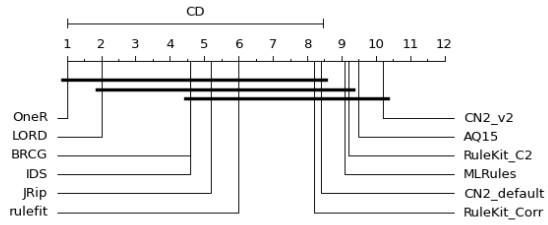
(c) BAcc for bigger binary datasets.



(d) BAcc for bigger multi-label datasets.



(e) Calc. time - smaller multi-label datasets.



(f) Calc. time - bigger binary datasets.

Figure 1: The Critical Difference diagrams for carried out experiments.

of including rules generated by these algorithms and implement the classification mechanism that these algorithms use. In our opinion, the LORD algorithm would also benefit from using a filtration approach.

3.2. Service functionalities

RuleMiner application offers many functionalities for data mining using rule induction, and it was designed to be user-friendly. The user can create projects, load and analyse datasets and induce rules for classification, regression and survival data. It not only offers the functionalities to run rule induction algorithms and explore the results (e.g., view the created rulesets, view characteristics of single rules and a whole ruleset), but it also allows the user to connect different rulesets, filter rules, manually edit them, and combine rules that were created in different ways (generated automatically, created manually, and induced using a user-driven approach). Thanks to this, the user has a wide range of functionalities to create the rule-based model that describes a dataset well, has good predictive capabilities, and includes rules relevant from the user's point of view. The changes can be verified on an ongoing basis, and their effect on the quality of the ruleset and the predictive capabilities can be monitored. The main functionalities groups available in the RuleMiner application are presented in Fig. 2, and the full description of functionalities is available in the documentation at our GitHub Wiki page (https://github.com/ruleminer/ruleminer/wiki).

The RuleMiner system offers functionality for generating reports, as it is shown in Fig. 2. This is a non-interactive part of the system, but it can be used to compare the results of different machine learning methods, not only rule-based ones. What is also worth noting is that RuleMiner offers a simple expert system that automatically configures the parameters of the rule induction algorithm based on the user's answers to defined questions. This allows the user to adjust the induction process to their

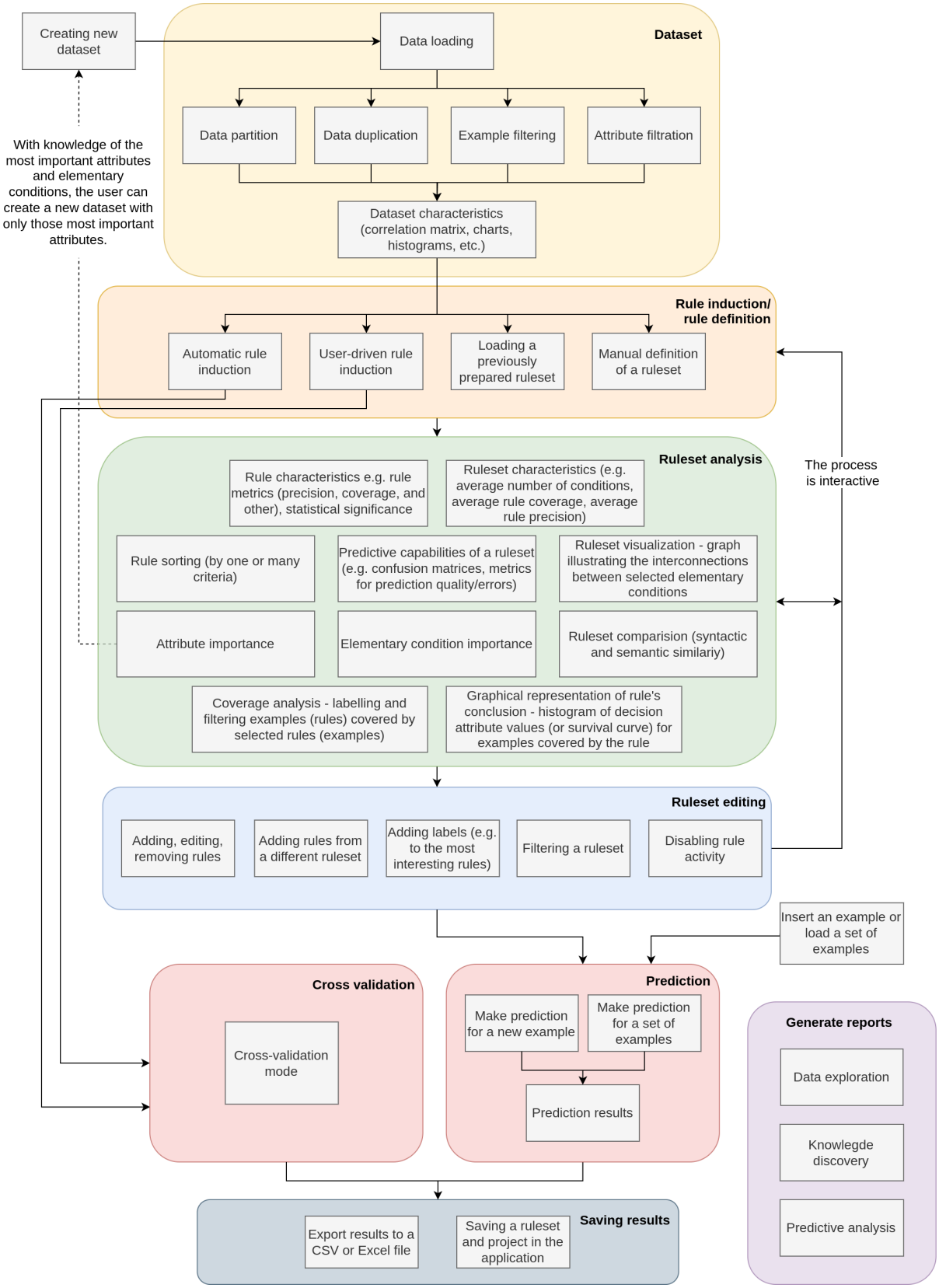


Figure 2: The most important functionalities available in the RuleMiner application and presentation of the proposed usage scenario.

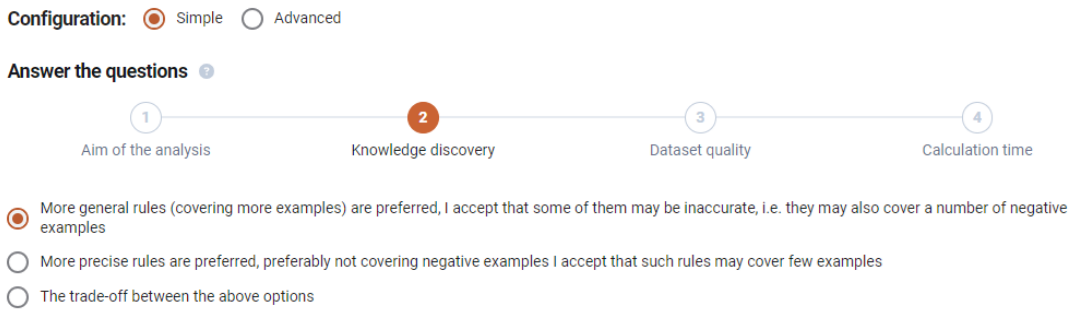


Figure 3: One of the stages of automatic rule induction configuration.

needs without exploring algorithm parameters. One of the questionnaire's windows is shown in Fig. 3.

3.3. Illustrative example

The illustrative example is based on a classification problem and a publicly available dataset that describes patients potentially infected with SARS-CoV-2 [39]. The goal was to diagnose patients based on questionnaires they filled out in the hospital. In this example, we are not focusing on analyzing the dataset but on demonstrating some of RuleMiner's functions based on this dataset.

Fig. 4 shows the RuleMiner main window with a generated ruleset sorted by three criteria. Users can analyse, sort and label the rules, evaluate the prediction quality and see the importance of attributes and conditions based on the generated ruleset [3] (Fig. 5 presents condition importance for one class).

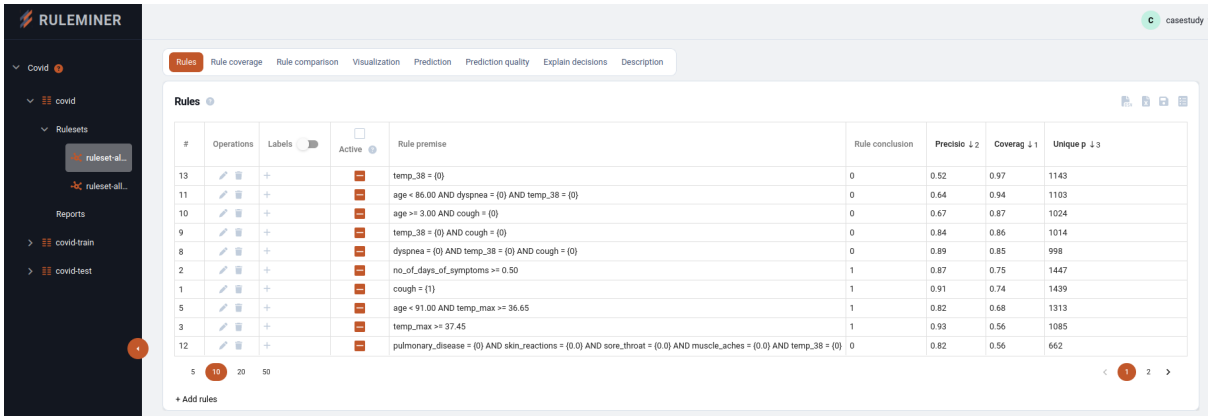


Figure 4: RuleMiner rules window.

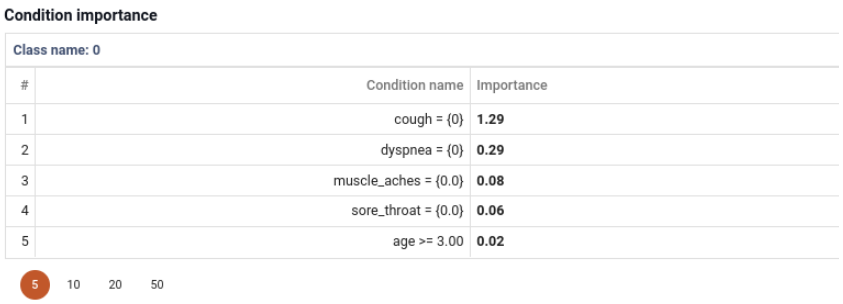


Figure 5: RuleMiner window with condition importance.

Then, it is possible to analyse the quality of predictions calculated on the train and test dataset (see Fig. 6).

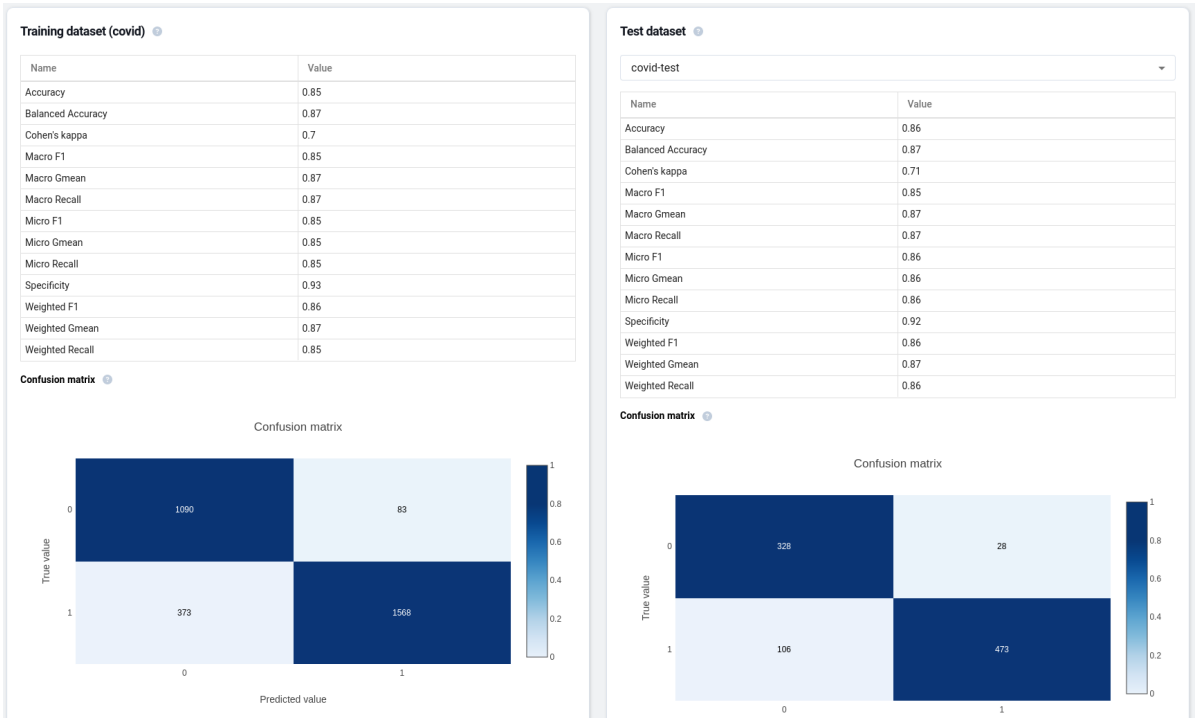


Figure 6: RuleMiner window with information about predictive capabilities.

Fig. 7 presents the analysis of coverage of examples by the two selected rules. In the presented case, the option *AND* was selected, so it means that the RuleMiner shows only the examples that are covered by both rules. Other analyses can also be performed.

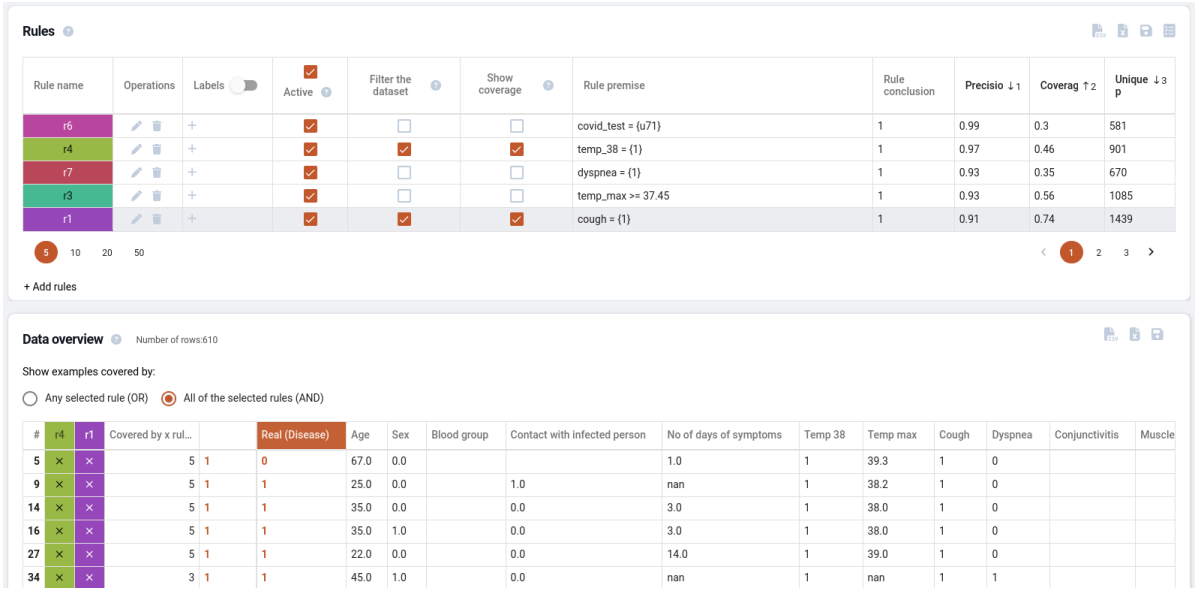


Figure 7: RuleMiner coverage window.

Finally, users can visualise how the rules from the ruleset are connected with each other, particularly how the elementary conditions are linked. This is demonstrated in Fig. 8. It's worth noting that when users select different conditions – not necessarily from the same rule – the application provides information about the hypothetical rule created in this manner (shown on the right side of Fig. 8).

The presented RulMiner windows represent only a small portion of the functionalities available in the RuleMiner application. However, they demonstrate the capabilities of the proposed system and its

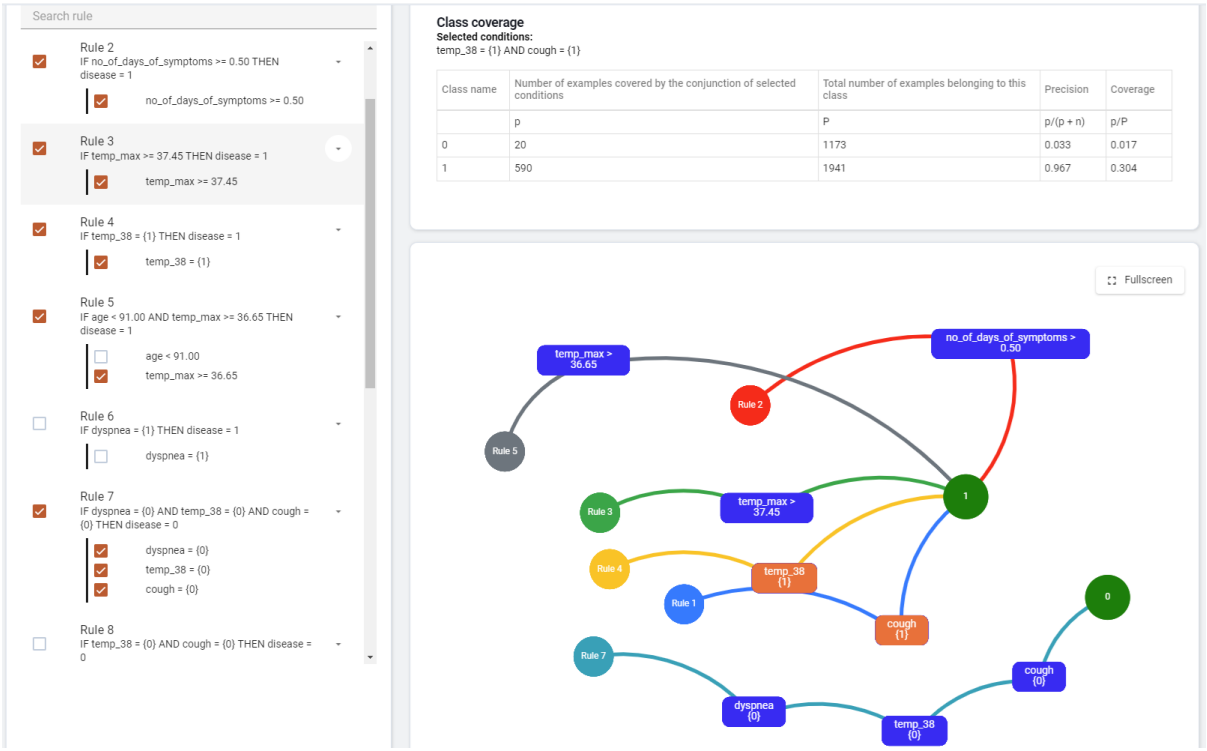


Figure 8: RuleMiner ruleset visualisation.

usability for analysing datasets using rulesets. Other available functionalities are presented in Fig. 2.

4. Conclusions and future work

We propose a RuleMiner web application for data analysis using rules. This system is a development of the RuleKit software, which has proven to be effective enough compared to other available solutions, as demonstrated by experimental results and a comprehensive report[6]. To ensure reproducibility, we have provided numerical results and datasets for other users to utilise in their experiments. In the future, we hope to enhance the comparison report by including outcomes from other methods that authors may submit to us.

When compared with other GUI tools for rule induction, RuleMiner has several advantages. First of all, it is the only visual tool for rule induction that is designed to enhance the capabilities of rule-based data analysis for three types of problems: classification, regression, and survival. In contrast, most other applications limit their functionality to a single type of rule induction, mostly classification rules.

RuleMiner's distinct advantage lies in its flexible and user-centric approach to rule creation. Users can use automatic rule induction in two variants: with parameters set based on a survey of the user's needs or by manually fine-tuning the parameters to suit specific requirements. The user can also load a previously prepared ruleset or manually define all of the rules. What truly sets RuleMiner apart, however, is its support for user-driven rule induction – users can define, e.g., what conditions or attributes should be forbidden and which should be preferred by the algorithm. Thanks to this, the algorithm can incorporate the user's unique domain expertise but still enables the discovery of novel rules that may be unknown to the user.

Because the RuleMiner application focuses solely on rule-based data analysis, it offers many functionalities for ruleset analysis that are not available in software where rule induction is merely one of many functionalities. Besides standard ruleset and rules metric presentation, it also gives insights

[6]https://github.com/ruleminer/ruleminer/blob/main/reports/comparing_RuleKit_with_other_methods/classification_summary.md

into the importance of attributes and conditions, ruleset graph-based visualizations, a visual tool for coverage analysis, and analysis of predictive capabilities, among other features.

An important advantage of RuleMiner is also the whole spectrum of functionalities for editing rulesets. Users can easily modify, add, disable, or remove rules, akin to the capabilities found in EasyMiner. Additionally, in RuleMiner, the user can combine rules from different rulesets, label rules, and filter rules by specific conditions. Importantly, all analyses available for initially generated rulesets can also be applied to any modified ruleset, offering flexibility in rule-based data analysis.

To conclude, the RuleMiner application offers not only a GUI for running RuleKit algorithms, functionalities to analyze induced rules or using created model for predictions but also gives the user the whole spectrum of tools to interact with created rules and personalise the induction process. Thanks to this, users can use their specific domain knowledge, experiment with ruleset modifications, and work with the system interactively. We believe that this high level of user engagement and customization is the significant advantage of the RuleMiner system.

The current version of the application has several useful functions that enable a complete rule-based data analysis process, but there is still room to add more. Right now, RuleMiner offers induction for classification, regression, and survival rules, but there are more RuleKit-based algorithms that could be included in the RuleMiner system in the future. For example, currently, there is no available induction of contrast sets or action rules that we want to implement into the GUI application. RuleMiner will be further developed both to provide the user with more tools for working with rules and data and to expand the number of available algorithms. If possible, we also want to add to the RuleMiner application import of MLRules and LORD rulesets and their classification mechanics, as these methods are the most promising in terms of the quality of generated rulesets and calculation time on large datasets. In the future, there will also be the functionality to load rulesets from any system, as long as the rules are in the specified format.

The RuleMiner application is currently available at https://app.ruleminer.ai/ and is in a stable version with ongoing development. The capacity for analysing datasets will expand in the future as computing resources increase. Any reported bugs are being fixed, and the application is undergoing continuous testing. We believe that the proposed RuleMiner application will be a valuable tool for analysing data using rules, serving a wide range of users.

References

[1] A. Gudyś, M. Sikora, Ł. Wróbel, RuleKit: A comprehensive suite for rule-based learning, Knowledge-Based Systems 194 (2020) 105480. doi:10.1016/j.knosys.2020.105480.

[2] M. Sikora, P. Matyszok, Ł. Wróbel, SCARI: Separate and conquer algorithm for action rules and recommendations induction, Information Sciences 607 (2022) 849–868. doi:10.1016/j.ins.2022.06.026.

[3] D. Macha, M. Kozielski, Ł. Wróbel, M. Sikora, RuleXAI—A package for rule-based explanations of machine learning model, SoftwareX 20 (2022). doi:10.1016/j.softx.2022.101209.

[4] M. Sikora, Ł. Wróbel, A. Gudyś, GuideR: A guided separate-and-conquer rule learning in classification, regression, and survival settings, Knowledge-Based Systems 173 (2019) 1–14. doi:10.1016/j.knosys.2019.02.019.

[5] J. Kulis, Ł. Wawrowski, Ł. Sędek, Ł. Wróbel, Ł. Słota, V. H. van der Velden, T. Szczepański, M. Sikora, Machine learning based analysis of relations between antigen expression and genetic aberrations in childhood B-cell precursor acute lymphoblastic leukaemia, Journal of Clinical Medicine 11 (2022) 2281.

[6] M. T. Ribeiro, S. Singh, C. Guestrin, Anchors: High-precision model-agnostic explanations, in: Proceedings of the AAAI Conference on Artificial Intelligence, volume 32, 2018.

[7] R. Michalski, On the Quasi-Minimal Solution of the Covering Problem, in: Proceedings of the V. International Symposium on Information Processing (FCIP), volume 3, Bled, Yugoslavia, 1969, pp. 123–125.

[8] J. Fürnkranz, Separate-and-Conquer Rule Learning, Artificial Intelligence Review 13 (1999) 3–54. doi:10.1023/A:1006524209794.
[9] J. Fürnkranz, D. Gamberger, N. Lavrač, Foundations of Rule Learning, Springer Science & Business Media, 2012.
[10] F. Janssen, J. Fürnkranz, Heuristic rule-based regression via dynamic reduction to classification, in: Twenty-Second International Joint Conference on Artificial Intelligence, Citeseer, 2011.
[11] Ł. Wróbel, A. Gudyś, M. Sikora, Learning rule sets from survival data, BMC Bioinformatics 18 (2017) 285. doi:10.1186/s12859-017-1693-x.
[12] J. Błaszczyński, R. Słowiński, M. Szeląg, Sequential covering rule induction algorithm for variable consistency rough set approaches, Information Sciences 181 (2011) 987–1002. doi:10.1016/j.ins.2010.10.030.
[13] J. Wang, G. Karypis, On mining instance-centric classification rules, IEEE Transactions on Knowledge and Data Engineering 18 (2006) 1497–1511. doi:10.1109/TKDE.2006.179.
[14] V. Q. P. Huynh, J. Fürnkranz, F. Beck, Efficient learning of large sets of locally optimal classification rules, Machine Learning 112 (2023) 571–610. doi:10.1007/s10994-022-06290-w.
[15] J. Friedman, B. Popescu, Predictive Learning via Rule Ensembles, The Annals of Applied Statistics 2 (2008). doi:10.1214/07-AOAS148.
[16] K. Dembczyński, W. Kotłowski, R. Słowiński, Maximum likelihood rule ensembles, in: Proceedings of the 25th International Conference on Machine Learning - ICML '08, ACM Press, Helsinki, Finland, 2008, pp. 224–231.
[17] K. Dembczyński, W. Kotłowski, R. Słowiński, ENDER: A statistical framework for boosting decision rules, Data Mining and Knowledge Discovery 21 (2010) 52–90. doi:10.1007/s10618-010-0177-7.
[18] J. Stefanowski, The bagging and n 2-classifiers based on rules induced by MODLEM, in: Proceedings of the International Conference on Rough Sets and Current Trends in Computing, Uppsala, Sweden, 2004, pp. 488–497.
[19] S. Dash, O. Gunluk, D. Wei, Boolean Decision Rules via Column Generation, in: Advances in Neural Information Processing Systems, volume 31, Curran Associates, Inc., 2018.
[20] H. Lakkaraju, S. H. Bach, J. Leskovec, Interpretable decision sets: A joint framework for description and prediction, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2016, pp. 1675–1684.
[21] R. C. Holte, Very Simple Classification Rules Perform Well on Most Commonly Used Datasets, Machine Learning 11 (1993) 63–90. doi:10.1023/A:1022631118932.
[22] J. Demšar, T. Curk, A. Erjavec, Č. Gorup, T. Hočevar, M. Milutinovič, M. Možina, M. Polajnar, M. Toplak, A. Starič, M. Štajdohar, L. Umek, L. Žagar, J. Žbontar, M. Žitnik, B. Zupan, Orange: Data mining toolbox in python, The Journal of Machine Learning Research 14 (2013) 2349–2353.
[23] E. Frank, M. A. Hall, I. H. Witten, The WEKA Workbench. Online Appendix for "Data Mining: Practical Machine Learning Tools and Techniques", Morgan Kaufmann, Fourth Edition, Morgan Kaufmann, 2016.
[24] M. Hahsler, B. Gruen, K. Hornik, Arules – A computational environment for mining association rules and frequent item sets, Journal of Statistical Software 14 (2005) 1–25. doi:10.18637/jss.v014.i15.
[25] A. Wojna, R. Latkowski, Rseslib 3: Open Source Library of Rough Set and Machine Learning Methods, in: Rough Sets, Springer International Publishing, Cham, 2018, pp. 162–176. doi:10.1007/978-3-319-99368-3_13.
[26] M. Rapp, E. L. Mencía, J. Fürnkranz, V.-L. Nguyen, E. Hüllermeier, Learning Gradient Boosted Multi-label Classification Rules, Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2020, Proceedings (2021) 124–140. doi:10.1007/978-3-030-67664-3_8.
[27] V. Arya, R. K. E. Bellamy, P.-Y. Chen, A. Dhurandhar, M. Hind, S. C. Hoffman, S. Houde, Q. V. Liao, R. Luss, A. Mojsilović, S. Mourad, P. Pedemonte, R. Raghavendra, J. Richards, P. Sattigeri, K. Shanmugam, M. Singh, K. R. Varshney, D. Wei, Y. Zhang, One Explanation Does Not Fit All: A

Toolkit and Taxonomy of AI Explainability Techniques, 2019. doi:10.48550/arXiv.1909.03012. arXiv:1909.03012.

[28] P. Clark, T. Niblett, The CN2 Induction Algorithm, Machine Learning 3 (1989) 261–283. doi:10.1023/A:1022641700528.

[29] J. Filip, T. Kliegr, PyIDS - Python Implementation of Interpretable Decision Sets Algorithm by Lakkaraju et al, 2016, in: RuleML+RR, 2019.

[30] W. W. Cohen, Fast effective rule induction, in: Twelfth International Conference on Machine Learning, Morgan Kaufmann, 1995, pp. 115–123.

[31] G. Holmes, M. Hall, E. Frank, Generating rule sets from model trees, in: Twelfth Australian Joint Conference on Artificial Intelligence, Springer, 1999, pp. 1–12.

[32] C. Singh, K. Nasseri, Y. S. Tan, T. Tang, B. Yu, Imodels: A python package for fitting interpretable models, Journal of Open Source Software 6 (2021) 3192. doi:10.21105/joss.03192.

[33] J. Alcalá-Fdez, L. Sánchez, S. García, M. J. del Jesus, S. Ventura, J. M. Garrell, J. Otero, C. Romero, J. Bacardit, V. M. Rivas, J. C. Fernández, F. Herrera, KEEL: A software tool to assess evolutionary algorithms for data mining problems, Soft Computing 13 (2009) 307–318. doi:10.1007/s00500-008-0323-y.

[34] M. R. Berthold, N. Cebron, F. Dill, T. R. Gabriel, T. Kötter, T. Meinl, P. Ohl, C. Sieb, K. Thiel, B. Wiswedel, KNIME: The Konstanz Information Miner, in: Data Analysis, Machine Learning and Applications, Springer, Berlin, Heidelberg, 2008, pp. 319–326.

[35] A. Wojna, K. Jachim, Ł. Kosson, Ł. Kowalski, D. Mański, M. Mański, K. Mroczek, K. Niemkiewicz, R. Piszczatowski, M. Próchniak, T. Romańczuk, P. Skibiński, M. Staszczyk, M. Szostakiewicz, L. Tur, D. Wójcik, M. Zuchniak, QMAK: Interacting with Machine Learning Models and Visualizing Classification Process, in: 18th Conference on Computer Science and Intelligence Systems, 2023, pp. 315–318.

[36] S. Vojíř, V. Zeman, J. Kuchař, T. Kliegr, EasyMiner.eu: Web framework for interpretable machine learning based on rules and frequent itemsets, Knowledge-Based Systems 150 (2018) 111–115. doi:10.1016/j.knosys.2018.03.006.

[37] S. Vojíř, T. Kliegr, Editable machine learning models? A rule-based framework for user studies of explainability, Advances in Data Analysis and Classification 14 (2020) 785–799. doi:10.1007/s11634-020-00419-2.

[38] J. Demšar, Statistical comparisons of classifiers over multiple data sets, Journal of Machine Learning Research 7 (2006) 1–30.

[39] J. Mika, J. Tobiasz, J. Zyla, A. Papiez, M. Bach, A. Werner, M. Kozielski, M. Kania, A. Gruca, D. Piotrowski, B. Sobala-Szczygieł, B. Włostowska, P. Foszner, M. Sikora, J. Polanska, J. Jaroszewicz, Symptom-based early-stage differentiation between SARS-CoV-2 versus other respiratory tract infections—Upper Silesia pilot study, Scientific Reports 11 (2021) 13580. doi:10.1038/s41598-021-93046-6.