EasyMiner/R Preview: Towards a Web Interface for Association Rule Learning and Classification in R

Stanislav Vojíř[1], Václav Zeman[1], Jaroslav Kuchař[2], and Tomáš Kliegr[1]

[1] Department of Information and Knowledge Engineering
Faculty of Informatics and Statistics
University of Economics, Prague, Czech Republic
firstname.lastname@vse.cz
[2] Web Intelligence Research Group, Faculty of Information Technology,
Czech Technical University in Prague
firstname.lastname@fit.cvut.cz

Abstract. EasyMiner is a web-based visual interface for association rule learning. This paper presents a preview of the next release, which uses the R environment as the data processing backend. EasyMiner/R uses the arules package to learn rules. It uses the Classifications Based on Associations (CBA) algorithm as a classifier and to perform rule pruning. Experimental results show that EasyMiner with the R-based backend is able to handle larger datasets than the previous version.

Keywords: association rules, R, web service, classification

1 Introduction

This paper describes a preview version of the next generation of the EasyMiner system for interactive association rule learning and classification. EasyMiner consists of an interactive web-based user interface and a web-service layer, which wraps association rule learning implementation. The system was first introduced at ECML'12 [1] with the LISp-Miner (`lispminer.vse.cz`) system as the association rule learning backend. In this paper, we introduce a new version, which uses the popular *arules* package [2] for the R environment. In addition to association rule learning, the new release allows to perform classification based on association rules using the Classifications Based on Associations (CBA) algorithm [3].

The benefits of the new version are as follows. The arules package, implementing the apriori algorithm [4], provides better performance on larger datasets. The addition of the CBA algorithm allows for new use cases. Apart from the support for the classification task, CBA can be used as a rule pruning algorithm. Reducing the number of rules on the output is vital for many applications, including business rule learning [5].

This paper is organized as follows. Section 2 gives a walk through the system's user interface. Section 3 describes the backend. A brief description of the CBA

component is given in Section 4. Evaluation of the system is covered by Section 5. The conclusions summarize the contribution and outline future work.

2 Workflow in the User Interface

As the first step, the user has to log in using local account or social networks. The user uploads the dataset in CSV or zipped CSV and selects mining backend (R or LISp-Miner). In the background this operation creates a named *miner* associated with a database table holding the raw data and an empty table holding the preprocessed data.

The user can optionally perform data preprocessing. This is especially needed for numerical attributes due to limitations of the apriori algorithm. The preprocessing is performed by the user dragging a field from the *Data fields palette* to the *Attributes palette* (Fig. 1A,B) and selecting preprocessing type (e.g. equidistant binning). This creates an *attribute* that can be used in the *Rule pattern*, out of a field in the input CSV file. In the background, the data are immediately processed. In order to facilitate processing of larger datasets, the system allows to skip the preprocessing step, using a verbatim copy of the fields in the input dataset as attributes.

In the main mining interface (Fig. 1), the user defines preprocessing instructions and the rule pattern. The definition is based on drag-and-drop operations the user drags an attribute from the attributes palette (Fig. 1B) and drops it into antecedent or consequent part of the rule pattern (Fig. 1C).

Finally, the user executes the task. This sends the task definition to the mining backend. The system supports ordering of the discovered rules (Fig. 1D) by values of interest measures. Selected rules can be stored in Rule Clipboard (Fig. 1E), the contents of which persists across multiple tasks on the same *miner*, or to the *Knowledge Base*, which persists across multiple miners.

The user interface layer is implemented in PHP (using Nette Framework) and JavaScript.

3 R backend

EasyMiner-Apriori-R is a REST service wrapper over the R arules library, which serves as a backend in EasyMiner, but can also be used as an independent web service.

This wrapper processes HTTP requests, transforms them to R scripts and forwards them to the R environment. The web service was implemented in the Scala language as a stand-alone application therefore it can be run on JVM (version 7+) without additional containers.

It provides the HTTP facade for sending association rule mining tasks in the GUHA AR PMML format, which is an extension of the PMML AssociationRules model[3] that supports both standard association rules mined by apriori and more expressive GUHA rules mined by LISp-Miner, the original backend in EasyMiner.

[3] `http://www.dmg.org/v4-0-1/AssociationRules.html`

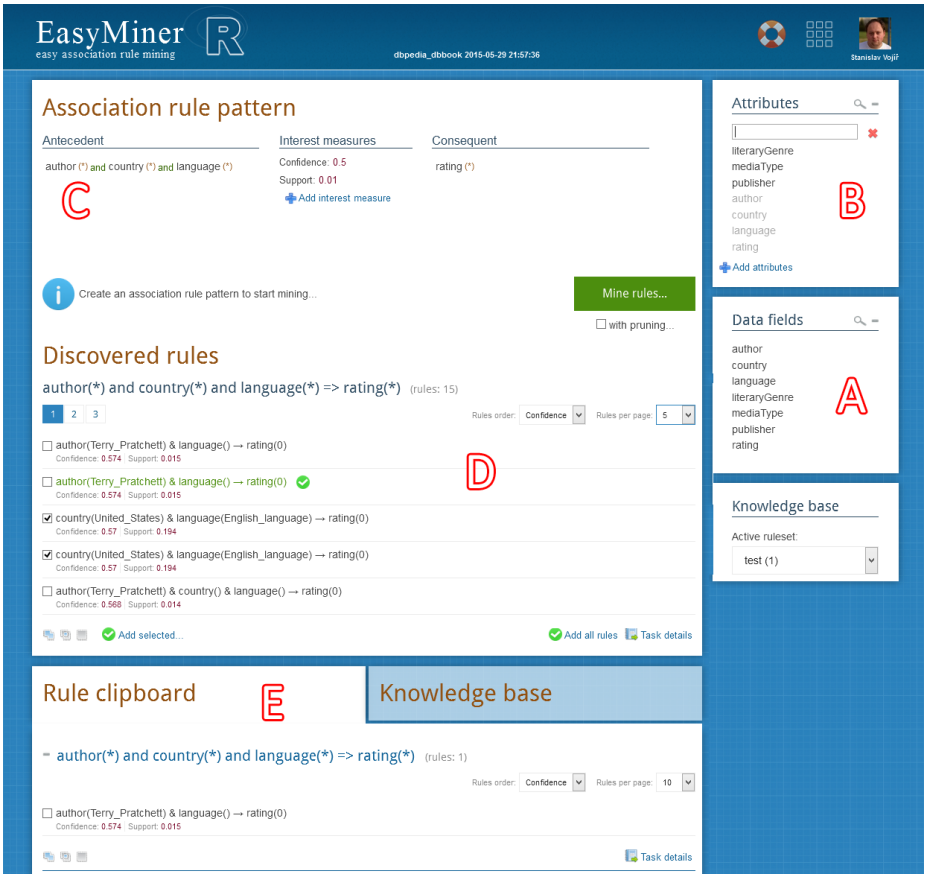

Fig. 1. User interface of EasyMiner/R

This PMML input is processed asynchronously and transformed to R scripts that use MySQL queries for data loading and itemset pruning, and the arules library for the association rules mining. Input data, which we want to mine association rules from, have to be saved in the MySQL database; so the PMML task input has to contain both information about a task (interest measure, antecedent and consequent definitions) and information about the connection to the database.

For the data transmission from the REST service to the R environment and vice versa we use the Rserve server. Any R script, sent to the R server, has to be completely initialized and all required libraries must be loaded for each request; this initializing part may take several seconds. In order to solve this problem we implemented an Rserve connection pooling system for the pre-initialization of R scripts, so if a user posts the mining task request to the server, the system is able to pull a prepared connection from the connection pool. The R apriori mining method is called immediately without waiting for an initialization. The

result of the mining process is a PMML output file where found association rules are saved

EasyMiner-Apriori-R is completely thread-safe and can handle several mining requests concurrently; the number of parallel connections depends on the actor system setting: the implementation uses Akka (`http://akka.io`) framework with Spray (`http://spray.io`).

4 Classification Based on Associations

CBA [3] is considered as the first algorithm for classification based on association rules. The algorithm was proposed in 1998, and it has multiple successor algorithms such as CPAR [6] or CMAR [7]. We selected the original algorithm rather than any of its successors, because it provides a desirable balance between accuracy and low rule count.

In general CBA proceeds as follows. The rules output by the apriori algorithm are sorted according to some criteria and then pruned – some rules are removed. Finally, a default rule is added at the bottom of the rule list ensuring that all instances will be covered. An unlabeled instance is classified by the top-ranked matching rule.

Since CBA only removes rules for the original list output by association rule learning, it can be also used for rule pruning. More compact rule sets have the advantage of better interpretability. The pruned rule set generated by CBA has the following desirable properties: a) *each training case is covered by the rule with the highest precedence among the rules that can cover the case,* and b) *every rule correctly classifies at least one training case.*

The CBA in EasyMiner/R follows the optimized M2 version of the algorithm [3]. Pessimistic pruning, an optional step in the original algorithm, is currently not included. The software is implemented in Java and wrapped with rJava into an R package. It is executed as an optional step within the R backend.

5 Evaluation

This section presents a comparison of the time requirements of EasyMiner/R with the previous version. We also empirically describe the effect of pruning on rule count and computation time.

Dataset. As the evaluation dataset we used the DBpedia Binary training dataset, generated from the ESWC 2014 Recommender Systems Challenge.[4] This dataset contains 72,371 rows containing the following attributes: *author* (2 551 unique values), *country* (77 unique values), *language* (53 unique values), *literaryGenre* (307 unique values), *mediaType* (32 unique values), *publisher* (676 unique values) and *rating* (values 0 and 1). The dataset is very sparse with many missing values. No pre-processing was performed. The benchmark dataset is made available on the `easyminer.eu` website.

[4] Dataset: `http://easyminer.eu/images/data/dbpedia_train_small.zip`

Task setting. The mining task was constrained so that the attribute *rating* is present in the consequent and the remaining attributes in the antecedent. The required minimal value of the *confidence* threshold was set to 0.5.

The count of discovered rules depends on missing value treatment. Columns denoted as *with pruning* contain the time requirements for solving the combined tasks of mining of association rules and the pruning using the CBA algorithm.

	rule count			backend-only		EasyMiner/R	
support	w/o miss.	w miss.	w miss. pruned	LISp-Miner	arules	mining	with prun.
0.010	79	163	54	3 s	6.2 s	5.4 s	33.8 s
0.009	95	186	68	6 s	6.4 s	5.4 s	27.8 s
0.008	112	213	73	16 s	6.2 s	5.4 s	31.7 s
0.007	144	295	90	27 s	6.3 s	5.5 s	31.7 s
0.006	187	397	107	1 m 10 s	6.3 s	5.5 s	35.6 s
0.005	256	552	141	4 m 38 s	6.3 s	5.7 s	35.5 s
0.004	396	765	184	28 m 04 s	6.5 s	6.0 s	37.8 s
0.003	602	1147	253	>5 h	6.5 s	8.6 s	43.3 s
0.002	1391	2699	430	>6 h	6.5 s	14.0 s	1 m 04.1 s
0.001	3394	6034	697	>6 h	6.7 s	15.1 s	1 m 59.0 s

Table 1. Time requirements of rule mining (confidence=0.5)

Benchmark setup. We evaluate three configurations: a) the time required by the previous backend (LISp-Miner) run on desktop, b) the time required by the current backend (arules) run on desktop, c) the time required if mining is run from the frontend from the user interface in a web browser. This allows us to demonstrate improvement over the previous version as well as to show what additional latencies of gains have been introduced by EasyMiner/R compared to running the new arules backend system directly on the user's computer.

All three setups exclude the time required to preprocess the datasets and import them to a database.

The time reported for EasyMiner/R essentially amounts to: transmission of the task from user's browser to the front-end server, serialization of the task in the frontend-server to PMML, transmission to the server, execution of the mining task, serialization of results to PMML, transmission to the front-end server, display of the first ten rules in the user's browser. The result reported was measured in Firefox 38.0.1 (average across three runs).

Hardware and software configuration. The evaluation was performed using the latest version of LISp-Miner (25.14.06) run on Intel Core i7-3930K @ 3.2GHz, 3.5GB RAM.

Results. The results depicted at Table 1 show that EasyMiner with the arules package as the backend is significantly faster with lower support thresholds. An interesting observation is that for tasks with pruning set to off that produce a smaller number of rules, the user will even get the results faster with

EasyMiner/R in than from the R console on her computer. This is due to a saving associated with the database connection pooling.

In case of tasks with pruning, the mining takes longer, but the resulting rule set is on this particular dataset up to nine times smaller, saving the time of the human analyst or computer system doing subsequent processing.

6 Conclusions and future work

EasyMiner/R available at `http://easyminer.eu` is an experimental academic system for association rule learning and building classifiers composed of association rules. The new version with R backend allows to handle larger datasets as empirically validated. Compared to the previous version using the LISp-Miner system, the new version does not allow to use expressive constructs such as disjunctions and negations when defining the rule pattern. Another limitation of the current version stemming from the use of the MySQL database is a limit on the maximum number of fields in the input dataset. We plan to remove the latter limitation in the next release, which will use a column-oriented database as the storage backend. Another much needed extension is execution of the benchmark on a representative sample of datasets.

EasyMiner/R is designed to be used as a complete integrated system, however, its individual components - the EasyMiner-Apriori-R[5] web service wrapper for R arules package, and the CBA implementation[6], can also be used independently.

Acknowledgment

The research and development of EasyMiner is performed under grant IGA 21/2013. The development of EasyMiner-R-Apriori-R component was supported by the European Union's 7th Framework Programme via the LinkedTV project (no. FP7-287911). The development of the CBA component was supported by CESNET grant no. 540/2014. Stanislav Stanislav Vojíř and Tomáš Kliegr were supported in writing this paper by the University of Economics, Prague within the "institutional support for long term research" scheme.

References

1. Škrabal, R., Šimůnek, M., Vojíř, S., Hazucha, A., Marek, T., Chudán, D., Kliegr, T.: Association rule mining following the web search paradigm. In Flach, P.r., Bie, T., Cristianini, N., eds.: Machine Learning and Knowledge Discovery in Databases. Volume 7524 of Lecture Notes in Computer Science. Springer Berlin Heidelberg (2012) 808–811

[5] `https://github.com/KIZI/EasyMiner-Apriori-R`
[6] `https://github.com/jaroslav-kuchar/rCBA`

2. Hahsler, M., Grün, B., Hornik, K.: arules - a computational environment for mining association rules and frequent item sets. Journal of Statistical Software **14**(15) (9 2005) 1–25
3. Liu, B., Hsu, W., Ma, Y.: Integrating classification and association rule mining. In: KDD'98. (1998) 80–86
4. Agrawal, R., Imielinski, T., Swami, A.N.: Mining association rules between sets of items in large databases. In: SIGMOD. (1993) 207–216
5. Kliegr, T., Kuchař, J., Sottara, D., Vojíř, S.: Learning business rules with association rule classifiers. In: RuleML. (2014)
6. Yin, X., Han, J.: CPAR: Classification based on predictive association rules. In Barbar, D., Kamath, C., eds.: SDM, SIAM (2003) 331–335
7. Li, W., Han, J., Pei, J.: CMAR: Accurate and efficient classification based on multiple class-association rules. In: Proceedings of the 2001 IEEE International Conference on Data Mining. ICDM '01, Washington, DC, USA, IEEE Computer Society (2001) 369–376