Bidirectional Action Rule Learning

Paweł Matyszok[1], Łukasz Wróbel[1,2(✉)], and Marek Sikora[1,2]

[1] Institute of Informatics, Silesian University of Technology, ul. Akademicka 16, 44-100 Gliwice, Poland
{pawel.matyszok,lukasz.wrobel,marek.sikora}@polsl.pl
[2] Institute of Innovative Technologies EMAG, ul. Leopolda 31, 40-189 Katowice, Poland

Abstract. Action rules specify recommendations which should be followed in order to transfer objects to the desired decision class. In this paper influence of employing information contained in source and target class examples in sequential covering based action rule induction method is examined. Results show that using source class for guiding the induction process produces best results.

Keywords: Action rules · Classification · Data mining

1 Introduction

Action rules are one of the ways to use rule-based representations to search for recommendations which indicate how the change of attribute values can cause the change in the assignment of examples to the given concept (decision class). There have been several proposals of algorithms for action rule induction, but neither presents one of the most popular and efficient approaches to the rule induction, which is the sequential covering (known also as separate-and-conquer) strategy [4]. The paper features a proposal of an algorithm for action rule induction by means of the sequential covering strategy.

The rule growing and pruning phases are guided by rule quality measures [4]. The possibility to use different quality measures allows generating more accurate or general rules, depending on the users' needs [17]. In comparison to other methods, the proposed algorithm is distinguishable by its possibility to generate action rules on the basis of numerical attributes without necessity of their previous discretization. The algorithm features two versions: the first approach builds action rules based on rules describing the examples in the source class (the class from which examples should be transferred out), while the second version is driven by rules selecting examples in the target class (the one to which the examples should be shifted).

The paper is organized as follows: Sect. 2 features related work on the action rule induction, Sect. 3 describes the proposed algorithm. Section 4 presents the results of the algorithm application in sample publicly available datasets with comparison to other action rule induction method, ARED. Section 5 gives the conclusions and presents the possible future work.

T. Czachórski et al. (Eds.): ISCIS 2018, CCIS 935, pp. 220–228, 2018.
https://doi.org/10.1007/978-3-030-00840-6_24

2 Related Work

First, the works in the field of action rule induction concentrate on generating action rules on the basis of the existing classification rules generated by means of algorithms based on the rough set theory [9,10]. Then algorithms for direct induction of action rules from data were proposed. Here it is worth to mention the algorithm of association action rule induction and the ARED algorithm [3,8]. All aforementioned approaches are based on the assumption that all possible rules which fulfill the condition of minimum support and confidence are generated. In [6] the authors proposed an algorithm which allowed to generate action rules from temporal decision tables.

Action rules present recommendations on the changes of attribute values but they do not indicate which operations cause the changes (e.g. recommendation "change blood sugar level from 95 to 80" does not show what kind of medicine should be taken to fulfill this recommendation). In this case the usability of action rules is understood as the analysis and identification of operations that must be done to change the values of the attributes occurring in the action rules premises. Such operations were called meta-actions, the analysis of dependencies between action rules and meta-actions was presented in [14,16]. Recently, Almardini et al. [1] have presented procedure paths as a sequence of procedures that a given patient undertakes to reach the desired treatment.

The action rule induction is part of a more general issue of the usability and actionability of the data exploration results. This issue has been raised more and more often [7]. In the case of rule-based representations, the issue of actions generation on the basis of classification rules, without necessity to generate action rules, was proposed in [15]. In other works [5,12,18], in turn, the authors discussed the methods of rule assessment from the point of view of their usability and actionability.

3 Approach Description

3.1 Basic Notion

An action rule [11] may be seen as the conjunction of two decision rules. To define the action rule, basic notions are presented in this section.

Let us consider that a finite set of examples T_r is given. Each example in this set can be described with a set of attributes $A \cup \{d\}$, where $a : T_r \rightarrow V_a$ for each $a \in A \cup \{d\}$. The set V_a is called *range* of attribute a. The elements of A are called conditional attributes, and the variable d is known as a decision attribute – its value is considered as an assignment of an example to a decision class. Conditional attributes can be of numeric or symbolic type. Symbolic attributes have discrete value. Numeric attributed are represented by real values or ranges. The decision attribute is always of symbolic type.

The conditional expression IF $w_1 \wedge w_2 \wedge \ldots \wedge w_k$ THEN $d = v$ is called decision rule. The conclusion of a decision rule is understood as an assignment of an example fulfilling the premise of this rule to the concept bound with the value of

decision attribute d. The premise of the decision rule is a conjunction of elementary conditions w_i. The form of an elementary condition used in the presented approach is $w_i \equiv a_i \in Z_i$, where a_i is the name of the conditional attribute and Z_i is the interval in this attribute range V_{a_i}. For symbolic attributes the elementary condition is simple $a_i = v_j$, where $v_j \in V_{a_i}$.

Let us consider following formula: IF $w_{1_1} \rightarrow w_{2_1} \wedge \ldots \wedge w_{1_k} \rightarrow w_{2_k}$ THEN $d = v_1 \rightarrow v_2$ which may be seen as a composition of two decision rules $r1, r2$ having mutually exclusive conclusions (i.e. $v_1 \neq v_2$). We will refer to such expression as an action rule. An action rule shows possible transition of examples from one class to another in some example set T_r after a suggested action in premise of the rule are undertaken, i.e. attribute values are changed according to actions gathered. The class on the left of the action rule conclusion will be often referred to as *source class* or *source concept* of given rule, while the class on the right of the premise is being called *target class* or *target concept* of the rule. We will refer to the left side of action rule (the $r1$ part of the composition) as *source rule*, while the right side (the $r2$ part of the composition) will be called *target rule*. Such naming convention clearly expresses the ability of source and target rule to select source and target class examples in the dataset.

The formula of the action rule may be rewritten in simpler form which maintains clarity of connection between the value change and the attribute: $(a_1, v_{a_{1_1}} \rightarrow v_{a_{1_2}}) \wedge \ldots \wedge (a_k, v_{a_{k_1}} \rightarrow v_{a_{k_2}}) \rightarrow (d = v_1 \rightarrow v_2)$.

The elementary condition of the form $(a_i, v_{a_{i_1}} \rightarrow v_{a_{i_2}})$ is called an *elementary action* (simply – an *action*). In the context of action rules as described in this section, the action should be understood as demand to influence the examples from dataset T_r in a way that will change value of an attribute a_i of examples covered by condition $a_i \in v_{a_{i_1}}$ such they will now fulfill the other condition $a_i \in v_{a_{i_2}}$. The conclusion of the action rule is the expected effect of applying the action rule premise to examples in the dataset. It describes change of concept from source to target class. In the context of the action rule induction, attributes are divided into two types: *flexible* and *stable*. Stable attributes cannot be used in any of the action induced (e.g. the date of birth cannot be changed), while flexible attributes can be subject of change by some action.

3.2 Action Rule Induction

In this section a covering action rule induction algorithm is presented. The algorithm utilizes a well known separate-and-conquer framework [4] and classification rule quality measures [17] to induce action rules. The algorithm of rule induction is divided into two phases: growing and pruning. The action rule is added to the resulting rule set, and all source class examples covered by left side of newly created action rules are removed from the training set. The examples of the target class are never removed from the training set. Rules are inducted until all examples of the source class in the training set T_r are covered by left sides of action rules inducted.

To grow an action rule is a process beginning with a rule having empty premise and established conclusion with a source class on the left side and a

target class on the right side. Two temporal classification rules are being maintained during action rule growing, one rule per class in conclusion. Iteratively (limited by the *min_cov* parameter), candidate for best elementary condition describing the source class is searched in the input dataset feature space and added to temporary rule responsible for selecting source class examples. The rule is assessed with quality measure q. After choosing the best elementary condition, attribute on which it is based on is extracted and new best elementary condition is being looked for, but this time with regard to target class and limited by attribute selected. Second temporary rule is used to choose the best condition. Both conditions are used to build an action, which is added to premise of action rule being build.

The process of pruning of the action rule begins when the rule is fully grown. Actions in the premise of the action rule are being iteratively trimmed or removed, as long as the quality of the rule measured by the function q does not decrease. To trim the action means that the action of the form $(a_i, v_{a_{i1}} \rightarrow v_{a_{i2}})$ becomes $(a, v_{a_{i1}})$. If the quality of the action rule after action trimming does not decrease, it is also checked if removal of whole action will have similar effect. The action rule premise is modified based on result of that test.

The proposed bidirectional algorithm of action rules induction covers the input dataset sequentially and simultaneously induces two action rules using sequential covering process described above. One action rule is induced according to the intention of the algorithm user: the rules are initialized with a conclusion having the source class on the left side and the target class on the right – we will refer to this rule as a *forward-rule*. The other rule starts with the conclusion constructed with the classes reverted – in the article text this rule will be described as *backward-rule*. Two copies of dataset are maintained during rule induction: one for the forward-rule, and one for the backward-rule. Both rules are sequentially grown, each on respective copy of input dataset.

After the rules are grown, the optional pruning step is performed. The examples covered by newly created action rules are removed from respective datasets, and the rules are collected in separate ruleset: *FAR* for forward-rules, and *BAR* for backward-rules. This procedure is repeated until both copies of input dataset are empty. Now the procedure of reverting the rules gathered in the *BAR* set is conducted. The goal of this procedure is to obtain action rules which are representing the transition of examples demand by the user of the method. The reverting procedure is quite simple: first, new conclusion for a rule is constructed by reversing the order of the classes in the original conclusion. Afterwards, the premise is traversed, and each action is reverted. If the action in the premise being reverted was trimmed, then new action is constructed with special symbol ALL on the left side, and the left value of the original action on the right side, i.e. reverting action of form $(a_i, v_{a_{i1}})$ will produce $(a_i, ALL \rightarrow v_{a_{i1}})$. In other cases, the reverting of the action is realized by switching places of attribute values on left and right side of the action. Both *FAR* and *BAR* rulesets are returned from the method.

The goal of the process of inducing rules with reverted conclusion is to steer the process of building the action rule by knowledge contained in examples representing the target class. Conceptually, the rules collected in the *FAR* ruleset are representing the idea of an example leaving the source class (because the attributes for the premise are selected based on the data assigned to the source class), while the reverted rules from the *BAR* ruleset represent concept of "achieving" the target class – because more emphasis in the premise of the backward-rule is put on the knowledge gathered in the target class examples.

4 Experiments

4.1 Experimental Setup

In the experimental part of the study, we compare Forward with the Backward method of action rule induction in order to determine which of these two approaches could be more beneficial for action rule learning. For both methods the pruning procedure is enabled and the *mincov* parameter is set to 5. During specialization and pruning of the rules the *Correlation* quality measure (q parameter) is used. The *Correlation* measure is defined as: $(pN - Pn)/\sqrt{PN(p+n)(P-p+N-n)}$ where: P (N) stands for the number of examples in the dataset that are covered (not covered) by the conclusion of r, p is the number of true positives, that is, the number of examples covered by both the premise and the conclusion of the rule r, and n is the number of false positives, that is, the number of examples covered by the premise, but not covered by the conclusion of the rule r. The measure evaluates the correlation coefficient between the predicted and the target labels. As empirical [4,17] studies show, it maintains a good balance between accuracy and comprehensibility of rules generated in covering schemas.

For comparison purposes, we also provide the results obtained with the custom implementation of the ARED algorithm [8]. The ARED method uses the concept of Pawlaks' information system [9] in which certain relationships between granules, defined by indiscernibility relation of objects, are identified. Some of these relationships are used to define the action rule. The ARED approach is based on the assumption that all possible rules are generated that meet the minimum support and minimum confidence constraints. For all examined datasets minimum support was set to 5 and the minimum confidence parameter was equal to 0.9.

Experiments were carried out on 11 publicly available datasets, mostly from the UCI repository. Although, the Forward and Backward methods are able to handle multi-class problems by employing the one-vs-all binarization schema, for simplicity we consider only two-class problems, choosing one of the class as the source and the second one as the target. As the ARED algorithm is not able to handle numerical attributes it was run on the discretized version of the datasets. The discretization was conducted using the entropy criterion. For the Forward, Backward and ARED methods all attributes of all tested datasets were marked as flexible.

Table 1. The characteristics of action rule sets generated by Forward, Backward and ARED algorithms: the total number of rules (#r), the average number of conditions ($\overline{c}$) and actions ($\overline{a}$) per rule, the average precision ($\overline{prec}$) and sensitivity ($\overline{sens}$) of the left/right side of the rule in the set. The *pval* row shows the p-value of the Wilcoxon signed rank test confronting Forward method with other approaches. The *Me* row contains median value over all datasets.

Data	Forward					Backward					ARED				
	#r	$\overline{c}$	$\overline{a}$	$\overline{prec}$	$\overline{sens}$	#r	$\overline{c}$	$\overline{a}$	$\overline{prec}$	$\overline{sens}$	#r	$\overline{c}$	$\overline{a}$	$\overline{prec}$	$\overline{sens}$
bre	11	3.3	0.4	.42/.80	.18/.77	10	3.1	2.6	.69/.81	.30/.42	75	1.0	1.0	.34/.72	.19/.24
cr-a	3	3.3	2.3	.81/.87	.85/.63	3	3.3	3.3	.82/.74	.72/.72	3344	3.6	3.1	.69/.86	.02/.05
cr-g	13	5.2	2.7	.50/.88	.47/.32	18	5.3	5.2	.67/.84	.18/.52	428	2.2	1.6	.34/.69	.07/.10
hun	5	3.6	1.2	.74/.92	.69/.38	4	2.2	2.2	.76/.82	.68/.88	18	1.2	1.1	.41/.70	.25/.40
lab	2	3.0	1.0	.94/.85	.62/.89	1	1.0	1.0	.82/.85	.45/.89	15	1.0	1.0	.68/.76	.42/.41
monk	7	1.7	1.1	.78/.95	.27/.41	6	1.5	1.3	.65/.73	.38/.41	25	1.3	1.2	.50/.62	.27/.32
pima	7	3.7	1.1	.55/.91	.80/.36	18	3.9	3.9	.56/.85	.62/.51	22	1.0	1.0	.37/.67	.30/.40
prnn	4	1.2	1.0	.79/.88	.76/.69	4	1.8	1.8	.85/.82	.83/.91	7	1.0	1.0	.71/.76	.34/.47
stat	6	5.7	2.5	.82/.80	.59/.70	6	3.8	3.8	.85/.84	.55/.73	13	1.0	1.0	.52/.59	.50/.61
tit	2	2.0	0.5	.66/.73	.49/.48	7	2.0	1.1	.79/.54	.92/.23	8	1.0	1.0	.45/.32	.09/.37
vot	3	1.3	1.0	.97/.89	.86/.93	1	1.0	1.0	.99/.92	.92/.97	110496	6.4	4.5	.90/.90	.04/.14
Me	5	3.3	1.1	.78/.88	.62/.63	6	2.2	2.2	.79/.82	.62/.72	22	1.0	1.0	.50/.70	.25/.37
pval						1.0	0.6	0.0	.45/.08	.95/.41	0.0	0.0	0.8	.02/.00	.00/.01

4.2 Results

In the experimental part of this study the examined algorithms are evaluated according to the: total number of action rules, average number of conditions per rule, average number of actions per rule, precision and sensitivity of the left and the right side of the rule. The precision of the rule r is defined as $\frac{p}{p+n}$ where p, n, P i N values have the same meaning as for *Correlation* measure. The higher precision of the left/right side of the action rule is, the more accurately rule covers examples from the source/target class. The sensitivity, expressed as $\frac{p}{P}$, estimates what part of the source/target class is covered by the left/right side of the action rule. The low value of the sensitivity usually reflects very specific rule which might not generalize well for new examples. The detailed characteristics of the action rule sets generated by the Forward, Backward and ARED algorithms are provided in Table 1.

According to the Wilcoxon signed rank test the only significant difference between the Forward and Backward methods is in the average number of actions per rule (the p-value is close to 0). On average, the number of action per rule for Backward is twice as large as for Forward – the median value for Backward is 2.2 whereas for Forward it equals 1.1.

In regard to remaining criteria the Forward and Backward behave very similarly, all the p-values of the Wilcoxon test are greater than 0.05. Both Forward and Backward are able to generate compact rule sets – for most datasets algorithms found less than 10 rules. The output rules usually contain 2–3 conditions among which 1–2 are actions.

Rules generated by Forward and Backward methods are characterized by relatively high values of precision and sensitivity of the left and the right side of the action rules. For most datasets, the average precision of rules is greater than 0.7. The sensitivity, on average, ranges from 0.2 to 0.9 with the median value over all datasets around 0.6. It can be also observed that for many datasets the precision of the right side of the action rule is higher than of the left side, indicating that action rules usually cover the target class more accurately than the source class.

Compared to the ARED algorithm, the Forward method seems to achieve better results according to the considered criteria. In most cases, the p-values of the Wilcoxon test are close to 0. The only exceptions are the average number of actions where both algorithms behave similarly and the average number of conditions where ARED tends to generate shorter rules. The greater number of rules for the ARED method results from the fact that ARED is based on the concept of learning all possible rules satisfying minimum confidence and support thresholds. As ARED usually generates very short rules, containing only one condition, the large number of rules often goes hand in hand with decreased average precision and sensitivity of rules.

5 Conclusions

In this paper two working action rule induction algorithms were presented. After analysis of the output of the presented methods, despite that the idea of employing knowledge gathered in target class examples embedded in Backward method is appealing, our recommendation is to use the Forward method. It was shown that both Forward and Backward method outperformed ARED method wherever number of resulting action rules or precision and sensitivity of source and target rule are considered. If the goal of using action rule induction method is to reach consistent and comprehensible ruleset, then approach presented in this paper is able to deliver demanded output.

Future work will focus on the development of measures and methods for assessing the quality of inducted action rules and sets. It is planned to adapt the measures dedicated to the quality evaluation of classification rules. Our work will focus on measures which assess the coverage and precision of a rule at the same time. In the case of large datasets or the necessity to reduce the number of rules, the methods of example selection, as well as rules filtering will be used [2,13]. A very important next step is the preparation of datasets where successive decision tables will reflect successive moments of time (control points). The tables will contain information about the application of certain actions to specific examples. A part of the datasets will be generated synthetically (e.g. the inverted pendulum problem), another part will describe a real-life problem (PersonALL project – see acknowledgment).

Acknowledgement. This work was partially supported by Polish National Centre for Research and Development (NCBiR) within the programme Prevention and Treatment of Civilization Diseases – STRATEGMED III, grant number STRATEGMED3/304586/5/NCBR/2017 (PersonALL). A part of the work was carried out within the statutory research project of the Institute of Informatics, BK-213/RAU2/2018.

References

1. Almardini, M., et al.: Reduction of readmissions to hospitals based on actionable knowledge discovery and personalization. In: Kozielski, S., Mrozek, D., Kasprowski, P., Małysiak-Mrozek, B., Kostrzewa, D. (eds.) BDAS 2015-2016. CCIS, vol. 613, pp. 39–55. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-34099-9_3
2. Blachnik, M.: Instance selection for classifier performance estimation in meta learning. Entropy **19**(11), 583 (2017)
3. Dardzinska, A.: Action Rules Mining, vol. 468. Springer, Cham (2012)
4. Fürnkranz, J., Gamberger, D., Lavrač, N.: Foundations of Rule Learning. Springer, Cham (2012)
5. Greco, S., Matarazzo, B., Pappalardo, N., Słowinski, R.: Measuring expected effects of interventions based on decision rules. J. Exp. Theor. Artif. Intell. **17**(1–2), 103–118 (2005)
6. Hajja, A., Raś, Z.W., Wieczorkowska, A.A.: Hierarchical object-driven action rules. J. Intell. Inf. Syst. **42**(2), 207–232 (2014)
7. He, Z., Xu, X., Deng, S.: Data mining for actionable knowledge: a survey (2005). arXiv preprint arXiv:cs/0501079
8. Im, S., Raś, Z.W.: Action rule extraction from a decision table: ARED. In: An, A., Matwin, S., Raś, Z.W., Ślęzak, D. (eds.) ISMIS 2008. LNCS (LNAI), vol. 4994, pp. 160–168. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-68123-6_18
9. Pawlak, Z.: Information systems theoretical foundations. Inf. Syst. **6**(3), 205–218 (1981)
10. Raś, Z.W., Tzacheva, A.A., Tsay, L.S., Giirdal, O.: Mining for interesting action rules. In: IEEE/WIC/ACM International Conference on Intelligent Agent Technology, pp. 187–193. IEEE (2005)
11. Ras, Z.W., Wieczorkowska, A.: Action-rules: how to increase profit of a company. In: Zighed, D.A., Komorowski, J., Żytkow, J. (eds.) PKDD 2000. LNCS (LNAI), vol. 1910, pp. 587–592. Springer, Heidelberg (2000). https://doi.org/10.1007/3-540-45372-5_70
12. Słowiński, R., Greco, S.: Measuring attractiveness of rules from the viewpoint of knowledge representation, prediction and efficiency of intervention. In: Szczepaniak, P.S., Kacprzyk, J., Niewiadomski, A. (eds.) AWIC 2005. LNCS (LNAI), vol. 3528, pp. 11–22. Springer, Heidelberg (2005). https://doi.org/10.1007/11495772_3
13. Stańczyk, U., Zielosko, B.: On combining discretisation parameters and attribute ranking for selection of decision rules. In: Polkowski, L., et al. (eds.) IJCRS 2017. LNCS (LNAI), vol. 10313, pp. 329–349. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-60837-2_28
14. Touati, H., Raś, Z.W., Studnicki, J., Wieczorkowska, A.A.: Mining surgical meta-actions effects with variable diagnoses' number. In: Andreasen, T., Christiansen, H., Cubero, J.-C., Raś, Z.W. (eds.) ISMIS 2014. LNCS (LNAI), vol. 8502, pp. 254–263. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-08326-1_26

15. Trépos, R., Salleb-Aouissi, A., Cordier, M.O., Masson, V., Gascuel-Odoux, C.: Building actions from classification rules. Knowl. Inf. Syst. **34**(2), 267–298 (2013)
16. Wang, K., Jiang, Y., Tuzhilin, A.: Mining actionable patterns by role models. In: 22nd International Conference on Data Engineering, pp. 16–16. IEEE (2006)
17. Wróbel, Ł., Sikora, M., Michalak, M.: Rule quality measures settings in classification, regression and survival rule induction-an empirical approach. Fundam. Inform. **149**(4), 419–449 (2016)
18. Zhu, H.M., Huang, W.D., Zheng, H.S.: Method for discovering actionable rule. In: Fourth International Conference on Fuzzy Systems and Knowledge Discovery, vol. 1, pp. 397–401. IEEE (2007)