Learning Business Rules with Association Rule Classifiers

Tomáš Kliegr[1,4], Jaroslav Kuchař[1,2], Davide Sottara[3], and Stanislav Vojíř[1]

[1] Department of Information and Knowledge Engineering,
Faculty of Informatics and Statistics
University of Economics, Prague, Czech Republic, first.last@vse.cz
[2] Web Engineering Group, Faculty of Information Technology
Czech Technical University in Prague
[3] Biomedical Informatics Department
Arizona State University, dsottara@asu.edu
[4] Multimedia and Vision Research Group
Queen Mary, University of London

Abstract. The main obstacles for a straightforward use of association rules as candidate business rules are the excessive number of rules discovered even on small datasets, and the fact that contradicting rules are generated. This paper shows that Association Rule Classification algorithms, such as CBA, solve both these problems, and provides a practical guide on using discovered rules in the Drools BRMS and on setting the ARC parameters. Experiments performed with modified CBA on several UCI datasets indicate that data coverage rule pruning keeps the number of rules manageable, while not adversely impacting the accuracy. The best results in terms of overall accuracy are obtained using minimum support and confidence thresholds. Disjunction between attribute values seem to provide a desirable balance between accuracy and rule count, while negated literals have not been found beneficial.

Keywords: association rules, rule pruning, business rules, Drools

1 Introduction

Association rule learning cannot be directly used for learning business rules, due to the excessive number of rules generated even for small datasets, and the lack of a rule conflict resolution strategy. However, if several techniques originally developed for association rule classification (ARC) are adopted, association rules can be used as classification business rules. ARC algorithms contain a rule pruning step, which significantly reduces the number of rules, and define a conflict resolution strategy for cases when one object is matched by multiple rules.

This paper has two focus areas. Due to the limited amount of prior work, in the first part of the paper we evaluate to what degree ARC algorithms meet the requirements of the business rule learning task and demonstrates how the discovered rules can be used in a Drools Business Rule Management System

(BRMS) system. The second part of the paper describes our implementation and experimental evaluation of a business rule learning system. In contrast to mainstream ARC algorithms, the system allows to learn disjunctive and negative rules. We hypothesize that the additional expressiveness could result in a rule set which is smaller, and thus more intelligible for the business user. Another modification is a simplification of the rule pruning phase.

This paper is organized as follows. Section 2 reviews related research. Section 3 presents a set of requirements on business rule learning algorithm and contrasts it with what ARC algorithms provide. Section 4 describes how rules learnt from data can be used in the Drools. Section 5 presents our experimental business rule learning system *brCBA*. Section 6 presents experimental evaluation on several datasets. Finally, Section 7 summarizes our findings, gives limitations of the presented work and outlines viable directions of future research.

2 Related work

There is a very limited amount of prior work on learning business rules from data. This paper is restricted to what we call *classification* business rules i.e. rules that assign a class (a type) to an object whenever its description matches the conditions contained in the rule's body. This corresponds to what is known in the rule learning literature as *classification rule* or *predictive rule*.

Association rule learning algorithms such as *apriori* [1] or FP-growth [3] can be used to learn conjunctive classification rules from data if the mining setup is constrained so that only the target class values can occur in the consequent of the rules. The GUHA method [7] is an alternative approach to mine association rules, which allows to learn also rules featuring negation and disjunction between attribute values.

The main obstacles for a straightforward use of association rules as candidate business rules are the excessive number of rules discovered even on small datasets, and the fact that contradicting rules are generated. Association Rule Classifier (ARC) algorithms provide an extension over association rule learning algorithms which address exactly these issues. These algorithms contain a rule pruning step, which significantly reduces the number of rules, and define a conflict resolution strategy for cases when one object is matched by multiple rules.

The first ARC algorithm dubbed CBA (Classification based on Associations) was introduced in 1998 by Liu et al. [5]. While there were multiple follow-up algorithms providing incremental improvements in classification performance (e.g. CPAR [15], CMAR [4] and MMAC [10]), the structure of most ARC algorithms follows that of CBA [13]: 1) learn association rules, 2) prune the set of classification rules, 3) classify new objects. Our proposed brCBA algorithm also follows this structure. It differs from CBA and other algorithms by using a GUHA-based algorithm in the "learn association rules" phase, which allows us to explore the effects of disjunction and negation on classification performance. To the best of our knowledge, the impact of the increased expressiveness added by these connectives on ARC performance has not yet been reported.

The output of association rule learning algorithms is determined typically by two parameters: minimum confidence and support thresholds on the training data. The confidence of a rule is defined as $a/(a + b)$, where a is the number of correctly classified objects, i.e. those matching rule antecedent as well rule consequent, and b is the number of misclassified objects, i.e. those matching the antecedent, but not the consequent. The support of a rule is defined as a/n, where n is the number of all objects (relative support), or simply as a (absolute support). The confidence threshold can be used to control the quality of the resulting classifier. While the authors of ARC classifiers report the confidence threshold used in their experimental setups (0.3 [10], 0.4 [9], 0.5 [5]), the impact of varying the value of this threshold on classifier performance has not yet been studied (to the best of our knowledge). To help guide the setting of ARC algorithms, we provide a detailed study of the effect of confidence threshold and support thresholds on the classification accuracy and rule count.

There is also a very limited work on effects of rule pruning. A qualitative review of rule pruning algorithms used in ARC are given e.g. in [13, 8]. The effect of pruning on the size of the rule set is reported in [5], which presents evaluation on 26 UCI datasets. The average number of rules per dataset without pruning was 35,140, with pruning the average number of rules was reduced to 69. However, this paper focuses on the evaluation of less commonly employed pessimistic pruning. We focus on evaluation of data coverage pruning, which is the most commonly used pruning algorithm (present, with some modifications, in CBA, CMAR and MMAC).

3 Business Rule Learning Requirements

The business rule learning workflow imposes some specific demands on the selection of a suitable rule learning algorithm. In this section, we discuss the compliance of ARC algorithms with some of the requirements that we have identified.

BRMS Supported Rule Expressiveness. The rules learnt are composed of a conjunction of constraints on attribute values in the antecedent, and a single value for the class attribute in the consequent. The operations performed by later steps in ARC execution, such as pruning or ranking, do not change the internal structure of the rules.

Example 1. Rule learnt on the Iris dataset.

$\ulcorner$petalLength=$\langle 3.95; 4.54)$ $\wedge$ petalWidth=$\langle 1.3; 1.54)$ $\rightarrow_{1,0.14}$ Class=Iris-versicolor$\urcorner$, where 1 is rule confidence and 0.14 (relative) rule support.

Rules, such as the one depicted in Example 1, can be translated into technical rule languages for execution inside a rule engine. In our earlier work [14] we presented the mapping to DRL, the format used by the open source BRMS system Drools.

Small number of output rules. Perhaps the biggest challenge in converting association rules to business rules is the fact that the number of discovered rules is often too large to be presented to a user. The two common strategies to solve this problem are rule grouping and rule pruning.

Rule grouping algorithms cluster the rules according to a predefined distance measure [12]. Most ARC algorithms use rule pruning. The details of the individual types of pruning algorithms is given e.g. in [13, 11, 8]. The most commonly used method according to these survey papers is *Data Coverage Pruning* (see Subs. 5.2).

Exhaustive set of rules. Most ARC algorithms use an exact association rule learning algorithm, either based on apriori or FP-Growth. These algorithms learn exhaustive set of rules matching predefined minimum confidence and minimum support thresholds [13].

However, some rules are removed in the pruning phase. Since pruning[5] removes only rules which cover objects which are already covered by another higher priority rule, the pruning typically affects only rules that would be viewed by the user as redundant.

Rule conflict resolution. Once association rules are generated and pruned, ARC algorithms use them to classify new objects. There are two fundamental approaches: *single rule* and *multiple rule* classification [13], depending on the number of rules that are involved in assigning a class to an object. The *single rule* classification used in CBA is described in Section 5.3 and subject to experimental evaluation as part of our implementation in Section 6. An overview of possible implementation in the Drools Rule Engine is present in Section 4.

Ability to control rule quality. The rule quality can be controlled by setting the minimum confidence (and support) thresholds. It should be noted that ARC algorithms try to cover every training object with at least one rule, for example, CBA ensures this by adding a default rule to the rule set. The default rule insertion needs to be omitted (ref. to Subs. 5.2) in order to allow the user to control the overall quality of the rule set.

4 Drools-based Rule Engine

The learning algorithm generates association rules which establish an implication between the antecedent and the consequent. In the case of classification rules, the consequent is the type of an individual object whose features have been matched by the antecedent. So, they can naturally be reinterpreted as business rules with the semantics of production rules. This allows to decouple recognition from decision making, resulting in more robust knowledge bases. Moreover, (production) rule engines can be considered commodity components: in particular, we have used the popular open source business logic platform Drools[6]. Drools is written in Java and relies on an object-oriented rule engine inspired from the RETE algorithm.

[5] Referring to the "database coverage" algorithm.

[6] http://drools.jboss.org

Listing 1.2. A Conflict Resolution Meta-Rule in Drools

```
rule 'Block by confidence'  @Direct
  when
    $m1 : Match( associationRole == 'premise', $t : tuple )
    $m2 : Match( this != $m1, associationRole == 'premise', tuple == $t,
                 confidence > $m1.confidence ||
                 confidence == $m1.confidence && support > $m1.support ||
                 antecedent < $m1.antecedent )
  then
    kcontext.cancelMatch( $m1 );
  end
```

In our implementation, we have created a simple, generic data model with two classes to model attributes and inferred types: `DrlObject` and `DrlAR` respectively. This allows to write rules such as the one in Listing 1.1.

Listing 1.1. A Sample Classification Rule in Drools

```
rule "rule_1" @associationRole(premise)
    @antecedent(4) @confidence(1) @support(0.06)
  when
    DrlObj( name == "petalLength", numVal >= 1 && < 1.59 )
    DrlObj( name == "petalWidth",  numVal >= 0.1 && < 0.34 )
    DrlObj( name == "sepalLength",
            numVal >= ( 4.3 && < 4.66 ) || ( >= 4.66 && < 5.02 ) )
    DrlObj( name == "sepalWidth", numVal >= 2.96 && < 3.2) )
  then
    DrlAR $type = new DrlAR( "rule_1", "Iris_Setosa", 4, 1, 0.06 );
    insertLogical( $type );
end
```

The rules are generated automatically from the output of the rule learner. Since the learner produces XML, we have applied an XSLT transformation to generate DRL, the Drools technical rule language. Notice that information such as confidence and support is retained as metadata and modelled using Java-like @annotations.

In order to implement the conflict resolution strategies mentioned in Section 5.2, we have exploited the "declarative agenda" feature of the rule engine. In a production rule engine, whenever one or more facts match the left-hand side of a rule, a rule activation is created and queued into an agenda. Activations are then consumed and the actions in the right-hand side are executed by the engine. Drools' declarative agenda allows to define rules that match and process the activations queued in the agenda itself. Such "meta-rules" are deployed into the same rule base as the standard rules. More specifically, entries in the agenda are instance of the class `Match`, which holds references to the rule that was activated as well as the tuple that caused the activation. Any metadata that is attached to the original rule is exposed by the engine as a virtual property of the activation, so that the meta-rule can constrain their value. Thanks to these capabilities, any conflict resolution strategy can be implemented with a single meta-rule, as shown in Listing 1.2. In our case, the activation of a rule with higher priority will cancel the activation of a rule with a lower priority for the same tuple.

5 brCBA - CBA for Business Rule Learning

In this section, we describe the setup used to perform the experimental evaluation. The implementation comes out of the seminal CBA algorithm. However, there are minor differences in individual steps, which are summarized in Table 1 and explained in the remainder of this section. Most importantly, brCBA uses for rule learning the LISp-Miner system[7], an implementation of the GUHA method, instead of the apriori algorithm.

stage	**CBA [5]**	**brCBA**
learning	conjunctive rules (apriori)	conj. rules, disjunctions between attribute values, negations (GUHA method)
pruning	pessimistic pruning (optional), data coverage, default rule replacement	no pruning, data coverage pruning
classification	complete	partial

Table 1. Comparison of CBA and brCBA

5.1 Rule Expressiveness

The mainstream systems for mining association rules employed in ARC, including CBA, output conjunctive association rules. The basic building block of an association rule is a literal.[8]

Definition 1. ***(literal)*** *A literal p is an attribute-value pair, taking the form of (A_i, v) in which A_i is an attribute and v a value. An object o satisfies a literal $p = (A_i, v)$ if and only if $o_i = v$, where o_i is the value of the i^{th} attribute of o.*

Definition 2. ***(rule)*** *A rule r, which takes the form of "$l_1 \wedge l_2, \wedge \ldots \wedge l_m \rightarrow c$", consists of a conjunction of literals $l_1, l_2, \ldots, l_m$, associated with a class label c. An object satisfies rule r's body if and only if it satisfies every literal in the rule. If object satisfies r's body, r predicts that the object is of class c. If a rule contains zero literal, its body is satisfied by any object.*

In brCBA we extend the original notion of literal present in Def. 1 to allow for disjunction between attribute values (dynamic binning) and negated literals.

Dynamic binning (disjunctions between attribute values). Typically value binning is performed during the preprocessing step, creating a modified data table which contains a smaller number of merged values. This approach may negatively impact the quality of the rule learning if the bins created are too narrow or too broad. In brCBA we extend the definition of literal to allow for dynamic binning, which merges multiple values during *rule learning* into a value range (an enumeration of values or an interval).

[7] `http://lispminer.vse.cz`

[8] We introduce the definition of literal and an association rule from [15] substituting the machine learning term "tuple" by term "object" common in the BRMS field.

Definition 3. ***(positive literal)*** *A positive literal p is an association of an attribute with a value range, taking the form of (A_i, V) in which A_i is an attribute and V is a value range. An object o satisfies a positive literal $p = (A_i, V)$ if and only if $o_i \in V$, where o_i is a value of the i^{th} attribute of object o.*

From the options offered by the LISp-Miner system, we consider two types of dynamic binning: **Subset** binning merges up to a prespecified number of values, while **Sequence** (Interval) binning merges up to a prespecified number of *adjacent* values [7]. Subset binning is typically applied on on nominal attributes, while adjacent value binning on numerical or ordinal attributes.

The maximum number of values to be merged is set by parameter λ (for both methods). The result of dynamic binning on an attribute is a set of literals. Unlike some greedy algorithms (such as the algorithm for grouping values in C4.5 [6]), the dynamic binning operator is exhaustive. For an attribute A_i with n distinct values, assuming that $n \geq \lambda$, sequence binning creates $\sum_{j=1}^{\lambda} n - j + 1$ literals, while subset binning $\sum_{j=1}^{\lambda} \binom{n}{j}$ literals.

Example 2. Binning
The discretization on the petalLength attribute from the Iris dataset was performed by creating equidistant bins during preprocessing[a]: $[1; 1.59)$, $[1.59; 3.95)$, $[3.95; 4.54)$, $[4.54; 5.13)$, $[5.13; 5.72)$. Interval binning set to maximum length λ=2 will create 9 literals: five literals corresponding the original values plus the following four: $[1; 1.59) \vee [1.59; 3.95)$, $[1.59; 3.95) \vee [3.95; 4.54]$, $[3.95; 4.54) \vee [4.54; 5.13)$, $[4.54; 5.13) \vee [5.13; 5.72)$.
An example rule featuring dynamically binned intervals: ⌜petalLength $= [4.54; 5.13) \vee \langle 5.13; 5.72) \rightarrow_{0.77,0.33}$ Class=Iris-versicolor⌝,

[a] Merging bins with too small support count into one bin.

Negation Considering negative literals in addition to the positive ones during rule mining produces a richer set of rules. It was previously conjectured that this could benefit the performance of ARC [2].

Definition 4. ***(negative literal)*** *A negative literal n is an association of an attribute with a value range, taking the form of (A_i, V) in which A_i is an attribute and V is a value range. An object o satisfies a negative literal $n = (A_i, V)$ if and only if $o_i \notin V$, where o_i is a value of the i^{th} attribute of o.*

Example 3. Rule with a negative literal
⌜¬petalLength=$[1; 1.59) \wedge$ petalWidth $[0.1; 0.34) \rightarrow_{1,0.05}$ Class=Iris-setosa⌝

5.2 Rule Pruning

CBA and brCBA use the *data coverage* rule pruning algorithm. This algorithm applies to a sorted list of ranked rules. Each rule is matched against the training

Algorithm 1 Data Coverage

```
Require: rules – sorted list of rules, T – set of objects in the training dataset
Ensure: rules – pruned list of rules

  rules := sort rules according to criteria on Fig. 1
  for all rule ∈ rules do
    matches:= set of objects from T that match both rule ant. and conseq.
    if matches==∅ then
      remove rule from rules
    else
      remove matches from T
    end if
  end for
  return rules
```

data. If a rule does not correctly classify any object, it is discarded. Otherwise, the rule is kept, and the objects correctly classified are removed (ref. to Alg. 1).

The output of rule pruning is a reduced set of rules, where the redundant rules have been removed. If there are two rules matching one training object, the weaker rule (acc. to Fig. 1) will be removed.

1. r_a is ranked higher if confidence of r_a is greater than that of r_b,
2. r_a is ranked higher if confidence of r_a is the same as confidence of r_b, but support of r_a is greater than that of r_b,
3. r_a is ranked higher if r_a has shorter antecedent (fewer conditions) than r_b.

Fig. 1. Rule ranking criteria. Tie-breaking conditions applied if antecedents of two rules r_a and r_b match the same object.

It should be noted that the original CBA classifier contains two additional pruning steps: a) pessimistic pruning and b) replacement of rules performing worse than the majority class baseline with the default rule predicting the majority class. Pessimistic pruning is not featured in our setup, since it was not found to improve performance [5]. The omission of the default rule pruning in brCBA gives the user the control over the quality of the rule set, which can be influenced by the minimum confidence parameter, obtaining a *partial classifier* (not all objects may be labeled).

5.3 Classification and Rule Conflict Handling

If an input object matches exactly one rule, the classification step is very simple – the class contained in the consequent of the rule is assigned to the object. However, the output of association rule learning contains all too often an excessive number of redundant and conflicting rules. Employing rule pruning alleviates the number of conflicts since the number of redundant rules is reduced. Nevertheless pruning does not ensure that rule conflict will not emerge.

Rule conflict occurs if for a given object, there are at least two rules r_a and r_b, whose antecedents match the object. In practical terms, handling rule conflict is of importance if the consequents of these two rules are different, i.e. the rules assign a different class.

Association rules readily come with several scores that could be used to define a priority. These are primarily confidence and support, however additional measures such as chi-square or lift can be computed. The problem is thus to select, or combine these metrics into a total order, which would allow to solve ties between individual rules. brCBA uses the same method as CBA. In the first step, rules are sorted according to confidence, support and rule length – in the same way as in the data coverage pruning (see Fig. 1). The conflict is resolved by selecting the consequent of the top-ranked rule matching the object.

6 Experiments

The purpose of the experimental evaluation was to assess the impact of the following settings of association rule classifiers in the context of partial classification: data coverage rule pruning, dynamic binning, negated literals, and confidence/support thresholds.

6.1 Setup

Datasets. Experiments were performed on Iris, Balance Scale and Glass datasets from the UCI repository[9], which are frequently used for benchmarking classification systems. The use of a smaller number of datasets than in most related work allows us to present a detailed qualitative analysis of the results.
Preprocessing. Numerical attributes were discretized using equidistant binning with custom merging of bins with small support.
Rule learning. To perform the experiments, we used the LISp-Miner system[10] for learning association rules. LISp-Miner allows to perform learning of negative and disjunctive rules. Disjunctive rules (dynamic binning) are learnt through the setting of the LISp-Miner coefficient feature on individual input attributes to *subset* or, respectively, *sequence* type. The maximum length parameter λ was set to 2.[11]
Rule pruning. To perform rule pruning we used our Java implementation of the data coverage algorithm. This algorithm does not have any parameters.
Conflict resolution. We used the conflict resolution according to Fig. 1.

[9] `http://archive.ics.uci.edu/ml/`
[10] `http://lispminer.vse.cz`
[11] The system allows to enter also the minimum length parameter, which was left set to 1. For experiments involving negative rules, the system was set to consider both positive and negative version for each literal. The remaining parameters of the LISp-Miner system were left at their default values.

6.2 Results

The experimental results achieved on individual datasets are depicted on Table 2-5 in terms of accuracy and rule count. Accuracy is computed as $correct/N$, where *correct* is the number of correct predictions and N the total number of objects.

Since brCBA is a partial classifier, it may not assign a label to all objects. For this reason, we also provide complementary results using precision, which we compute as $correct/N_{cov}$, where N_{cov} is the number of covered (classified) objects. The plots depicted on Figure 2-5 provide accuracy and precision at minimum confidence levels 0.5, 0.6, 0.7, 0.8, 0.9 and 1.0 along with the average number of unclassified objects ($N - N_{cov}$).

All results are reported using ten fold cross validation with macro averaging.

	not pruned				pruned			
	without binning		sequence 1-2		without binning		sequence 1-2	
confidence	rules	accuracy	rules	accuracy	rules	accuracy	rules	accuracy
0.5	96	0.940	972.2	0.940	20	0.920	17	**0.953**
0.6	87	0.940	903.6	0.940	19	0.920	17	**0.953**
0.7	83	0.940	839.6	0.940	17	0.920	17	**0.953**
0.8	76	0.940	734.7	0.940	17	0.920	15	0.947
0.9	68	0.900	603.2	0.940	15	0.880	14	0.940

Table 2. Dataset: Iris, minimum support threshold: 7 objects (5.18%)

	not pruned				pruned			
	without binning		subset 1-2		without binning		subset 1-2	
confidence	rules	accuracy	rules	accuracy	rules	accuracy	rules	accuracy
0.6	124	**0.891**	11947	0.758	78	0.870	153	0.779
0.7	86	0.875	8462	0.826	70	0.864	153	0.779
0.8	50	0.790	4881	0.838	50	0.782	153	0.779
0.9	24	0.547	2193	0.838	24	0.547	153	0.779
1.0	1	0.047	1001	0.811	1	0.047	99	0.758

Table 3. Dataset: Balance Scale, minimum support threshold: 10 objects (1.78%)

Minimum support and confidence thresholds. Experimental results show that the lower minimum support threshold is generally associated with improved accuracy. This is demonstrated on Table 5.

For Iris and Balance Scale datasets the precision and accuracy do not react to an increase of minimum confidence within a certain interval (Figure 2-4). This phenomenon is encountered without respect to whether the pruning is turned on or off. This can be explained by the fact that the mining output for a given minimum confidence threshold contains also the higher confidence rules. If these higher confidence rules cover all test objects that are covered by the lower confidence rules, due to the conflict resolution strategy used the lower confidence

	not pruned				pruned			
	positive only		with negations		positive only		with negations	
confidence	rules	accuracy	rules	accuracy	rules	accuracy	rules	accuracy
0.5	58.3	0.529	1418.8	0.492	25.8	**0.534**	44.3	0.519
0.6	31.8	0.464	838.5	0.492	21.1	0.464	42.4	0.492
0.7	10.3	0.290	416.7	0.449	8.4	0.286	29.3	0.444
0.8	2.4	0.117	195.6	0.225	1.8	0.117	11.9	0.225
0.9	0.4	0.010	63.8	0.071	0.2	0.010	1.8	0.071

Table 4. Dataset: Glass, minimum support threshold: 10 objects (5.18%)

		not pruned		pruned	
Dataset, task	support	rules	accuracy	rules	accuracy
iris	7 (4.7%)	87	0.940	19	0.920
”	2 (1.3%)	168	0.947	21	0.913
”	1 (0.7%)	291	**0.967**	23	0.927
iris, sequence 1-2	7 (4.7%)	904	0.940	17	0.953
”	2 (1.3%)	1661	0.953	19	**0.960**
”	1 (0.7%)	2653	**0.960**	19	**0.960**
glass	10 (4.7%)	32	0.464	21	0.464
”	2 (0.9%)	2374	**0.622**	68	0.608
balance scale	10 (1.7%)	124	**0.891**	78	0.870
”	2 (0.4%)	558	0.841	216	0.714
balance scale, subset 1-2	10 (1.7%)	11947	0.758	153	**0.779**

Table 5. Impact of miminum support treshold. minimum confidence 0.6.

rules are never applied. The minimum confidence threshold thus starts to have effect once it removes rules which cover objects uncovered by any other higher confidence rule.

A Similar effect can be observed for the minimum support threshold. An optimal support threshold of 1% is reported in [5], [9] gives 2%, while [10] suggests 2% or 3%. Our results indicate that the best results are obtained with support threshold set to 1 object.[12]

Pruning. Experimental results show that pruning is an effective tool for reducing the number of rules without significantly affecting classification accuracy and precision. Without pruning, confidence and support thresholds need to be carefully chosen in order to balance number of rules and performance (Table 2-5). Pruning ensures a manageable number of rules even for low threshold values. For example, the best performing setup on iris dataset achieves accuracy of 0.967 with 291 rules, no test object is left unclassified. Pruning reduces the number of rules to only 23 with a slight drop in accuracy due to an increase in the number of unclassified objects (Fig. 2).

Negation and Dynamic Binning. Experiments performed on the Glass and Iris datasets explore the effect of negation (ref. to Table 4 and Fig. 4). The results

[12] This setup is referred to in the literature as “no support” mining.

show that involving negation in rule learning phase significantly increases the computational demands of the rule learner used, while the results are generally unaffected in terms of accuracy, and inflated in terms of rule count.

Sequence binning was performed on the Iris dataset, which contains only numerical attributes. The results for a higher minimum support thresholds indicate that sequence binning slightly improves performance (Table 2) while simultaneously decreasing rule count. While overall the best accuracy of 0.967 is achieved without binning (Table 5), the result obtained with a pruned set of rules featuring dynamically created bins (0.960) is only slightly worse, but is composed of a much smaller set of rules (19 vs 291). For the Balance Scale dataset, which contains nominal attributes, subset binning was performed. This highly computationally intensive operation did not provide accuracy improvement (Table 3).
Comparison with other algorithms. To compare with earlier reported results for CBA, the first two brCBA columns report results from runs, which were generated with similar rule learning settings of 50% min. confidence and 1% min. support thresholds, no dynamic binning and no negation. There is, however, some difference in data preprocessing of numerical attributes – with brCBA we used equidistant binning (see Example 2).

The results depicted on Table 6 indicate that the in terms of accuracy, brCBA with no pruning gives the best performance by thin margin on the iris dataset, but lags behind significantly on the glass dataset. Comparing runs with pruning, the additional pruning steps in the "full" CBA provide better accuracy. And, according to the comparison with the rule count reported in [5], even smaller rule count.

It should be emphasized that the conclusions drawn above are only indicative due to a small number of datasets involved in the benchmark.

	previous results [4, 15]					brCBA	
dataset	c4.5	ripper	cmar	cpar	cba	not pr.	pruned
iris	0.953	0.940	0.940	0.94.7	0.947	**0.967**	0.927
glass	0.687	0.691	0.701	**0.744**	0.739	0.622	0.612

Table 6. Comparison with other systems – accuracy.

7 Conclusion

This paper investigated the possibility of learning classification business rules from data using association rule learning algorithms.

We introduced brCBA, a modification of the CBA algorithm, which omits the default rule classification. This enabled us to demonstrate the sensitivity of rule count and accuracy on the minimum confidence and support thresholds. Also, our modified implementation used a more expressive rule learning system, which allowed to study the effect of involving rules with disjunction and negations.

Our experimental evaluation on several UCI datasets lead to the following recommendations for business rule learning with ARC algorithms:

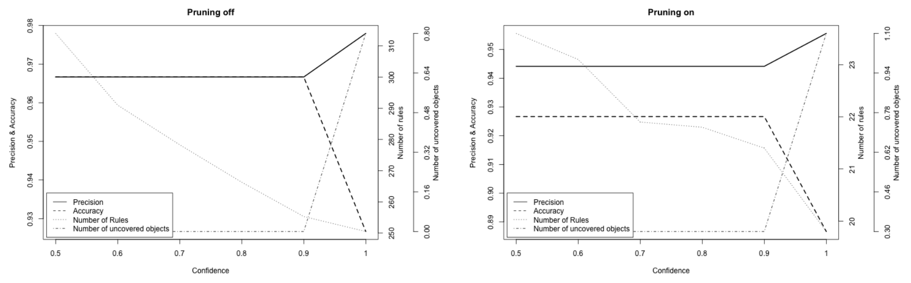

Fig. 2. Effect of pruning. Setting: Iris dataset, minimum support threshold 1

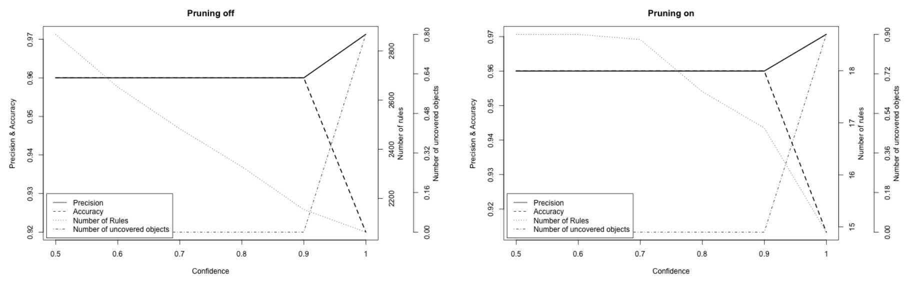

Fig. 3. Effect of dynamic binning on numerical attributes (sequence of length 2). Setting: Iris dataset, minimum support 1, dynamic binning on

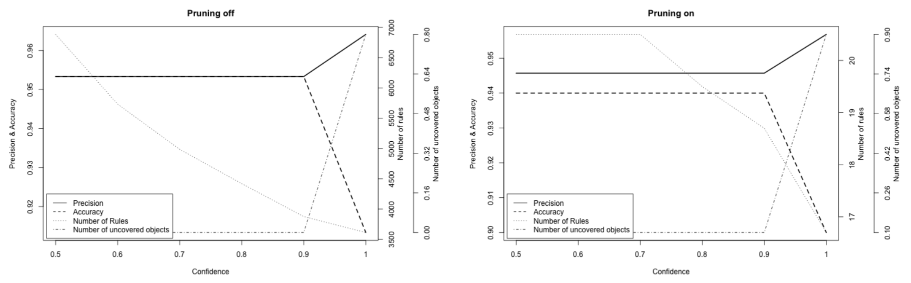

Fig. 4. Effect of including negative literals. Setting: Iris dataset, minimum support threshold 1

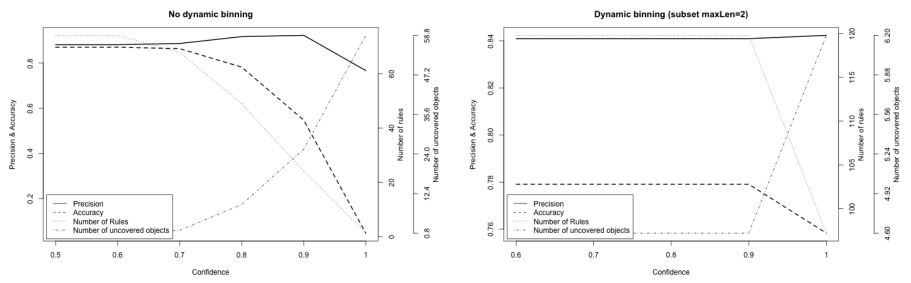

Fig. 5. Effect of dynamic binning on nominal attributes (subset of length 2). Setting: minimum support threshold 10, pruning on, Balance Scale dataset.

– The lowest confidence and support thresholds produce the best results. Since low threshold values have adverse effect on computational tractability, the setting of these thresholds is constrained by the available computational resources.
– Omission of important rules by pruning is a marginal, if any, issue, since pruned rule set maintains the accuracy of the original rule set on test data. Since pruning was at the same time found to significantly reduce the rule count, it is suitable for a business rule pruning setup.
– Involving higher expressiveness rules is not recommended given the substantial increase in computational demands and a negligible positive effect on accuracy and rule count (as opposed to default run with pruning).

It should be noted that the applicability of these recommendation is limited by the small number of the datasets involved in the experimental evaluation. Additionally, we have shown that the rule ranking algorithm used in CBA can be easily implemented as a rule conflict handling method in the Drools BRMS system, providing a complete workflow from data to actionable business rules.

As a future work, we plan to create an experimental web-based system that would allow to perform business rule learning with ARC algorithms. Also, we would like to further explore the topic of dynamic binning (disjunctions between values of one attribute), which provided promising results. It would be also interesting to perform additional experiments on a larger number of datasets.

Acknowledgment

The authors would like to thank the anonymous reviewers for their insightful comments. This work is supported by the European Union under grant no. FP7-ICT-2011-7 LinkedTV (Television Linked To The Web) and by the University of Economics in Prague by grants no. IGA 20/2013 and institutional support (IP 400040).

References

1. Rakesh Agrawal, Tomasz Imielinski, and Arun N. Swami. Mining association rules between sets of items in large databases. In *SIGMOD*, pages 207–216. ACM Press, 1993.
2. Maria-Luiza Antonie and Osmar R. Zaïane. Mining positive and negative association rules: An approach for confined rules. In *Proceedings of the 8th European Conference on Principles and Practice of Knowledge Discovery in Databases*, PKDD '04, pages 27–38, New York, NY, USA, 2004. Springer-Verlag New York, Inc.
3. Jiawei Han, Jian Pei, Yiwen Yin, and Runying Mao. Mining frequent patterns without candidate generation: A frequent-pattern tree approach. *Data Min. Knowl. Discov.*, 8(1):53–87, January 2004.
4. Wenmin Li, Jiawei Han, and Jian Pei. CMAR: accurate and efficient classification based on multiple class-association rules. In *ICDM'01*, pages 369–376, 2001.
5. Bing Liu, Wynne Hsu, and Yiming Ma. Integrating classification and association rule mining. In *KDD'98*, pages 80–86, 1998.

6. J. Ross Quinlan. *C4.5: Programs for Machine Learning.* Morgan Kaufmann, 1993.
7. Jan Rauch and Milan Šimůnek. An alternative approach to mining association rules. *Foundation of Data Mining and Knowl. Discovery*, 6:211–231, 2005.
8. Fadi Thabtah. Pruning techniques in associative classification: Survey and comparison. *Journal of Digital Information Management*, 4(3), 2006.
9. Fadi Thabtah, Peter Cowling, and Yonghong Peng. The impact of rule ranking on the quality of associative classifiers. In Max Bramer, Frans Coenen, and Tony Allen, editors, *Research and Development in Intelligent Systems XXII*, pages 277–287. Springer London, 2006.
10. Fadi Thabtah, Peter Cowling, and Yonghong Peng. Multiple labels associative classification. *Knowledge and Information Systems*, 9(1):109–129, 2006.
11. Fadi A. Thabtah. A review of associative classification mining. *Knowledge Eng. Review*, 22(1):37–65, 2007.
12. H. Toivonen, M. Klemettinen, P. Ronkainen, K. Htnen, and H. Mannila. Pruning and grouping discovered association rules. In *ECML'95 Workshop on statistics, Machine Learning and Knowledge Discovery in Databases*, pages 47–52, 1995.
13. K. Vanhoof and B. Depaire. Structure of association rule classifiers: a review. In *Intelligent Systems and Knowledge Engineering (ISKE), 2010 International Conference on*, pages 9–12, Nov 2010.
14. Stanislav Vojíř, Tomáš Kliegr, Andrej Hazucha, Radek Skrabal, and Milan Šimunek. Transforming association rules to business rules: Easyminer meets drools. In Paul Fodor, Dumitru Roman, Darko Anicic, Adam Wyner, Monica Palmirani, Davide Sottara, and François Lévy, editors, *RuleML (2)*, volume 1004 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2013.
15. Xiaoxin Yin and Jiawei Han. CPAR: Classification based on predictive association rules. In *Proceedings of the SIAM International Conference on Data Mining*, pages 369–376, San Franciso, 2003. SIAM Press.