A Brief Overview of Rule Learning

Johannes Fürnkranz[1] and Tomáš Kliegr[2](✉)

[1] Department of Computer Science, TU Darmstadt, Hochschulstraße 10, 64289 Darmstadt, Germany
juffi@ke.informatik.tu-darmstadt.de

[2] Department of Information and Knowledge Engineering, University of Economics, Prague, nám. Winstona Churchilla 4, 13067 Prague, Czech Republic
tomas.kliegr@vse.cz

Abstract. In this paper, we provide a brief summary of elementary research in rule learning. The two main research directions are descriptive rule learning, with the goal of discovering regularities that hold in parts of the given dataset, and predictive rule learning, which aims at generalizing the given dataset so that predictions on new data can be made. We briefly review key learning tasks such as association rule learning, subgroup discovery, and the covering learning algorithm, along with their most important prototypes. The paper also highlights recent work in rule learning on the Semantic Web and Linked Data as an important application area.

1 Introduction

Rule-based methods are a popular class of techniques in machine learning and data mining [19]. They share the goal of finding regularities in data that can be expressed in the form of an IF-THEN rule. Depending on the type of rule that should be found, we can discriminate between *descriptive rule discovery*, which aims at describing significant patterns in the given dataset in terms of rules, and *predictive rule learning*. In the latter case, one is often also interesting in learning a collection of the rules that collectively cover the instance space in the sense that they can make a prediction for every possible instance. In the following, we will briefly introduce both tasks and point out some key works in this area.

While in some application areas rule learning algorithms are superseded by statistical approaches such as Support Vector Machines (SVMs). An emerging use case for rule learning is the Semantic Web, whose representation is built on rule-based formalisms. We give a brief overview of recent papers in this domain, focusing on algorithms for completing large linked open data knowledge bases, such as DBpedia or YAGO.

This paper is organized as follows. Section 2 covers descriptive rule discovery algorithms, with emphasis on subgroup discovery and association rule mining. Section 3 discusses predictive rule discovery. This section includes the topic of classification by association rules, providing a connection to descriptive rule learning. The seminal algorithms of the rule learning field, including RIPPER

N. Bassiliades et al. (Eds.): RuleML 2015, LNCS 9202, pp. 54–69, 2015.
DOI: 10.1007/978-3-319-21542-6_4

and CN2, are presented in Section 4. Section 5 focuses on recent work in rule learning on the Semantic Web and Linked Data. The conclusion highlights some advantages of rule learning compared to its arguably biggest rival – decision tree learning, and points at emerging research in the linked data domain.

2 Descriptive Rule Discovery

In descriptive rule discovery, the key emphasis lies on finding rules that describe patterns and regularities that can be observed in a given dataset. In contrast to predictive rule learning (Section 3), the focus lies on finding individual rules. Consequently, evaluation does typically not focus on predictive performance, but on the statistical validity of the found rules. Predominant in the literature are two main tasks, namely subgroup discovery, where a given property of interest is analyzed (supervised learning), and association rule discovery, where arbitrary dependencies between attributes can be considered (unsupervised learning).

2.1 Subgroup Discovery

The task of subgroup discovery was defined by Klösgen [29] and Wrobel [59] as follows: *Given a population of individuals and a property of those individuals that we are interested in, find population subgroups that are statistically 'most interesting', e.g., are as large as possible and have the most unusual statistical (distributional) characteristics with respect to the property of interest.*

Thus, a subgroup may be considered as an IF-THEN rule that relates a set of independent variables to a target variable of interest. The condition of the rule (the *rule body* or *antecedent*) typically consists of a conjunction of Boolean terms, so-called *features*, each one constituting a constraint that needs to be satisfied by an example. If all constraints are satisfied, the rule is said to *fire*, and the example is said to be *covered* by the rule. The *rule head* (also called the *consequent* or *conclusion*) consists of a single class value, which is predicted in case the rule fires. In the simplest case, this is a binary target class c, and we want to find one or more rules that are predictive for this class.

In the literature, one can also find several closely related tasks, where the head of the rule does not only consist of a single binary attribute. Examples include mining for subgroup discovery, contrast sets [4], correlated pattern mining [40], mining for emerging patterns [11], exceptional model mining, and others. For more information, we refer to Kralj Novak et al. [30] and Zimmermann and De Raedt [64], who present unifying frameworks for these approaches.

The rule bodies typically consist of features that test for the presence of a particular attribute value or, in the case of numerical attributes, of an inequality that requires that the observed value is above or below a threshold. More expressive constraints include *set-valued attributes* (several values of the same attribute can be observed in the training examples), *internal disjunctions* (only one of several values of the same attribute needs to be present), *hierarchical attributes* (certain values of the attributes subsume other values), etc. Conjunctive combinations of features may be viewed as statements in propositional logic

function FindPredictiveRule (Examples)

Input: *Examples*, a set of positive and negative examples for a class c.

//initialize the rule body
$rb \leftarrow \emptyset$
// repeatedly find the best refinement
repeat
    build refinements $R \leftarrow \{rb' \mid rb' = rb \wedge f, \text{ for some feature } f\}$
    evaluate all $rb' \in R$ according to some quality criterion
    $rb =$ the best refinement in R
until rb satisfies a stopping criterion
    or covers no examples

Output: rule $(c \leftarrow R)$

Fig. 1. Greedy search for a predictive rule

(propositional rules). If relations between features can be considered (i.e., if propositions can be formulated in first-order logic), we speak of first-order rules.

Top-Down Hill-Climbing Algorithm. Figure 1 shows a simple greedy hill-climbing algorithm for finding a single predictive rule. It starts with an empty rule body and successively adds new conditions. For adding a condition, it tries all possible additions and evaluates them with a heuristic quality criterion, which typically depends on the number of covered and uncovered examples that belong to the class c (positive examples) or do not belong to c (negative examples). A few important ones are (assume that p out of P positive examples and n out of N negative examples are covered by the rule):

Laplace estimate (Lap $= \frac{p+1}{p+n+2}$) computes the fraction of positive examples in all covered examples, where each class is initialized with 1 virtual example in order to penalize rules with low coverage.

m-estimate (m $= \frac{p+m \cdot P/(P+N)}{p+n+m}$) is a generalization of the Laplace estimate which uses m examples for initialization, which are distributed according to the class distribution in the training set [7].

information gain (ig $= p \cdot (\log_2 \frac{p}{p+n} - \log_2 \frac{p'}{p'+n'}$), where p' and n' are the number of positive and negative examples covered by the rule's predecessor) is Quinlan's (1990) adaptation of the information gain heuristic used for decision tree learning. The main difference is that this only focuses on a single branch (a rule), whereas the decision tree version tries to optimize all branches simultaneously.

correlation and χ^2 ($\text{corr} = \frac{p(N-n)-(P-p)n}{\sqrt{PN(p+n)(P-p+N-n)}}$) computes the four-field correlation of covered/uncovered positive/negative examples. It is equivalent to a χ^2 statistic ($\chi^2 = (P+N)\,\text{corr}^2$).

An exhaustive overview and theoretical comparison of various search heuristics in coverage space, a variant of ROC space, can be found in [18].

In the simplest case, conditions are added until the rule covers no more negative examples. In practical applications, we may want to stop earlier in order to avoid overfitting. In this case, a separate *stopping criterion* may be used in order to stop the refinement process when a certain quality threshold for the learned rule is satisfied, or the rule set may be optimized on an independent pruning set [16].

A greedy hill-climbing search is quite likely to get stuck in a local optimum. However, it is fairly straight-forward to generalize this algorithm so that different search strategies can be employed (e.g., beam search [9] or best-first search) or that not only one but multiple rules are returned (typically the top-k rules for some value of k).

2.2 Association Rule Discovery

An association rule is a rule where certain properties of the data in the body of the rule are related to other properties in the head of the rule.

A typical application example for association rules are product associations. For example, the rule

$$\texttt{bread}, \texttt{butter} \rightarrow \texttt{milk}, \texttt{cheese}$$

specifies that people who buy bread and butter also tend to buy milk and cheese.

The importance of an association rule is often characterized with two measures:

Support measures the fraction of all rows in the database that satisfy both, body and head of the rule. Rules with higher support are more important.

Confidence measures the fraction of the rows that satisfy the body of the rule, which also satisfy the head of the rule. Rules with high confidence have a higher correlation between the properties described in the head and the properties described in the body.

If the above rule has a support of 10% and a confidence of 80%, this means that 10% of all people buy bread, butter, milk, and cheese together, and that 80% of all people who buy bread and butter also buy milk and cheese.

Apriori Algorithm. The discovery of association rules typically happens in two phases, which were pioneered in the Apriori algorithm [2]. First, all *frequent itemsets* (i.e., conditions that cover a certain minimum number of examples) are found. In a second pass, these are then converted into association rules.

For finding all frequent itemsets, Apriori generates all rules with a certain minimum frequency in parallel with a so-called *level-wise search*, as shown in

function FreqSet (Examples)

Input: *Examples*, described with a set of binary features, so-called *Items*.

// the first iteration consists of all single items $k = 1$
$C_1 = Items$

//loop until no nor candidate items left **while** $C_k \neq \emptyset$ **do**

// remove all infrequent items from C_k
// (requires check on database of *Examples*)
$S_k = C_k \setminus \{$ all infrequent itemsets in $C_k\}$

// generate new candidates
$C_{k+1} = \{$ all sets with $k+1$ elements that
can be formed by uniting two itemsets in $S_k\}$

$C_{k+1} = C_{k+1} \setminus \{$ all itemsets for which not all subsets of size k
are contained in $S_k\}$

$S = S \cup S_k$
$k = k + 1$

endwhile

Output: S, the set of all frequent itemsets

Fig. 2. Find all Frequent Itemsets

Figure 2. The level-wise search first generates all frequent itemsets of size one, then all frequent itemsets of size two, and so on, thereby performing a breadth-first search. However, from each iteration to the next, a large number of possible extensions can be pruned because of the *anti-monotonicity* of the frequency of the itemsets (their *support*). This essentially means that if a conjunction of conditions is extended with a new condition, the resulting rule body will only cover a subset of the examples covered by the original rule body. Thus, when computing C_{k+1}, the set of candidate itemsets of size $k+1$, we only need to consider itemsets that result as a combination of two itemsets of size k which overlap in $k-1$ items. For example, if the two itemsets $\{A, B, C\}$ and $\{B, C, D\}$ are in S_3, the itemset $\{A, B, C, D\}$ will be in C_4. It may be later removed if either one of its subsets of size 3 is not frequent (if, e.g., $\{A, C, D\}$ is not contained in S_3), or if the subsequent check on the dataset shows that it is itself not frequent.

The resulting frequent itemsets are then used for constructing rules in a post-processing phase. The key idea here is to try all possible ways of using an implication sign to separate a frequent itemset into items that are used in the rule body and items that are used in the rule head, and keeping only those where the resulting association rule has a certain minimum strength (confidence). This can, again, be sped up considerably using a similar idea to the anti-monotonicity of the support.

Apriori Successors. While the second phase of Apriori remains almost unchanged, a number of alternative algorithms, such as Eclat [62] or FP-Growth [25], have been proposed for the frequent itemset discovery phase. Mining for *closed* frequent itemsets proposed by Pasquier et al. [46] is another optimization. A frequent itemset P is closed if P is included in no other itemset that has the same support as P.

In recent years there was a growing interest in approaches that support parallel execution of frequent itemset mining in order to harness modern multi-core architectures. PLCM [45] and MT-Closed [38] are parallel implementations of two fastest algorithms LCMv2 [56] and DCI Closed [37] according to the FIMI'04 workshop[1], which provided a benchmark of submitted frequent itemset mining implementations [44]. The recently proposed ParaMiner [44] algorithm yields comparable execution times to PLCM and MT-Closed, while it allows to mine not only for closed frequent itemsets, but also for additional types of patterns such as connected relational graphs and gradual itemsets.

For surveys of frequent set mining and association rule discovery we refer the reader to [22,63]. A freely accessible implementations of multiple frequent itemset mining implementations can be found at http://borgelt.net/fpm.html, ParaMiner is also made available by the authors under an open license.

Connections to Mathematical Logic and Statistics. The notion of association rules was introduced already in mid 1960's by Petr Hájek in the frame of development of the GUHA method (abbrev. of General Unary Hypothesis Automaton) [23]. The purpose was to automatically generate large number of (statistical) hypotheses which had the form of association rules. These hypotheses are automatically verified using a number of criteria, including Chi-square and Fisher statistical tests and what is now known as support and confidence. The hypotheses that pass the criteria are represented as (true) logical formulas of *observational calculi*, a theoretical framework for exploratory data analysis combining logic and mathematical statistics. Example of such a formula is:

$$\texttt{bread(brown)} \wedge \texttt{butter(yes)} \Longrightarrow_{B,p} \texttt{milk(skimmed)} \wedge \texttt{cheese(french)}$$

This example features the *founded implication* quantifier $\Longrightarrow_{B,p}$, which asserts that the support of the rule is at least B instances and the confidence is at least p. Observational calculi are further studied by Rauch [54]. One practical result is the introduction of deduction rules, which allow to identify redundant hypotheses a to deal with domain knowledge.

A maintained implementation of GUHA method is LISp-Miner, which is freely available from lispminer.vse.cz. This software supports the distinct GUHA features such as negated literals, e.g. $\neg$`bread(brown)`, and disjunctions, e.g. `bread(brown)` $\vee$ `butter(yes)`, or `cheese(french` $\vee$ `dutch)`. The higher expressiveness leads to a considerable increase in computational cost [28]. Kliegr et al. [28] suggested that GUHA may find use in business rule learning, where a lower number of more expressive rules can be desirable.

[1] http://fimi.ua.ac.be/fimi04/

3 Predictive Rule Learning

Whereas descriptive rule discovery aims at finding individual rules that capture some regularities and patterns of the input data, the task of predictive rule learning is to generalize the training data so that predictions for new examples are possible. As individual rules will typically only cover part of the training data, we will need to enforce completeness by learning an unordered rule set or a decision list.

An *unordered rule set* is a collection of individual rules that collectively form a classifier. In contrast to a decision list, the rules in the set do not have an inherent order, and all rules in the set have to be tried for deriving a prediction for an example. This may cause two types of problems that have to be resolved with additional algorithms:

Multiple rules fire: More than one rule can fire on a single example, and these rules can make contradicting predictions. This type of conflict is typically resolved by preferring rules that cover a higher fraction of training examples of their class (typically estimated with Laplace correction). This is equivalent to converting the rule set into a decision list that is ordered according to this evaluation heuristic. More elaborate tie breaking schemes, such as using the *Naive Bayes* algorithm, or inducing a separate rule set for handling these conflicts (*double induction* [32]) have also been tried.

No rules fire: It may also occur that no rule fires for a given example. Such cases are typically handled via a so-called *default rule*, which typically predicts the majority class. Again, more complex algorithms, such as FURIA [26] trying to find the closest rule (*rule stretching* [13]) have been proposed.

A rule set in which all rules predict the same class needs to be complemented with an (implicit) default rule that predicts the other class in case none of the previous rules fires (very much like the closed world semantics in PROLOG). If all rules are conjunctive, such rule sets may be interpreted as a definition in *disjunctive normal form* for this class.

In contrast to an unordered rule set, a *decision list* has an inherent order, which makes classification quite straightforward. For classifying a new instance, the rules are tried in order, and the class of the first rule that covers the instance is predicted. If no induced rule fires, a *default rule* is invoked, which typically predicts the majority class of the uncovered training examples. Decision lists are particularly popular in *inductive logic programming* [10,12], because PROLOG programs may be considered to be simple decision lists, where all rules predict the same concept.

Both decision trees and rule sets are often learned with the same or very similar strategies. The two most popular strategies for learning rule sets may be viewed as extensions of the association rule and subgroup discovery algorithms discussed in the previous section, and are discussed in the following.

3.1 Classification by Association

A prototypical instantiation of this framework is *associative classification*, as exemplified by the CBA rule learning algorithm [35,36]. This type of algorithm typically uses a conventional association rule discovery algorithm, such as Apriori [2], to discover a large number of patterns. From these, all patterns that have the target class in the head are selected, and only those are subsequently used for inducing a rule set. This is formed by sorting the patterns according to some heuristic function and adding the best to the rule set.

A variety of successor systems have been proposed that follow the same principal architecture [e.g., 5,27,31,43,60]. Sulzmann and Fürnkranz [55] compare various approaches for combining association rules into a rule-based theory. Azevedo and Jorge [3] propose to generate an ensemble of rule sets instead of a single rule set.

CBA and its direct successors such as CMAR are restricted to nominal attributes. If the dataset contains numeric (quantitative) attributes, these attributes need to be discretized e.g. using the minimum description length principle [14]. This is a severe limitation compared to many other learning algorithms which natively handle numerical attributes.

As in association rule discovery, there are approaches to associative classification that employ fuzzy logic to alleviate this problem. A recent example of such an approach is the FARC-HD algorithm [1]. Alcala-Fdez et al. [1] also provide a benchmark comparing their algorithm against the C4.5 decision tree learner as well as against multiple association rule classification algorithms including CBA, CBA2, CPAR and CMAR. The results show that FARC-HD provides a slight improvement in average accuracy across the basket of 25 datasets but at a several orders of magnitude higher computational cost. The benchmark also reveals large differences in the size of the rule set among classifiers. While CBA achieves slightly smaller accuracy than its successor algorithms CPAR and CMAR, it produces a notably smaller number of rules.

Free implementations of CBA, CMAR and CPAR are available at http://cgi.csc.liv.ac.uk/~frans/KDD/Software/. A good survey of associative classification and related algorithms can be found in [6].

3.2 Covering Algorithm

An alternative approach, the so-called *covering* or *separate-and-conquer* algorithm, relies on repeatedly learning a single rule (e.g., with a subgroup discovery algorithm). After a new rule has been learned, all examples that are covered by this rule are removed. This is repeated until all examples are covered or a given stopping criterion fires. A simple version of this so-called *covering* algorithm is shown in Figure 3, a survey of this family of algorithms can be found in [17]. The members of this family differ mostly in the way the FindPredictiveRule method is implemented.

procedure COVERING (Examples,Classifier)

Input: *Examples*, a set of positive and negative examples for a class c.

// initialize the rule set
$R = \emptyset$

//loop until no more positive examples are covered
while not all positive examples are covered **do**
    // find the best rule for the current examples
    r = FINDPREDICTIVERULE (Examples)

    // check if we need more rules
    if $R \cup r$ is good enough
    then break while

    // remove covered examples and add rule to rule set
    Examples = Examples \ { examples covered by r}
    $R = R \cup r$
endwhile

Output: the learned rule set R

Fig. 3. The covering algorithm for finding a rule set

4 Well-Known Rule Learning Algorithms

AQ can be considered as the original covering algorithm. Its original version was conceived by Ryszard Michalski in the sixties [39], and numerous versions and variants of the algorithm appeared subsequently in the literature. AQ uses a top-down beam search for finding the best rule. It does not search all possible specializations of a rule, but only considers refinements that cover a particular example, the so-called *seed example*. This idea is basically the same as the use of a bottom clause in inductive logic programming [10,41,42].

CN2 [8,9] employs a beam search guided by the Laplace or m-estimates, and the above-mentioned likelihood ratio significance test to fight overfitting. It can operate in two modes, one for learning rule sets (by modeling each class independently), and one for learning decision lists.

FOIL [51] was the first relational learning algorithm that received attention beyond the field of inductive logic programming. It learns a concept with the covering loop and learns individual concepts with a top-down refinement operator, guided by information gain. The main difference to previous systems is that FOIL allowed the use of first-order background knowledge. Instead of only being able to use tests on single attributes, FOIL could employ tests that compute relations between multiple attributes, and could also introduce new variables in the body of a rule.

RIPPER was the first rule learning system that effectively countered the overfitting problem via *incremental reduced error pruning* [16]. It also added a post-processing phase for optimizing a rule set in the context of other rules. The key idea is to remove one rule out of a previously learned rule set and try to re-learn it not only in the context of previous rules (as would be the case in the regular covering rule), but also in the context of subsequent rules. RIPPER is still state-of-the-art in inductive rule learning. A freely accessible re-implementation can be found in the WEKA machine learning library [58] under the name of JRIP.

OPUS [57] was the first rule learning algorithm to demonstrate the feasibility of a full exhaustive search through all possible rule bodies for finding a rule that maximizes a given quality criterion (or heuristic function). The key idea is the use of *ordered search* that prevents that a rule is generated multiple times. This means that even though there are $l!$ different orders of the conditions of a rule of length l, only one of them can be taken by the learner for finding this rule. In addition, OPUS uses several techniques that prune significant parts of the search space, so that this search method becomes feasible. Follow-up work has shown that this technique is also an efficient alternative for association rule discovery, provided that the database to mine fits into the memory of the learning system.

5 Applications in Linked Data and Semantic Web

While research in machine learning currently tends to move away from learning logical concept representations towards statistical learning algorithms, rules are still used in many application areas. A particularly important case is the Semantic Web, whose representation is built on rule-based formalisms. As it is difficult to manually write a complete set of rules for representing knowledge, rule learning algorithms have great potential in supporting automation of this process.

Inductive logic programming algorithms are one obvious candidate for this purpose, because they allow to operate in more expressive, relational logical frameworks such as RDF[2] or OWL[3], which form the backbone of the Semantic Web [33,34]. However, their expressiveness has to be paid for with a high computational complexity. Compared to approaches based on inductive logic programming (ILP), APRIORI and its successors are not only much more efficient, but also they do not require counter examples [20], on which most ILP approaches rely. This is important because semantic knowledge bases such as DBpedia (http://dbpedia.org) do not contain negative statements. Additionally, since they are built under the *open world assumption*[4], the negative statements cannot be directly inferred. It was observed that semantic reasoners may not provide meaningful results on real open world knowledge bases yet for another reason: these crowd-sourced resources contain errors. A single erroneous fact can cause the RDFS reasoner to infer an incorrect statement [49].

[2] http://www.w3.org/TR/rdf-primer/
[3] http://www.w3.org/TR/owl2-primer/
[4] A statement which is not present in the knowledge base is not necessarily false.

A current use case demonstrating advantages of association rule learning in the linked data domain is the completion of the large DBpedia knowledge base. Association rules were applied to infer missing types for entities in [48] and to perform schema induction (infer new classes) in [61]. These approaches for DBpedia completion directly use the Apriori algorithm, which implies limitations stemming from the inherently relational setting of linked data. AMIE [20] is a state-of-the-art algorithm that extends the association rule learning principles allowing to mine Horn clauses such as

$$\texttt{hasAdvisor(x, y)} \wedge \texttt{graduateFrom(x, z)} \implies \texttt{worksAt(y, z)}$$

AMIE is reported to be highly computationally efficient, it processes entire DBpedia in less than 3 minutes and the larger YAGO2 ontology (www.mpi-inf.mpg.de/yago/) in 4 minutes. In contrast, the authors report that in their benchmark state-of-the-art ILP approaches did not finish within days.

Rule learning may not only support the construction of Semantic Web resources, but, conversely, the Semantic Web may also serve as a source for background knowledge in many data mining tasks. For example, Paulheim and Fürnkranz [50] have shown that unsupervised feature generation from various knowledge sources in the Linked Open Data (LOD) cloud may yield interesting and useful features. One can even go as far as trying to mine databases that have no inherent background knowledge. For example, Paulheim [47] used LOD knowledge for trying to find explanation for common statistics such as the quality-of-living index of cities.

This short survey shows that rule learning algorithms can be with success directly applied to large linked datasets available on the "Semantic Web". Apart from the inference of new facts or identification of errors in semantic knowledge bases, it was recently suggested that association rule learning can serve e.g. for schema alignment between ontologies [21]. There is an ongoing research into specialized approaches tailored for RDF datasets which opens new opportunities as well as challenges.

6 Conclusion

This paper provided a brief introduction to rule learning, mainly focusing on the best-known algorithms for descriptive and predictive rule learning. Whereas the main goal of association rule and subgroup discovery is to discover single rules that capture patterns in parts of the data, the main task of classification by association and the covering strategy for learning predictive rule sets and decision lists is to be able to generalize the training data so that predictions on new data can be made. In comparison with other popular classification algorithms such as Support Vector Machines, predictive rule learning together with decision trees has the advantage of easy interpretability. The individual rules that comprise the classifier can be explained to a human expert.

Obviously, this brief survey is far from complete. Other techniques for generating rule sets are possible. For example, rules can be generated from induced

decision trees. Standard algorithms for learning decision trees (such as C4.5 [53]) are quite similar to the covering algorithm for learning decision lists in that the aim of extending a decision tree with another split is to reduce the class impurity in the leaves (usually measured by entropy or the Gini index). However, whereas a decision tree split is chosen to optimize all successor branches simultaneously, a rule learning heuristic only focuses on a single rule. As a result, rule sets are often more compact than decision trees. Consequently, a rule set can be considerably simplified during the conversion of a decision tree to a set of rules [52,53]. For example, Frank and Witten [15] suggested the PART algorithm, which tries to integrate this simplification into the tree induction process by focusing only on a single branch of a tree.

The APRIORI algorithm [2], which provides means to discover association rules in large datasets, is considered as one of the major advancements in data mining technology in the seminal book of Hastie et al. [24]. Its recent successors, such as the LCM group of algorithms provide further improvements in terms of computational efficiency. Other algorithms, such as PARAMINER provide generic framework allowing to discover not only frequent itemsets but also other types of patterns. The performance of parallel implementations of association rule learning stimulates novel applications on large datasets that are becoming freely available as part of the linked open data initiative. Examples of such efforts include completion of semantic knowledge bases with new facts.

Acknowledgment. Tomáš Kliegr was partly supported by the Faculty of Informatics and Statistics, University of Economics, Prague within "long term institutional support for research activities" scheme and grant IGA 20/2013.

References

1. Alcala-Fdez, J., Alcala, R., Herrera, F.: A fuzzy association rule-based classification model for high-dimensional problems with genetic rule selection and lateral tuning. IEEE Transactions on Fuzzy Systems **19**(5), 857–872 (2011)
2. Agrawal, R., Imielinski, T., Swami, A.N.: Mining association rules between sets of items in large databases. In: Buneman, P., Jajodia, S. (eds.) Proceedings of the ACM International Conference on Management of Data (SIGMOD 1993), Washington, D.C., pp. 207–216 (1993)
3. Azevedo, P.J., Jorge, A.J.: Ensembles of jittered association rule classifiers. Data Mining and Knowledge Discovery **21**(1), 91–129 (2010). Special Issue on Global Modeling using Local Patterns
4. Bay, S.D., Pazzani, M.J.: Detecting group differences: Mining contrast sets. Data Mining and Knowledge Discovery **5**(3), 213–246 (2001)
5. Bayardo Jr., R.J.: Brute-force mining of high-confidence classification rules. In: Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining (KDD 1997), pp. 123–126 (1997)
6. Bringmann, B., Nijssen, S., Zimmermann, A.: Pattern-based classification: a unifying perspective. In: Knobbe, A., Fürnkranz, J. (eds.) Proceedings of the ECML/PKDD 1909 Workshop From Local Patterns to Global Models (LeGo 1909), Bled, Slovenia, pp. 36–50 (2009)

7. Cestnik, B.: Estimating probabilities: a crucial task in Machine Learning. In: Aiello, L. (ed.) Proceedings of the 9th European Conference on Artificial Intelligence (ECAI 1990), Pitman, Stockholm, Sweden, pp. 147–150 (1990)
8. Clark, P., Boswell, R.: Rule induction with CN2: Some recent improvements. In: Kodratoff, Y. (ed.) Machine Learning – EWSL-91. LNCS, vol. 482, pp. 151–163. Springer, Heidelberg (1991)
9. Clark, P., Niblett, T.: The CN2 induction algorithm. Machine Learning **3**(4), 261–283 (1989)
10. De Raedt, L.: Logical and Relational Learning. Springer-Verlag (2008)
11. Dong, G., Li, J.: Efficient mining of emerging patterns: discovering trends and differences. In: Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 1999), San Diego, CA, pp. 43–52 (1999)
12. Džeroski, S., Lavrač, N. (eds.): Relational Data Mining: Inductive Logic Programming for Knowledge Discovery in Databases. Springer-Verlag (2001)
13. Eineborg, M., Boström, H.: Classifying uncovered examples by rule stretching. In: Rouveirol, C., Sebag, M. (eds.) ILP 2001. LNCS (LNAI), vol. 2157, pp. 41–50. Springer, Heidelberg (2001)
14. Fayyad, U.M., Irani, K.B.: Multi-interval discretization of continuous-valued attributes for classification learning. In: Proceedings of the 13th International Joint Conference on Artificial Intelligence (IJCAI 1993), pp. 1022–1029 (1993)
15. Frank, E., Witten, I.H.: Generating accurate rule sets without global optimization. In: Shavlik, J. (ed.) Proceedings of the 15th International Conference on Machine Learning (ICML 1998), pp. 144–151. Morgan Kaufmann, Madison (1998)
16. Fürnkranz, J.: Pruning algorithms for rule learning. Machine Learning **27**(2), 139–171 (1997)
17. Fürnkranz, J.: Separate-and-conquer rule learning. Artificial Intelligence Review **13**(1), 3–54 (1999)
18. Fürnkranz, J., Flach, P.A.: ROC 'n' rule learning - Towards a better understanding of covering algorithms. Machine Learning **58**(1), 39–77 (2005)
19. Fürnkranz, J., Gamberger, D., Lavrač, N.: Foundations of Rule Learning. Springer-Verlag (2012)
20. Galárraga, L.A., Teflioudi, C., Hose, K., Suchanek, F.: AMIE: association rule mining under incomplete evidence in ontological knowledge bases. In: Proceedings of the 22nd International Conference on World Wide Web (WWW 2013), Switzerland, pp. 413–422 (2013)
21. Galárraga, L.A., Preda, N., Suchanek, F.M.: Mining rules to align knowledge bases. In: Proceedings of the 2013 Workshop on Automated Knowledge Base Construction (AKBC 2013), pp. 43–48. ACM, New York (2013)
22. Goethals, B.: Frequent set mining. In: Maimon, O., Rokach, L. (eds.) The Data Mining and Knowledge Discovery Handbook, 2nd edn., pp. 321–338. Springer-Verlag (2010)
23. Hájek, P., Holena, M., Rauch, J.: The GUHA method and its meaning for data mining. Journal of Computer and System Sciences **76**(1), 34–48 (2010). Special Issue on Intelligent Data Analysis
24. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning. Springer Series in Statistics. Springer, New York (2001)
25. Han, J., Pei, J., Yin, Y., Mao, R.: Mining frequent patterns without candidate generation: A frequent-pattern tree approach. Data Mining and Knowledge Discovery **8**(1), 53–87 (2004)

26. Hhn, J., Hllermeier, E.: Furia: an algorithm for unordered fuzzy rule induction. Data Mining and Knowledge Discovery **19**(3), 293–319 (2009)
27. Jovanoski, V., Lavrač, N.: Classification rule learning with APRIORI-C. In: Brazdil, P.B., Jorge, A.M. (eds.) EPIA 2001. LNCS (LNAI), vol. 2258, pp. 44–51. Springer, Heidelberg (2001)
28. Kliegr, T., Kuchař, J., Sottara, D., Vojíř, S.: Learning business rules with association rule classifiers. In: Bikakis, A., Fodor, P., Roman, D. (eds.) RuleML 2014. LNCS, vol. 8620, pp. 236–250. Springer, Heidelberg (2014)
29. Klösgen, W.: Explora: a multipattern and multistrategy discovery assistant. In: Fayyad, U.M., Piatetsky-Shapiro, G., Smyth, P., Uthurusamy, R. (eds.) Advances in Knowledge Discovery and Data Mining, chap. 10, pp. 249–271. AAAI Press (1996)
30. Kralj Novak, P., Lavrač, N., Webb, G.I.: Supervised descriptive rule discovery: A unifying survey of contrast set, emerging pattern and subgroup mining. Journal of Machine Learning Research **10**, 377–403 (2009)
31. Li, W., Han, J., Pei, J.: CMAR: accurate and efficient classification based on multiple class-association rules. In: Proceedings of the IEEE Conference on Data Mining (ICDM 2001), pp. 369–376 (2001)
32. Lindgren, T., Boström, H.: Resolving rule conflicts with double induction. Intelligent Data Analysis **8**(5), 457–468 (2004)
33. Lisi, F.: Building Rules on Top of Ontologies for the Semantic Web with Inductive Logic Programming. Theory and Practice of Logic Programming **8**(3), 271–300 (2008)
34. Lisi, F., Esposito, F.: An ilp perspective on the semantic web. In: Bouquet, P., Tummarello, G. (eds.) Semantic Web Applications and Perspectives - Proceedings of the 2nd Italian Semantic Web Workshop (SWAP-05), pp. 14–16. University of Trento, Trento (2005)
35. Liu, B., Hsu, W., Ma, Y.: Integrating classification and association rule mining. In: Agrawal, R., Stolorz, P., Piatetsky-Shapiro, G. (eds.) Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining (KDD 1998), pp. 80–86 (1998)
36. Liu, B., Ma, Y., Wong, C.K.: Improving an association rule based classifier. In: Zighed, D.A., Komorowski, J., Żytkow, J.M. (eds.) PKDD 2000. LNCS (LNAI), vol. 1910, pp. 504–509. Springer, Heidelberg (2000)
37. Lucchese, C.: DCI closed: a fast and memory efficient algorithm to mine frequent closed itemsets. In: Proceedings of the IEEE ICDM 2004 Workshop on Frequent Itemset Mining Implementations (FIMI 2004) (2004)
38. Lucchese, C., Orlando, S., Perego, R.: Parallel mining of frequent closed patterns: harnessing modern computer architectures. In: Proceedings of the 7th IEEE International Conference on Data Mining (ICDM 2007), pp. 242–251 (2007)
39. Michalski, R.S.: On the quasi-minimal solution of the covering problem. In: Proceedings of the 5th International Symposium on Information Processing (FCIP-69) (Switching Circuits), vol. A3, Bled, Yugoslavia, pp. 125–128 (1969)
40. Morishita, S., Sese, J.: Traversing itemset lattice with statistical metric pruning. In: Proceedings of the 19th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS 2000), pp. 226–236. ACM (2000)
41. Muggleton, S.H.: Inverse entailment and Progol. New Generation Computing **13**(3,4), 245–286 (1995). Special Issue on Inductive Logic Programming
42. Muggleton, S.H., De Raedt, L.: Inductive Logic Programming: Theory and methods. Journal of Logic Programming **19–20**, 629–679 (1994)

43. Mutter, S., Hall, M., Frank, E.: Using classification to evaluate the output of confidence-based association rule mining. In: Webb, G.I., Yu, X. (eds.) AI 2004. LNCS (LNAI), vol. 3339, pp. 538–549. Springer, Heidelberg (2004)
44. Negrevergne, B., Termier, A., Rousset, M.C., Mhaut, J.F.: Para miner: a generic pattern mining algorithm for multi-core architectures. Data Mining and Knowledge Discovery **28**(3), 593–633 (2014)
45. Negrevergne, B., Termier, A., Rousset, M.C., Mhaut, J.F., Uno, T.: Discovering closed frequent itemsets on multicore: parallelizing computations and optimizing memory accesses. In: Proceedings of the International Conference on High Performance Computing and Simulation (HPCS 2010), pp. 521–528 (2010)
46. Pasquier, N., Bastide, Y., Taouil, R., Lakhal, L.: Discovering frequent closed itemsets for association rules. In: Beeri, C., Bruneman, P. (eds.) ICDT 1999. LNCS, vol. 1540, pp. 398–416. Springer, Heidelberg (1998)
47. Paulheim, H.: Generating possible interpretations for statistics from linked open data. In: Simperl, E., Cimiano, P., Polleres, A., Corcho, O., Presutti, V. (eds.) ESWC 2012. LNCS, vol. 7295, pp. 560–574. Springer, Heidelberg (2012)
48. Paulheim, H., Browsing linked open data with auto complete. In: Proceedings of the Semantic Web Challenge co-located with ISWC 2012. Univ., Mannheim, Boston (2012)
49. Paulheim, H., Bizer, C.: Type inference on noisy rdf data. In: Alani, H., Kagal, L., Fokoue, A., Groth, P., Biemann, C., Parreira, J.X., Aroyo, L., Noy, N., Welty, C., Janowicz, K. (eds.) ISWC 2013, Part I. LNCS, vol. 8218, pp. 510–525. Springer, Heidelberg (2013)
50. Paulheim, H., Fürnkranz, J.: Unsupervised feature construction from linked open data. In: Proceedings of the ACM International Conference Web Intelligence, Mining, and Semantics (WIMS 2012) (2012)
51. Quinlan, J.R.: Learning logical definitions from relations. Machine Learning **5**, 239–266 (1990)
52. Quinlan, J.R.: Generating production rules from decision trees. In: Proceedings of the 10th International Joint Conference on Artificial Intelligence (IJCAI 1987), pp. 304–307. Morgan Kaufmann (1987)
53. Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufmann, San Mateo (1993)
54. Rauch, J.: Observational Calculi and Association Rules, Studies in Computational Intelligence, vol. 469. Springer (2013)
55. Sulzmann, J.N., Fürnkranz, J.: A comparison of techniques for selecting and combining class association rules. In: Knobbe, A.J. (ed.) Proceedings of the ECML/PKDD 2008 Workshop From Local Patterns to Global Models (LeGo 2008), Antwerp, Belgium, pp. 154–168 (2008)
56. Uno, T., Kiyomi, M., Arimura, H.: LCM ver. 2: efficient mining algorithms for frequent/closed/maximal itemsets. In: Proceedings of the IEEE ICDM 2004 Workshop on Frequent Itemset Mining Implementations (FIMI 2004) (2004)
57. Webb, G.I.: OPUS: An efficient admissible algorithm for unordered search. Journal of Artificial Intelligence Research **5**, 431–465 (1995)
58. Witten, I.H., Frank, E.: Data Mining - Practical Machine Learning Tools and Techniques with Java Implementations, 2nd edn. Morgan Kaufmann Publishers (2005)
59. Wrobel, S.: An algorithm for multi-relational discovery of subgroups. In: Komorowski, J., Żytkow, J.M. (eds.) PKDD 1997. LNCS, vol. 1263, pp. 78–87. Springer, Heidelberg (1997)

60. Yin, X., Han, J.: CPAR: classification based on predictive association rules. In: Proceedings SIAM Conference on Data Mining (SDM 2003) (2003)
61. Völker, J., Niepert, M.: Statistical schema induction. In: Antoniou, G., Grobelnik, M., Simperl, E., Parsia, B., Plexousakis, D., De Leenheer, P., Pan, J. (eds.) ESWC 2011, Part I. LNCS, vol. 6643, pp. 124–138. Springer, Heidelberg (2011)
62. Zaki, M.J., Parthasarathy, S., Ogihara, M., Li, W.: New algorithms for fast discovery of association rules. In: Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining (KDD 1997), Newport, CA, pp. 283–286 (1997)
63. Zhang, C., Zhang, S.: Association Rule Mining –Models and Algorithms. Springer (2002)
64. Zimmermann, A., De Raedt, L.: Cluster grouping: From subgroup discovery to clustering. Machine Learning **77**(1), 125–159 (2009)