Genetic Search for Feature Selection in Rule Induction Algorithms

Jasmina Novakovic, Milomir Minic, Alempije Veljovic

Abstract – **This paper presents the impact of genetic search for selection of features in rule induction algorithms. Seven rule induction algorithms: JRip, ConjuctiveRule, DecisionTable, OneR, PART, Ridor and ZeroR are used. Compared to rule induction algorithms without genetic search for feature selections, this approach can improves classification accuracy of classifiers. Evaluation of selecting features with genetic search is fast compare with wrapper approach.**

Keywords **- classification accuracy, genetic search, rule induction algorithms.**

I. Introduction

In knowledge discovery, one of the main tasks considered is supervised classification, where learning process is provided with a set of training examples of target classes. Each example corresponds to a single object to be classified and is described by a finite of features. The goal of learning is to discover a rule or a function (in machine learning often called a hypothesis) which maps such descriptions into those classes [1], [2].

An algorithm, which consists of knowledge representation (learned from some training set) and the strategy of its usage, forms a classifier, which can be used to predict classes of new coming objects. Classification accuracy is typical measure used to evaluate classifier's performance. Several algorithms have been proposed over the years for inducing various knowledge representations and various classifiers [3], [1], [4], [5]. For many classification problems those algorithms are very effective, but they do not always lead to satisfactory classification accuracy in more complex and difficult cases. As it is shown in theoretical studies and confirmed in empirical comparative studies there is no single best algorithm to be used for all data sets. It means that every algorithm has its own area of superiority and is "specialised" to solve some classes of learning problems.

Our research interest includes classifiers by using rules. Rules are one of the most popular type of knowledge used in practice, mainly due to the fact that they are very expressive and human-readable at the same time. A number of algorithms has been developed to induce such rules [6], [3], [1], [4].

The main aim of this paper is to experimentally verify, we intend to study the impact of genetic search on classification accuracy of rule induction algorithms.

This paper is organised as follows. In the next section we briefly describe the rule induction algorithms. Section 3 contains general issues concerning genetic search. Section 4 presents experimental evaluation. Final section contains discussion of the obtained results and some closing remarks.

II. Rule Induction Algorithms

One type of classification algorithms is rule induction algorithms. Rule induction algorithms generate a model as a set of rules, which are in the standard form of IF-THEN rules. We have identified and tested seven rule algorithms for their suitability in handling nominal class prediction, which is the common type of class attribute for the medical datasets.

One of the most primitive learning scheme, **ZeroR**, predicts the majority class in the training data for problems with a categorical class value, and the average class value for numeric prediction problems. It is useful for generating a baseline performance that other learning schemes are compared to. In some cases, it is possible that other learning schemes perform worse than ZeroR, an indicator of substantial overfitting.

The **ConjuctiveRule** algorithm implements a single conjunctive rule learner that can predict for numeric and nominal class labels. A rule consists of antecedents "AND"ed together and the consequent (class value) for the classification/regression. In this case, the consequent is the distribution of the available classes (or mean for a numeric value) in the dataset.

The **DecisionTable** algorithm builds a decision rule using a simple decision table majority classifier [7]. It summarizes the dataset with a 'decision table' which contains the same number of attributes as the original dataset.

OneR or "One Rule" is a simple algorithm proposed by [8]. It builds one rule for each attribute in the training data and then selects the rule with the smallest error rate as its 'one rule'.

Jasmina Novakovic, Faculty of Public Administration, Megatrend University, Bulevar umetnosti 29, 11070 Novi Beograd, Srbija (phone: 381-11-2092111, e-mail: jnovakovic@megatrend.edu,rs).

Milomir Minic, Faculty of Public Administration, Megatrend University, Bulevar umetnosti 29, 11070 Novi Beograd, Srbija (phone: 381-11-2092111, e-mail: mminic@megatrend.edu,rs).

Alempije Veljovic, Technical Faculty Cacak, University in Kragujevac, Svetog Save 65, 32000 Čačak, Srbija, (phone: 381 032/342-101), e-mail: (alempije@beotel.rs).

PART is a separate-and-conquer rule learner proposed by [9]. PART producing sets of rules called 'decision lists' which are ordered set of rules. This algorithm builds a partial C4.5 decision tree in each iteration and makes the "best" leaf into a rule. PART is a combination of C4.5 [10] and RIPPER [11] rule learning.

Ridor is the implementation of a RIpple-DOwn Rule learner. It generates the default rule first and then the exceptions for the default rule with the least (weighted) error rate. Then it generates the "best" exceptions for each exception and iterates until pure. Thus it performs a tree-like expansion of exceptions and the leaf has only default rule but no exceptions. The exceptions are a set of rules that predict the class other than class in default rule. IREP is used to find out the exceptions.

JRip implements a propositional rule learner, Repeated Incremental Pruning to Produce Error Reduction (RIPPER), which was proposed by [11], [12] as an optimized version of IREP [13]. Ripper builds a ruleset by repeatedly adding rules to an empty ruleset until all positive examples are covered. Rules are formed by greedily adding conditions to the antecedent of a rule (starting with empty antecendent) until no negative examples are covered. After a ruleset is constructed, an optimization postpass massages the ruleset so as to reduce its size and improve its fit to the training data.

III. Genetic Search

Genetic algorithms are adaptive search techniques based on the principles of natural selection in biology [14]. Genetic algorithms employ a population of competing solutions—evolved over time—to converge to an optimal solution. The solution space is searched in parallel, which helps in avoiding local optima. An outline of the genetic algorithm is shown in Fig. 1.

```
Input:
        $P$:            Randomly initialized population
        $p_c$, $m_r$:   Crossover and Mutation rates
        $MAXGEN$:       Maximum number of generations
        $n$:            Population size, $|P|$
Output:
        $x$:            Best individual from current $P$
Method:
        EvaluateFitness($P$)
        while
(generation<$MAXGEN$) $\wedge$ (NotConvergence) do
                $M \leftarrow$ Recombine($P$)
                $O \leftarrow$ Crossover($M$, $p_c$)
                $O \leftarrow$ Mutate($O$, $m_r$)
                EvaluateFitness($P$)
                $P \leftarrow$ Select($P$,$O$)
                generation $\leftarrow$ generation + 1
        endwhile
```

Figure 1. Genetic algorithm.

Solution for feature selection is typically a fixed length binary string representing a feature subset—the value of each position in the string represents the presence or absence of a particular feature. The algorithm is an iterative process where each successive generation is produced by applying genetic operators such as crossover and mutation to the members of the current generation. Crossover combines different features from a pair of subsets into a new subset. Mutation changes some of the values (thus adding or deleting features) in a subset randomly. The application of genetic operators to population members is determined by their fitness. For feature selection, it means how good a feature subset is with respect to an evaluation strategy. If feature subsets are better than they have a greater chance of being selected to form a new subset through crossover or mutation.

IV. Experiment and Results

Congressional voting records data set was used for evaluating different rule induction algorithms with genetic search. This data set includes votes for each of the U.S. House of Representatives Congressmen in 1984, on the 16 key votes identified by the CQA, and classify as Republican or Democrat. The CQA lists nine different types of votes: voted for, paired for, and announced for (these three simplified to yea), voted against, paired against, and announced against (these three simplified to nay), voted present, voted present to avoid conflict of interest, and did not vote or otherwise make a position known (these three simplified to an unknown disposition). Congressional voting records data set taken from the UCI repository of machine learning databases.

Features of this data set are: handicapped-infants, water-project-cost-sharing, adoption-of-the-budget resolution, physician-fee-freeze, el-salvador-aid, religious-groups-in-schools, anti-satellite-test-ban, aid-to-nicaraguan-contras, mx-missile, immigration, synfuels-corporation-cutback, education-spending, superfund-right-to-sue, crime, duty-free-exports and export-administration-act-south-africa. All features can only have two values: *y* or *n*. The numbers of classes are two: democrat and republican.

In our experiments are used rule induction algorithms with genetic search for feature selections. Parameters for genetic search are following:

- the probability of crossover, which is the probability that two population members will exchange genetic material is set on 0,6;
- the number of generations to evaluate is 20;
- the probability of mutation occurring is 0.033;
- the population size, the number of individuals (attribute sets) in the population, is set on 20;
- the random seed is set on 1;
- there is no start point for the search.

CFS Subset Evaluator was used. It evaluates the worth of a subset of attributes by considering the individual predictive ability of each feature along with the degree of redundancy between them. Subsets of features that are highly correlated with the class while having low intercorrelation are preferred. Settings for this evaluator are:

- Identify locally predictive attributes. Iteratively adds attributes with the highest correlation with the class as long as there is not already an attribute in the subset that has a higher correlation with the attribute in question.

- Counts for missing values are distributed across other values in proportion to their frequency.

TABLE 1: NUMBERS OF RULES AND CONFUSION MATRIX BY RULE INDUCTION ALGORITHMS.

Classifier	**Numbers of rules**	**Confusion Matrix**
ConjunctiveRule	1	a b <-- classified as 247 20 | a = democrat 3 165 | b = republican
DecisionTable	3	a b <-- classified as 253 14 | a = democrat 7 161 | b = republican
OneR	1	a b <-- classified as 253 14 | a = democrat 5 163 | b = republican
PART	4	a b <-- classified as 257 10 | a = democrat 9 159 | b = republican
Ridor	5	a b <-- classified as 250 17 | a = democrat 5 163 | b = republican
ZeroR	1	a b <-- classified as 267 0 | a = democrat 168 0 | b = republican
JRip	2	a b <-- classified as 253 14 | a = democrat 5 163 | b = republican

The purpose of the experiments described in this section is to empirically test the claim that genetic search can improve the accuracy of rule induction algorithms. Classification accuracy was estimated using ten-fold crossvalidation on each data set. Genetic search selected four attributes for this data set: 3 (adoption-of-the-budget-resolution), 4 (physician-fee-freeze), 10 (immigration) and 11 (synfuels-corporation-cutback). These attributes used rule induction algorithms. Table 1 shows numbers of rules and confusion matrix by rule induction algorithms.

The same higher value for classification accuracy had three rule induction algorithms: OneR, PART and JRip. Table 2 shows produced rules by these algorithms.

TABLE 2: RULES FOR CONGRESSIONAL VOTING RECORDS DATA SET.

Classifiers	**Rules**
OneR	(physician-fee-freeze = y) => Class = republican ELSE => Class=democrat
PART	(physician-fee-freeze = n) => Class = democrat (synfuels-corporation-cutback = n) => Class = republican (adoption-of-the-budget-resolution = n) => Class = republican ELSE=> Class = democrat
JRip	(physician-fee-freeze = y) => Class=republican ELSE => Class=democrat

Fig. 2 shows classification accuracy for seven classification algorithms (JRip, ConjuctiveRule, DecisionTable, OneR, Part, Ridor and ZeroR) with and without genetic search (GS). In our experiment we used evaluator namely CFS Subset Evaluator.

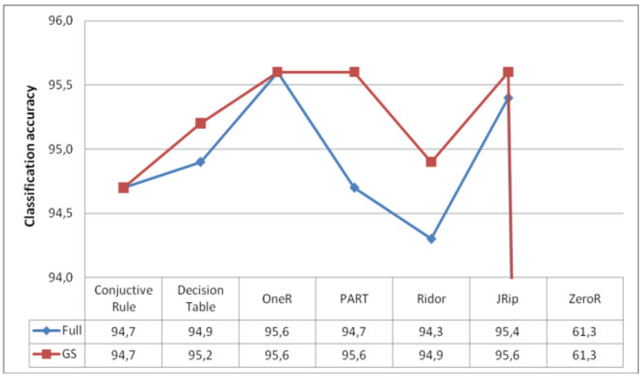

Figure 2. Classification accuracy of rule induction algorithms with and without genetic search.

Genetic search degrades the accuracy of none classifier for this data set. Genetic search improves the accuracy of four classifiers: JRip, Ridor, DecisionTable and PART and maintains the accuracy of tree classifiers: ConjuctiveRule, OneR and ZeroR. Accuracy for this data set is the worst with ZeroR. It is expectancy result, because ZeroR is one of the most primitive learning scheme. For this data set genetic search improves the accuracy of PART the most.

V. CONCLUSIONS

The experiments presented in this article show that classification accuracy with genetic search improves or maintains for seven rule induction algorithms. Genetic search degrades the accuracy of none classifier for this data set. Genetic search improves the accuracy of four classifiers: JRip, Ridor, DecisionTable and PART and maintains the accuracy of tree classifiers: ConjuctiveRule, OneR and ZeroR. Evaluation of selecting features with genetic search is fast compare with wrapper approach.

Some improvements of the selecting methods presented here are possible. The algorithms and data sets will be selected according to precise criteria: classify algorithms and several data sets, either real or artificial, with nominal, binary and continuous features. These conclusions and recommendations will be tested on larger data sets using various classification algorithms in the near future.

REFERENCES

[1] R.S. Michalski, I. Bratko, M. Kubat (eds.), *Machine learning and data mining*, John Wiley & Sons, 1998.

[2] T.M. Mitchell, *Machine learning*, 1997.

[3] W. Klosgen, J.M. Zytkow (eds.), *Handbook of data mining and knowledge discovery*, Oxford Press, 2002.

[4] J. Stefanowski, *Algorithms of rule induction for knowledge discovery*. (In Polish), Habilitation Thesis published as Series Rozprawy no. 361, Poznan Univeristy of Technology Press, Poznan 2001.

[5] L. Kuncheva, *Combining pattern classifiers: Methods and algorithms*, Wiley, 2004.

[6] J.W. Grzymala-Busse, LERS — a system for learning from examples based on rough sets. In: Slowinski R. (ed.) *Intelligent Decision Support. Handbook of Applications and Advances of the Rough Sets Theory*. Kluwer, 1992, 3–18.

[7] R. Kohavi, The Power of Decision Tables. In: *Machine Learning*, 1995, pp 174-189.

[8] R.C. Holt, Very simple classification rules perform well on most commonly used datasets. In: *Machine Learning*, 1993, 11:63-91.

[9] I.H. Witten, and E. Frank, *Data Mining: Practical machine learning tools and techniques*, Morgan Kaufmann, San Francisco, 2005.

[10] R. Quinlan, C4.5: *Programs for machine learning*, Morgan Kaufmann, San Mateo, 1993.

[11] W.W. Cohen, Fast effective rule induction. In: *Machine Learning*, Lake Tahoe, California, 1995.

[12] W.W. Cohen, Learning rules that classify e-mail. In: *Machine Learning in Information Access*, pp 18-25, 1996.

[13] J. Furnkranz, G. Widmer, Incremental Reduced Error Pruning. In: *Machine Learning*, Morgan Kaufmann, New Jersey, 1994, pp 70-77.

[14] J.H. Holland, *Adaption in natural and artificial systems*, University of Michigan Press, Ann Arbor, MI, 1975.