Turkish Journal of Electrical Engineering and Computer Sciences

Volume 21 | Number 5 Article 1

1-1-2013

A rule induction algorithm for knowledge discovery and classification

ÖMER AKGÖBEK

TÜBİTAK

Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2013) 21: 1223 – 1241

doi:10.3906/elk-1202-27

A rule induction algorithm for knowledge discovery and classification

Ömer AKGÖBEK*
Department of Industrial Engineering, Zirve University, 27260 Gaziantep, Turkey

Received: 07.02.2012 • **Accepted:** 16.04.2012 • **Published Online:** 12.08.2013 • **Printed:** 06.09.2013

Abstract: Classification and rule induction are key topics in the fields of decision making and knowledge discovery. The objective of this study is to present a new algorithm developed for automatic knowledge acquisition in data mining. The proposed algorithm has been named RES-2 (Rule Extraction System). It aims at eliminating the pitfalls and disadvantages of the techniques and algorithms currently in use. The proposed algorithm makes use of the direct rule extraction approach, rather than the decision tree. For this purpose, it uses a set of examples to induce general rules. In this study, 15 datasets consisting of multiclass values with different properties and sizes and obtained from the University of California, Irvine, have been used. Classification accuracy and rule count have been used to test the proposed method. This method presents an alternative 3-step method to classify categorical, binary, and continuous data by taking advantage of algorithms for data mining classification and decision rule generation. The method aims at improving the classification accuracy of the algorithms that extract the decision rules. Experimental studies were conducted on the benchmark datasets and the results of the comparisons with some known algorithms for decision rule generation have shown that the proposed method performs classification with a higher accuracy and generates fewer rules.

Key words: Knowledge discovery, rule extraction, classification, data mining

1. Introduction

Data mining is described as the process of accessing information in large databases and extracting generalized information, or, in other words, searching, revealing, and providing correlations in large databases that will enable us to make estimations about the future with the help of computer programs. Data mining covers basics like statistics, databases, programming techniques, and high performance processing, as well as all works for formulizing and applying induction procedures that would extract significant and useful information from available data [1]. Structural patterns and hidden knowledge in the large databases can be represented in various ways, such as decision tables, decision trees, association rules, and classification rules. Algorithms are needed for generating rules that determine the description of the concepts to be learned. However, the description is only one of many possible interpretations of the training data and may impose a completely different meaning that is far from the meaning of the concept [2].

A major problem encountered in the design of learning algorithms is the generation of a complex description from noisy examples. Learning from noise-corrupted data may result in a large number of complicated decision rules describing trivial instances. Hence, the resulting concept description may not reflect general situations. Such a case is referred to as 'overfitting', which refers to a tendency to force the rule induced from the

*Correspondence: omer.akgobek@zirve.edu.tr

training data to agree with the data too closely, at the cost of generalization for other examples. Poor concept description is another cause of overfitting. Researchers have studied the methods to overcome the overfitting caused by noise for a long time [2,3].

In recent years, many algorithms in the literature have been developed in order to discover the relationships between the input data and the output data for generating rules. In these algorithms, approaches such as divide and conquer, association rules, covering rule induction, decision trees, and naïve Bayes are used [1–13]. Apart from these, there are also other algorithms that generate rules using techniques such as neural networks, ant colony optimization, genetic algorithms, particle swarm optimization, fuzzy logic, and support vector machines [14–19]. In this study, the covering rule induction method has been used in order to obtain generalized rules.

Tan et al. [20] presented a hybrid evolutionary algorithm using genetic algorithms and a support vector machine for the attribute selection. Elalfi et al. [21] proposed an algorithm for extracting accurate and intelligible rules from databases via a trained artificial neural network using genetic algorithms. Kahramanli and Allahverdi [22] presented a method that extracts rules from trained adaptive neural networks using artificial immune systems. The authors iterated the purpose of the study as to develop a new adaptive function and a new method for rule generation. In their study, the results of the proposed algorithm on 2 benchmark datasets were compared with the results obtained from the previous works in the literature. Rodríguez et al. [14] presented an efficient distributed genetic algorithm for classification rule extraction based on the island model and enhanced it for scalability with data training partitioning in data mining. In this study, they calculated the acquisition rate, rule count, and standard deviation for many datasets. Cohen et al. [7] introduced an automatic, general, decision-tree–based framework for instance-space decomposition with a contrasted population miner rule using grouped gain-ratio. Thabtah and Cowling [11] proposed an algorithm that resolves the overlapping between rules in the classifier by generating rules that do not share training objects during the training phase, resulting in a more accurate classifier. Their results, obtained from experimenting on 20 binary, multiclass, and multilabel datasets, show that the proposed technique is able to produce classifiers that contain rules associated with multiple classes. Cios and Kurgan [12] described a hybrid inductive machine learning algorithm called CLIP4. In this algorithm, data are first partitioned into subsets using a tree structure and then production rules are generated only from the subsets stored at the leaf nodes. The unique feature of the algorithm is the generation of rules that involve inequalities. Moreover, this algorithm-generated model of the data consists of well-generalized rules, and it ranks the attributes and selectors that can be used for feature selection. Thabtah et al. [9] proposed a new associative classification method called multiclass classification based on association rules (MCAR). This method takes advantage of vertical format representation and uses an efficient technique for discovering frequent items based on recursively intersecting the frequent items of size n to find potential frequent items of size $n+1$. Coenen and Leng [10] showed that the choice of appropriate values for the support and confidence thresholds could have a significant effect on the accuracy of classifiers obtained by classification association rule mining (CARM) algorithms. They examined the effect of the choices on the predictive accuracy of the CARM methods, showed that the accuracy can almost always be improved by a suitable choice of parameters, and described a hill-climbing method for finding the best parameter settings. Su et al. [8] proposed the differential evolution (DE)/quantum-inspired differential evolution (QDE) algorithm for the discovery of classification rules. This algorithm combines the characteristics of the conventional DE algorithm and the QDE algorithm. Based on some strategies of the DE and QDE algorithms, the DE/QDE algorithm can directly cope with the continuous, nominal attributes without discretizing the continuous attributes in the preprocessing step. The comparisons of

the DE/QDE algorithm with Ant-Miner and CN2 on 6 datasets from the University of California, Irvine (UCI) datasets showed competitive results.

These algorithms generate general concept definitions from special examples with the help of special procedures. These algorithms are classified into 2 main groups. The first is decision-tree–based algorithms and the other is rule-based algorithms.

Decision tree is one of the data mining and machine learning techniques with applications in various fields. A decision tree is a tree structure representation of the given decision problem, such that each nonleaf node is associated with one of the decision variables, each branch from a nonleaf node is associated with a subset of the values of the corresponding decision variable, and each leaf node is associated with a value of the target variable [23]. Decision-tree–based algorithms usually use the information entropy measure to grow a decision tree by searching for a feature that gives the maximum information gain [3]. The procedure of growing a decision tree continues by dividing examples into smaller subsets until the training examples are correctly classified based on a user-specified termination criterion. An example of the first-type algorithm is the incremental decision (ID) family of algorithms, such as ID3 [24] and C4.5 [6].

In real-world applications, training examples are usually insufficient to define a concept description uniquely. Therefore, learning algorithms need the flexibility to produce different generalizations from given examples. In decision-tree–based algorithms, the description of a subset of examples in a leaf node of a tree is uniquely described as a series of feature tests from the root to the bottom of a tree. This approach does not have the flexibility of describing a target concept in different ways.

Rule-based algorithms have the ability to generate multiple descriptions of a concept. An example is the algorithm quasi-optimal (AQ)15, where the empirical learning was treated by Michalski [25] as the general covering problem. The basic term of a cover, as used in the AQ family of algorithms, implies that there may be multiple covers to include positive training examples. This resulted in the development of procedures that produce a quasi-optimal solution in polynomial time. Generally, AQ algorithms follow a greedy heuristic that tries to include/exclude as many as possible of the positive/negative examples when searching for a complex.

There are many information display forms of data mining classification algorithms that are used as output formats. Among them are decision rules, decisions lists, decision trees, inductive logic programs, and neural networks. Decisions rules are the simplest display form of these output formats: *If Condition Then Class*. Here, the 'If' part is composed of attribute–value pairs with the help of AND/OR logical operators and the 'Then' part shows the corresponding class value of the given condition. Generally, these decision rules are extracted during the training process through data mining classification algorithms in such a manner that all of the training set is covered. The purpose of extracting decision rules from training data is to obtain generalized rules that define the classes from the special data. The generalization process ends when new rules cannot be generated or the specified termination criteria are met. The extracted decision rules are used to classify the samples in the test set. The output formats of the data mining algorithms, such as decision trees and neural networks, can be easily converted to decision rules.

The proposed algorithm will be described and exemplified on a simple dataset in the next section. The comparisons of the proposed algorithm with the other algorithms in the literature over 15 datasets are given in Section 3. Finally, the results are discussed in Section 4.

2. The RES-2 algorithm for rule discovery

In this section, the rule extraction system algorithm RES-2 for solving classification problems will be introduced. The algorithm can be readily applied on the datasets that contain continuous, categorical, and binary attributes. The continuous attributes in datasets are discretized using Fayyad and Irani's entropy-based discretization method [26]. Fayyad and Irani have extended the method of binary discretization in CART [27] and C4.5, and they introduced multiinterval discretization using a minimal description length technique. In this method, the data are discretized into 2 intervals for a candidate cut point, and the resulting class information entropy is calculated. A binary discretization is determined by selecting the cut point for which the entropy is minimal amongst all of the candidates. The binary discretization is applied recursively, always selecting the best cut point. A minimum description length criterion is applied to decide when to stop the discretization [28]. The rule generation procedure of RES-2 is given in the Figure.

```
Step-1.  Convert continuous values to categorical values.
Step-2.  n = 1  // the number of attributes used in the combination
Step-3.  REPEAT
         Stage I
            ALL_CLASSIFIED=FALSE
            FOR I = 1 TO Number_Of_Samples
               IF  I^th example is unclassified THEN
                  Pick only one value for each attribute in the example set and  and
                  form combinations with n number of attributes .
                  S = 0, Rule_Generated = False
                  DO
                     S = S + 1
                     S^th combination is applied to all examples in the example set and
                     T is set to the number of the classes corresponding to this combination.
                     IF  T = 1 THEN
                        The combination is turned into a rule.
                        Produced rule is added to the rule set.
                        The examples which are classified by rule are marked.
                        Rule_Generated=True
                     END IF
                  WHILE S < Number_Of_Combinations AND Rule_Generated
                  Rule which classifies most examples is selected.
               END IF
            END FOR
            IF all samples classified THEN ALL_CLASSIFIED = TRUE
               ELSE n = n + 1
            END IF
         UNTIL  n > Number_of_Attribute OR ALL_CLASSIFIED
         Stage II
            Rule which classifies most examples is selected.
            If there is more than one rule representin g the same example,
            pick up most general one. The others are eliminated.
         Stage III
            Rules which can be combined using the OR operator are determined and combined.
Step-4.  Rules covered by others are eliminated and the most general rules are selected.
```

Figure. Rule generation procedure of the RES-2 algorithm.

The algorithm checks all of the samples one by one in the training set beginning from the first sample, and the samples classified by the generated rule are sorted out at each stage. Thanks to this process, the operation time is guaranteed to be shorter and the same rules are prevented from being regenerated. When a sample is dealt with, first it is checked to see whether the combination of the single attribute–value pair generates a rule. If this combination becomes a rule, then all of the samples covered by this rule are marked and the other combinations are checked. Otherwise, the other combinations are checked directly. This operation continues

until a rule that classifies the sample in question is generated. Because of this, all of the samples included in the training set can be classified by the generated rules. This means that the training set can be classified with 100% accuracy by this algorithm.

2.1. Rule representation of the algorithm

In the context of the classification task of data mining, discovered knowledge is often expressed in the form of IF–THEN rules, as follows [29]:

IF Conditions THEN Class

The rule antecedent (IF part) contains a set of conditions, usually connected by a logical conjunction operator (AND). We will refer to each rule condition as a term, such that the rule antecedent is a logical conjunction of terms in the form of *IF term1 AND term2 AND ... THEN Class.* Each term is a triple $<$ *attribute, operator, value* $>$, such as $<$ *Temperature = High* $>$.

The rule consequent (THEN part) specifies the class predicted for cases whose predictor attributes satisfy all of the terms specified in the rule antecedent. From a data mining viewpoint, this kind of knowledge representation has the advantage of being intuitively comprehensible for the user, as long as the number of discovered rules and the number of terms in the rule antecedents are not large. More particularly, a rule is given as:

IF $(x_1 \leq t_1)$ *AND ... AND* $(x_n \geq t_n)$ *THEN C,*

where x_i are continuous variables, t_i are real numbers, and C is a class designating a concept, such as the class of healthy patients. Note that discrete variables are special cases of continuous variables. For instance, *If* $x_2 = 3.0$ *And* $x_1 \geq 2.0$ *And* $x_1 \leq 5.0$ *Then Class = Yes.*

At the third stage of the proposed algorithm, the rules are combined with the OR logical operator to decrease the number of rules. For instance, *If* $x_3 =$ *Normal Or* $(x_2 = 3.0$ *And* $x_1 \geq 2.0$ *And* $x_1 \leq 5.0)$ *Then Class = Yes.*

2.2. Rule generation methodology of the algorithm

The developed method achieves the rule generation process in 3 stages. These stages are given below:

1. Creating combinations and checking whether or not they correspond to only one class. Adding the rule that corresponds to only one class to the rule set. Thus, the generated rule set classifies the training set with 100% accuracy.
2. Choosing the most comprehensive rule among the inherited ones and deleting the repeating rules from the database. This enables fewer rule generations.
3. Decreasing the number of rules to a minimum level by combining the terms that are in the antecedent part of the rules with the same classification, not only with the AND operator but also with the OR operator.

The first stage consists of the same number of steps as the attribute number included in the sample set. At the first step, the value of each attribute in the current sample is taken one by one and checked to see whether it constitutes a rule on its own. The condition constituting a rule needs to belong to only one class and the relevant attribute–value pair is set as a rule. Otherwise, no action is taken. This process is repeated for all of the attribute–value pairs that are in the sample set. All of the rules generated in this stage are added to the rule base. The structure of the rules obtained is as follows:

If $Attribute_1 = Value_{1,1}$ Then $Decision = Class_1$.

This way, it is made possible to obtain the most general rules that contain the smallest amount of attribute–value pairs and that represent the sample set. In this stage, the samples that contain missing values are not removed from the sample set, but it is made sure that the missing values are ignored. At this point, the values of 2 attributes in the current sample are taken and combined with the logical AND operator in order to check whether they correspond to only one class. If they correspond to only one class, the conditions are made a rule and are added to the rule base; otherwise, no transaction is conducted. This process is repeated for all of the dual combinations. The structure of the rules obtained as a result of this stage is as follows:

If $Attribute_1 = Value_{1,1}$ AND $Attribute_2 = Value_{2,1}$ Then $Decision = Class_n$.

In the third step, it is explored whether triple attribute–value pairs form a rule. This way, the procedures are repeated in k steps in a sample set consisting of k attribute–values. This stage continues until all of the samples are classified and all of the rules generated are added to the rule base. The longest rule, which may be formed in a sample set that contains k attributes, is possible in the format below:

If $Attribute_1 = Value_{1,a}$ AND $Attribute_2 = Value_{2,b}$ And ... And $Attribute_k = Value_{k,c}$
Then $Decision = Class_n$.

However, the possibility of such a rule is very small. Such a rule is named a special rule rather than a general rule and represents only one sample.

This stage continues until all of the rules in the sample set are classified. As no pruning procedure is used, the sample set is classified by the rule set with 100% accuracy.

The second stage is the elimination of unnecessary and repeating rules in the rule base. The repeating rules or the rules that are covered by another rule are eliminated from the rule base in such a way that there remains in the end only one rule. It is aimed that the number of the rules generated with the help of this feature be at a minimum rate.

In the third stage, those rules in the rule set that have the same classification can be combined with the OR logical operator. There are 2 exemplary rules relating to this case. There is the repeating $X_1 = Medium$ *condition* in the antecedent part of both of the rules.

If $X_1 = Medium$ AND $X_2 = High$ Then Decision = YES
If $X_1 = Medium$ AND $X_2 = Short$ Then Decision = YES

These 2 rules can be combined with the IN or OR logical operators and made into a single rule:

If $X_1 = Medium$ AND X_2 IN *[High, Short]* Then $Decision = YES$

or

If $X_1 = Medium$ AND ($X_2 = High$ *OR* $X_2 = Short$) Then Decision = YES

After this stage, those rules covered by the other rule could be eliminated from the rule set. The second of the 2 rules given below is eliminated as it is covered by the first rule.

If $X_1 = Normal$ OR $X_2 = Low$ Then $Decision = Yes$
If $X_1 = Normal$ Then $Decision = Yes$

2.3. Characteristics of the algorithm

The characteristics of this algorithm can be described as follows:

- The algorithm does not perform any calculation in the sample set (such as entropy, Gini index, etc.) and it does not generate a decision tree. It has quite a simple rule discovery method, as it directly extracts rules beginning from the first sample. This characteristic provides great convenience in terms of programming logic.
- The algorithm can also take action according to the samples that contain a missing value. That some samples in the sample set contain missing values is inevitable. For this, the samples containing missing values in the training set are not eliminated from the sample set, but instead they are ignored during the rule extraction.
- Rules can be generated from the sample set not only for the categorical attributes but also for the attributes containing numerical and binary values.
- It allows for the range definitions for continuous values to be made. With the help of this feature, it is made possible to obtain more general rules.

2.4. Application of the algorithm

In Table 1 [30], a sample set is given that consists of 4 attributes and 14 samples in order to decide whether an investment project can be implemented or not. The RES-2 algorithm has been applied step by step in order to understand the rule extraction process of the algorithm and to obtain rules from the sample set.

Step 1.

As all of the values in the sample set are categorical and as they do not contain any numerical value, discretization is not performed.

Step 2.

First Stage starts with the number of attributes in combination as n = 1. In this stage, the rule extraction process is performed according to the combinations of the single attribute–value pairs. For this, the unclassified samples will be tried in order.

Step 3.

The first sample is chosen as a principle. This sample is composed of the attribute–values of {Global_risk:*High*, Profitableness:*Important*, Return_time:*Long*, Investment_level: *High*}. As each value corresponds to more than one class in the sample set when the combinations of the single values in this example are tried one by one, it cannot generate a rule on its own, and the next sample is tried.

The second example is composed of the attribute–values of {Global_risk:*High*, Profitableness:*Important*, Return_time:*Long*, Investment_level:*Low*} . These values, as well, cannot generate rules on their own and so the next sample is tried.

The third example is composed of the attribute–values of {Global_risk:*Low*, Profitableness:*Important*, Return_time:*Long*, Investment_level: *High*} . It is checked to see whether all of these values correspond to only one class in the sample set. As the first value, {Global_risk:*Low*} , corresponds to the *YES* class in the samples numbered 3, 7, 12, and 13, it generates the following rule.

Table 1. Investment training examples.

Example no.	Global_risk	Profitableness	Return_time	Investment_level	Class
1	High	Important	Long	High	No
2	High	Important	Long	Low	No
3	Low	Important	Long	High	Yes
4	Medium	Medium	Long	High	Yes
5	Medium	Small	Short	High	Yes
6	Medium	Small	Short	Low	No
7	Low	Small	Short	Low	Yes
8	High	Medium	Long	High	No
9	High	Small	Short	High	Yes
10	Medium	Medium	Short	High	Yes
11	High	Medium	Short	Low	Yes
12	Low	Medium	Long	Low	Yes
13	Low	Important	Short	High	Yes
14	Medium	Medium	Long	Low	No

Rule 1: ***IF*** *Global_risk = Low* ***THEN*** *Class = YES*

With this rule, the samples numbered 3, 7, 12, and 13 are classified and then marked as never to be paid attention to. As the other values in the third sample do not generate a rule on their own, the next unclassified sample is tried. As none of the attribute–value pairs in the other remaining samples correspond to only one class on their own, they cannot generate a rule and the next stage is performed.

Dual combinations of the attribute–values in the samples that are not classified in the second stage are formed and it is checked to see whether these combinations will generate rules or not.

The first sample that is not classified is taken. This sample consists of the attribute–values of {Global_risk:*High*, Profitableness:*Important*, Return_time:*Long*, Investment_level:*High*} . Dual combinations of these values are applied on the sample set respectively. As the {Global_risk:*High*, Return_time:*Long*} combination of these are included in the NO class, it generates the following rule.

Rule 2: ***IF*** *Global_risk = High* ***AND*** *Return_time = Long* ***THEN*** *Class = NO*

With this rule, the samples numbered 1, 2, and 8 are classified. These samples are marked as never to be paid attention to later on. As the remaining combinations do not correspond to only one class, they cannot generate rules on their own.

The fourth sample that is not classified is chosen. This sample consists of the attribute–values of {Global_risk:*Medium*, Profitableness:*Medium*, Return_time:*Long*, Investment_level:*High*} . As the {Global_risk:*Medium*, Investment_level:*High*} of the dual combinations of these values is included only in one class, it generates the following rule.

Rule 3: ***IF*** *Global_risk = Medium* ***AND*** *Investment_level = High* ***THEN*** *Class = YES*

With this rule, the samples numbered 4, 5, and 10 are classified and these samples are marked.

The sixth sample, which is the first sample that is not classified, is taken. This sample consists of the attribute–values of {Global_risk:*Medium*, Profitableness:*Small*, Return_time:*Short*, Investment_level:*Low*} . As the {Global_risk:*Medium*, Investment_level:*Low*} of the dual combinations of these values corresponds to only one class, it generates the following rule.

Rule 4: ***IF*** *Global_risk = Medium* ***AND*** *Investment_level = Low* ***THEN*** *Class = NO*

With this rule, the samples numbered 6 and 14 are classified and these samples are marked as never to be paid attention to again.

Up to this stage, there remain the samples numbered 9, 10, and 11 as unclassified. The first one of these, the ninth sample, consists of the attribute-values of {Global_risk:*High*, Profitableness:*Small*, Return_time:*Short*, Investment_level:*High*} . As the {Return_time:*Short*, Investment_level:*High*} of the dual combinations of these values is included only in 1 class in all of the sample sets, it generates the following rule:

Rule 5: ***IF*** *Return_time = Short* ***AND*** *Investment_level = High* ***THEN*** *Class = YES*

With this rule, the samples numbered 5, 9, 10, and 13 are classified.

Up to this stage, there remains the sample numbered 11 as unclassified. This sample consists of the attribute–values of {Global_risk:*High*, Profitableness:*Medium*, Return_time:*Short*, Investment_level:*Low*} . As the {Global_risk:*High*, Return_time:*Short*} of the dual combinations of these values corresponds to only one class, the following rule is obtained.

Rule 6: ***IF*** *Global_risk = High* ***AND*** *Return_time = Short* ***THEN*** *Class = YES*

With this rule, the samples numbered 9 and 11 are classified.

Up to this stage, 6 rules have been generated and all of the samples have been classified with these rules. Which samples are classified by which rule can be seen in Table 2.

Table 2. Examples classified by the obtained rules.

Rule no	Classified samples by rule	Unclassified samples
1	3 – 7 – 12 – 13	1 – 2 – 4 – 5 – 6 – 8 – 9 – 10 – 11 – 14
2	1 – 2 – 8	4 – 5 – 6 – 9 – 10 – 11 – 14
3	4 – 5 – 10	6 – 9 – 11 – 14
4	6 – 14	9 – 11
5	5 – 9 – 10 – 13	11
6	9 – 11	–

Step 4.

The samples classified by the rule numbered 5 are also classified by the other rules as well. The rule numbered 5 classified the samples numbered 5, 9, 10, and 13. As the samples numbered 5 and 10 are classified by rule 3, the sample numbered 9 by rule 6, and the sample numbered 13 by rule 1, rule 5 is cancelled. The rule set obtained after these processes is given in Table 3.

Table 3. Rule set that is generated by RES-2.

Rule no.	Rule definition
1	IF Global_risk = Low THEN Class = YES
2	IF Global_risk = High AND Return_time = Long THEN Class = NO
3	IF Global_risk = Medium AND Investment_level = High THEN Class = YES
4	IF Global_risk = Medium AND Investment_level = Low THEN Class = NO
5	IF Global_risk = High AND Return_time = Short THEN Class = YES

3. Experimental evaluation of RES-2

Datasets were used in the study to test the success of the developed method. A dataset consists of a set of already classified samples ($S = s1, s2, \ldots$). Each sample ($s_i = x_1, x_2, \ldots$) is a vector and x_i is defined as

the features or attributes of this sample. The dataset can be expanded using a $C = c_1, c_2, ...$ vector. Here, c_i designates the class of each sample.

Cross-validation is one of the most commonly used methods for testing the results. The predictive accuracy is measured by a well-known 10-fold cross-validation method for the used datasets. In this study, each dataset is divided into 2, as the training set and the test set, according to the 10-fold cross-validation method. The sample set is divided into 10 parts with this method, where 1 of those parts is the test set (10% of the sample set), the remaining parts are taken as the training set (90% of the sample set), and the problem is run 10 times. In the first run, the first 10% is taken as the test set; in the second run, the second 10% is taken as the test set; .... and in the final run, the last 10% is taken as the test set.

The rule base was created from training sets using the RES-2 algorithm, and these rule sets were applied to the test sets and the accuracies were calculated. During the test procedure, each sample in the test dataset is dealt with in the given order. Rules in the rule set are searched for the sample one by one. If the sample is accurately classified by a rule, the number of samples covered by the rule set is increased by one; otherwise, it is added to the set of inaccurately classifieds. This procedure is applied to all of the samples in the test dataset and the number of accurately classified samples is determined. The accuracy is then calculated by dividing this achieved number by the total number of samples in the test set. A repeat is made for the sample set divided into 10 parts according to the 10-fold cross-validation method and the accuracy of the sample set is calculated by taking the mean of the values achieved as a result.

Software was developed for the RES-2 algorithm using the Delphi programming language. Datasets existing in the text format were transferred to the Oracle database in order to run search procedures on the datasets faster using SQL expressions. Samples with incomplete values were not removed from the dataset, but they were ensured to be omitted by a special code integrated in the prepared software. The program was run on a PC computer with a Pentium IV 3.0 GHz processor and 1 GB RAM to compare results and results were obtained.

3.1. Characteristics of the used datasets

In this study, 15 datasets that contain the multiclass values of different characteristics and sizes obtained from the UCI repository of machine learning databases have been used in order to test the performance of the proposed method. The properties of the datasets used [attribute type, number of examples, number of attributes, number of classes, and missing values (%)] are given in Table 4, where it can be seen that the datasets have 2 to 19 classes and 4 to 61 characteristics. The attributes consist of Integer, Real, Categorical, and a mix of these, and their number of samples in the dataset vary between 148 and 12,960 (for more details, see http://archive.ics.uci.edu/ml/datasets.html).

3.2. Classification accuracy

The accuracy of the values obtained is calculated using Eq. (1). The accuracy is defined as the algorithm's rate of correctly classifying data that were not previously encountered.

$$Accuracy(\%) = \frac{\text{No. of test examples covered by the rule set}}{\text{Total no. of test examples}} x100 \quad (1)$$

The classification accuracy is obtained with the division of the test set chosen from the sample set and the samples correctly classified by the rule set by the total number of samples according to the 10-fold cross-

Table 4. Features of the example sets.

Dataset	Attribute type	Number of examples	Number of attributes	Number of classes	Percent of missing values
Balance-scale	Categorical	625	4	3	0.0
Breast cancer	Categorical	286	9	2	0.35
Breast cancer w.	Integer	699	10	2	0.23
Chess	Categorical	3196	36	2	0.0
CRX	Categorical, Integer, Real	690	15	2	0.65
Dermatology	Categorical, Integer	366	34	6	0.06
Diabetes	Categorical, Integer	768	8	2	0.0
Hepatitis	Categorical, Integer, Real	155	19	2	5.67
Iris	Real	150	4	3	0.0
Lympograph	Categorical	148	18	8	0.0
Mushroom	Categorical	8124	22	2	1.39
Nursery	Categorical	12,960	8	5	0.0
Soybean-Large	Categorical	307	35	19	6.56
Tic-Tac-Toe	Categorical	958	9	2	0.0
Vote	Categorical	435	16	2	4.14

validation method. The arithmetic average of the values obtained after the 10 repeats is taken and the average accuracy is calculated.

3.3. Results obtained using RES-2

Four criteria were used to test the results of the algorithms: the mean accuracy, number of rules, standard deviation, and process time. The results obtained from the RES-2 algorithm according to the 10-fold cross-validation method are given in Table 5.

Table 5. The results obtained by RES-2.

Dataset	Maximum accuracy (%)	Minimum accuracy (%)	Standard deviation	Average accuracy (%)	Number of rules	Execution time (s)
Balance-scale	91.94	50.00	13.90	79.43	17.86 ± 1.25	0.34
Breast cancer	94.74	77.78	5.13	86.18	22.50 ± 0.96	0.35
Breast cancer w.	98.44	93.31	1.46	95.74	18.14 ± 1.64	0.33
Chess	99.38	97.81	0.58	98.76	17.00 ± 1.1	103.21
CRX	95.52	81.54	4.30	89.05	39.43 ± 2.26	2.43
Dermatology	97.50	91.43	2.10	95.29	16.50 ± 1.12	19.23
Diabetes	88.59	73.68	4.40	79.26	17.43 ± 1.68	0.78
Hepatitis	93.33	79.42	5.08	84.53	12.57 ± 1.40	0.40
Iris	100.00	90.91	3.68	97.61	5.29 ± 0.70	0.22
Lympograph	98.50	66.67	10.52	81.85	23.83 ± 3.44	0.59
Mushroom	100.00	100.00	0.0	100.00	18.14 ± 2.03	79.56
Nursery	88.39	99.59	4.45	94.47	80.11 ± 3.09	180.27
Soybean-Large	100.00	77.78	7.23	94.14	20.14 ± 1.73	15.34
Tic-Tac-Toe	100.00	93.65	2.30	97.07	24.57 ± 1.18	0.62
Vote	100.00	95.35	1.25	97.44	10.43 ± 0.90	0.55

3.4. Performance comparison with other algorithms

For each dataset shown in Table 4, the performance of the RES-2 algorithm was compared with very well-known algorithms often used for benchmarking purposes in the data mining and machine learning literature.

All of the selected datasets belong to real-world domains and come from the well-known UCI Machine Learning Repository from UCI [31]. Real-world domains are useful because they come from real-world problems that we do not always understand and are therefore actual problems for which we would like to improve the performance. The comparisons with respect to the average accuracy and the number of rules induced by the RES-2 algorithm and some algorithms are given in Tables 6–20.

Table 6. Comparison of the RES-2 algorithm with other classifiers for the balance-scale dataset.

Algorithm	Average accuracy	Number of rules	References
RES-2	79.43	17.86 ± 1.25	This study
MCAR	77.54	16	Thabtah et al. [9]
PAT-DTL (AGM)	74.40	-	Kang and Sohn [32]
CBA	65.66	15	Thabtah et al. [9]
Ripper	64.56	17	Thabtah et al. [9]
C4.5	64.32	33	Thabtah et al. [9]

Table 7. Comparison of RES-2 with other classifiers for the Breast Cancer Wisconsin dataset.

Algorithm	Average accuracy	Number of rules	References
CBA	98.84	45	Thabtah and Cowling [11]
CPOM-NB	97.42	1	Cohen et al. [7]
NBTree	96.56	28	Cohen et al. [7]
RES-2	96.54	19.33 ± 1.70	This study
MCAR	96.48	61	Thabtah et al. [9]
SVM+Pr	96.30	5.1	Martens et al. [33]
Ant-Miner	96.04	6.2 ± 0.25	Su et al. [8]
RMR	95.92	60	Thabtah and Cowling [11]
Ripper	95.42	6	Thabtah and Cowling [11]
G-REX	95.10	2.2	Martens et al. [33]
Trepan	95.00	5.4	Martens et al. [33]
CN2	94.88	18.6 ± 0.45	Su et al. [8]
C4.5	94.66	14	Thabtah and Cowling [11]
DE/QDE	92.68	11.8 ± 1.08	Su et al. [8]

Table 8. Comparison of RES-2 with other classifiers for the Breast Cancer dataset.

Algorithm	Average accuracy	Number of rules	References
RES-2	86.18	22.50 ± 0.96	This study
C5.0	75.80	9	Pham and Afify [34]
DE/QDE	75.52	6.30 ± 1.19	Su et al. [8]
Ant-Miner	75.28	7.10 ± 0.31	Su et al. [8]
PAT-DTL (JS)	73.43	-	Kang and Sohn [32]
Rules-6	72.60	10	Pham and Afify [34]
Ant-Miner w/o rule pruning	70.69	19.60 ± 0.22	Su et al. [8]
PAT-DTL (AGM)	69.23	-	Kang and Sohn [32]
Rules-3 Plus	68.40	40	Pham and Afify [34]
CN2	67.69	55.40 ± 2.07	Su et al. [8]

Table 9. Comparison of RES-2 with other classifiers for the Chess dataset.

Algorithm	Average accuracy	Number of rules	References
Rules-3Plus	99.0	108	Pham and Afify [34]
RES-2	98.8	17.00 ± 1.1	This study
Rules-6	98.5	31	Pham and Afify [34]
C5.0	97.2	21	Pham and Afify [34]
ID3	93.9	-	Nielsen et al. [35]
C4.5	92.9	-	Nielsen et al. [35]
DIR	92.9	-	Nielsen et al. [35]

Table 10. Comparison of RES-2 with other classifiers for the CRX dataset.

Algorithm	Average accuracy	Number of rules	References
MOEA	93.48	-	Setzkorn and Paton [18]
C4.5	91.79	-	Carvalho and Freitas [36]
C4.5/GA-large-SN	90.40	-	Carvalho and Freitas [36]
Double C4.5	90.02	-	Carvalho and Freitas [36]
RES-2	89.05	39.43 ± 2.26	This study
C4.5/GA-small	88.94	-	Carvalho and Freitas [36]
SVM	83.92	-	Setzkorn and Paton [18]
CBA	83.50	-	Chan et al. [37]
GARC	82.50	-	Chan et al. [37]
EROL	81.63	-	Setzkorn and Paton [18]
S-NBTree	76.76	-	Wang et al. [19]
NBTree	75.42	-	Wang et al. [19]

Table 11. Comparison of RES-2 with other classifiers for the Dermatology dataset.

Algorithm	Average accuracy	Number of rules	References
GP	96.60	-	Bojarczuk [17]
DIMLP	95.70	-	Bologna [16]
RES-2	94.88	17.33 ± 1.25	This study
MLP	94.70	-	Bologna [16]
Ant-Miner	94.29	7.30 ± 0.15	Su et al. [8]
DE/QDE	91.53	11.90 ± 2.02	Su et al. [8]
CN2	90.38	18.50 ± 0.47	Su et al. [8]
C4.5	89.10	-	Bojarczuk [17]
BGP	86.20	-	Bojarczuk [17]
Ant-Miner w/o rule pruning	83.05	25.90 ± 0.31	Su et al. [8]

Table 12. Comparison of RES-2 with other classifiers for the Diabetes dataset.

Algorithm	Average accuracy	Number of rules	References
C4.5	85.82	20	Thabtah and Cowling [11]
RMR	78.79	65	Thabtah and Cowling [11]
RES-2	78.71	18.29 ± 0.45	This study
Ripper	76.04	4	Thabtah and Cowling [11]
CBA	75.34	36	Thabtah and Cowling [11]
CORE	75.34	-	Tan et al. [20]
NavieBayes	75.09	-	Tan et al. [20]
Cal5	75.00	-	Tan et al. [20]
CART	74.50	-	Tan et al. [20]
GGP	72.60	-	Tan et al. [20]
AC^2	72.40	-	Tan et al. [20]
CN2	71.10	-	Tan et al. [20]

Table 13. Comparison of RES-2 with other classifiers for the Hepatitis dataset.

Algorithm	Average accuracy	Number of rules	References
Ant-Miner w/o rule pruning	92.50	6.80 ± 0.13	Su et al. [8]
DE/QDE	90.97	4.30 ± 0.64	Su et al. [8]
Ant-Miner	90.00	3.40 ± 0.16	Su et al. [8]
CN2	90.00	7.20 ± 0.25	Su et al. [8]
RES-2	83.22	12.48 ± 0.64	This study
C4.5/GA-large-SN	82.52	-	Carvalho and Freitas [36]
TFPC	81.20	72.9	Coenen and Leng [10]
CMAR	81.00	153.1	Coenen and Leng [10]
C4.5/GA-small	79.36	-	Carvalho and Freitas [36]
Double C4.5	66.16	-	Carvalho and Freitas [36]
CBA	57.80	15.8	Coenen and Leng [10]

Table 14. Comparison results of RES-2 with other classifiers for the IRIS dataset.

Algorithm	Average accuracy	Number of rules	References
REGAL	99.00	11	Rodríguez et al. [14]
RES-2	97.61	5.00 ± 0.76	This study
Trepan	96.20	6.7	Martens et al. [33]
SVM+Pr	96.00	7	Martens et al. [33]
CPOM-NB	96.00	2	Cohen et al. [7]
EDGAR	96.00	14	Rodríguez et al. [14]
CLIP4	95.60	4	Cios and Kurgan [12]
MCAR	95.32	31	Thabtah et al. [9]
C4.5	95.10	4.3	Martens et al. [33]
G-REX	94.80	4.0	Martens et al. [33]
Ripper	94.66	4	Thabtah and Cowling [11]
NBTree	94.00	4	Cohen et al. [7]
RMR	93.87	15	Thabtah and Cowling [11]
CBA	93.25	18	Thabtah and Cowling [11]

Table 15. Comparison of RES-2 with other classifiers for the Lympograph dataset.

Algorithm	Average accuracy	Number of rules	References
SIM	86.20	-	Luukka [38]
Rules-6	86.00	15	Pham and Afify [34]
RES-2	81.85	24.17 ± 3.53	This study
CN2	81.60	-	Luukka [38], Bologna [16]
MLP	81.60	-	Luukka [38], Bologna [16]
C4.5	81.08	12	Thabtah et al. [9]
DIMLP	80.40	-	Luukka [38], Bologna [16]
Rules-3 Plus	80.00	24	Pham and Afify [34]
Ripper	77.02	6	Thabtah et al. [9]
CBA	76.38	38	Thabtah et al. [9]
MCAR	76.02	49	Thabtah et al. [9]
C5.0	76.00	7	Pham and Afify [34]

Table 16. Comparison of RES-2 with other classifiers for the Mushroom dataset.

Algorithm	Average accuracy	Number of rules	References
CLIP4	100.00	2	Cios and Kurgan [12]
RES-2	100.00	18.14 ± 2.03	This study
Ripper	99.90	14	Thabtah and Cowling [11],
			Thabtah et al. [9]
C4.5	99.77	44	Thabtah and Cowling [11],
			Thabtah et al. [9]
ID3	99.10	-	Nielsen et al. [35]
DIR	98.90	-	Nielsen et al. [35]
PDG1	98.80	-	Nielsen et al. [35]
MCAR	97.56	53	Thabtah et al. [9]
CBA	91.29	38	Thabtah and Cowling [11],
			Thabtah et al. [9]

Table 17. Comparison of RES-2 with other classifiers for the Nursery dataset.

Algorithm	Average accuracy	Number of rules	References
EDGAL	99.00	270 ± 28.41	Rodríguez et al. [14]
REGAL	98.00	309 ± 22.41	Rodríguez et al. [14]
NBTree	95.92	139	Cohen et al. [7]
RES-2	94.47	80.11 ± 3.09	This Study
CPOM-NB	94.21	15	Cohen et al. [7]
CBA	90.10	78.4	Coenen and Leng [10]
CMAR	88.30	298.6	Coenen and Leng [10]
NPGA	78.25	7	Dehuri and Mall [15]
TFPC	77.80	39.7	Coenen and Leng [10]
INPGA	76.65	7	Dehuri and Mall [15]
SGA	76.20	7	Dehuri and Mall [15]

Table 18. Comparison of RES-2 with other classifiers for the Soybean-Large dataset.

Algorithm	Average accuracy	Number of rules	References
RES-2	94.14	20.14 ± 1.73	This study
CBA	91.00	-	Coenen and Leng [10]
CMAR	90.80	-	Coenen and Leng [10]
TFPC	89.10	-	Coenen and Leng [10]
PDG1	88.20	-	Nielsen et al. [35]
NB	87.70		Nielsen et al. [35]
PAT-DTL(JKL)	87.41	-	Kang and Sohn [32]
DIR	85.90	-	Nielsen et al. [35]

Table 19. Comparison of RES-2 with other classifiers for the Tic-Tac-Toe dataset.

Algorithm	Average accuracy	Number of rules	References
CBA	100.00	25	Thabtah and Cowling [11]
RMR	100.00	26	Thabtah and Cowling [11]
MCAR	100.00	27	Thabtah et al. [9]
EDGAR	99.00	87 ± 14.45	Rodríguez et al. [14]
DE/QDE	98.85	10	Su et al. [8]
CN2	97.38	39.70 ± 2.52	Su et al. [8]
RES-2	97.10	24.86 ± 1.64	This study
Ripper	96.97	9	Thabtah and Cowling [11]
REGAL	91.00	50 ± 13.12	Rodríguez et al. [14]
C4.5	83.71	95	Thabtah and Cowling [11]
CPOM-NB	76.51	7	Cohen et al. [7]
NBTree	75.67	51	Cohen et al. [7]
Ant-Miner	73.04	8.50 ± 0.62	Su et al. [8]

Table 20. Comparison of RES-2 with other classifiers for the Vote dataset.

Algorithm	Average accuracy	Number of rules	References
RES-2	97.44	10.43 ± 0.90	This study
Rules-3 Plus	97.00	33	Pham and Afify [34]
C5.0	97.00	5	Pham and Afify [34]
Rules-6	95.60	10	Pham and Afify [34]
CLIP4	94.00	9.7	Cios and Kurgan [12]
MCAR	88.70	85	Thabtah et al. [9]
RMR	88.70	84	Thabtah and Cowling [11]
C4.5	88.27	4	Thabtah and Cowling [11], Thabtah et al. [9]
Ripper	87.35	4	Thabtah and Cowling [11], Thabtah et al. [9]
CBA	86.91	40	Thabtah and Cowling [11], Thabtah et al. [9]

When the results shown in these tables are examined, it can be seen that the proposed RES-2 algorithm both classifies the test set with high accuracy and generates a limited number of rules according to the rules that it obtains from the given training set. For instance, the RES-2 algorithm produced the highest results for a total of 5 datasets: 86.18% accuracy for Breast Cancer, 97.44% for Vote, 94.14% for Soybean-Large, 79.43% for

Balance-Scale, and 100% for Mushroom. It produced the second highest results for the Iris and Chess datasets at 97.61% and 98.8%, respectively, and it produced the third highest results for the Diabetes, Lympograph, and Dermatology datasets. It produced results with high levels of accuracy for the other datasets, as well.

4. Conclusion

In this article, a new algorithm directly producing rules with a simple and at the same time effective rule search mechanism has been presented. This algorithm explores all of the cases that could generate rules using the search technique, without performing any calculations in the rule generation stage, and turns them into rules. The search technique is based on testing whether the attributive–value combinations that could be rules correspond to only one class or not. Thus, it generates all of the rules and ensures that the most general rules are chosen. Fewer rules are generated by eliminating unnecessary and repeated rules at the end of the procedure. Furthermore, it classifies the training set with 100% accuracy, as it does not use a pruning method. According to the results obtained using the benchmark datasets taken from real life, the proposed method produces better results than many other algorithms in terms of both the accuracy and rule number. In the future, it may be possible for the algorithm to eliminate unqualified rules and generate fewer rules by applying a pruning procedure and calculating the quality of the obtained rules. Additional issues to be further studied include examining how the proposed algorithm can be implemented using other classification methods such as support vectors machines, ant colony optimization, or Bayesian networks.

Acknowledgment

All of the datasets were obtained from the University of California at Irvine's Repository of Machine Learning Databases and Domain Theories, managed by Patrick M Murphy.

References

[1] Ö. Akgöbek, Y.S. Aydın, E. Öztemel, M.S. Aksoy, "A new algorithm for automatic knowledge acquisition in inductive learning", Knowledge-Based Systems, Vol. 19, pp. 388–395, 2006.

[2] K.J. Cios, N. Liu, L.S. Goodenday, "Generation of diagnostic rules via inductive machine learning", Kybernetes, Vol. 22, pp. 44–56, 1993.

[3] D. Fournier, B. Cremilleux, "A quality index for decision tree pruning", Knowledge-Based Systems, Vol. 15, pp. 37–43, 2002.

[4] D.T. Pham, S. Bigot, S.S. Dimov, "A rule merging technique for handling noise in inductive learning", Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science, Vol. 218, pp. 1255–1268, 2004.

[5] D.T. Pham, S. Bigot, S.S. Dimov, "Rules-5: A rule induction algorithm for classification problems involving continuous attributes", Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science, Vol. 217, pp. 1273–1285, 2003.

[6] J.R. Quinlan, C4.5: Programs for Machine Learning", Morgan Kaufmann, San Mateo, CA, USA, 1993.

[7] S. Cohen, L. Rokach, O. Maimon, "Decision-tree instance-space decomposition with grouped gain-ratio", Information Sciences, Vol. 177, pp. 3592–3612, 2007.

[8] H. Su, Y. Yang, L. Zhao, "Classification rule discovery with DE/QDE algorithm", Expert Systems with Applications, Vol. 37, pp. 1216–1222, 2010.

[9] F. Thabtah, P. Cowling, S. Hammoud, "Improving rule sorting, predictive accuracy and training time in associative classification", Expert Systems with Applications, Vol. 31, pp. 414–426, 2006.

[10] F. Coenen, P. Leng, "The effect of threshold values on association rule based classification accuracy", Data and Knowledge Engineering, Vol. 60, pp. 345–360, 2007.

[11] F.A. Thabtah, P.I. Cowling, "A greedy classification algorithm based on association rule", Applied Soft Computing, Vol. 7, pp. 1102–1111, 2007.

[12] K.J. Cios, L.A. Kurgan, "CLIP4: Hybrid inductive machine learning algorithm that generates inequality rules", Information Sciences, Vol. 163, pp. 37–83, 2004.

[13] Ö. Akgöbek, "A hybrid approach for improving the accuracy of classification algorithms in data mining", Energy Education Science and Technology Part A: Energy Science and Research, Vol. 29, pp. 1039–1054, 2012.

[14] M. Rodríguez, D.M. Escalantea, A. Peregrín, "Efficient distributed genetic algorithm for rule extraction", Applied Soft Computing, Vol. 11, pp. 733–743, 2011.

[15] S. Dehuri, R. Mall, "Predictive and comprehensible rule discovery using a multiobjective genetic algorithm", Knowledge-Based Systems, Vol. 19, pp. 413–421, 2006.

[16] G. Bologna, "A model for single and multiple knowledge based networks", Artificial Intelligence in Medicine, Vol. 28, pp. 141–163, 2003.

[17] C.C. Bojarczuk, H.S. Lopes, A.A. Freitas, E.L. Michalkiewicz, "A constrained-syntax genetic programming system for discovering classification rules: Application to medical datasets", Artificial Intelligence in Medicine, Vol. 30, pp. 27–48, 2004.

[18] C. Setzkorn, R.C. Paton, "On the use of multi-objective evolutionary algorithms for the induction of fuzzy classification rule systems", BioSystems, Vol. 81, pp. 101–112, 2005.

[19] L.M. Wang, X.L. Li, C.H. Cao, S.M. Yuan, "Combining decision tree and naive Bayes for classification", Knowledge-Based Systems, Vol. 19, pp. 511–515, 2006.

[20] K.C. Tan, E.J. Teoh, Q. Yu, K.C. Goh, "A hybrid evolutionary algorithm for attribute selection in data mining", Expert Systems with Applications, Vol. 36, pp. 8616–8630, 2009.

[21] E. Elalfi, R. Haque, M.E. Elalami, "Extracting rules from trained neural network using GA for managing E-business", Applied Soft Computing, Vol. 4, pp. 65–77, 2004.

[22] H. Kahramanli, N. Allahverdi, "Rule extraction from trained adaptive neural networks using artificial immune systems", Expert Systems with Applications, Vol. 36, pp. 1513–1522, 2009.

[23] K.M. Osei-Bryson, "Post-pruning in decision tree induction using multiple performance measures", Computers and Operations Research, Vol. 34, pp. 3331–3345, 2007.

[24] J.R. Quinlan, "Learning efficient classification procedures and their application to chess end games", In: R.S. Michalski, J.G. Carbonell, T.M. Mitchell, Eds., Machine Learning: An Artificial Intelligence Approach, Morgan Kaufmann, Los Altos, CA, USA, pp. 463–482, 1983.

[25] R.S. Michalski, "A theory and methodology of inductive learning, machine learning - an artificial intelligence approach", In: R.S. Michalski, J.G. Carbonell, T.M. Mitchell, Eds., Machine Learning: An Arti?cial Intelligence Approach, Morgan Kaufmann, Los Altos, CA, USA, pp. 83–134, 1983.

[26] U.M. Fayyad, K.B. Irani, "Multi-interval discretization of continuous-valued attributes", 13th International Joint Conference on Artificial Intelligence, pp. 1022–1027, 1993.

[27] L. Breiman, J.H. Fiedman, R.A. Olshen, C.J. Stone, Classification and Regression Trees, Wadsworth, Belmont, CA, USA, 1984.

[28] C.H. Lee, "A Hellinger-based discretization method for numeric attributes in classification learning", Knowledge-Based Systems, Vol. 20, pp. 419–425, 2007.

[29] K.A. Toh, "An error-counting network for pattern classification", Neurocomputing, Vol. 71, pp. 1680–1693, 2008.

[30] M.M. Oprea, "Inductive learning applied to knowledge acquisition for an expert systems", 35th Year of Petroleum-Gas University Activity, 2002.

[31] C.J. Blake, C.J. Merze, UCI Repository of Machine Learning Database (Machine Readable Data Repository), Department of Information and Computer Science, University of California, Irvine, 1999. Available at http://archive.ics.uci.edu/ml/datasets.html.

[32] D.K. Kang, K. Sohn, "Learning decision trees with taxonomy of propositionalized attributes", Pattern Recognition, Vol. 42, pp. 84–92, 2009.

[33] D. Martens, B. Baesens, T.V. Gestel, J. Vanthienen, "Comprehensible credit scoring models using rule extraction from support vector machines", European Journal of Operational Research, Vol. 183, pp. 1466–1476, 2007.

[34] D.T. Pham, A.A. Afify, "RULES-6: A simple rule induction algorithm for supporting decision making", 31st Annual Conference of IEEE Industrial Electronics Society, pp. 2184–2189, 2005.

[35] J.D. Nielsen, R. Rumi, A. Salmerón, "Supervised classification using probabilistic decision graphs", Computational Statistics and Data Analysis, Vol. 53, pp. 1299–1311, 2009.

[36] D.R. Carvalho, A.A. Freitas, "A hybrid decision tree/genetic algorithm for data mining", Information Sciences, Vol. 163, pp. 13–35, 2004.

[37] G. Chan, H. Lui, L. Yu, Q. Wei, X. Zhang, "A new approach to classification based on association rule mining", Decision Support Systems, Vol. 42, pp. 674–689, 2006.

[38] P. Luukka, "Similarity classifier using similarity measure derived from Yu's norms in classification of medical data sets", Computers in Biology and Medicine, Vol. 37, pp. 1133–1140, 2007.