

A Softened Formulation of Inductive Learning and Its Use for Coronary Disease Data

Janusz Kacprzyk and Grażyna Szkatula

Systems Research Institute, Polish Academy of Sciences
ul. Newelska 6, 01-447 Warsaw, Poland
{szkatulg,kacprzyk}@ibspan.waw.pl

Abstract. We present an improved inductive learning method to derive classification rules that correctly describe most of the examples belonging to a class and do not describe most of the examples not belonging to this class. The problem is represented as a modification of the set covering problems solved by a genetic algorithm. Its is employed to medical data on coronary disease, and the results seem to be encouraging.

1 Introduction

In learning from examples, traditionally, we seek a classification rule satisfying *all* positive examples and *no* negative examples. These requirements are often too strict, and some softening may help. Here, we follow Zadeh's computing with words and perceptions paradigm (cf. Zadeh and Kacprzyk [20]), using fuzzy logic at the level of soft problem formulation, and non-fuzzy techniques at the solution level of its solution. We postulate: (1) *a partial completeness*, i.e. that the classification rule must correctly describe (have the same attribute values), say, *most* of the positive examples, (2) *a partial consistency*, i.e. that the classification rule must describe, say, *almost none* of the negative examples, (3) *convergence*, i.e. the classification rule must be derived in a *finite* number of steps, (4) the classification rule of a *minimal "length"* is to be found, e.g. with the minimum number of attributes (or, more generally being "simple").

Examples are described (cf. Michalski [19]) by a set of K "attribute - value" pairs $e = \bigwedge_{j=1}^K [a_j \# v_j]$ where a_j denotes attribute j with value v_j and $\#$ is a relation as, e.g., $=$, $<$, $>$, \approx , \geq , etc. For instance, for the attributes: *height*, *color_of_hair*, *color_of_eyes*, we can describe the look of a person as $[height = "high"] \wedge [color_of_hair = "blond"] \wedge [color_of_eyes = "blue"]$.

We propose a modified inductive learning procedure based on Michalski's [19] star-type methodology, related to our previous work (cf. Kacprzyk and Szkatula [10 – 18]). A pre-processing of data is first performed based on an analysis of how frequent the values of the particular attributes occur. These frequencies are used to define *typical values* by deriving weights associated with those values. The problem is a modification of set covering, and solved by a genetic algorithm (IP2_GA).

2 A Softened Problem Formulation of Inductive Learning

Sets of examples U and attributes $A = \{a_1, \dots, a_K\}$ are finite. $V_{a_j} = \{v_{i_1}^{a_j}, \dots, v_{i_j}^{a_j}\}$ is a domain of a_j , $j=1, \dots, K$, $V = \bigcup_{j=1, \dots, K} V_{a_j}$. $f: U \times A \rightarrow V$ is function, called an *informational function*, $f(e, a_j) \in V_{a_j}$, $\forall a_j \in A$, $\forall e \in U$. Each $e \in U$, with K attributes, $A = \{a_1, \dots, a_K\}$, is written $e = \bigwedge_{j=1}^K [a_j = v_i^{a_j}]$, where $v_i^{a_j} = f(e, a_j) \in V_{a_j}$ denotes attribute a_j taking on value $v_i^{a_j}$ for example e . An e in (1) is composed of K "attribute-value" pairs, denoted $s_j = [a_j = v_i^{a_j}]$ (selectors). The conjunction of $l \leq K$ "attribute-value" pairs, i.e.

$$C^I = \bigwedge_{j \in I} s_j = \bigwedge_{j \in I} [a_j = v_i^{a_j}] = [a_{j_1} = v_i^{a_{j_1}}] \wedge \dots \wedge [a_{j_l} = v_i^{a_{j_l}}] \quad (1)$$

where $I = \{j_1, j_2, \dots, j_l\} \subseteq \{1, \dots, K\}$ is called a *complex*.

Let us have example e and a complex $C^I = [a_{j_1} = v_i^{a_{j_1}}] \wedge \dots \wedge [a_{j_l} = v_i^{a_{j_l}}]$ that corresponds to the set of indices $I = \{j_1, \dots, j_l\} \subseteq \{1, \dots, K\}$. The set of indices $\{j_1, \dots, j_l\}$ is equivalent to a vector $x = [x_j]^T$, $j=1, \dots, K$, such that $x_j = 1$ if a selector $s_j = [a_j = v_i^{a_j}]$ occurs in C^I , and 0 otherwise. Complex C^I covers example e if all conditions on attributes given as selectors are covered by (equal to) the values of the respective attributes in e , i.e. $f(C^I, a_j) = f(e, a_j), \forall j \in I$.

Now, a_d is a decision attribute and $V_{a_d} = \{v_{i_1}^{a_d}, \dots, v_{i_d}^{a_d}\}$ is a domain of a_d . Each example $e \in U$ is described by a set of attributes $\{a_1, a_2, \dots, a_K\} \cup \{a_d\}$. So, attribute a_d determines a partition $\{Y_{v_{i_1}^{a_d}}, Y_{v_{i_2}^{a_d}}, \dots, Y_{v_{i_d}^{a_d}}\}$ of set U , where $Y_{v_{i_t}^{a_d}} = \{e \in U : f(e, a_d) = v_{i_t}^{a_d}\}$, $v_{i_t}^{a_d} \in V_{a_d}$ for $t=1, \dots, d$. Set $Y_{v_{i_t}^{a_d}}$ is called the t -th *decision class* (for $v_{i_t}^{a_d} \in V_{a_d}$), $Y_{v_{i_1}^{a_d}} \cup \dots \cup Y_{v_{i_d}^{a_d}} = U$, $Y_{v_{i_t}^{a_d}} \cap Y_{v_{i_j}^{a_d}} = \emptyset$ for $i \neq j$.

Suppose that we have a set of *positive* and *negative examples*:

$$S_P(Y_{v_{i_t}^{a_d}}) = \{e \in U : f(e, a_d) = v_{i_t}^{a_d}\} \quad (2)$$

$$S_N(Y_{v_{i_t}^{a_d}}) = \{e \in U : f(e, a_d) \neq v_{i_t}^{a_d} \text{ and } \forall e' \in S_P(Y_{v_{i_t}^{a_d}}) \\ \exists a_j \in P, f(e, a_j) \neq f(e', a_j)\} \quad (3)$$

and $S_P(Y_{v_{i_t}^{a_d}}) \cap S_N(Y_{v_{i_t}^{a_d}}) = \emptyset$ and $S_P(Y_{v_{i_t}^{a_d}}) \neq \emptyset$, $S_N(Y_{v_{i_t}^{a_d}}) \neq \emptyset$.

The rule "IF C^I THEN $[a_d = v_{i_t}^{a_d}]$ " is called an "elementary" rule for class $Y_{v_{i_t}^{a_d}}$, where C^I is a description of example in terms of attributes $a_j, j \in I$, and this example belongs to class $Y_{v_{i_t}^{a_d}}$. We consider the classification rules:

$$\text{IF } C^{I_1} \cup \dots \cup C^{I_L} \text{ THEN } [a_d = v_{i_t}^{a_d}] \quad (4)$$

with: $I_1, \dots, I_L \subseteq \{1, \dots, K\}$, $C^{I_l} = \bigwedge_{j \in I_l} [a_j = v_i^{a_j}]$, $l = 1, \dots, L$.

Suppose we have P positive examples, $e^m \in S_P(Y_{v_{i_t}^{a_d}})$, $m = 1, \dots, P$, and N negative examples, $e^n \in S_N(Y_{v_{i_t}^{a_d}})$, $n = 1, \dots, N$. For each a_j , each possible value occurs at some intensity (frequency). If it occurs more frequently in the positive examples and less frequently in the negative ones, then it is somehow typical and should rather appear in the rule sought. So, we introduce the function, for each a_j , $j = 1, \dots, K$ and $v \in V_{a_j}$

$$g_j(v) = \frac{1}{P} \sum_{m=1}^P \delta(e^m, v) - \frac{1}{N} \sum_{n=1}^N \delta(e^n, v) \quad (5)$$

for each $v \in V_{a_j}$, where: $\delta(e^m, v) = \begin{cases} 1 & \text{for } v_i^{a_j} = v \\ 0 & \text{otherwise} \end{cases}$, and: $e^m \in S_P$,

$v_i^{a_j} = f(e^m, a_j) \in V_{a_j}$; and analogously for $\delta(e^n, v)$. So, we may express to what degree the particular values $v \in V_{a_j}$ of attribute a_j occurs more often in the positive than negative examples.

We assume that $g_j(v)$ is used as a weight of value $v \in V_{a_j}$ of each a_j (cf. Kacprzyk and Szkatula [15, 16]).

Example e_W with weights is $e_W = \bigwedge_{j=1}^K [a_j = v_i^{a_j}; g_j(v_i^{a_j})]$, i.e. is a conjunction of weighted selectors, $s_j^W = [a_j = v_i^{a_j}; g_j(v_i^{a_j})]$, that is

$$C_W^I = \bigwedge_{j \in I \subseteq \{1, \dots, K\}} s_j^W \quad (6)$$

and is called a *weighted complex*. Notice that for C_W^I x has the elements $x_j = 1$ for $j \in I$, while, for $j \in \{1, 2, \dots, K\} \setminus I$, $x_j = 0$. For C_W^I its *weighted length* is:

$$d_W(C_W^I) = \sum_{j \in I} (1 - g_j(v_i^{a_j})) \cdot x_j + \sum_{j \in \{1, 2, \dots, K\} \setminus I} (1 - g_j(v_i^{a_j})) \cdot x_j = \sum_{j=1}^K (1 - g_j(v_i^{a_j})) \cdot x_j \quad (7)$$

which reflects a higher relevance of those values of attributes which occur more often in the positive than in negative examples.

The *length of the weighted classification rule* $R_W = C_W^{I_1} \cup \dots \cup C_W^{I_L}$ is

$$d_{R_W}(C_W^{I_1} \cup \dots \cup C_W^{I_L}) = \max_{l=1, \dots, L} d_W(C_W^{I_l}) \quad (8)$$

We look for an optimal classification rule $R_W^* = C_W^{I_1^*} \cup \dots \cup C_W^{I_L^*}$ such that

$$\min_{I_1, \dots, I_L} d_{R_W}(C_W^{I_1} \cup \dots \cup C_W^{I_L}) \quad (9)$$

As the (exact) solution of (11) is very difficult, an auxiliary problem is solved (cf. Kacprzyk and Szkatula [11]) whose solution is in general very close but much easier, i.e. an $R_W^* = C_W^{I_1^*} \cup \dots \cup C_W^{I_L^*}$ is sought such that

$$\min_{I_1} d_W(C_W^{I_1}), \dots, \min_{I_L} d_W(C_W^{I_L}) \quad (10)$$

3 Solution by Using the IP2_GA Method

For $e^p \in S_P$, and all the negative examples $e^{P+n} \in S_N$, $n = 1, \dots, N$, we construct a 0-1 matrix $Z_{N \times K} = [z_{nj}]$, $n = 1, \dots, N$, $j = 1, \dots, K$, defined as

$$z_{nj} = \begin{cases} 1 & \text{for } f(e^p, a_j) = f(e^{P+n}, a_j) \\ 0 & \text{for } f(e^p, a_j) \neq f(e^{P+n}, a_j) \end{cases} \quad (11)$$

whose rows correspond to the consecutive negative examples $e^{P+n} \in S_N$, $n = 1, \dots, N$ and columns to the attributes a_1, \dots, a_K ; $z_{nj} = 1$ occurs if a_j takes on different values

in the positive and negative example, i.e. $f(e^p, a_j) \neq f(e^{p+n}, a_j)$, and $z_{nj} = 0$ otherwise. There are no rows with all elements equal 0 since the sets of positive and negative examples are disjoint (and non-empty). Thus, for any positive and negative example there is always at least one attribute with a different value in these examples.

Consider now the following inequality

$$\sum_{j=1}^K z_{nj} x_j \geq \gamma_n, \quad n = 1, \dots, N \quad (12)$$

where $\gamma = [\gamma_1, \dots, \gamma_N]^T$ is a 0-1 vector, and $x_j \in \{0,1\}$, for $j = 1, \dots, K$.

Any vector x satisfying $Zx \geq \gamma$ (12) determines therefore uniquely some complex such that the partial completeness and consistence are satisfied. It describes at least one example from the set of positive examples, and it does not describe most of the examples from the set of negative examples. If vector x does not describe the n -th negative example, then $\gamma_n = 1$; and $\gamma_n = 0$ otherwise.

The minimization in (10), using inequality (12), is

$$\min_{x: Zx \geq \gamma} \sum_{j=1}^K (1 - g_j(v_i^{a_j})) \cdot x_j \quad (13)$$

The minimization over the set of indices I_l may be replaced by the minimization with respect to x which yields [cf. (7)] an $R_W^* = C_W^{I_1^*} \cup \dots \cup C_W^{I_L^*}$ such that

$$\min_{x: Z^1 x \geq \gamma} d_W(C_W^{I_1}), \dots, \min_{x: Z^L x \geq \gamma} d_W(C_W^{I_L}) \quad (14)$$

Each minimization with respect to x in (14) is therefore equivalent to the determination of a 0-1 vector x^* which uniquely determines the complex of the shortest weighted length. On the other hand, the satisfaction of $Zx \geq \Lambda$ (Λ is a unit vector) guarantees that such a complex would not describe all negative examples. If rules defining class $Y_{v_i^{a_d}}^{a_d}$ must describe *almost none* of the negative examples,

problem (13) can be written as a modification of the set covering problem

$$\min_{x, \gamma} \sum_{j=1}^K c_j x_j, \text{ subject to } \sum_{j=1}^K z_{nj} x_j \geq \gamma_n, \quad n = 1, \dots, N \quad (15)$$

with an additional constraint: $\sum_{n=1}^N \gamma_n \geq N - rel$, where $c_j = (1 - g_j(v_i^{a_j}))$, $z_{nj} \in \{0,1\}$,

$x_j \in \{0,1\}$, $j = 1, \dots, K$, $\gamma = [\gamma_1, \dots, \gamma_N]^T$, $\gamma_n \in \{0,1\}$, $rel \geq 0$.

This is the same as the original set covering problem with the exception that no more than rel rows are uncovered. Then, clearly no more than rel rows can be deleted. We may, in deleting rows, lose some information about the problem. This reduction

cannot always be applied. In the set covering problem (cf. Beasley and Chu [4]) there is only constraint, and $\gamma = [\gamma_1, \dots, \gamma_N]^T$ is a unit vector. Problem (15) is the problem of covering at least $N\text{-rel}$ rows of an N -row, K -column, zero-one matrix (z_{nj}) by a subset of the columns at minimal cost c_j . We define $x_j = 1$ if column j with cost $c_j > 0$ is in the solution, and $x_j = 0$ otherwise. Then, most rows (at least $N\text{-rel}$ rows) are covered by at least one column. It always has a feasible solution (a unit vector x of K element), due to the required disjointness of the sets of positive and negative examples and the way the matrix Z was constructed.

So, we seek a 0-1 vector x at minimum cost and a 0-1 vector $\gamma = [\gamma_1, \dots, \gamma_N]^T$ which determines the covered rows, $\gamma_n = 1$ if row n is covered by solution x , and $\gamma_n = 0$, otherwise. By assumption, at least $N\text{-rel}$ rows must be covered by x .

The set covering problem is a well-known NP-complete combinatorial optimization problem. A number of optimal and faster heuristic algorithms have been proposed, cf. Grossman and Wool [9]. Beasley and Chu [4] presented a genetic algorithm, with modified operations.

For solving (14) we propose a new procedure, IP2_GA, based on a genetic algorithm. We assume that the classification rules must correctly describe *most of the examples*, at least $A_{learning}$; the measure of classification accuracy $A_{learning}$ is the percentage of examples correctly classified. We assume a K -bit binary string which represents a potential solution, K is the number of variables. One for bit j implies that column j is in solution x^l , i.e. that x_j^l is in the solution. In IP2_GA in each iteration all solutions are evaluated with respect to their completeness and consistency.

The fitness of an individual solution x is

$$eval(x) = f(x) - g_{\max} \frac{K}{N} \sum_{n=1}^N f_n(x) \quad (16)$$

$$f(x) = \sum_{j=1}^K c_j x_j ; \quad f_n(x) = \begin{cases} 0 & \text{for } \sum_{j=1}^K z_{nj} \cdot x_j > 0 \\ 1 & \text{for } \sum_{j=1}^K z_{nj} \cdot x_j = 0 \end{cases} ; g_{\max} = \max\{g_j : j = 1, \dots, K\},$$

$n=1, \dots, N$; x_j is the value of column j in the string and c_j is the cost of column j .

The structure of a new population is chosen by a stochastic universal sampling (cf. Baker [1]). The consecutive steps of IP2_GA are:

Step 1. Initialize: $S = S_p$, i.e. the whole set of examples is initially assumed to contain the positive ones, S_N is a set of negative examples, and $R_W^* = \emptyset$, i.e. the initial set of complexes is assumed empty, iteration $j = 0$, given parameter $rel \geq 0$.

Step 2. Iteration $j = j + 1$. Determine the weights G by analyzing (pre-processing) of the examples due to (4).

Step 3. Determine an appropriate starting point; a good one may be a centroid (cf. Kacprzyk and Szkatula [13, 15]) that is a (possibly non existing) example in which the attributes take on values that occur most often in the positive examples and seldom in the negative examples. In the set of positive examples we find the closest positive example e^P to centroid, as the starting point for the next iterations.

Step 4. For the e^P we form the matrix $Z_{N \times K} = [z_{nj}]$, $n = 1, \dots, N$, $j = 1, \dots, K$, due to (13) and form a modification of the set covering problem, due to (15).

Step 5. We apply a genetic algorithm.

Step 1'. Set $t = 1$. Generate an initial population of random solutions $P(t) = \{x^1, x^2, \dots, x^P\}$, and evaluate the fitness $eval(x^l)$ of individuals in the population.

Step 2'. For the first solutions a crossover operator is applied. Two solutions are chosen and form two new solutions.

Step 3'. A mutation operator is applied to each solution in the population.

Step 4'. The new solution generated by the crossover and mutation procedures may not be feasible. We evaluate the fitness $eval(x^l)$ of new individuals in the population.

Step 5'. If a termination condition is satisfied, then STOP, and the best solution is the one with the smallest fitness; otherwise, go to Step 6'.

Step 6'. Select a new population $P(t+1)$ from $P(t)$ and return to Step 2'.

The 0-1 vector $x^* = [x_1^*, \dots, x_K^*]^T$ found determines uniquely the complex $C_W^{I_j^*}$, and the 0-1 vector $\gamma = [\gamma_1, \dots, \gamma_N]^T$ determines the fulfilled constraints. The complex can not describe more than rel examples (given $rel \geq 0$). Now, we can go to Step 6.

Step 6. Include the complex $C_W^{I_j^*}$ found in Step 5 into the classification rule sought R_W^* (i.e. that with the minimal weighted length), $R_W^* := R_W^* \cup C_W^{I_j^*}$, and discard from the set of examples S all examples covered by that complex.

Step 7. If the set of examples S remaining is *small enough*, STOP and the rule sought is R_W^* , is the one sought; otherwise, return to Step 2.

4 Using IP2_GA to Solve a Coronary Heart Disease Problem

Several factors of blood, both morphological as well as of plasma, can indicate some illness. Only very basic blood examination is unfortunately so far widely considered though, e.g., blood viscosity may change due to some physical and psychical conditions of people, both ill and well.

We have 90 examples, either ill or healthy persons, 60 of them are a training set and 30 are for testing. The 12 blood factors (attributes) are measured: $lk1$ - blood viscosity for coagulation quickness 230/s, $lk2$ - blood viscosity for coagulation quickness 23/s, $lk3$ - blood viscosity for coagulation quickness 15/s, $lp1$ - plasma

viscosity for coagulation quickness 230/s, *lp2* - plasma viscosity for coagulation quickness 23/s, *agr* - aggregation level of red blood cells, *fil* - blood cells capacity to change shape, *fib* - fibrin level in plasma, *ht* - hematocrite value, *sas* - sial acid rate in blood serum, *sak* - sial acid rate in blood cells, *ph* - acidity of blood.

The learning problem is formulated as to find classification rules into the classes:

class 1: no coronary disease,

class 2: coronary disease.

We use IP2_GA (with elements of a genetic algorithm) and IP2_GRE (with elements of a greedy algorithm), and assume that the classification rules must correctly describe *most of the learning examples* belonging to class 1 and 2, at least $A_{learning}$, by assumption. The results are shown in Tables 1 and 2, and we denote: IP2_GRE1 - $A_{learning} = 100\%$, IP2_GRE2 - $A_{learning} \geq 97\%$, IP2_GA1 - $A_{learning} = 100\%$ and IP2_GA2 - $A_{learning} \geq 90\%$. The percentage of correct classifications is the measure of classification accuracy, in percentage. The computational results are given below. A better classification accuracy for testing examples was obtained by using the classification rules correctly describes *most* of the training examples.

Table 1. Parameters describing the process of finding a classification rule for class 1/2

Algorithm	Number of iterations for class 1/class 2	Number of selectors in rule for class 1/class 2
IP2_GRE1	16/19	43/55
IP2_GRE2	13/17	2633
IP2_GA1	18/19	46/49
IP2_GA2	13/19	26/37

Table 2. Some parameters describing the process of classification the patients

Algorithm	$A_{learning}$ %, by assumption	Classification accuracy, $A_{testing}$ achieved
IP2_GRE1	100 %	90.0 %
IP2_GRE2	at least 97 %	96.7 %
IP2_GA1	100 %	73.4 %
IP2_GA2	at least 90 %	96.7 %

5. Concluding Remarks

We proposed an improved inductive learning procedure IP2_GA, based on a genetic algorithm. Results seem to be very encouraging.

Bibliography

1. Baker J.E. (1987) Reducing bias and inefficiency in the selection algorithm. In: Genetic Algorithms and Their Applications: Proceedings of the 2nd International Conference on Genetic Algorithms. (ed.) J.J. Grefenstette, 14-21, LEA, Cambridge, MA.
2. Balas E. (1980) Cutting planes from conditional bounds: a new approach to set covering. *Mathematical Programming Study* 12, 19-36.
3. Balas E. and Padberg M.W. (1979) Set partitioning - A survey. In: N. Christofides (ed.) *Combinatorial Optimisation*, Wiley, New York.
4. Beasley J.E., Chu P.C. (1994) *A genetic algorithm for the set covering problem*. Technical Report, The Management School, Imperial College.
5. Beasley J.E. (1996) A genetic algorithm for the set covering problem. *European Journal of Operational Research* 94, 392-404.
6. Christofides N. and Korman S. (1975) A computational survey of methods for the set covering problem. *Management Science* 21, 591-599.
7. Chvatal V. (1979) A greedy heuristic for the set-covering problem. *Math. of Oper. Res.* 4 (3) 233-235.
8. Garfinkel R. S. and Nemhauser G.L. (1978) *Integer programming*. John Wiley & Sons, New York-London-Sydney-Toronto.
9. Grossman T. and Wool A. (1995) Computational experience with approximation algorithms for the set covering problem. Working paper, Theoretical Division and CNLS, Los Alamos National Laboratory.
10. Kacprzyk J., Szkatuła G. (1994) Machine learning from examples under errors in data, Proceedings of Fifth International Conference in Information Processing and Management of Uncertainty in Knowledge-Based Systems IPMU'94 Paris France, Vol.2, 1047-1051
11. Kacprzyk J. and Szkatuła G. (1995) Machine learning from examples under errors in data. In B. Bouchon-Meunier, R.R. Yager and L.A. Zadeh (eds.): *Fuzzy Logic and Soft Computing*, World Scientific, Singapore, 1995, pp. 31 - 36.
12. Kacprzyk J. and Szkatuła G. (1996) An algorithm for learning from erroneous and incorrigible examples, *Int. J. of Intelligent Syst.* 11, 565 - 582.
13. Kacprzyk J. and Szkatuła G. (1997a) An improved inductive learning algorithm with a preanalysis of data", In Z.W. Raś and A. Skowron (eds.): *Foundations of Intelligent Systems* (Proceedings of 10th ISMIS'97 Symposium, Charlotte, NC, USA), Springer-Verlag, Berlin, 157 - 166.
14. Kacprzyk J. and Szkatuła G. (1997b) Deriving IF-THEN rules for intelligent decision support via inductive learning", in N.Kasabov et al. (eds.): *Progress in Connectionist-Based Information Systems* (Proceedings of ICONIP'97, ANZIS'97 and ANNES'97 Conference, Dunedin, New Zealand), Springer, Singapore, vol. 2, 818 - 821.
15. Kacprzyk J. and Szkatuła G. (1998) IP1 - An Improved Inductive Learning Procedure with a Preprocessing of Data. In L. Xu, L.W. Chan, I. King and A. Fu (eds.): *Intelligent Data Engineering and Learning. Perspectives on Financial Engineering and Data Mining* (Proceedings of IDEAL'98, Hong Kong), Springer, Hong Kong, pp. 385-392, 1998.
16. Kacprzyk J. and Szkatuła G. (1999) An inductive learning algorithm with a preanalysis of data. *Int. J. of Knowledge - Based Intelligent Engineering Systems*, vol. 3, 135-146.
17. Kacprzyk J. and Szkatuła G. (2002) An integer programming approach to inductive learning using genetic algorithm. In: Proceedings of the 2002 Congress on Evolutionary Computation, CEC'02, Honolulu, Hawaii, pp. 181-186.
18. Kacprzyk J. and Szkatuła G. (2002) An integer programming approach to inductive learning using genetic and greedy algorithms. In L.C. Jain and J. Kacprzyk (eds.): *New Learning Paradigms in Soft Computing*, Physica-Verlag, Heidelberg and New York, 2002, pp. 322-366.
19. Michalski R.S. (1983) A theory and methodology of inductive learning. In: R. Michalski, J. Carbonell and T.M. Mitchell (Eds.), *Machine Learning*. Tioga Press.
20. Zadeh L.A. and J. Kacprzyk (1999) *Computing with Words in Information/Intelligent Systems*. Vol. Foundations, Vol. 2 Applications, Physica-Verlag, Heidelberg and New York.