Threshold rules decision list

Marcin Blachnik
Silesian University of Technology, Department of Material Engineering and Metallurgy, Krasińskiego 8 , 40-019 Katowice, Poland
e-mail: marcin.blachniki@polsl.pl

Włodzisław Duch
Nicolaus Copernicus University, Department of Informatics, Grudziądzka 5,Toruń, Poland & Nanyang Technological University, School of Computer Engineering, Singapore
Google: Duch

Tadeusz Wieczorek
Silesian University of Technology, Department of Material Engineering and Metallurgy, Krasińskiego 8, 40-019 Katowice, Poland
e-mail: tadeusz.wieczorek@polsl.pl

Abstract

Understanding data is one of most important problems. Popular crisp logic rules are easy to understand and compare, however for some datasets the number of extracted rules is very large, what affect reduction of generalization and makes the system less transparent. Another solution are fuzzy logic rules, which are much more flexible, however they don't support symbolic and nominal attributes. Alternative systems for rules extraction base on prototype rules, this type of rules drives from similarity base learning. Presented threshold rules algorithm extracts form data small number of ordered rules, which are very accurate. Numerical experiments on real data show the usefulness of such approach as an alternative to neurofuzzy models.

Keywords: artificial intelligence, expert systems, decision support, knowledge discovery.

1. Introduction

Nowadays we are witnesses of flash development of computer science. Speed of popular, cheep processors is so high that even science-fiction writers of eighties couldn't imagine such situation; moreover increasing sizes of data storage allow collecting enormous numbers of data. Access to different information by the agency of Internet allows collecting information about our habits and preferences which can be used by economists to produce better commodity.

Computerization development entered into medicine and industry where data taken from different sensors and results of measurements are no longer saved on paper rather on computer data storage.

All this things make our society full of information unfortunately this is not correlated with knowledge which we have about all this subjects [11]. In this case people are determined to introduce and develop new methods allowing exploring the data and discovering knowledge. The only possibility to create such system is to develop an algorithm which can learn from known examples and then understanding how this algorithm works analyze what it has learned. Most popular of such algorithms are decision trees like C4.5, CART or SSV [13][8][2] tree and rules systems like AQ or CN2 [5]. All this methods works very well producing sets of crisp logical rules, however they have some limits, a specially when optimal decision border is linear crossing different attributes. In this situation all this methods produce step like decision border with large set of complicate rules. Much better are fuzzy rules (F-rules) [12] which allow obtaining linear decision border, and real word application shows that this approach is very successful. Unfortunately also this group of methods is not free of disadvantages. Usually comparison of obtained rules and their transparency is very difficult and not clear, and the last, but maybe one of most important problems are different types of features. F-rules works very well on continues and liner attributes but real word applications often require calculations on symbolic or nominal attributes, which F-rules don't support. This makes F-rules impracticable for large subset of applications, like for example in analyzing gene expressions.

Another and alternative way to understand the data derives form similarity-based learning framework (SBL) [5] - prototype rules (P-rules). This group of methods looks similar to F-rules however seems more general, being not restricted in applications. P-rules base on fixing prototypes and during analyzes compares unknown case with set of earlier defined reference vectors. Basically there are two different types of rules [1]

- nearest neighbour (or nearest prototype) rule
- threshold rules

First group of methods try to find prototype or prototypes which are closest to test vector and on this basis the output class is assigned as closest prototype label, second group of methods – threshold rules – use a distance or similarity measure and a threshold, finally assigning output class, the some as prototypes class, for all vectors which falls into the hypersphere surrounding this prototype (for Euclidian distance) and defined by this threshold. One way to create such system are heterogeneous decision trees [9] where, attributes are distances between each training vector and all other training vectors, so data analyzed by decision tree is a square matrix where number of columns and rows are equal to number of vectors.

The aim of this work is to present new model of threshold rules, creating ordered decision list, where individual prototypes rules may overlap. This may leads to reduction number of rules. Next section describes how P-rules supports different types of attributes, third section present threshold rules decision list algorithm, in forth section results of experiments are presented and last section derive perspectives on further improvements.

2. Distance functions and its possibilities

One of most important elements that are the base of P-rules methods are distance functions, or similarity functions but in this paper we were concentrating on utilization of distance functions. The reason of such choice is Heterogeneous Distance Functions (HDF) [16] which allows operating on all types of attributes.

In real word very often happen that datasets which have to be analyzed are made up of mixture of continues, discrete, nominal and symbolic features, and this becomes one of very important limitation for large group of data analyzing tools.

Usually most of algorithms require one type of data, and this force to use complicated preprocessing methods like discretization, or linearization algorithms which unnecessary increase model variance.

In SBL framework exists HDF which join two types of distance functions, typical and popular distance functions like Minkowski distance function (1) used for continues or ordered discrete values attributes and probability distance functions [1] described by equation (2) applied for symbolic features.

$$D_{Mink}(\mathbf{x},\mathbf{r})^{\alpha} = \sum_{i=1}^{n} |x_i - r_i|^{\alpha} \tag{1}$$

$$D_{VMD}(\mathbf{x},\mathbf{r})^{\alpha} = \sum_{i=1}^{n}\sum_{j=1}^{C} \left| p(c_j / x_i) - p(c_j / r_i) \right|^{\alpha} \tag{2}$$

where **x** and **r** are respectively data and reference vectors *n* is number of features, *C* is number of classes, and α – value of exponent . $p(c_j / x_i)$ and $p(c_j / r_i)$ are work out as

$$p(c_j \mid x_i) = \frac{Nx_{ij}}{Nx_i} \tag{3}$$

Where N_i is number of instances in training set that have value *x* for attribute *i,* N_{ij} is the some as N_i but for class *j*.

Both this types of distance functions are additive so the overall distance function can be calculated as sum of both this types of measures depending on attribute types (4)

$$D(\mathbf{x},\mathbf{r})^{\alpha} = D_{Mink}(\mathbf{x_a},\mathbf{r_a})^{\alpha} + D_{VDM}(\mathbf{x_b},\mathbf{r_b})^{\alpha} \tag{4}$$

where $\mathbf{x_a}$ and $\mathbf{r_a}$ are subsets of continues attributes of vector **x** and **r** and $\mathbf{x_b}$ and $\mathbf{r_b}$ are subsets of vector **x** and **r** of symbolic features. Unfortunately these advantages require future normalization, that each distance component has the some or comparable meaning.

Another advantage of P-rules is possibility to have influence on shape of decision border. In most of distance functions, specially this presented here exist α parameter which have significance influence on differences between different distance components. Changing α value from 1 to *inf.* different shapes of constant value lines are obtained, see fig 1. For α equal 1 rhomboidal shape is obtained, for $\alpha=2$ spherical and higher α lead to more rectangular shape, where for $\alpha=inf$ constant distance lines reach full square shape.

This feature of P-rules can be very helpful and allow generating crisp logical rules if it is necessary. Also fuzzy rules can be extracted from datasets and this problem was discussed in [7]. This advantage makes P-rules more general so different types of models can be written as P-rules concept.

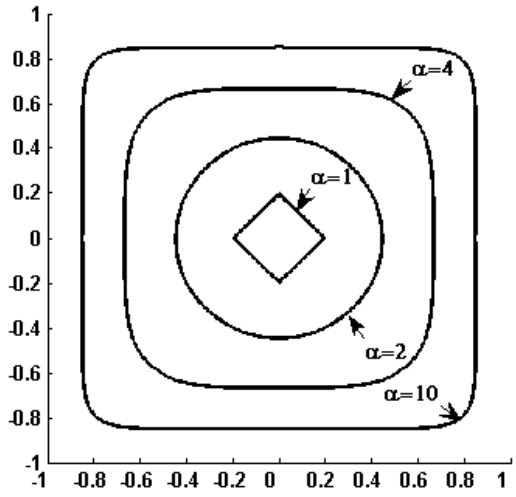

Fig. 1 Influence α parameter on shape of constant distance border

3. Threshold rules

Like it was introduced earlier threshold rules assign output class on the basis how fare is query vector from prototypes. The value of the threshold can be adopted, or it can be set to direct value, usually 0.5. The simplest example of threshold rules are radial basis neural networks where depending on the value of sum of neuron activations if it exceed threshold value the corresponding output of network is set to 1. This concept was presented in [4]. Another possible conception of threshold rules are heterogonous forests of decision trees [9]. The idea is to exchange or extend future set by the set of attributes arise as distance between each data vector and other training vectors and applies it to typical decision tree algorithm.

Presented in this paper approach combine heterogeneous decision trees with rules induction algorithms.

3.1. Threshold rules decision list algorithm

This algorithm is a simple method for prototype rule extraction; it searches for prototypes which can describe data assigning suitable threshold value for each of them. As the output algorithm creates list of decision rules from the most general to the most detailed. Because obtain rules may overlap this list of rules should be read in inverse order beginning from most detailed then if this rule don't fulfil the condition the next more general rule should be verified, at the end if non of the conditions was fulfilled the output class is assigned by the *else* condition as the opposite to this last most general rule fig. 2.

This algorithm is described just for two class problem C=[1 2] however it can be easily extended for more the two class by multiple starts for each class. The microcode of this threshold rules decision list (TRDL) method is presented on fig.3

TRDL creating rules use all training vectors as possible prototypes and for each of prototypes algorithm search for best threshold and highest criterion value. Then the prototype with highest criterion value is stored in memory with its threshold and corresponding class. In the next step all correctly classified vectors are removed form current training set, and the process of creating new rule is replayed with all misclassified vectors from prototype class and all vectors from remaining classes, however still whole dataset is analyzed as possible prototypes.

This condition ensures improvement of generated rule and allow for higher generalization. This is happening because optimal position of prototype doesn't have to be one of currently misclassified vectors; moreover usually it is one of other vectors from the some class.

Process of rule generation is stopped if any wrong classified vector does not exist or if exist just one misclassified instance. This second condition secure before unnecessary rule generation. However it still doesn't ensure any generalization, usually leading to data over fitting. The way to defeat this problem is cross validation.

Using 10 fold cross validation test, for each fold whole decision list is created. Finally on each level of the list appropriate condition (accuracy, balanced accuracy, etc.) is verified and at the end the optimal number of rules can be found as the one which maximize appropriate criterion.

One of the most important elements is the *CalcQuality* function. Results presented in the next section were obtained with the criterion of balanced accuracy. This function was used both, defining threshold, and determining how well is current rules set.

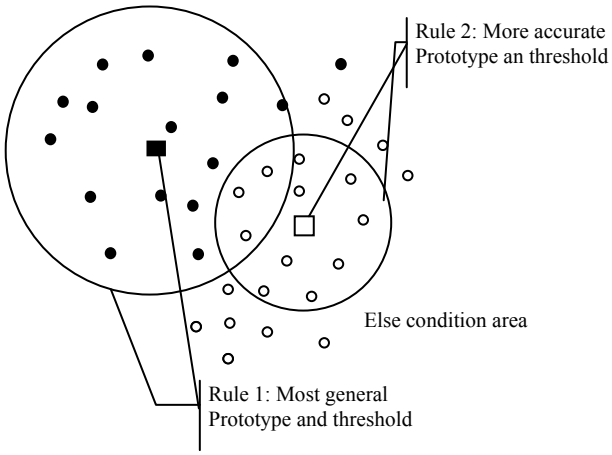

Fig 2. Example of threshold rules: Rule 1 – most general; Rule 2 – more accurate; and Else area

```
function [P,TH]=FindThresholdRules(T)
imput:
T – training set
output:
P – set of prototypes
TH – set of thresholds for appropriate prototypes
internal variables:
S – set of wrong classified vectors
p – index of training vector considered as possible prototype
D –distance matrix NxN where N-size(T)
begin
S=T;
D=dist(T,T);                    %calculate distance between each training vectors
i=0;
while false
  i=i+1;
  for C=0:1                     %for each class
    if size(S(C))==1,
          C=~C or breake;       %if  one misclassified vector of class C then change class or breake
    endif;
    for p=1:T(C)                %for each training vector form class C
        [THtmp,Qtmp]=CalcQuality( D(p),S(C),T(~C)); %for each prototype of class C calc Threshold and Quality of possible
                                %rule covering wrong classified vectors from C and not covering training vectors from
                                %remaining class
          if Qtmp>Quality
              TH(i)=THtmp;      %remember threshold value
              Quality=Qtmp;     %remember best quality value
              P(i)=T(p);        %remember current prototype
          endif
      endfor
  endfor;
  S=Classify(T,P,TH)            %classify training set using current set of rules, return all wrong classified vectors
  if isempty(S)
      return
  end;
endwhile;
end;
```

Fig. 3. Microcode of Threshold rules decision list algorithm

Most of real word datasets has mixed type of attributes, discrete, symbolic, nominal and continues and this is solved by heterogeneous distance functions. Unfortunately this determine that all of analyzed features, understand as distance between

prototype and other vectors, have positive defined continues values. This makes the problem of defining threshold value very important. So in this algorithm an important problem is computational cost. Reducing time is obtained by limitation number of possible tests which have to be taken to get best threshold value. As possible threshold are considered all pairs of neighbour vectors in the distance meaning which fulfil condition that closer to the prototype vector has the same class as prototype, and further, next vector is from remaining class. This condition allows finds optimal threshold for each considered prototype without losing any optimization abilities.

Like it was earlier written as criterion was used balanced accuracy, calculated by equation (5)

$$cv = \frac{N_{C+}}{N_C} + \frac{N_{\overline{C}-}}{N_{\overline{C}}} \tag{5}$$

where N_{C+} are all vectors from class C that falls into the hypersphere defined by the rule, N_C are all vectors with label C, and $N_{\overline{C}}$, $N_{\overline{C}-}$ are number of vectors form remaining class, and all vectors form remaining class which don't fulfil condition defined by the rule respectively.

4. Real datasets experiments

To compare results obtained with TRDL with other well known and popular methods, Weka software was used with four different and popular classification algorithms. Two of them were methods allowing for rule extraction: C4.5 decision tree and Decision Table algorithm. And other two methods are popular and parameter free classification algorithms: nearest neighbour and Naïve Bayes algorithm. Also to compare obtained results with neuro fuzzy rule extraction system NefClass system was used **Błąd! Nie można odnaleźć źródła odwołania.**.

For tests six different datasets was used each as two class problem.

4.1. Datasets

First of used datasets was *Appendicitis* which has 7 attributes and 106 different cases 85 from class one and 21 from class two, each describing one person with appendicitis. From this dataset 2 most relevant features was selected using SSV tree and for this two features all test was performed.

Another dataset was *Cleveland Heart Disease* with five continues attributes and eight discrete. This dataset has 303 vectors describing healthy and sick persons.

Ionosphere is a set of two different types of radar signals reflected from ionosphere; this dataset has 351 vectors with 34 attributes,

Lancet dataset …..????

For tests also *Pima Indians Diabetes* dataset was used. This dataset characterize Indians seek for diabetes with 768 vectors described by eight features where 500 cases of healthy and 268 cases of unhealthy people.

The last dataset was set of instances of breast cancer from Wisconsin hospital (*Wisconsin Breast Cancer*) where two classes are: malignant tumour 241 cases (34,5%), and 458 (65,5%) cases of benign tumour. Each vector has 9 discrete features.

4.2. Classification results

To increase credibility of our tests 10 fold cross validation test was performed, and results are presented in table 1 as means of accuracy obtained in each fold. Marked as bold are best results obtained for each dataset.

As it can be seen in Tab 1, presented here TRDL algorithm usually was in the top group of compared algorithms, and even ones lead. But what is also important the mean of obtain results proof that presented algorithm use to lead in general having highest mean accuracy.

Table 1. Classification results

10 x CV	1NN	C4,5	Naive Bayes	Decision Table	NefClass	Treshold Rules	Num of Proto,
Appendictis	81	88	**88,82**	88,09	87,73	88,12	4
Cleveland heart-disease	77,1	77,76	**83,46**	82,13	82,82	82,21	1
Ionosphere	86,33	**91,46**	82,62	89,18	81,78	89,7	12
Lancet	94,23	**95,23**	93,78	94,23	90,18	94,23	6
Pima indians-diabetes	70,17	73,83	**76,31**	73,31	73,83	70,84	2
Wisconsin breast-cancer	96,16	96,17	96,31	95,43	94,86	**97,78**	1
mean	84,17	87,08	86,88	87,06	85,2	**87,15**	

5. Conclusions and future works

Presented in this paper rule extraction algorithm is a simple method leading to analyze data with small number of rules. Obtain results on real word datasets present their high abilities especially in comparison with alternative F-rules generation algorithms.

Advantages of presented in threshold rules decision list can be described as:

- flexibility

- support all type of attributes
- allow to obtain different types of rules, depending on constructor requirements: prototype, crisp, or F-rules [7]
- changing distance function may lead to better data adjustment, the same improving generalization
- small number of accurate rules

All this benefits make this algorithm very interesting and promising tool for data analyze.

However there are lots of improvements which can be done. One way to increase generalization abilities and reduce number of errors is applying feature selection algorithms which can be build into the algorithm. Because in the distinction to kNN algorithm this type of P-rules don't require common feature space, so each rule may operate on different subset of attributes. Especially in all distance methods feature selection may have important influence on obtained results. Another interesting expand of TRDL algorithm are different criterion algorithms.

Presented in this paper criterion is very simple and allow just for local judgement of classification abilities, much better should be all methods driven form decision trees. In this field lot of different criterion was discovered and without any important modifications can be used in this TRDL method. These groups of criterions are coefficients based on information entropy like InformationGain form ID3 tree or InformationGainRation form C4.5 [13] and SSV [8] criterion which directly optimize sensitivity and specificity value.

Unfortunately this algorithm has some limitations, for example undesirable is computational cost. In presented here algorithm the criterion function is called for every pair of points from different classes, and for each training vector taken as prototype, on each level of new rule extraction. This makes this algorithm rather slow, and difficult or even impossible to use on large datasets with high number of vectors. Another problem is distance calculation which also has to be evaluated many times, however for small datasets this shortcoming can be easily removed by storing in memory whole distance matrix.

All discussed in this section improvements may further increase TRDL abilities, and presented here solutions are just the first step in developing this method.

References

[1] E. Blanzieri, F. Ricci "Probability Based Metrics for Nearest Neighbor Classification and Case Based Reasoning", Proc. 3rd Int. Conf. on Case-Based Reasoning, Munich, August 1999

[2] L. Breiman, J.H. Friedman, R.A. Oslhen, and C.J. Stone, Classification and Regression Trees. Belmont, CA: Wadsworth International Group, 1984

[3] R.O. Duda, P.E. Hart, D.G. Stork, Pattern Classification, New York: John Wiley & Sons, 2nd ed, 2001.

[4] W. Duch, Similarity based methods: a general framework for classification, approximation and association. Control and Cybernetics 29 (4), pp. 937-968, 2000.

[5] W. Duch, R. Setiono, J.M. Zurada, Computational intelligence methods for understanding of data. Proc. of the IEEE ,Vol. 92, No. 5, 771- 805, 2004

[6] W. Duch, K. Grudziński "Prototype based rules - a new way to understand the data". Proc. of IJCNN 2001, Washington D.C. USA, pp. 1858-1863.

[7] W. Duch M. Blachnik "Fuzzy rule-based system derived from similarity to prototypes", Springer Lecture Notes in Computer Science, vol. 3316, pp 912-917, 2004.

[8] K. Grąbczewski, W. Duch "The separability of split value criterion" 5'th Conference Neural Network and Soft Computing, Zakopane, Poland 2000

[9] K. Grąbczewski and W. Duch, "Heterogenous forests of decision trees". Springer Lecture Notes in Comp. Sci. vol. 2415, pp. 504-509, 2002.

[10] C.J. Mertz and P.M. Murphy, UCI repository of machine learning databases, www.ics.uci.edu/pub/machine-learning-databases.

[11] W. Pedrycz, Fuzzy set technology in knowledge discovery, Fuzzy Sets and Systems 98 (1998) 279-290

[12] A. Piegat "Modelowanie I sterowanie rozmyte" AOW Exit, Warszawa 2003

[13] J.R. Quinlan, C4.5: Programs for machine learning. San Mateo, Morgan Kaufman 1993

[14] A.J. Walker, S.S. Cross, R.F. Harrison, Visualization of biomedical datasets by use of growing cell structure networks: a novel diagnostic classification technique. Lancet Vol. 354, pp. 1518-1522, 1999.

[15] A. Webb, Statistical Pattern Recognition. John Wiley and Sons 2002.

[16] D.R. Wilson, T.R. Martinez, "Improved Heterogeneous Distance Functions", J. of Artificial Intelligence Research 6, pp. 1-34, 1997.

[17] D. Nauck , F. Klawonn , R. Kruse, Foundations of Neuro-Fuzzy Systems, John Wiley & Sons, New York, NY, 1997