A Simple Fuzzy Classifier based on Inconsistency Analysis of Labeled Data

János Abonyi, Hans Roubos*
University of Veszprem, Department of Process Engineering
P.O. Box 158, H-8201 Veszprem, Hungary
Phone: +36-88-422-022.4209, Fax: ++36-88-422-022.4171
email:abonyij@fmt.vein.hu, web: www.fmt.vein.hu/softcomp
* Delft University of Technology,
Faculty of Information Technology and Systems, Control Engineering Laboratory
P.O. Box 5031, 2600 GA Delft, the Netherlands
Phone: +31-15-278.3371, Fax: +31-15-278.6679
email: hans@ieee.org

ABSTRACT: An extremely simple fuzzy classifier is identified based on the inconsistency analysis of labelled training data. The method was applied to the COIL challenge 2000 *Direct Mail* problem and resulted in 121 selected caravan policies within the first 800 selected customers. As this result is identical to the result of the winner of the competition, the presented method is an example for how the *try the simplest first* approach can be effective in real-life problems.

KEYWORDS: inconsistency analysis, COIL challenge 2000, fuzzy classifier, model initialisation, direct mail

INRODUCTION

Data mining and knowledge discovery have become very important in our society where the amount of data doubles almost every year. Special data-warehousing companies are founded and today's Internet contains terra-bytes of data. However, the information content is usually low and difficult to extract, e.g. search engines have major problem in finding the right information. Similar problems are found in the *Direct Mail* problems, where a company selects potential customers out of their database to send directed mail in order to have a high response rate. This year, the COIL-challenge is a direct-mailing problem subject to the selection of potential customers of caravan policy based on both sociogeographic and personalised data.

We threat the COIL problem as a classification problem and develop compact and accurate fuzzy classifiers from the labelled observation data. Fuzzy classifiers are idealy suited to provide interpretable solutions to users, since it handles imprecise data and the resulting rules are interpretable, i.e. the semantic structure provides insight into the classifier structure and decision making process. In recent years, many researchers developed algorithms for designing fuzzy systems for prediction, control, and pattern recognition. Most of these data-driven algorithms, however, are designed for accuracy and often result in complex non-interpretable rule bases.

We describe an algorithm for obtaining accurate but also interpretable fuzzy rule-based classifiers from labelled observation data. In the first step, the structure of the model is initialised based on the statistical analysis of the labelled data and straightforward data-mining tools like feature selection methods. After the feature selection step, the algorithm transforms the inconsistency analysis of the features into fuzzy sets. Finally, the classifier is optimised for accuracy by adapting the parameters of the resulting model.

THE COIL CHALLENGE PROBLEM

The development of the proposed inconsistency based method was motivated by the COIL challenge problem. The task of the competition was the prediction of which customers are potentially interested in a caravan insurance policy based on both sociogeograhic and personalised data. Moreover, the actual or potential customers had to be described. This is a typical data-mining problem; if the company has a better understanding of their potential customers then they are able to gain better profit out of their direct mailing campaigns. A data-based decision support system will help to reduce some of the expenses of the marketing project by using a model of their possible customers. For model building, data of 4000 customers was available. As this data was highly inconsistent, standard data-mining tools like decision-trees showed poor performance on this problem.

FUZZY CLASSIFIER STRUCTURE

We apply fuzzy classification rules that describe each a certain class of customers. The fuzzy classifier has unity rule-consequences, because each rule represents a special class (group) of the data. The degree of fulfilment of each rule relates to the truth-value, i.e. the membership grade of a pattern to the rule's class:

$$R_i: \text{ If } x_1 \text{ is } A_{i1} \text{ and } \ldots\ x_n \text{ is } A_{in} \text{ then } Class_i \tag{1}$$

where A_{ij} denotes the antecedent fuzzy set defined for the $j=1,\ldots,n$ -th feature in the $i=1,\ldots,M$ -th rule. The classifier output is calculated based on the degree of activation of the rules:

$$\beta_i(\mathbf{x})=\prod_{j=1}^{n} A_{ij}\left(x_j\right). \tag{2}$$

A crisp decision is made by taking the class belonging to the fuzzy-rule with the maximum degree of activation

$$y = Class(\max(\boldsymbol{\beta}(\mathbf{x}))) \tag{3}$$

where $\boldsymbol{\beta}(\mathbf{x})=\left[\beta_1(\mathbf{x}),\ldots,\beta_M(\mathbf{x})\right]$ is the vector of normalised degree of activation.
The certainty degree of this decision is $CF=\max(\boldsymbol{\beta}(\mathbf{x}))$. As a direct mailing problem means the choice of a predefined number of costumes that could be the member of a given class, the customers are shorted based on the CF, and the first predefined number of customer that are the member of the interested class is chosen.

MODEL INITIALIZATION

The inconsistency of the training set motivated the development of the model initialisation method proposed in this section.

INCONSISTENCY COUNT AND FEATURE SELECTION

The first step of the modelling procedure is the selection of a set of relevant features based on a feature ranking. Reducing the feature dimensionality initially improves the generalisation ability of the model. However, when a particular reduction of the pattern dimensionality is reached, the modelling performance highly degrades. Several alternative models differing in the amount of features are proposed based on this ranking. A model with a minimal amount of features but enough information to handle the classification problem is selected.

The training data consists of labelled classes y having a pattern $\mathbf{x}$ constituted with n features from a subset $X_{feature}$ of the original features. Two cases $\left(\mathbf{x}^i, y^i\right)$ and $\left(\mathbf{x}^k, y^k\right)$ are inconsistent if both have the same patterns $\mathbf{x}^i=\mathbf{x}^k$ but different associated classes $y^i \neq y^k$. The *inconsistency count* I_i of a given set of the same patterns $\mathbf{x}^i$ can be defined as number $n_{inc,i}$ of all inconsistent cases for matching pattern minus the largest number of cases in one of the classes from this set of inconsistent cases.
The inconsistency criteria defined on the reduced data set $T_{X_{feature}}$ is calculated as $J_{inc}\left(T_{X_{feature}}\right)=\sum I_i$.
This criterion is an open-loop feature selection that is independent from the applied model structure because it is based only on the labelled data. Moreover, it satisfies the monotonicty property, $J_{inc}\left(T^{+}_{X_{feature}}\right)\leq J_{inc}\left(T_{X_{feature}}\right)$, where $T^{+}_{X_{feature}}$ denotes a lager subset than $T_{X_{feature}}$.

INITIALISATION OF THE FUZZY MEMBERSHIP FUNCTIONS

In data-based fuzzy modelling, the fuzzy sets are often obtained by a clustering algorithm that obtains multivariable membership functions defined in the product space of the features. This is done by identification of fuzzy regions where the system can be approximated by single fuzzy rules. Univariate membership functions are then generated by projecting these multivariable membership functions onto the input variables and subsequently the projections are approximated by parametric functions. The algorithm proposed in this paper changes the order of the membership calculation and the projection. This means, firstly the data is projected onto the features and after this projection step the univariate membership functions are calculated.
This results in the following initialisation of the membership functions:

$$A_{ij}\left(x_j\right)=\frac{n_i\left(x_j\right)}{n\left(x_j\right)} \tag{4}$$

where $n_i\left(x_j\right)$ denotes the number of data that is in the class j, while $n\left(x_j\right)$ represents the number of all data that has property x_j.

APPLICATION TO THE COIL PROBLEM

THE INITIAL MODEL

The application of the previously proposed feature selection algorithm resulted in a selection of 5 variables:

feature 5: ***Customer main type***
***feature* 16:** **High level education**
***feature* 47:** **Purchasing power class**
***feature* 51:** **Contribution to car policies**
***feature* 59:** **Contribution to fire policies**

When the membership functions defined on these features as shown in Figure 1 are independently applied, approximately 10-12% hitrate is achieved. With the combination of these features, this result is further improved to 15.89% hitrate on the whole training set and 14.62% (selection of 117 caravan owners) on the previously unknown validation set.

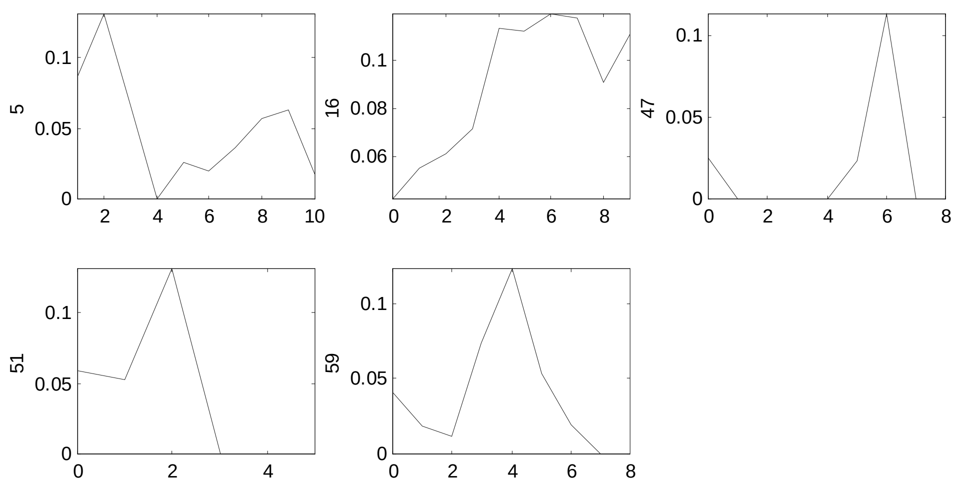

Figure 1: The frequency distribution on the selected features

As at the time of writing of the report the validation set is known, it has been turned out that the usage of ***Customer main type*** does not improve the classification performance. Without this feature, the application of the model results in 121 selected caravan policies within the first 800 selected customers. As this result is identical to the result of the winner of the competition, the presented method is an example for how the *try the simplest first* approach can be effective in real-life problems. Table I. summarises the performances when different features are used.

Table I: Performance of the fuzzy classifier by using different features

Features	*train set (tot=348)*	*test set (tot=238)*
16, 47, 51, 59	**194**	**121**
16, 47, 59	189	120
47, 59	190	113
all (1-85)	190	102

MODEL SIMPLIFICATION

The product of the membership functions is used in the fuzzy clasifier which allows us to normalise these functions in order to have normal fuzzy sets

$$A_{ij}^{*}\left(x_j\right)=\frac{A_{ij}\left(x_j\right)}{\max\limits_{k}\max\limits_{x_j} A_{kj}\left(x_j\right)} \qquad (5)$$

To improve interpretability and simplify the decision system, the membership functions are approximated as piecewise linear fuzzy sets that can be decomposed into triangular and trapezoidal membership functions (Figure 2.). This simplified and highly interpretable model results in 14.37% hitrate (selection of 115 owners) on the validation set.

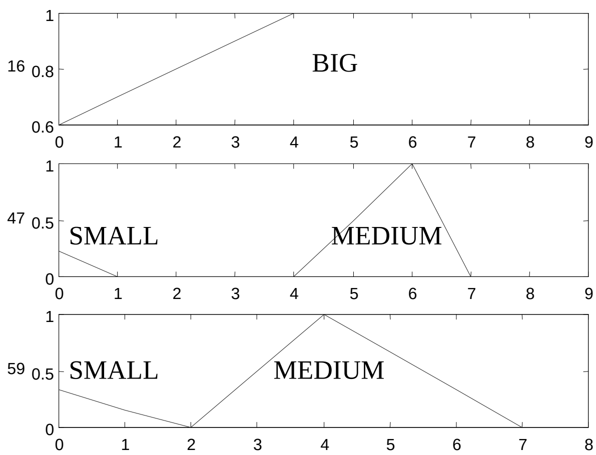

Figure 2: Shape of the extracted fuzzy sets

This model is easily interpretable as the rule of the model is:

If

High level education **is** *BIG* **and**
Purchasing power class **is (***MEDIUM* **or** *SMALL***) and**
Contribution to fire policies **is (***MEDIUM* **or** *SMALL***)**

Then OWNER

WHAT WAS OUR MISTAKE ?

The proposed simple fuzzy model initialisation technique resulted in the same performance as the solution of the winner of the COIL competition. Unfortunately, as most of the participants due to the competitive atmosphere, we also overtrained the previously presented initial model. To improve the performance of the classification system, we applied the previously presented inconsistency based approach to obtain multivariable fuzzy sets. By using such multidimensionality sets, the information loss that results form the projection step is avoided. Based on some heuristic search and optimisation, we obtained a more complex rule-base defined on the product-space of the above mentioned variables. The rules had the following form:

> IF **Purchasing power class is between 4 and 8** and **Contribution to car policies=4** and **Contribution to fire policies=4** then **the response rate is approximately 30%**

We were quite optimistic with this model, because the high response rate of the resulted classification model which is shown in Figure 3. On the training-set this model gives approximately 25% hitrate. Unfortunately, this complex multidimensional model showed poor performance on the unknown validation set (10% hitrate). The reason of this large deviation is that it was possible to pick up a lot of owners that had small inconsistency but they number was also small. Hence, the probability of the relative number of owners differs in the validation and training data was big. We took this risk and we have failed.

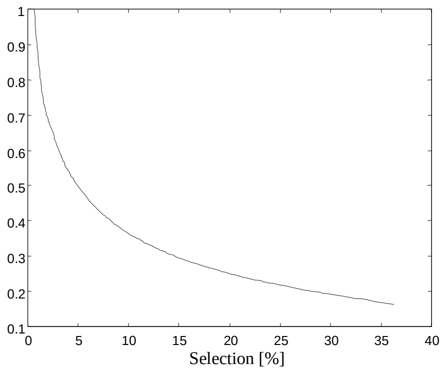

Figure 3. Response rate of the classification model

COLCLUSIONS

We presented an extremely simple fuzzy classifier identified based on inconsistency analysis of labelled training data.
The first step of the modelling procedure was the selection of a set of relevant features based on the ranking of the features according to the inconsistency of the training data. A new membership function initialisation procedure was presented, where the data is projected onto the features and after this projection step the univariate membership functions are calculated based on the inconsistency of the projected data.
The method has been applied to the COIL challenge 2000 problem and resulted in 121 selected caravan policies within the first 800 selected customers. As this result is identical to the result of the winner of the competition, the presented method is an example for how the *try the simplest first* approach can be effective in real-life problems.

IMPLEMENTATION

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% "Fuzzy" model based on freq. analysis
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%______________________________________
%Load the data
clear all
close all
rand('state',0);                     %Training data
train=load('ticdata2000.txt');
out=train(:,end);
N=length(train);
test=load('ticeval2000.txt');     %Test data
tout=load('tictgts2000.txt');
Nt=length(test);
%Features
feat=[16 47 51 59];
nf  =length(feat);
train=[train(:,feat)];
test=[test(:,feat)];
%Model structure
Model=zeros(nf,45);
score=ones(size(out));
tscore=ones(size(tout));
%_______________________________
%Main
for fi=1:nf
   %Calculation of the freq.
    val=min(train(:,fi)):max(train(:,fi));
    nv=length(val);
    dim=zeros(nv,1);
   for i=1:nv
      index=find(train(:,fi)==val(i));
      if isempty(index)
         dim(i)=0;
       else
          dim(i)=sum(out(index))/size(index,1);
       end
    end
    dim=dim/(max(dim)); %normalization of the mem. functions
    %application of the model
       for i=1:nv
           index=find(train(:,fi)==val(i));
           score(index)=score(index)*dim(i);
           index=find(test(:,fi)==val(i));
           tscore(index)=tscore(index)*dim(i);
       end
    %save the model parameters
        Model(fi,1)=min(val);
        Model(fi,2)=max(val);
        for i=1:length(dim)
            Model(fi,i+2)=dim(i);
        end
        subplot(3,ceil(nf/3),fi);
        plot(val,dim);
        ylabel(feat(fi));
        axis([min(train(:,fi)) max(train(:,fi)) min(dim) max(dim)])
end

[c, inds]=sort(-score); %-
Nc=round(length(out)*0.2);
inds=inds(1:Nc);
err=sum(out(inds))/Nc*100
sel=sum(out(inds))

[c, inds]=sort(-tscore); %-
inds=inds(1:800);
terr=sum(tout(inds))/800*100
tsel=sum(tout(inds))
```