A Neuro-Fuzzy Approach in the Prediction of Financial Stability and Distress Periods

Giovanis, eleftheios

10 August 2008

A Neuro-Fuzzy Approach in the Prediction of Financial Stability and Distress Periods

Eleftherios Giovanis

Abstract

The purpose of this paper is to present a neuro-fuzzy approach of financial distress pre-warning model appropriate for risk supervisors, investors and policy makers. We examine a sample of the financial institutions and electronic companies of Taiwan Security Exchange (TSE) from 2002 through 2008. We present an adaptive neuro-fuzzy system with triangle and Gaussian membership functions. We conclude that neuro-fuzzy model presents almost perfect forecasts for financial distress periods as also very high forecasting performance for financial stability periods, indicating that ANFIS technology is more appropriate for financial credit risk control and management and for the forecasting of bankruptcy and distress periods. On the other hand we propose the use of both models, because with Logit and generally with discrete choice models we can examine and investigate the effects of the inputs or the independent variables, while we can simultaneously use ANFIS for forecasting purposes. The wise and the most scientific option are to combine both models and not taking only one of them.

Keywords: Financial distress; ANFIS; Neuro-Fuzzy; Fuzzy rules; Fuzzy membership functions; triangle; Gaussian; MALTAB

1. Introduction

Previous studies used various approaches for the financial and bankruptcy modeling formulation. Platt and Platt (2002), and Cheng *et al*. (2006) used a Logit model to analyze pre-warning model and to a build financial distress model, while Zhang *et al*. (1998), and O'leary (1998) used artificial neural networks. Their findings support the superiority of artificial intelligence approaches. A significant study was made by Cheng *et al*. (2006). The authors study a pre-warning financial distress model for the TSE listed companies and they apply a binary logit and a fuzzy regression model with triangular membership function. Their results support fuzzy regression, where the correctly predicted percentage of fuzzy regression is 90.98 percent versus Logit regression which predicts correctly the 90.30 percent. In this case we present only the results of neuro-fuzzy approach, as we get similar forecast with Logit and Probit regressions. Furthermore, we show that neuro-fuzzy forecasts are significant superior to the findings of Cheng *et al*. (2006) who predict correctly at 90.98 using fuzzy regression, indicating that the combination of neural networks and fuzzy logic can be a superior approach.

2. Methodology

In this section we present the variables which are used in the analysis and a short description and definition of them (Cheng *et al.,* 2006). In table 1 we present some variables which canbe obtained in order to build a neuro-fuzzy system. The dependent dummy binary variable expresses the financial stage, where takes the value 1 if the specific company in the certain time period is on financial distress and value 0 if is characterized by financial stability.

Table 1. Financial statement ratios

Category	Financial variables	Definition of financial variables	Symbol
Financial structure	Shareholders' equity to total assets ratio (%)	Total shareholders' equity/total assets	x_1
	Debt to total assets ratio (%)	Total liabilities/total assets	x_2
	Permanent capital to fixed assets ratio (%)	(shareholders' equity + long debt)/ fixed assets	x_3
Liquidity	Current assets (%)	Current assets/current liabilities	x_4
Cash	Cash flow ratio (%)	Net cash flow from operation/ current liabilities	x_5
Asset utilization	Accounts receivable turnover	Sales/average accounts receivable	x_6
	Fixed asset turnover	Sales/average fixed assets	x_7
	Total asset turnover	Sales/average total assets	x_8
Profitability	Returns on assets (%)	Net income + interest expense (1 – tax rate)/ average total assets	x_9
	Return on common equity (%)	Net income/average shareholders' equity	x_{10}
	Pre-tax profit to capital (%)	Pre-tax income/capital	x_{11}
	Earnings per share	(after-tax income – preferred dividends)/the weight numbers of stock	x_{12}

The inputs we take in our fuzzy system are the same with those used in the tudy of Chen *et al.* (2006), which are the variables x_5 and x_9 from table 1. More inputs can be obtained, or more linguistic terms, but the results are not changed significant, as also the computation time is reduced in a great degree.

Jang (1993) and Jang and Sun (1995) introduced the adaptive network-based fuzzy inference system (ANFIS). We incorporate three linguistic terms {low,medum,high}. More linguistic terms can be introduced, as very low and very high, but the forecasting performance is almost the same, indicating that we can simplify the procedure by taking less linguistic terms and less rules. The rules are 9 because we have two inputs with three linguistic terms and it is 3*3=9. These rules are

IF returns are low OR cash flow is low THEN $f_1=p_1x_1 + q_1x_2 + r_1$

IF returns are low OR cash flow is medium THEN $f_2=p_2x_1 + q_2x_2 + r_2$

IF returns are low OR cash flow is high THEN $f_3=p_3x_1 + q_3x_2 + r_3$

IF GDP is medium OR cash flow is low THEN $f_4=p_4x_1 + q_4x_2 + r_4$

IF returns are medium OR cash flow is medium THEN $f_5=p_5x_1 + q_5x_2 + r_5$

IF returns are medium OR cash flow is high THEN $f_6=p_6x_1 + q_6x_2 + r_6$

IF returns are high OR cash flow is low THEN $f_7=p_7x_1 + q_7x_2 + r_7$

IF returns are high OR cash flow is medium THEN $f_8=p_8x_1 + q_8x_2 + r_8$

IF returns are high OR cash flow rate *is high THEN* $f_9=p_9x_1 + q_9x_2 + r_9$

, where returns denotes the returns on assets. Basically, there are two types of fuzzy set operation that are usually used in the antecedent rule, which are *AND* and *OR*. Mathematically, the *AND* operator can be realized using *Min* or *Product* operation while *OR* can be realized using *Max* or *Algebraic* sum operator. Also there is a confusion here as many scientists use *probabilistic* sum than *algebraic* sum. We choose the *OR* operator, because we assume that a financial distress might take place, if one of the to inputs activate the firing strength and is not nessecary that for example we need both low cash flow and returns on assets in order for a distress period to take place. We take the *Max* operator. Because we have nine rules and two inputs in the case we examine the steps for ANFIS system computation are:

In the first layer we generate the membership grades

$$O_i^1 = \mu_{A_i}(x_1),\ \mu_{B_i}(x_2) \qquad (1)$$

, where x_1 and x_2 are the inputs. In layer 2 we generate the firing strengths or weights

$$O_i^2 = w_i = \prod_{j=1}^{m} (\mu_{A_i}(x_1), \mu_{B_i}(x_2)) = \\ ORmethod\,(\mu_{A_i}(x_1), \mu_{B_i}(x_2)) \\ = \max(\mu_{A_i}(x_1), \mu_{B_i}(x_2)) \tag{2}$$

In layer 3 we normalize the firing strengths. Because we have nine rules will be:

$$O_i^3 = \overline{w_i} = \frac{w_i}{w_1 + w_2 + \ldots + w_8 + w_9} \tag{3}$$

In layer 4 we **c**alculate rule outputs based on the consequent parameters.

$$O_i^4 = y_i = \overline{w_i} f = \overline{w_i}(p_i x_1 + q_i x_2 + r_i) \tag{4}$$

In layer 5 we take the sum all the inputs from layer 4

$$O_i^5 = \sum_i y_i = \sum_i \overline{w_i} f = \overline{w_i}(p_i x_1 + q_i x_2 + r_i) \tag{5}$$

In the last layer the consequent parameters can be solved for using a least square algorithm as:

$$Y = X \cdot \theta \tag{6}$$

, where X is the matrix

$$X = [w_1 x + w_1 + w_2 x + w_2 + \ldots + w_9 x + w_9] \tag{7}$$

, where x is the matrix of inputs and θ is a vector of unknown parameters as:

$$\theta = [p_1, q_1, r_1, p_2, q_2, r_2, \ldots, p_9, q_9, r_9]^T \tag{8}$$

, where T indicates the transpose. For the first layer and relation (1) we use the triangular membership function. The symmetrical triangular function is defined as:

$$\mu_{ij}(x_j; a_{ij}, b_{ij}) = \begin{Bmatrix} 1 - \dfrac{|x_j - a_{ij}|}{b_{ij}/2}, if\, |x_j - a_{ij}| \le \dfrac{b_{ij}}{2} \\ 0, otherwise \end{Bmatrix} \tag{9}$$

The symmetrical Gaussian membership function is defined as:

$$\mu_{ij}(x_j; c_{ij}, \sigma_{ij}) = \exp\left(-\frac{(x_j - c_{ij})^2}{2\sigma^2_{ij}}\right) \quad (10)$$

, where c_{ij} is the center parameter and σ_{ij} is the spread parameter. In order to find the optimized antecedent parameters we use backpropagation algorithm.

The process for the initial values is as follows. For example for cash flow we take three samples. The first one accounts for values between the minimum and the mean value of cash flow for linguistic term *low*. For the term *medium* we take the sample for values between the mean and the third quartile. Finally for the last linguistic term *high* we take the sample for values ranging between the third quartile and the maximum value. From these samples we take the mean and standard deviation corresponding to center and base parameters respectively. The learning rates for triangle function have been set up at 0.8 for all parameters and the number of maximum epochs at 20.

3. Data

We use data from a sample of electronic companies and financial institutions listed in TSE Securities and Futures Institute Network from 2002 through 2008. We should mention that we obtained a sample of these companies and not all of them. Specifically our estimation sample is constituted by 179 companies. Also when we refer to financial institutions we mean all companies as banks, financial services, insurance companies, brokerage and others. We use the period 2002-2006 as the in-sample period or training data period and the period 2007-2008 is used for predictions in out-of sample period, or the testing data period, which we are mainly interesting about.

4. Empirical results

In tables 2 and 3 the forecasts in the training data period are reported and ANFIS with triangle membership function has a slightly higher overall percentage of 96.92 in relation with 92.90 per cent of Gaussian. On the other hand, with triangle function we predict 96.29 per cent the financial distress periods in the out-of-sample period as also we predict at 98.84 per cent correct the financial stability period. With Gaussian function we predict correct at 94.44 and 91.89 the financial distress and stability periods respectively. Furthermore, our findings are significant superior to those of Cheng *et al.* (2006) as we mentioned above suggesting that neuro-fuzzy can be a much more powerful forecasting tool than the simple fuzzy logic or fuzzy regressions or neural networks. The combination of both fuzzy logic and neural networks can have excellent results and very high forecasting power.

Table 2. Prediction results of ANFIS with triangle function for in-of sample period

Actual	Prediction		
	Financial distress	Financial stability	Correctly percentage rate
Financial distress	192	2	98.96
Financial stability	24	628	96.31
Overall percentage			96.92

Table 3, Prediction results of ANFIS with Gaussian function for in-of sample period

Actual	Prediction		
	Financial distress	Financial stability	Correctly percentage rate
Financial distress	191	3	98.45
Financial stability	57	595	91.25
Overall percentage			92.90

Table 4. Prediction results of ANFIS with triangle function for out-of sample period

Actual	Prediction		
	Financial distress	Financial stability	Correctly percentage rate
Financial distress	52	2	96.29
Financial stability	3	256	98.84
Overall percentage			98.40

Table 5. Prediction results of ANFIS with Gaussian function for out-of sample period

Actual	Prediction		
	Financial distress	Financial stability	Correctly percentage rate
Financial distress	51	3	94.44
Financial stability	21	238	91.89
Overall percentage			92.33

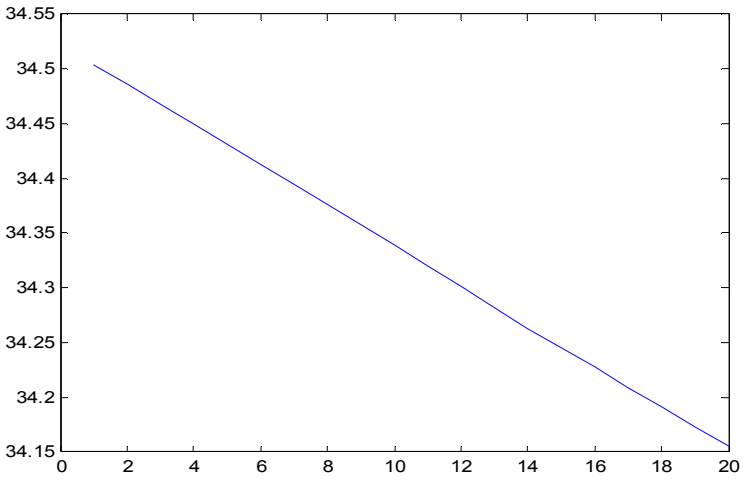

Figure 1. Example of error reduction through the training process with ANFIS and Gaussian membership function after 20 epochs

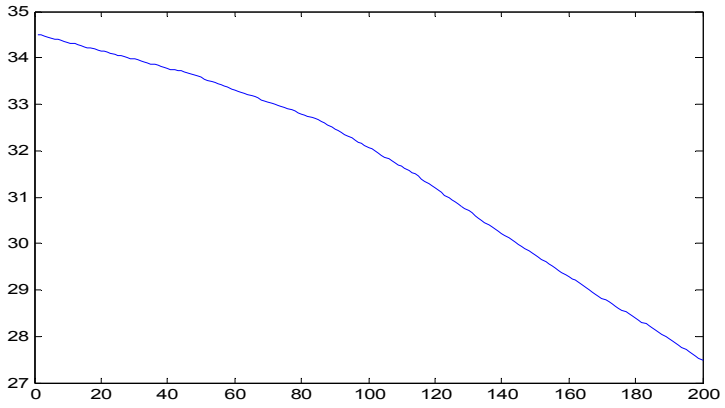

Figure 2. Example of error reduction through the training process with ANFIS and Gaussian membership function after 200 epochs

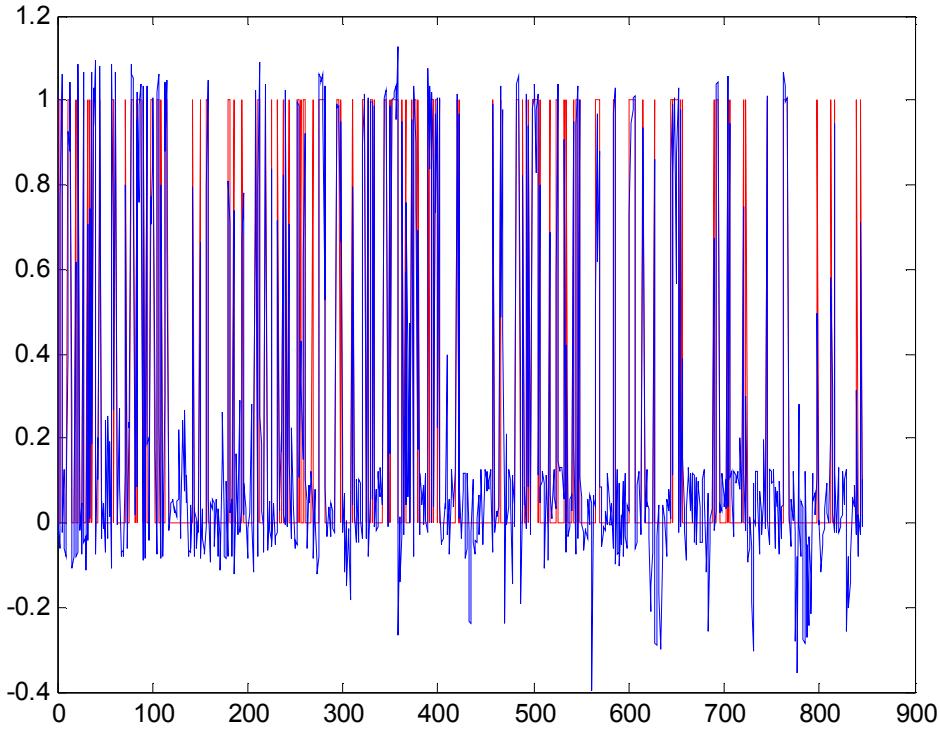

Figure 3. In-sample forecasts with ANFIS and triangle membership function

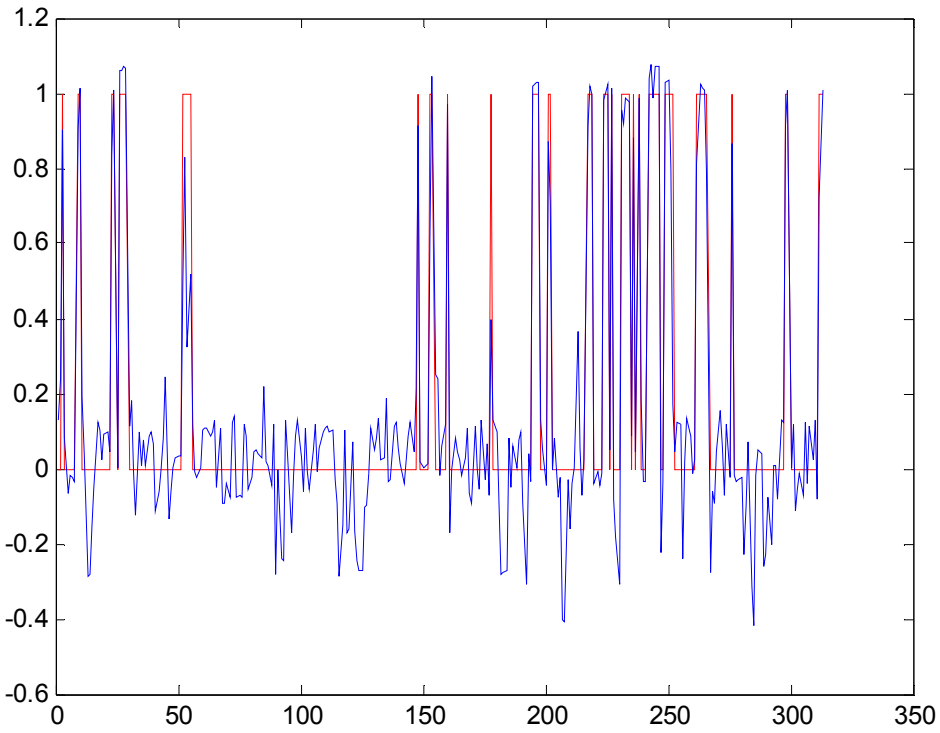

Figure 4. Out-of-sample forecasts with ANFIS and triangle membership function

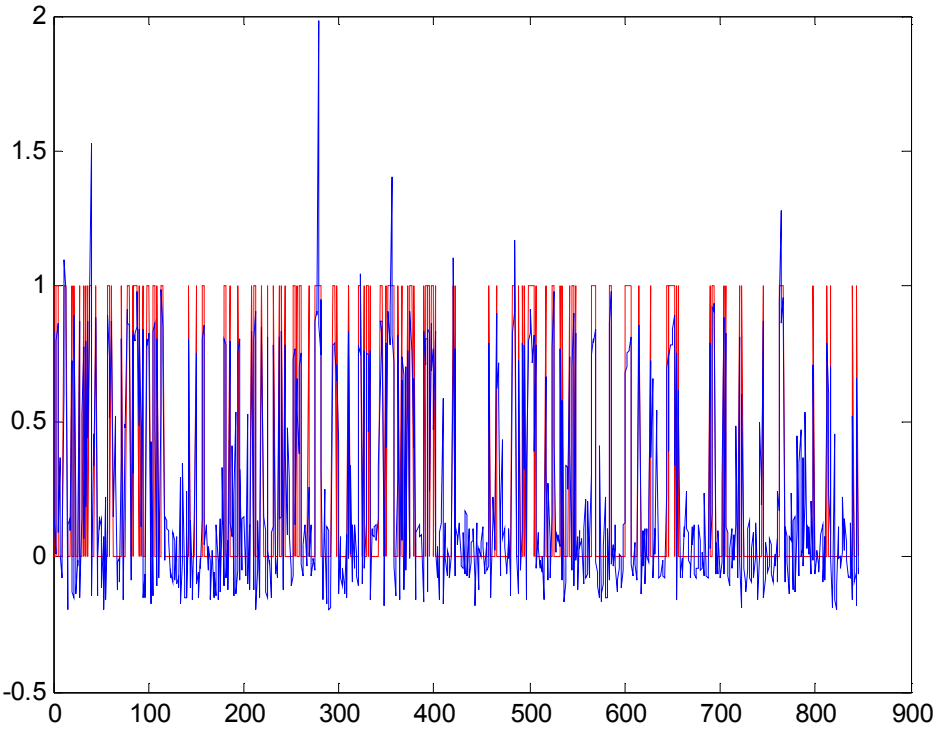

Figure 5. In-sample forecasts with ANFIS and Gaussian membership function

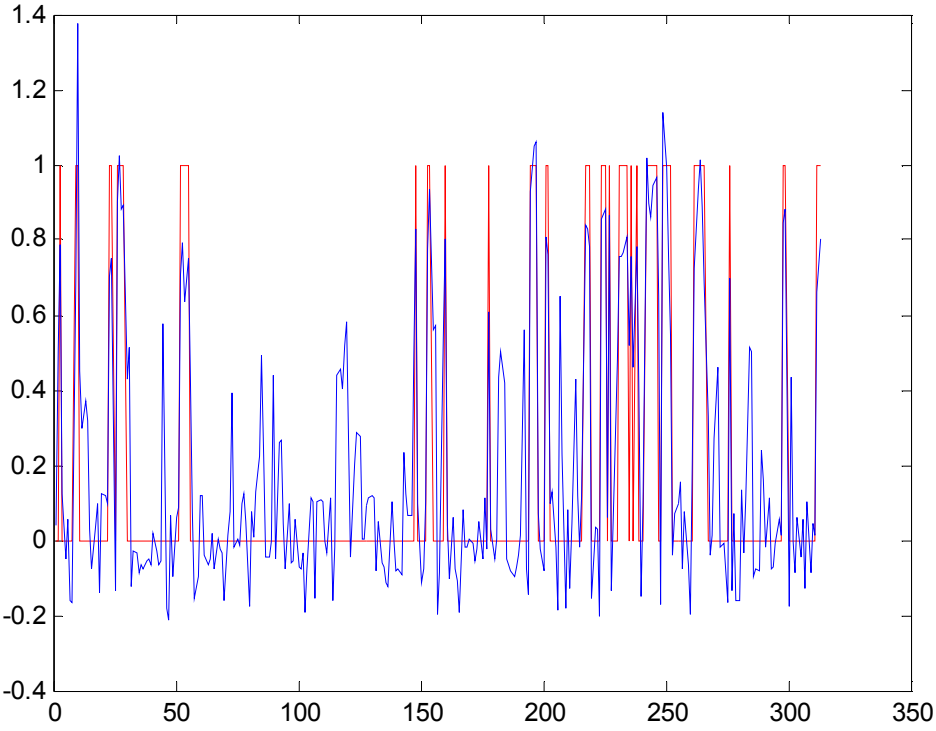

Figure 6. Out-of-sample forecasts with ANFIS and Gaussian membership function

Conclusions

We proposed and examined a financial distress model using ANFIS technology. Our findings support its utility and ANFIS can be a remarkable tool for financial crisis and distress prediction and not only. There is a huge field and gap in economics and finance where the neuro-fuzzy has not yet been practiced, tested or examined.

References

Cheng, W. Y., Su E. and Li, S. J. (2006). A Financial distress pre-warning study by fuzzy regression model of TSE-listed companies. *Asian Academy of Management Journal of Accounting and Finance*, Vol. 2, No. 2, pp. 75-93

Jang, J.-S.R. (1993). ANFIS: Adaptive-Network-based Fuzzy Inference Systems. *IEEE Transactions on Systems*, Man, and Cybernetics, Vol. 23, No. 3, pp. 665-685

Jang, J.-S. R. and Sun, C. T. (1995). Neuro-fuzzy Modeling and Control. *Proceedings of the IEEE*, Vol. 83, No. 3, pp. 378-406, March

O'leary, D.E. (1998). Using neural networks to predict corporate failure. *International Journal of Intelligent Systems in Accounting, Finance and Management,* Vol. 7*,* pp. 187–197.

Platt, H. D. and Platt, M. B. (2002). Predicting corporate financial distress: Reflections on choice-based sample bias. *Journal of Economics and Finance,* Vol. 26*,* pp. 184–199.

Zhang, G., Patuwo, B. E., and Hu, M. Y. (1998). Forecasting with artificial neural networks: The state of the art. *International Journal of Forecasting,* Vol. 14*,* pp. 35–62.

Appendix

MATLAB routine for ANFIS

```
%function []=neuro_fuzzy(x,y)

clear all;
load file.mat
% data is consisted by 3 columns, the first two concern the input
% variables and the last column the output variable
train_period=313
y=data(1:end- train_period,end)
x=data(1:end- train_period,1:2)
% y is the output dummy variable
% x is the matrix of inputs

em=3     % Select the operator, 1 for min which is for AND operator,
2 for product, which is for
         % AND operator and 3 for for max, which is for OR operator


memb=1  % 1 for triangular membership function, 2 for Gaussian, 3 for
        % sigmoid

% Set up the learning rates of your choice for center, bases and
parameters r

lr_center_cash=0.8
lr_center_ret=0.8
lr_base_cash=0.8
lr_base_ret=0.8
lr_r=0.8

% Take the inputs
ret=x(:,1)
cash=x(:,2)

% Take the dimensions of input-output data
[t nj]=size(y)
[nk ni]=size(x)

% Take center and bases values based on mean and standard deviation
respectively


% In the case you get zero sigma you can write the following. The
same is
% followed for the sigma in the ithe inputs
%if sigma_1==0 | sigma_2==0 |sigma_3==0
 %   sigma_1==0.025
%sigma_2==0.025
%sigma_3==0.025
%end



% Values for Unemployment rate
W_ret = sort(ret)

% Find the third quartile (75%)
```

```
Q3_ret = median(W_ret(find(W_ret>median(W_ret))));
e_high=find(ret>=Q3_ret & ret<=max(ret))
e_medium=find(ret>=mean(ret) & ret<=Q3_ret)
e_low=find(ret>=min(ret) & ret<=mean(ret) )


% find the values correspond to linguistic terms
e_high=ret(e_high)
e_medium=ret(e_medium)
e_low=ret(e_low)

% Find the mean and simga (standard deviation)
ce_1=mean(e_low)
ce_2=mean(e_medium)
ce_3=mean(e_high)

sigmae_1=std(e_low)
sigmae_2=std(e_medium)
sigmae_3=std(e_high)

W_cash = sort(cash)
Q3_cash = median(W_cash(find(W_cash>median(W_cash))));
ind_high=find(ret>=Q3_cash & cash<=max(cash))
ind_medium=find(cash>=mean(cash) & cash<=Q3_cash)
ind_low=find(cash>=min(cash) & cash<=mean(cash) )


% find the values correspond to linguistic terms
ind_high=cash(ind_high)
ind_medium=cash(ind_medium)
ind_low=cash(ind_low)


% Find the mean and simga (standard deviation)
cash_1=mean(ind_low)
cash_2=mean(ind_medium)
cash_3=mean(ind_high)


sigma_cash_1=std(ind_low)
sigma_cash_1=3
sigma_cash_2=std(ind_medium)
sigma_cash_3=std(ind_high)


if memb==1
% Take memebership degrees-grades


mf_ret_low=trimf(ret,[-sigmae_1/2+ce_1 ce_1 sigmae_1/2+ce_1]);     %
LOW
mf_ret_medium=trimf(ret,[-sigmae_2/2+ce_2 ce_2 sigmae_2/2+ce_2 ]); %
MEDIUM
mf_ret_high=trimf(ret,[-sigmae_3/2+ce_3 ce_3 sigmae_3/2+ce_3])

mf_cash_low=trimf(cash,[-sigma_cash_1/2+cash_1 cash_1
sigma_cash_1/2+cash_1]);     % LOW
```

```
mf_cash_medium=trimf(cash,[-sigma_cash_2/2+cash_2 cash_2
sigma_cash_2/2+cash_2 ]); % MEDIUM
mf_cash_high=trimf(cash,[-sigma_cash_3/2+sigma_cash_3 sigma_cash_3
sigma_cash_3/2+sigma_cash_3])

elseif memb==2

mf_ret_low=gaussmf(ret,[sigmae_1 ce_1]);     % LOW
mf_ret_medium=gaussmf(ret,[sigmae_2 ce_2 ]); % MEDIUM
mf_ret_high=gaussmf(ret,[sigmae_3 ce_3])


mf_cash_low=gaussmf(cash,[sigma_cash_1 cash_1]);     % LOW
mf_cash_medium=gaussmf(cash,[sigma_cash_2 cash_2 ]); % MEDIUM
mf_cash_high=gaussmf(cash,[sigma_cash_3 sigma_cash_3])

end

if em==1              % Min operator for AND rule
w1=min([mf_ret_low mf_cash_low]')     % LOW-LOW
w2=min([mf_ret_low mf_cash_medium]')  % LOW-MEDIUM
w3=min([mf_ret_low mf_cash_high]')    % LOW-HIGH
w4=min([mf_ret_medium mf_cash_low]')  % MEDIUM-LOW
w5=min([mf_ret_medium mf_cash_medium]')  % MEDIUM-MEDIUM
w6=min([mf_ret_medium mf_cash_high]')  % MEDIUM-HIGH
w7=min([mf_ret_high mf_cash_low]')  % HIGH-LOW
w8=min([mf_ret_high mf_cash_medium]')  % HIGH-MEDIUM
w9=min([mf_ret_high mf_cash_high]')  % HIGH-HIGH


elseif em==2      % product operator for AND rule
w1=(mf_ret_low.*mf_cash_low)'  % LOW-LOW
w2=(mf_ret_low.*mf_cash_medium)'  % LOW-MEDIUM
w3=(mf_ret_low.*mf_cash_high)'  % LOW-HIGH
w4=(mf_ret_medium.*mf_cash_low)'  % MEDIUM-LOW
w5=(mf_ret_medium.*mf_cash_medium)'  % MEDIUM-MEDIUM
w6=(mf_ret_medium.*mf_cash_high)'  % MEDIUM-HIGH
w7=(mf_ret_high.*mf_cash_low)'  % HIGH-LOW
w8=(mf_ret_high.*mf_cash_medium)'  % HIGH-MEDIUM
w9=(mf_ret_high.*mf_cash_high)'  % HIGH-HIGH


elseif em==3      % max operator for OR rule
w1=max([mf_ret_low mf_cash_low]')     % LOW-LOW
w2=max([mf_ret_low mf_cash_medium]')  % LOW-MEDIUM
w3=max([mf_ret_low mf_cash_high]')    % LOW-HIGH
w4=max([mf_ret_medium mf_cash_low]')  % MEDIUM-LOW
w5=max([mf_ret_medium mf_cash_medium]')  % MEDIUM-MEDIUM
w6=max([mf_ret_medium mf_cash_high]')  % MEDIUM-HIGH
w7=max([mf_ret_high mf_cash_low]')  % HIGH-LOW
w8=max([mf_ret_high mf_cash_medium]')  % HIGH-MEDIUM
w9=max([mf_ret_high mf_cash_high]')  % HIGH-HIGH
end



for j=1:t
if (w1(:,j)==0 & w2(:,j)==0 & w3(:,j)==0 & w4(:,j)==0 & w5(:,j)==0&
w6(:,j)==0& w7(:,j)==0& w8(:,j)==0& w9(:,j)==0)
```

```
nw1(:,j)=0;nw2(:,j)=0;nw3(:,j)=0;nw4(:,j)=0;
nw5(:,j)=0;nw6(:,j)=0;nw7(:,j)=0;nw8(:,j)=0;nw9(:,j)=0;
else
nw1(:,j)=w1(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7
(:,j)+w8(:,j)+w9(:,j));
nw2(:,j)=w2(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7
(:,j)+w8(:,j)+w9(:,j));
nw3(:,j)=w3(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7
(:,j)+w8(:,j)+w9(:,j));
nw4(:,j)=w4(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7
(:,j)+w8(:,j)+w9(:,j));
nw5(:,j)=w5(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7
(:,j)+w8(:,j)+w9(:,j));
nw6(:,j)=w6(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7
(:,j)+w8(:,j)+w9(:,j));
nw7(:,j)=w7(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7
(:,j)+w8(:,j)+w9(:,j));
nw8(:,j)=w8(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7
(:,j)+w8(:,j)+w9(:,j));
nw9(:,j)=w9(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7
(:,j)+w8(:,j)+w9(:,j));

end

end


X=[nw1.*ret';nw2.*ret';nw3.*ret';nw4.*ret';nw5.*ret';nw6.*ret';nw7.*r
et';nw8.*ret';nw9.*ret';...
nw1.*cash';nw2.*cash';nw3.*cash';nw4.*cash';nw5.*cash';
nw6.*cash';nw7.*cash';nw8.*cash';nw9.*cash';...
nw1;nw2;nw3;nw4;nw5;nw6;nw7;nw8;nw9];

params=pinv(X')*y % [p1 q1 r1……p27,q27,r27]
y1=X'*params;
e=y1-y;
error=sum(sum(e.^2))
tt=length(params)
Error=1/2*mse(e)

centers_1=[ce_1;ce_2;ce_3]
centers_2=[cash_1;cash_2;cash_3]

bases_1=[sigmae_1;sigmae_2;sigmae_3]
bases_2=[sigma_cash_1;sigma_cash_2;sigma_cash_3]

bases=[bases_1;bases_2 ]


centers=[centers_1;centers_2 ]

W=[mf_ret_low';mf_ret_medium';mf_ret_high';mf_cash_low';mf_cash_mediu
m';mf_cash_high']

[nk,ni]=size(x);
[n_rules,ni]=size(centers);
for i=1:n_rules

center_ret(i,:)=centers(i,1)
```

```
center_cash(i,:)=centers(i,2)
bases_ret(i,:)=bases(i,1)
bases_cash(i,:)=bases(i,2)
end

maxepochs=20
SSE_goal=10;
epochs=0
e.field=e
W.field=W

% Start the error backpropagation algorithm
while (epochs<maxepochs) & (error>SSE_goal)



for i=1:n_rules


eta_base1{i,:}=lr_base_ret
eta_base2{i,:}=lr_base_cash
eta_center1{i,:}=lr_center_ret
eta_center2{i,:}=lr_center_cash


end
eta_base=[eta_base1;eta_base2 ]

eta_center=[eta_center1;eta_center2 ]

eta_r{:,:}=lr_r

if memb==1
    for i=1:n_rules
    for j=1:ni



    ind_rule{:,i}=find((x(:,j)-centers(i,:))<=(bases(i,:)/2));

    end
end


x.field=x
 y1.field=y1

for kk=1:n_rules


delta_center{:,kk}=(y1.field(ind_rule{1,kk})*e.field(ind_rule{1,kk})'
*...
    (((2*sign(x.field(ind_rule{:,kk}))-centers(kk,:)))/bases(kk,:)))

delta_base{:,kk}=(y1.field(ind_rule{1,kk})*e.field(ind_rule{1,kk})'*.
..
    (((1-W.field(ind_rule{:,kk})))))/centers(kk,:))';
```

```
delta_r{:,:}=(e.field(ind_rule{1,kk}))'*(W.field(ind_rule{:,kk}));

del_center{:,kk}=-
((eta_center{kk,:}/(2*nk))*sum(delta_center{:,kk}));


del_base{:,kk}=-((eta_base{kk,:}/(2*nk))*sum(delta_base{:,kk}));

del_r{:,:}=-((eta_r{:,:}/(2*nk))*sum(delta_r{:,:}))

del_center_num(kk,:)=(del_center{:,kk})'

del_base_num(kk,:)=(del_base{:,kk})'

del_r_num(:,:)=(del_r{:,:})'

        end



elseif memb==2
for i=1:n_rules
    for j=1:ni



    ind_rule{:,i}=find((x(:,j)-centers(i,:))<=(bases(i,:)^2/2));

    end
end

x.field=x
y1.field=y1




for kk=1:n_rules


delta_center{:,kk}=(y1.field(ind_rule{1,kk})*e.field(ind_rule{1,kk})'
)'*...
    ((((x.field(ind_rule{1,kk})-centers(kk,:)))/bases(kk,:)^2))

delta_base{:,kk}=(y1.field(ind_rule{1,kk})*e.field(ind_rule{1,kk})')'
*...
    ((((x.field(ind_rule{1,kk})-centers(kk,:))).^2/bases(kk,:)^3))

delta_r{:,:}=(e.field(ind_rule{1,kk}))'*(W.field(ind_rule{:,kk}));


del_center{:,kk}=(-
((eta_center{kk,:})/(2*nk))*sum(delta_center{:,kk}));
```

```
del_base{:,kk}=(-((2*eta_base{kk,:})/(2*nk))*sum(delta_base{:,kk}));

del_center_num(kk,:)=(del_center{:,kk})'

del_base_num(kk,:)=(del_base{:,kk})'

del_r{:,:}=-((eta_r{:,:}/(2*nk))*sum(delta_r{:,:}))


del_r_num(:,:)=(del_r{:,:})'

end
end
centers=centers+del_center_num
bases=bases+del_base_num
y1.field=y1.field+del_r_num

center_ret=centers(1:3,1)
center_cash=centers(4:6,1)
bases_ret=bases(1:3,1)
bases_cash=bases(4:6,1)


if memb==1


mf_ret_low= trimf (ret,[-bases_ret(1,1)/2+center_ret(1,1)
center_ret(1,1)...
    bases_ret(1,1)/2+center_ret(1,1)]); % LOW
mf_ret_medium= trimf (ret,[-bases_ret(2,1)/2+center_ret(2,1)
center_ret(2,1)...
    bases_ret(2,1)/2+center_ret(2,1)]); % MEDIUM
mf_ret_high= trimf (ret,[-bases_ret(3,1)/2+center_ret(3,1)
center_ret(3,1)...
    bases_ret(3,1)/2+center_ret(3,1)]); % HIGH


mf_cash_low= trimf (cash,[-bases_cash(1,1)/2+center_cash(1,1)
center_cash(1,1)...
  bases_cash(1,1)/2+center_cash(1,1) ]); % LOW
mf_cash_medium= trimf (cash,[-bases_cash(2,1)/2+center_cash(2,1)
center_cash(2,1)...
  bases_cash(2,1)/2+center_cash(2,1)  ]); % MEDIUM
mf_cash_high= trimf (cash,[-bases_cash(3,1)/2+center_cash(3,1)
center_cash(3,1)...
  bases_cash(3,1)/2+center_cash(3,1) ]); % HIGH



elseif memb==2


mf_ret_low=gaussmf(ret,[bases_ret(1,1) center_ret(1,1)  ]);      %
LOW
```

```
mf_ret_medium=gaussmf(ret,[bases_ret(2,1) center_ret(2,1)  ]);   %
MEDIUM
mf_ret_high=gaussmf(ret,[bases_ret(3,1) center_ret(3,1)  ]);     %
HIGH


mf_cash_low=gaussmf(cash,[bases_cash(1,1) center_cash(1,1)   ]);
% LOW
mf_cash_medium=gaussmf(cash,[bases_cash(2,1) center_cash(2,1)   ]);
% MEDIUM
mf_cash_high=gaussmf(cash,[bases_cash(3,1) center_cash(3,1)  ]);
% HIGH
end

if em==1              % Min operator for AND rule
w1=min([mf_ret_low mf_cash_low]')      % LOW-LOW
w2=min([mf_ret_low mf_cash_medium]')  % LOW-MEDIUM
w3=min([mf_ret_low mf_cash_high]')    % LOW-HIGH
w4=min([mf_ret_medium mf_cash_low]')  % MEDIUM-LOW
w5=min([mf_ret_medium mf_cash_medium]')  % MEDIUM-MEDIUM
w6=min([mf_ret_medium mf_cash_high]')  % MEDIUM-HIGH
w7=min([mf_ret_high mf_cash_low]')  % HIGH-LOW
w8=min([mf_ret_high mf_cash_medium]')  % HIGH-MEDIUM
w9=min([mf_ret_high mf_cash_high]')  % HIGH-HIGH


elseif em==2      % product operator for AND rule
w1=(mf_ret_low.*mf_cash_low)'  % LOW-LOW
w2=(mf_ret_low.*mf_cash_medium)'  % LOW-MEDIUM
w3=(mf_ret_low.*mf_cash_high)'  % LOW-HIGH
w4=(mf_ret_medium.*mf_cash_low)'  % MEDIUM-LOW
w5=(mf_ret_medium.*mf_cash_medium)'  % MEDIUM-MEDIUM
w6=(mf_ret_medium.*mf_cash_high)'  % MEDIUM-HIGH
w7=(mf_ret_high.*mf_cash_low)'  % HIGH-LOW
w8=(mf_ret_high.*mf_cash_medium)'  % HIGH-MEDIUM
w9=(mf_ret_high.*mf_cash_high)'  % HIGH-HIGH


elseif em==3      % max operator for OR rule
w1=max([mf_ret_low mf_cash_low]')      % LOW-LOW
w2=max([mf_ret_low mf_cash_medium]')  % LOW-MEDIUM
w3=max([mf_ret_low mf_cash_high]')    % LOW-HIGH
w4=max([mf_ret_medium mf_cash_low]')  % MEDIUM-LOW
w5=max([mf_ret_medium mf_cash_medium]')  % MEDIUM-MEDIUM
w6=max([mf_ret_medium mf_cash_high]')  % MEDIUM-HIGH
w7=max([mf_ret_high mf_cash_low]')  % HIGH-LOW
w8=max([mf_ret_high mf_cash_medium]')  % HIGH-MEDIUM
w9=max([mf_ret_high mf_cash_high]')  % HIGH-HIGH
end



for j=1:t
if (w1(:,j)==0 & w2(:,j)==0 & w3(:,j)==0 & w4(:,j)==0 & w5(:,j)==0&
w6(:,j)==0& w7(:,j)==0& w8(:,j)==0& w9(:,j)==0)
nw1(:,j)=0;nw2(:,j)=0;nw3(:,j)=0;nw4(:,j)=0;
nw5(:,j)=0;nw6(:,j)=0;nw7(:,j)=0;nw8(:,j)=0;nw9(:,j)=0;
else
```

```
nw1(:,j)=w1(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7
(:,j)+w8(:,j)+w9(:,j));
nw2(:,j)=w2(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7
(:,j)+w8(:,j)+w9(:,j));
nw3(:,j)=w3(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7
(:,j)+w8(:,j)+w9(:,j));
nw4(:,j)=w4(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7
(:,j)+w8(:,j)+w9(:,j));
nw5(:,j)=w5(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7
(:,j)+w8(:,j)+w9(:,j));
nw6(:,j)=w6(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7
(:,j)+w8(:,j)+w9(:,j));
nw7(:,j)=w7(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7
(:,j)+w8(:,j)+w9(:,j));
nw8(:,j)=w8(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7
(:,j)+w8(:,j)+w9(:,j));
nw9(:,j)=w9(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7
(:,j)+w8(:,j)+w9(:,j));

end

end



X_train=[nw1.*ret';nw2.*ret';nw3.*ret';nw4.*ret';nw5.*ret';nw6.*ret';
nw7.*ret';nw8.*ret';nw9.*ret';...
nw1.*cash';nw2.*cash';nw3.*cash';nw4.*cash';nw5.*cash';
nw6.*cash';nw7.*cash';nw8.*cash';nw9.*cash';...
nw1;nw2;nw3;nw4;nw5;nw6;nw7;nw8;nw9];
x=x.field

params_train=pinv(X_train')*y % [p1 p2 p3 p4 p5 q1 q2 q3 q4 q5 r1 r2
r3 r4 r5]
y1=X_train'*params_train;
r=params_train(end-8:end,:)
outp=y1-y;
error=sum(sum(outp.^2))
epochs=epochs+1
array_y ( epochs )= error;
index_epochs ( epochs ) = epochs;
end

antecedent_par=[nw1',nw2',nw3',nw4',nw5',nw6',nw7',nw8',nw9']
consequent_par=y1

output_2=sum(antecedent_par'*consequent_par)/sum(sum(antecedent_par))
out= output_2

for kkk=1:nk
    if y1(kkk,:)>out
        S_in_sample(kkk,:)=1

    elseif y1(kkk,:)<out
        S_in_sample(kkk,:)=0
    end
end

% Compuation for clasiffication table preparation
```

```
Actual_positive_in_sample=find(y==1)
Actual_negative_in_sample=find(y==0)

Predicted_positive_in_sample=find(S_in_sample==1)
Predicted_negative_in_sample=find(S_in_sample==0)

Sum_actual_positive_in_sample=length(Actual_positive_in_sample)
Sum_actual_negative_in_sample=length(Actual_negative_in_sample)

Sum_predicted_positive_in_sample=length(Predicted_positive_in_sample)
Sum_predicted_negative_in_sample=length(Predicted_negative_in_sample)

Total_predicted_in_sample=find(y==S_in_sample)

Perc=(length(Total_predicted_in_sample)/nk)*100

W_in_sample=y(Total_predicted_in_sample)

Positive_in_sample=find(W_in_sample==1)
Negative_in_sample=find(W_in_sample==0)


Final_predicted_Positive_in_sample=length(Positive_in_sample)/Sum_act
ual_positive_in_sample

Final_predicted_Negative_in_sample=length(Negative_in_sample)/Sum_act
ual_negative_in_sample

Final_predicted_Positive_in_sample=Final_predicted_Positive_in_sample
*100

Final_predicted_Negative_in_sample=Final_predicted_Negative_in_sample
*100

Total_performance_in_sample=((length(Positive_in_sample) +
length(Negative_in_sample))/nk)*100


clear nw1
clear nw2
clear nw3
clear nw4
clear nw5
clear nw6
clear nw7
clear nw8
clear nw9


load file.mat
y_tes=data(end-train_period+1:end,end)
x_tes=data(end- train_period+1:end,1:2)
 ret=x_tes(:,1)
cash=x_tes(:,2)

t1=length(y_tes)
if memb==1
```

```
mf_ret_low= trimf (ret,[-bases_ret(1,1)/2+center_ret(1,1)
center_ret(1,1)...
    bases_ret(1,1)/2+center_ret(1,1)]); % LOW
mf_ret_medium= trimf (ret,[-bases_ret(2,1)/2+center_ret(2,1)
center_ret(2,1)...
    bases_ret(2,1)/2+center_ret(2,1)]); % MEDIUM
mf_ret_high= trimf (ret,[-bases_ret(3,1)/2+center_ret(3,1)
center_ret(3,1)...
    bases_ret(3,1)/2+center_ret(3,1)]); % HIGH


mf_cash_low= trimf (cash,[-bases_cash(1,1)/2+center_cash(1,1)
center_cash(1,1)...
  bases_cash(1,1)/2+center_cash(1,1) ]); % LOW
mf_cash_medium= trimf (cash,[-bases_cash(2,1)/2+center_cash(2,1)
center_cash(2,1)...
  bases_cash(2,1)/2+center_cash(2,1)  ]); % MEDIUM
mf_cash_high= trimf (cash,[-bases_cash(3,1)/2+center_cash(3,1)
center_cash(3,1)...
  bases_cash(3,1)/2+center_cash(3,1) ]); % HIGH



elseif memb==2
mf_ret_low=gaussmf(ret,[bases_ret(1,1) center_ret(1,1)  ]);      %
LOW
mf_ret_medium=gaussmf(ret,[bases_ret(2,1) center_ret(2,1)  ]);   %
MEDIUM
mf_ret_high=gaussmf(ret,[bases_ret(3,1) center_ret(3,1)  ]);     %
HIGH


mf_cash_low=gaussmf(cash,[bases_cash(1,1) center_cash(1,1)   ]);
% LOW
mf_cash_medium=gaussmf(cash,[bases_cash(2,1) center_cash(2,1)   ]);
% MEDIUM
mf_cash_high=gaussmf(cash,[bases_cash(3,1) center_cash(3,1)  ]);
% HIGH
end

if em==1              % Min operator for AND rule
w1=min([mf_ret_low mf_cash_low]')     % LOW-LOW
w2=min([mf_ret_low mf_cash_medium]')  % LOW-MEDIUM
w3=min([mf_ret_low mf_cash_high]')    % LOW-HIGH
w4=min([mf_ret_medium mf_cash_low]')  % MEDIUM-LOW
w5=min([mf_ret_medium mf_cash_medium]')  % MEDIUM-MEDIUM
w6=min([mf_ret_medium mf_cash_high]')  % MEDIUM-HIGH
w7=min([mf_ret_high mf_cash_low]')  % HIGH-LOW
w8=min([mf_ret_high mf_cash_medium]')  % HIGH-MEDIUM
w9=min([mf_ret_high mf_cash_high]')  % HIGH-HIGH


elseif em==2      % product operator for AND rule
w1=(mf_ret_low.*mf_cash_low)'  % LOW-LOW
w2=(mf_ret_low.*mf_cash_medium)'  % LOW-MEDIUM
w3=(mf_ret_low.*mf_cash_high)'  % LOW-HIGH
w4=(mf_ret_medium.*mf_cash_low)'  % MEDIUM-LOW
w5=(mf_ret_medium.*mf_cash_medium)'  % MEDIUM-MEDIUM
w6=(mf_ret_medium.*mf_cash_high)'  % MEDIUM-HIGH
```

```
w7=(mf_ret_high.*mf_cash_low)'  % HIGH-LOW
w8=(mf_ret_high.*mf_cash_medium)'  % HIGH-MEDIUM
w9=(mf_ret_high.*mf_cash_high)'  % HIGH-HIGH


elseif em==3      % product operator for OR rule
w1=max([mf_ret_low mf_cash_low]')     % LOW-LOW
w2=max([mf_ret_low mf_cash_medium]')  % LOW-MEDIUM
w3=max([mf_ret_low mf_cash_high]')    % LOW-HIGH
w4=max([mf_ret_medium mf_cash_low]')  % MEDIUM-LOW
w5=max([mf_ret_medium mf_cash_medium]')  % MEDIUM-MEDIUM
w6=max([mf_ret_medium mf_cash_high]')  % MEDIUM-HIGH
w7=max([mf_ret_high mf_cash_low]')  % HIGH-LOW
w8=max([mf_ret_high mf_cash_medium]')  % HIGH-MEDIUM
w9=max([mf_ret_high mf_cash_high]')  % HIGH-HIGH
end

for j=1:t1
if (w1(:,j)==0 & w2(:,j)==0 & w3(:,j)==0 & w4(:,j)==0 & w5(:,j)==0&
w6(:,j)==0& w7(:,j)==0& w8(:,j)==0& w9(:,j)==0)
nw1(:,j)=0;nw2(:,j)=0;nw3(:,j)=0;nw4(:,j)=0;
nw5(:,j)=0;nw6(:,j)=0;nw7(:,j)=0;nw8(:,j)=0;nw9(:,j)=0;
else
nw1(:,j)=w1(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7
(:,j)+w8(:,j)+w9(:,j));
nw2(:,j)=w2(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7
(:,j)+w8(:,j)+w9(:,j));
nw3(:,j)=w3(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7
(:,j)+w8(:,j)+w9(:,j));
nw4(:,j)=w4(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7
(:,j)+w8(:,j)+w9(:,j));
nw5(:,j)=w5(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7
(:,j)+w8(:,j)+w9(:,j));
nw6(:,j)=w6(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7
(:,j)+w8(:,j)+w9(:,j));
nw7(:,j)=w7(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7
(:,j)+w8(:,j)+w9(:,j));
nw8(:,j)=w8(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7
(:,j)+w8(:,j)+w9(:,j));
nw9(:,j)=w9(:,j)/(w1(:,j)+w2(:,j)+w3(:,j)+w4(:,j)+w5(:,j)++w6(:,j)+w7
(:,j)+w8(:,j)+w9(:,j));

end

end
X_test=[nw1.*ret';nw2.*ret';nw3.*ret';nw4.*ret';nw5.*ret';nw6.*ret';n
w7.*ret';nw8.*ret';nw9.*ret';...
nw1.*cash';nw2.*cash';nw3.*cash';nw4.*cash';nw5.*cash';
nw6.*cash';nw7.*cash';nw8.*cash';nw9.*cash';...
nw1;nw2;nw3;nw4;nw5;nw6;nw7;nw8;nw9];

yf1=X_test'*params_train;

for kkk=1:t1
    if yf1(kkk,:)>out
        S_out_sample(kkk,:)=1

    elseif yf1(kkk,:)<out
        S_out_sample(kkk,:)=0
    end
```

```
end

Actual_positive=find(y_tes==1)
Actual_negative=find(y_tes==0)

Predicted_positive=find(S_out_sample==1)
Predicted_negative=find(S_out_sample==0)

Sum_actual_positive=length(Actual_positive)
Sum_actual_negative=length(Actual_negative)

Sum_predicted_positive=length(Predicted_positive)
Sum_predicted_negative=length(Predicted_negative)

Total_predicted=find(y_tes==S_out_sample)

WWW=y_tes(Total_predicted)

Positive=find(WWW==1)
Negative=find(WWW==0)

ccc=0.01
Final_predicted_Positive=length(Positive)/Sum_actual_positive

Final_predicted_Negative=length(Negative)/Sum_actual_negative

Final_predicted_Positive=Final_predicted_Positive*100

Final_predicted_Negative=Final_predicted_Negative*100

Total_performance=((length(Positive) + length(Negative))/t1)*100
% Classification Table
re = '========================';
li= '------------------------';
sp = '  ';
disp([re ' Classification Table ' re])
disp([li '         True         ' li])
disp([ '                   1                                   0
Total ' ])
disp([li ' --------------------' li])
disp(sprintf('               %f:                          %f    %f',
length(Positive_in_sample),...
    Sum_actual_positive_in_sample-length(Positive_in_sample) ))
disp(sprintf('               %f:                          %f    %f',
Sum_actual_negative_in_sample-...
    length(Negative_in_sample) , length(Negative_in_sample)))
disp([li ' --------------------' li])
disp(sprintf('Sum            %f:                          %f    %f',
Sum_actual_positive_in_sample,...
    Sum_actual_negative_in_sample, nk))
disp([li ' --------------------' li])

disp(sprintf('Correct Predict Financial Stage 1 (Crisis)
%f',...
    Final_predicted_Positive_in_sample))

disp(sprintf('Correct Predict Financial Stage 2 (No Crisis)
%f',...
```

```
    Final_predicted_Negative_in_sample))

disp(sprintf('Overall Prediction Performance
%f',...
    Total_performance_in_sample))

disp(blanks(1)')

disp([re ' Classification Table ' re])
disp([li '         True         ' li])
disp([ '                    1                                    0
Total ' ])
disp([li ' ---------------------' li])
disp(sprintf('               %f:                          %f    %f',
length(Positive),...
    Sum_actual_positive-length(Positive)))
disp(sprintf('               %f:                         %f    %f',
...
    Sum_actual_negative-length(Negative), length(Negative)))
disp([li ' ---------------------' li])
disp(sprintf('Sum            %f:                         %f    %f',
Sum_actual_positive ,...
    Sum_actual_negative, t1))
disp([li ' ---------------------' li])

disp(sprintf('Correct Predict Financial Stage 1 (Crisis)
%f', Final_predicted_Positive))

disp(sprintf('Correct Predict Financial Stage 2 (No Crisis)
%f', Final_predicted_Negative))

disp(sprintf('Overall Prediction Performance
%f', Total_performance))

figure, plot(y,'-r'); hold on; plot(y1,'-b');
xlabel('Periods')
ylabel('Values')
%title('In_sample forecasts')
h1 = legend('Actual','forecasts',1);
figure, plot(y_tes,'-r'); hold on; plot(yf1,'-b');
xlabel('Periods')
ylabel('Values')
%title('Out_of_sample forecasts')
h = legend('Actual','forecasts',1);
figure, plot (index_epochs , array_y );
xlabel('epochs')
ylabel('Error')
title('Number of Epochs')
```