Papre 250-26

Using SAS® Enterprise Miner™ for Forecasting

Kattamuri S. Sarma

Introduction:

This paper shows how to use SAS Enterprise Miner™ to develop forecasting models. As an illustration, a model is presented to forecast the entire current quarter's Gross Domestic Product (GDP) based on monthly indicators available during only the first half of the quarter.

The paper shows, step-by-step, how to set up a forecasting project, use different nodes of the SAS Enterprise Miner™ to train and validate the models, and display the results. SAS Enterprise Miner™ consists of a number of nodes for data cleaning, exploratory data analysis, model development and validation, scoring and forecasting. Only a few of these nodes are used in this project. In particular, the focus here is the Neural Network node. The Decision Tree node can also be used for exploratory data analysis and modeling, but it is not included in the discussion due to space limitation. It is hoped that the material presented here serves as an introduction to the SAS Enterprise Miner™. The methodology can easily be extended to other applications such as predicting response to direct mail in marketing, forecasting potential losses of prospective credit card customers, etc.

Forecasting Current Quarter GDP.

The current quarter model developed here is based on monthly indicators released during the quarter. The monthly indicators are: (1) Non-farm payroll employment, (2) Average weekly hours in private nonagricultural establishments, (3) Index of industrial Production, and (4) Real retail sales. By the middle of each quarter all the four indicators are published for the first month of the quarter. So we forecast the four indicator variables for the remaining two months of the quarter using a monthly model. The quarterly average of each indicator variable is then calculated using the actual value for the first month and the predicted values for the second and third months. We use these quarterly averages of the indicator variables as inputs to a quarterly equation, which forecasts current quarter GDP sixty days ahead of the official release. Both the monthly model and the quarterly forecasting equation are developed using the Neural Network node of the SAS Enterprise Miner™.

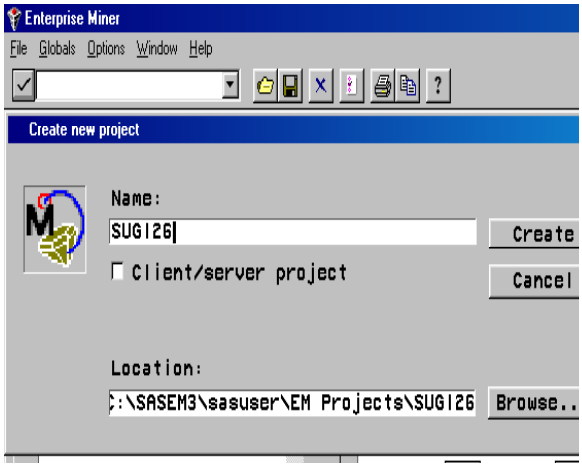
In order to make the process clearer, let us review how current quarter forecasts are made for a specific quarter using this model. Suppose we are interested in forecasting GDP growth for the second quarter. The quarter begins on April 1, and ends on June 30. By May 15th or 17th we get published values of all the four indicator variables for the month of April. So we forecast the indicator variables for May and June. A monthly Vector Auto Regression (VAR) Model is used for this purpose. We calculate the average of each indicator variable for the second quarter using the published data for April and the forecasted data for May and June. Then we feed the quarterly averages of the indicator variables into the quarterly equation to get a forecast of GDP growth for the second quarter. The actual estimate of the second quarter GDP is released by the US Department of Commerce during the third week of July. Therefore the forecasts obtained from the model have a lead-time of sixty days.

Setting up the Forecasting Project in SAS Enterprise Miner.

In order to start the forecasting project we follow these simple steps:

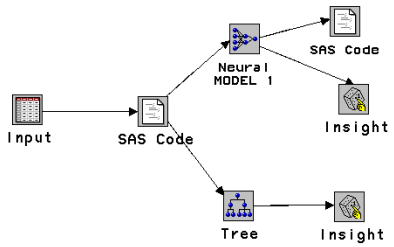
(1) Open the Enterprise Miner™.
(2) From the Menu bar select
File -> New -> Project.
The project window opens (See Diagram 1)
(3) Type the project name, and push the button "Create."

Diagram1: Starting the Project



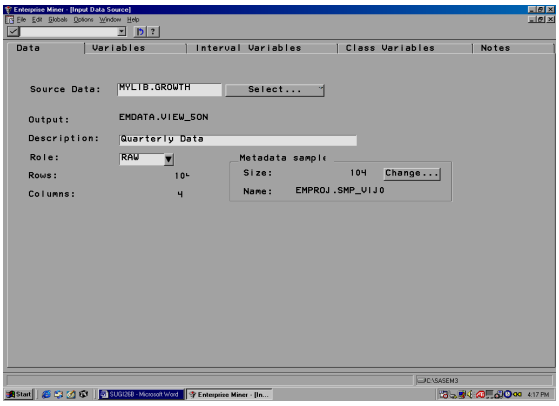
When you push the "Create" button a project window (not shown here) opens. It is split into two sub-windows side-by-side. On the left there is Project name, and underneath it there is a diagram name. Initially the name of the diagram is "untitled." We changed the name to "Quarterly." By double clicking on the diagram name the second window on the right is activated. Here we draw the network diagram for the project. By dragging the node icons into this window and connecting them we build a network of nodes as shown in Diagram 2 below.

Diagram 2: Network diagram for the quarterly model.



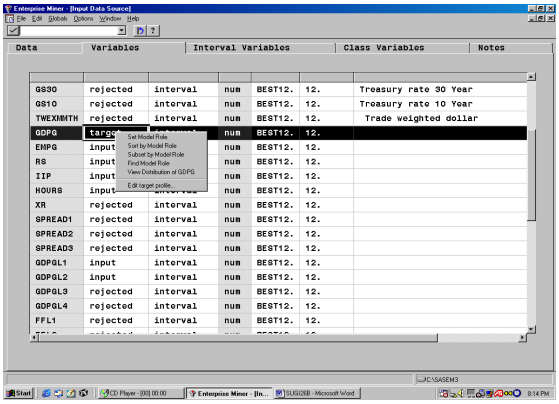
The first node in Diagram 2 is the input node. In this node we specify whether the role of the data set is "raw," "train," "validate," "test," or "score." Diagram 3 shows the input node after it is opened by double clicking on it. On the top of this window there are 5 tabs: Data, Variables, Interval Variables, Class Variables, and Notes. Below there are three text boxes for Source Data, Description, and Role. First we specify the source of the data, libref.name, add some descriptive text, and assign a role to it. In this example, the source data is mylib.growth, The Description is "Quarterly Data," and the role assigned is "Raw." In this window there are push buttons to select the source data and change the size of the metadata.

Diagram 3: Input Node



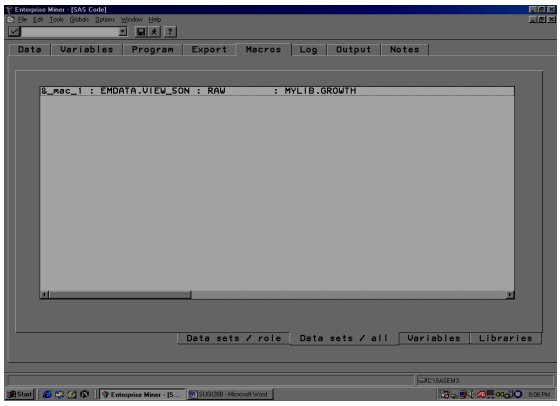
By selecting the Variables tab we can see a list of variables as in Diagram 4. By selecting any variable we can assign it a model role such as "target," "input," or "reject." In this example the variable *gdpg* (GDP growth) is assigned the model role of target, and the variables *empg, iip, hours,* and *rs* (retail sales), *gdpgl1, and gdpgl2* are assigned the role of "input." The variable *time* is given the role of "id" and all other variables are "rejected."

Diagram 4: Input Node: Assigning model roles to variables



The data for this project consists of quarterly observations from the third quarter of 1974 to the second quarter of 2000. Observations from the third quarter of 1974 to the first quarter of 1998 are used for model development (training), and the remaining observations are used for validation. If we were to partition the data into development and validation samples randomly, we could have used the Data Partition node of the Enterprise Miner. Such random partitioning is not appropriate for time series data since the order of observations cannot be changed. In order to preserve the chronological order of the observations in the training and validation samples, we do the partitioning in the SAS code node.

Diagram 5: SAS Code Node:



When we open the SAS code node window there are tabs for Variables, Program, Export, Macros, Log, Output and Notes. When we select the Macros tab a set of under-tabs appears. and we select Data sets /all. This brings up a window with a list of the imported data sets and the macro names given to them by the Enterprise Miner™ (Diagram 5). In the example considered here the data set name is mylib.growth. The Enterprise Miner™ has assigned the macro name &_MAC_1 to this raw data set. Next we have to identify the roles of the data sets to be created in the SAS code node. These data sets will be exported to the next node. To add the data sets for export we select the Export tab, and push the "Add" button. A pop-up menu opens. In it we select the data sets we intend to create. For this project we need to create a "Training" data set, and a "Validation" data set. We first click on "Train," and add another data set for "validation" by clicking "Add" and "Validate". The Enterprise Miner has assigned the names &_TRA and &_VAL to these data sets. These macro names are used in the Program in the SAS Code node. Diagrams 6 and 7 show the selection of the roles of the exported data sets. Diagram 8 shows the SAS code we entered in the SAS Code node.

Diagram 6: Selecting data set names for exported data sets

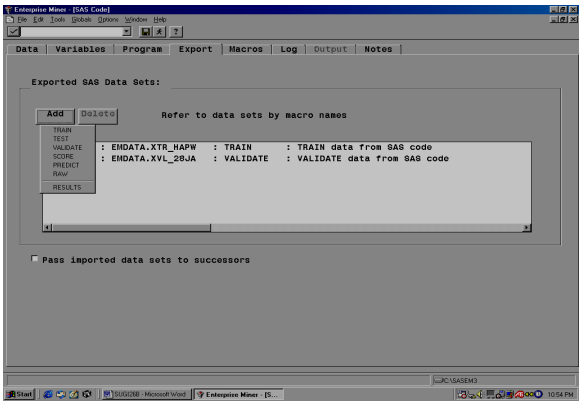


Diagram 7: Exported data sets

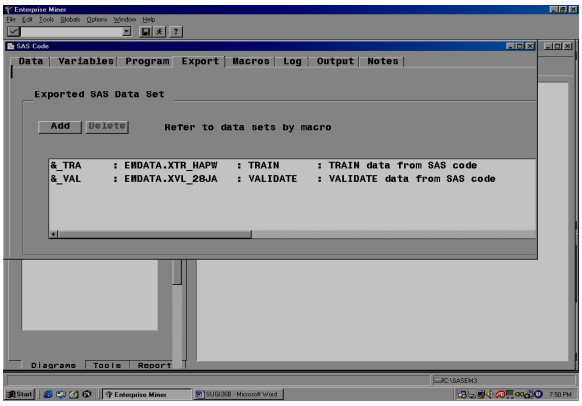


Diagram 8: SAS code:

```
Data &_TRA &_VAL;
 set &_MAC_1;
 if time le 1998.1 then output &_TRA ;
 if time ge 1998.2 then output &_VAL;
run;
```

A neural network for forecasting Current Quarter GDP.

The next node is the Neural Network node. A neural network with one hidden layer with 3 neurons can be represented by the following equation.

$$gdpg = w_0 + w_1 H_{11} + w_2 H_{12} + w_3 H_{13} \quad (1)$$

Where, $H_{11}, H_{12,}$ and H_{13} are the outputs of the first, second, and third neurons in the hidden layer. In this example there is only one hidden layer with three neurons. One can add more hidden layers and/or increase the number of neurons in each hidden layer. The number of parameters increases as we increase the number of hidden layers and/or hidden neurons. The output (target) variable is $gdpg$. The output of the jth neuron in the ith hidden layer, represented by H_{ij} is a function of the inputs. The inputs are first combined using a combination function (η_{ij}). The combination function can be linear as shown in equation (2) below, or radial. For the forecasting model developed here we use the following linear combination function.

$$\eta_{ij} = w_{0ij} + w_{1ij} empg + w_{2ij} hours + w_{3ij} iip + w_{4ij} rs + w_{5ij} gdpgl1 + w_{6ij} gdpgl2 \quad (2)$$

In the example considered here we standardized the inputs using the standardization technique "standard deviation." Hence the inputs in equation (2) should be interpreted as standardized inputs, not as actual inputs.

The inputs are: $empg$ (growth in non-farm payroll employment), $hours$ (average weekly hours private nonagricultural establishments), IIP (index of industrial production), rs (real retail sales), $gdpgl1$ (GDP growth in the previous quarter), and $gdpgl2$ (GDP growth two quarters before).

In the terminology of Generalized Linear Models equation (2) is called the linear predictor. In neural networks it is called the combination function. The output of each neuron is a transformation of the combination function. This transformation is achieved by an activation function. The neural network node in SAS Enterprise Miner offers a choice of different activation functions. We tried hyperbolic tangent and Elliot. Elliot has given better forecasts. It is of the following form.

$$H_{ij} = \frac{\eta_{ij}}{1 + |\eta_{ij}|} \quad (3)$$

where H_{ij} is the output of the jth neuron in ith hidden layer. The "intercept" terms, w_0 and w_{0ij} in equations (1) and (2) are called "bias."

Table 1
Estimated Weights in the Neural Network

From	To	Weight
EMPG	H11	11.6601
GDPGL1	H11	-5.7695
GDPGL2	H11	-5.6517
HOURS	H11	0.2445
IIP	H11	-0.6599
RS	H11	1.0239
EMPG	H12	-0.1841
GDPGL1	H12	0.0421
GDPGL2	H12	0.0195
HOURS	H12	0.0250
IIP	H12	-0.1842
RS	H12	-0.2210
EMPG	H13	-6.4627
GDPGL1	H13	3.7112
GDPGL2	H13	10.0720
HOURS	H13	26.9298
IIP	H13	8.3042
RS	H13	-9.7779
BIAS	H11	10.9206
BIAS	H12	0.8189
BIAS	H13	9.0886
H11	GDPG	2.4496
H12	GDPG	-14.7791
H13	GDPG	1.3070
BIAS	GDPG	7.1458

Note: The inputs in Table 1 are standardized Inputs.

Below we show how the weights given in Table I are used in equation (1), (2) and (3) to calculate the predicted value of the target variable (GDP growth.)

First the variables are standardized as follows:

```
S_EMPG    = -0.8835 + 0.4413 * empg
S_GDPGL1  = -0.8396 + 0.2759 * gdpgl1
S_GDPGL2  = -0.8109 + 0.2719 * gdpgl2
S_HOURS   =  0.1841 + 0.8466 * hours
S_IIP     = -0.4302 + 0.1495 * iip
S_RS      = -0.3107 + 0.1678 * rs
```

These standardized inputs are substituted in the combination functions given in equation (2) for each observation in the validation data set.

η_{11} = 11.6601*S_EMPG-5.7695*S_GDPGL1
-5.6517*S_GDPGL2 +0.2445*S_HOURS
-0.6599*S_IIP+1.0239*S_RS
+10.9206

η_{12} = -0.1841*S_EMPG+0.0421*S_GDPGL1
+0.0195*S_GDPGL2+0.0250*S_HOURS
-0.1842*S_IIP-0.2210*S_RS
+0.8189

η_{13} = -6.4628*S_EMPG+3.7118*S_GDPGL1
+10.0720*S_GDPGL2+26.9297*S_HOURS
+8.3042*S_IIP-9.7779*S_RS
+ 9.0886

The output of each neuron is calculated by applying the activation function:

H11 = η_{11} / (1.0 + ABS(η_{11}))

H12 = η_{12} / (1.0 + ABS(η_{12}))

H13 = η_{13} / (1.0 + ABS(η_{13}))

Finally the predicted value of the target is calculated for each quarter in the validation data set as follows:

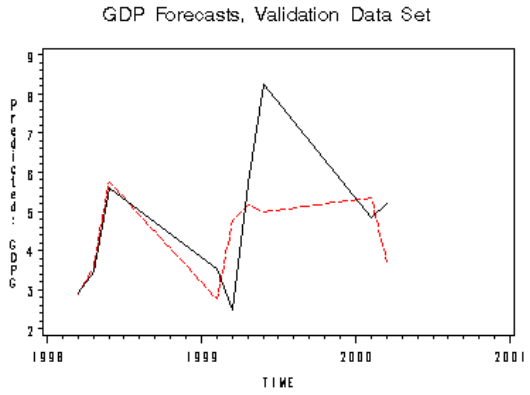
```
P_GDPG = 2.4496*H11-14.7791*H12
         +1.3070*H13
         +7.1458
```

In this example no activation function is used to calculate the predicted value from the output of the hidden nodes. SAS code is written in the final SAS Code node to plot the actuals and predicted values from the validation data set. The plot is saved as a gif file, and displayed in Diagram 9.

The code that generated the plot file is reproduced below:

```
filename outfile 'c:\Gdp\graph1.gif' ;
goptions gsfmode=replace gsfname=outfile
device=gif373;
proc gplot data=&_mac_2 ;
 title h=1.5 'GDP Forecasts, Validation
Data Set';
 symbol1 c = r v=none i=join l=3;
 symbol2 c=bl v=none i=join;
 plot p_gdpg*time=1 gdpg*time=2 / overlay
frame
 haxis=time ;
run;
quit;
```

Diagram 9:



A Vector Autoregression Model of Monthly Indicators.

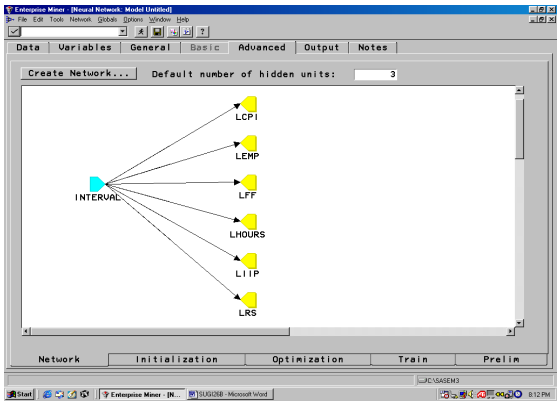
As pointed out earlier, when we make a current quarter GDP forecast at the middle of the quarter we have data for only one month of the current quarter. In order to make quarterly averages of the monthly indicators we need data on the indicators for the remaining two months in the quarter. Since these data do not exist at the time of the forecast we forecast them using a Vector Auto Regression (VAR) model. The VAR can be represented by the following set of equations:

$$y_{it} = \sum_{l=1}^{L}\sum_{j=1}^{6}\phi_{jt-l}\,y_{jt-l}$$
$$i = 1,\ldots 6 \qquad (4)$$

Where y_{it} is a monthly indicator for current month (t), and y_{jt-l} is the j^{th} monthly indicator lagged l months. In the example used here there are 6 monthly indicators. They are: Index of industrial production (IIP),

real retail sales (RS), average weekly hours of non-farm workers (hours), non-farm payroll employment (EM), Consumer price index excluding food and fuel (CPI), and Federal Funds Rate (FF). All variables are logarithms of the original variables. The VAR model shown in (4) above is also estimated by the Neural Network node. The neural network diagram is shown in Diagram 10.

Diagram 10: Vector Auto Regression Model for forecasting monthly indicators.



Note that the input node is connected to each target directly without any hidden nodes.

In order to construct this network we have to first activate the advanced user interface from the "General" tab in the neural network node. By selecting the "Advanced" tab we get a condensed network diagram, which by default includes a hidden layer. By deleting the hidden layer, a direct connection is established between the inputs and targets. There are 6 targets and the inputs are the lagged values of the targets. Initially all targets are in a single node. This would give the same set of coefficients for all the target variables. Although the inputs are the same for each target, in a VAR model each equation should have a different set of coefficients. This feature can be imposed by opening the target node (by double clicking on it), and selecting the "Variables" tab and pushing the "Transfer" button. A list of all the target variables (Six in this example) appears. We select all the target variables and choose "multiple new nodes" from the pop-up menu. This creates one node for each target as shown in Diagram 10.

This network represents a Vector Auto Regression Model. There is no activation function and a linear combination function is used at each node. It is easy to add an activation function at each node, and hidden nodes can also be added to get a non-linear VAR Model.

To make the model operational, the monthly VAR model is first run, and the indicator variables are predicted one period at a time. A quarterly data set is created with quarterly averages of the monthly variables. This data is scored by the score code generated by the Neural Network node of the quarterly model. These tasks can be done either in the SAS code node or in the SAS Program window.

References

(1) Kattamuri S. Sarma. 1991, *Forecasting Current Quarter GNP: A Comparison of Alternative Methods*, Paper presented at the Eleventh International Symposium on Forecasting, New York, June 11.

(2) SAS Institute Inc., *Data Mining Using Enterprise Miner™ Software: A Case Study Approach, First Edition*, Cary, NC: SAS Institute Inc., 2000.

Kattamuri S. Sarma, Ph.D.
Ecostat Research Corp.
61 Hawthorne Street
White Plains, NY 10603
(914) 428-8733. Fax: 914-428-4551.
Email: KSSarma@worldnet.att.net