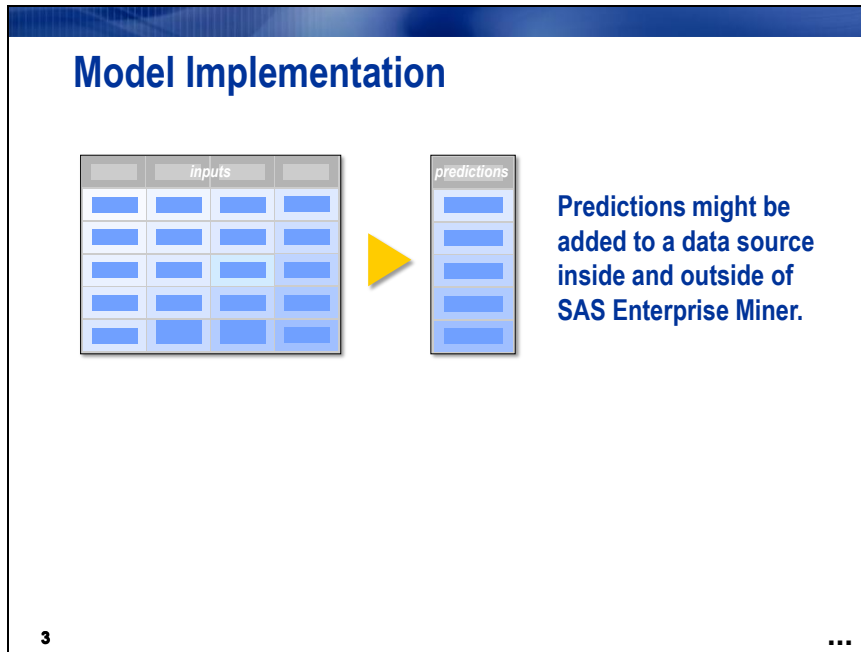


Chapter 7 Model Implementation

0.1	Introduction.....	Error! Bookmark not defined.
0.2	A Section Title	Error! Bookmark not defined.
	Demonstration: <Type title of demo here.>	Error! Bookmark not defined.
	Exercises	Error! Bookmark not defined.
0.3	Chapter Summary.....	Error! Bookmark not defined.
0.4	Solutions	Error! Bookmark not defined.
	Solutions to Exercises	Error! Bookmark not defined.
	Solutions to Student Activities (Polls/Quizzes)	Error! Bookmark not defined.

7.1 Introduction



After you train and compare predictive models, one model is selected to represent the association between the inputs and the target. After it is selected, this model must be put to use. The contribution of SAS Enterprise Miner to model implementation is a scoring recipe capable of adding predictions to any data set structured in a manner similar to the training data.

SAS Enterprise Miner offers two options for model implementation.

- *Internally scored data sets* are created by combining the Score tool with a data set identified for scoring.
 - ✎ A copy of the scored data set is stored on the SAS Foundation server assigned to your project. If the data set to be scored is very large, you should consider scoring the data outside the SAS Enterprise Miner environment and use the second deployment option.
- *Scoring code modules* are used to generate predicted target values in environments outside of SAS Enterprise Miner. SAS Enterprise Miner can create scoring code in the SAS, C, and Java programming languages. The SAS language code can be embedded directly into a SAS Foundation application to generate predictions. The C and Java language code must be compiled. The C code should compile with any C compiler that supports the ISO/IEC 9899 International Standard for Programming Languages -- C.

7.2 Internally Scored Data Sets

To create an internally scored data set, you need to define a Score data source, integrate the Score data source and Score tool into your process flow diagram, and (optionally) relocate the scored data set to a library of your choice.



Creating a Score Data Source

Creating a score data source is similar to creating a modeling data source.

1. Right-click **Data Sources** in the Project panel and select **Create Data Source**. The Data Source Wizard opens.
2. Select the table **ScorePVA97NK** in the AAEM library.
3. Select **Next >** until you reach Data Source Wizard -- Step 6 of 7 Data Source Attributes.
4. Select **Score** as the role.

The screenshot shows the 'Data Source Wizard -- Step 6 of 7 Data Source Attributes' dialog box. On the left is a vertical sidebar with a large orange 'i' icon and a blue arrow pointing down, along with several smaller orange icons. The main area contains the following fields:

- Name :** SCOREPVA97NK
- Role :** Score (selected in a dropdown menu)
- Segment :** (empty text box)
- Notes :** (empty text box)

At the bottom right are three buttons: '< Back', 'Next >', and 'Cancel'.

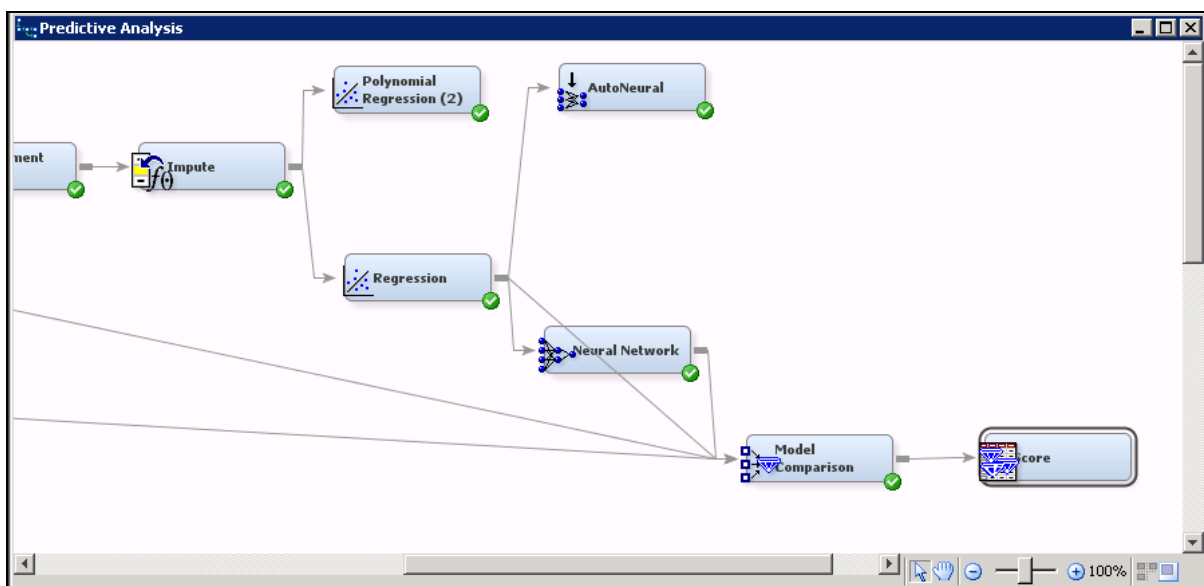
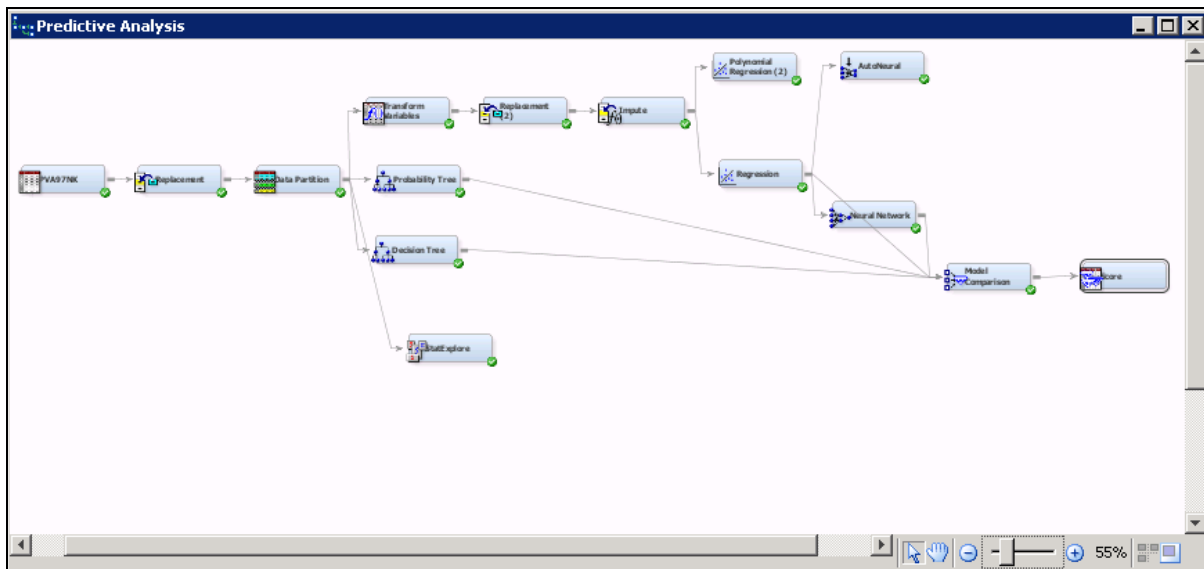
5. Select **Next >** to move to Step 7.
6. Select **Finish**. You now have a data source that is ready for scoring.



Scoring with the Score Tool

The Score tool attaches model predictions from a selected model to a score data set.

1. Select the **Assess** tab.
2. Drag a **Score** tool into the diagram workspace.
3. Connect the **Model Comparison** node to the **Score** node as shown.

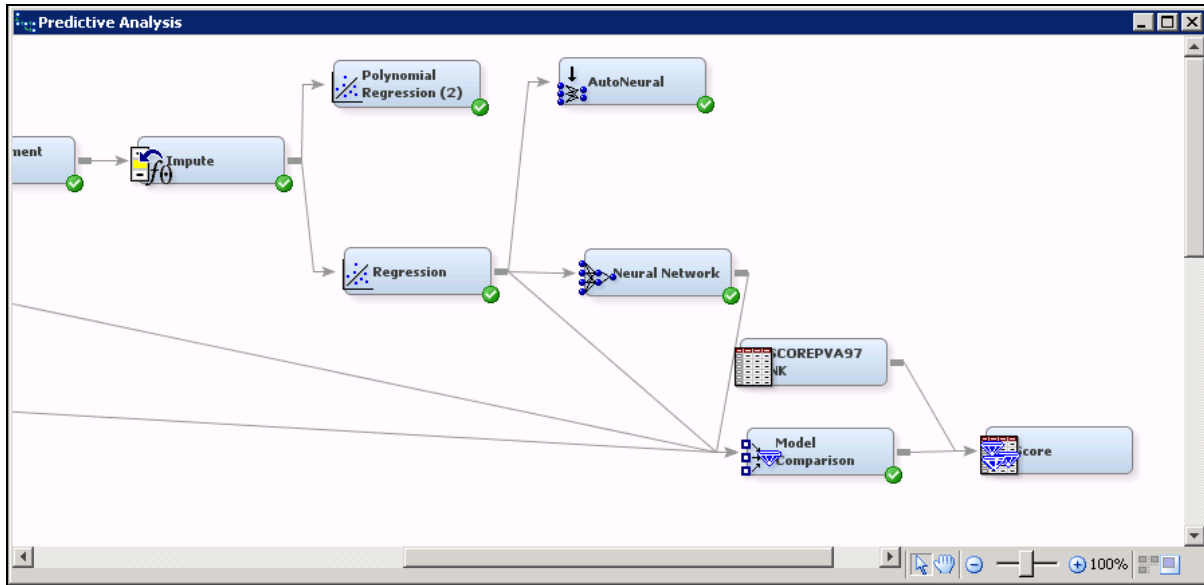


The Score node creates predictions using the model deemed best by the Model Comparison node (in this case, the Regression model).



If you want to create predictions using a specific model, either delete the connection to the Model Comparison node of the models that you do not want to use, or connect the Score node directly to the desired model and continue as described below.

4. Drag the **ScorePVA97NK** data source into the diagram workspace.
5. Connect the **ScorePVA97NK** data source node to the **Score** node as shown.



6. Run the Score node and view the results. The Results - Score window opens.

Optimized SAS Code

```

*-----*
* EM SCORE CODE;
* VERSION: 6.1;
* GENERATED BY: student;
* CREATED: 08JUL2009:16:22:48;
*-----*
* TOOL: Input Data Source;
* TYPE: SAMPLE;
* NODE: Ids;
*-----*
* TOOL: Extension Class;
* TYPE: MODIFY;
* NODE: Repl;
*-----*
* TOOL: Partition Class;
* TYPE: SAMPLE;
* NODE: Part2;
*-----*

```

Output

```

*-----*
User:      Student
Date:      July 08, 2009
Time:      16:22:56
*-----*
* Training Output
*-----*

```

Variable Summary

Role	Measurement Level	Frequency Count
SEGMENT	INTERVAL	1
TARGET	BINARY	1

SAS Code

```

*-----*
* EM SCORE CODE;
* VERSION: 6.1;
* GENERATED BY: student;
* CREATED: 08JUL2009:16:22:48;
*-----*
* TOOL: Input Data Source;
* TYPE: SAMPLE;
* NODE: Ids;
*-----*
* TOOL: Extension Class;
* TYPE: MODIFY;
* NODE: Repl;
*-----*
* ;
* Variable: DemMedIncome ;
* ;
Label DEP_DemMedIncome = #DemMedIncome;

```

Output Variables

Variable Name	Creator	Variable Label	Function	Type
D_TARGETB	Reg	Decision: T...	DECISION	C
EM_CLASS...	Score	Prediction f...	CLASSIFIC...	C
EM_EVENT...	Score	Probability f...	PREDICT	N
EM_PROBA...	Score	Probability...	PREDICT	N
EM_SEG...	Score	'	TRANSFORMN	
EP_TARGET...	Reg	Expected Pr...	ASSESS	N
I_TargetB	Reg	Into: TargetB	CLASSIFIC...	C
LOG_GiftAv...	Trans		TRANSFORMN	
LOG_GiftC...	Trans		TRANSFORMN	
P_TargetB0	Reg	Predicted: T...	PREDICT	N
P_TargetB1	Reg	Predicted: T...	PREDICT	N
U_TargetB	Reg	Unnormaliz...	CLASSIFIC...	N
_WARN...	Reg	Warnings	ASSESS	C
b_TargetB	MdlComp		TRANSFORMN	

7. Maximize the Output window.

The item of greatest interest is a table of new variables added to the Score data set.


8. Go to line 150.







Score Output Variables			
Variable Name	Function	Creator	Label
D_TARGETB	DECISION	Reg	Decision: TargetB
EM_CLASSIFICATION	CLASSIFICATION	Score	Prediction for TargetB
EM_EVENTPROBABILITY	PREDICT	Score	Probability for level 1 of TargetB
EM_PROBABILITY	PREDICT	Score	Probability of Classification
EM_SEGMENT	TRANSFORM	Score	'
EP_TARGETB	ASSESS	Reg	Expected Profit: TargetB
I_TargetB	CLASSIFICATION	Reg	Into: TargetB
LOG_GiftAvgAll	TRANSFORM	Trans	
LOG_GiftCnt36	TRANSFORM	Trans	
P_TargetB0	PREDICT	Reg	Predicted: TargetB=0
P_TargetB1	PREDICT	Reg	Predicted: TargetB=1
U_TargetB	CLASSIFICATION	Reg	Unnormalized Into: TargetB
WARN	ASSESS	Reg	Warnings
b_TargetB	TRANSFORM	MdlComp	

9. Close the Results window.






Exporting a Scored Table (Self-Study)

1. Select **Exported Data** ⇨  from the Score node Properties panel.

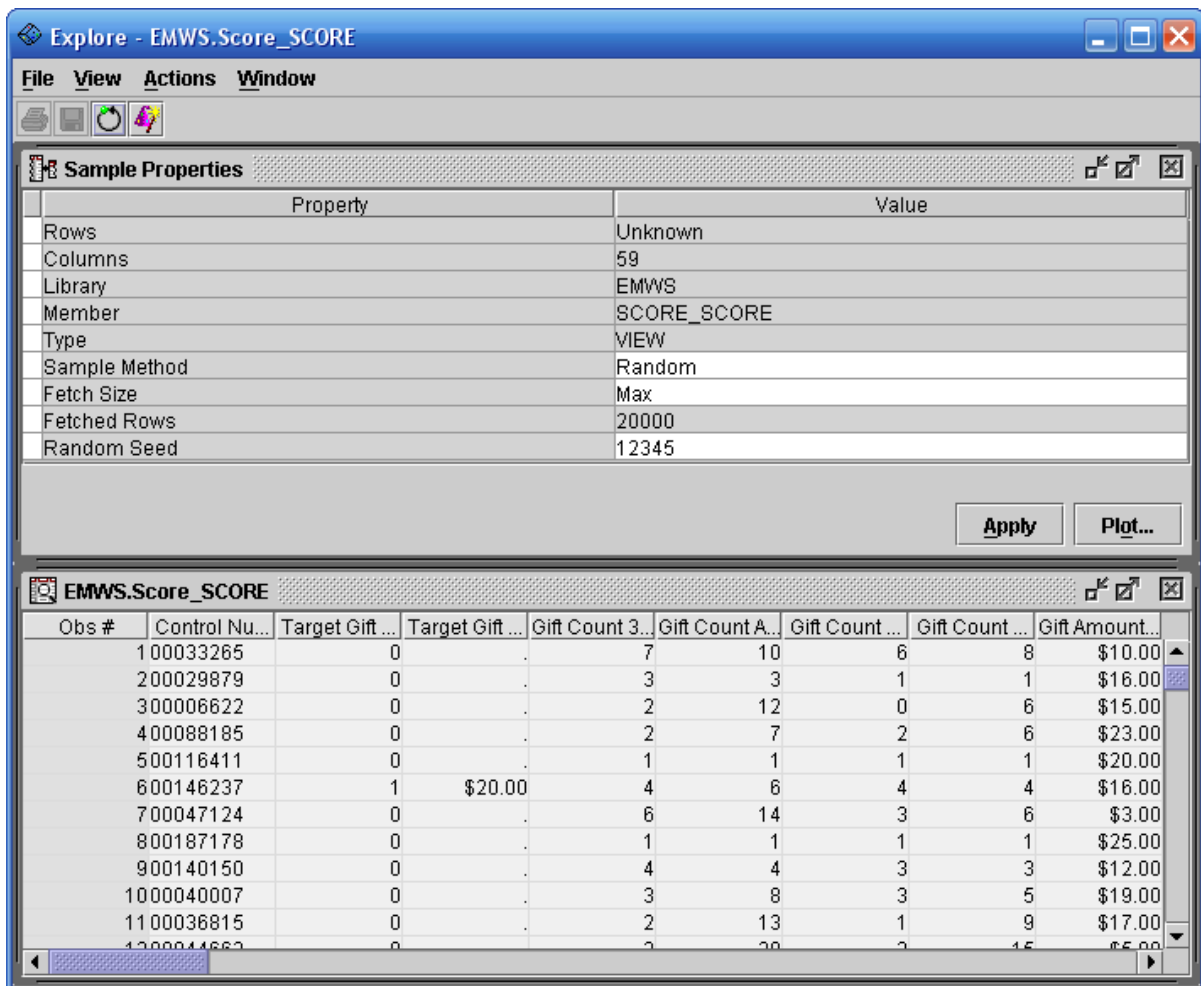
Property	Value
General	
Node ID	Score
Imported Data	
Exported Data	 
Notes	
Train	
Variables	
Type of Scored Data	View
Use Fixed Output Name	Yes
Hide Variables	No
Hide Selection	

The Exported Data - Score dialog box opens.

Exported Data - Score			
Port	Table	Role	Data Exists
TRAIN	EMWS.Score_TRAIN	Train	Yes
VALIDATE	EMWS.Score_VALIDATE	Validate	Yes
TEST	EMWS.Score_TEST	Test	No
SCORE	EMWS.Score_SCORE	Score	Yes
<div>     </div>			

2. Select the **Score** table.

3. Select **Explore...**. The Explore window opens with a 20,000-row sample of the **EMWS2.Score_SCORE** data set (the internal name in SAS Enterprise Miner for the scored **ScorePVA97NK** data).



The screenshot shows the SAS Explore window titled "Explore - EMWS.Score_SCORE". The window has a menu bar (File, View, Actions, Window) and a toolbar. Below the toolbar is the "Sample Properties" tab, which displays a table of properties and values. Below this is the "EMWS.Score_SCORE" tab, which displays a data table with 10 columns and 13 rows of data.

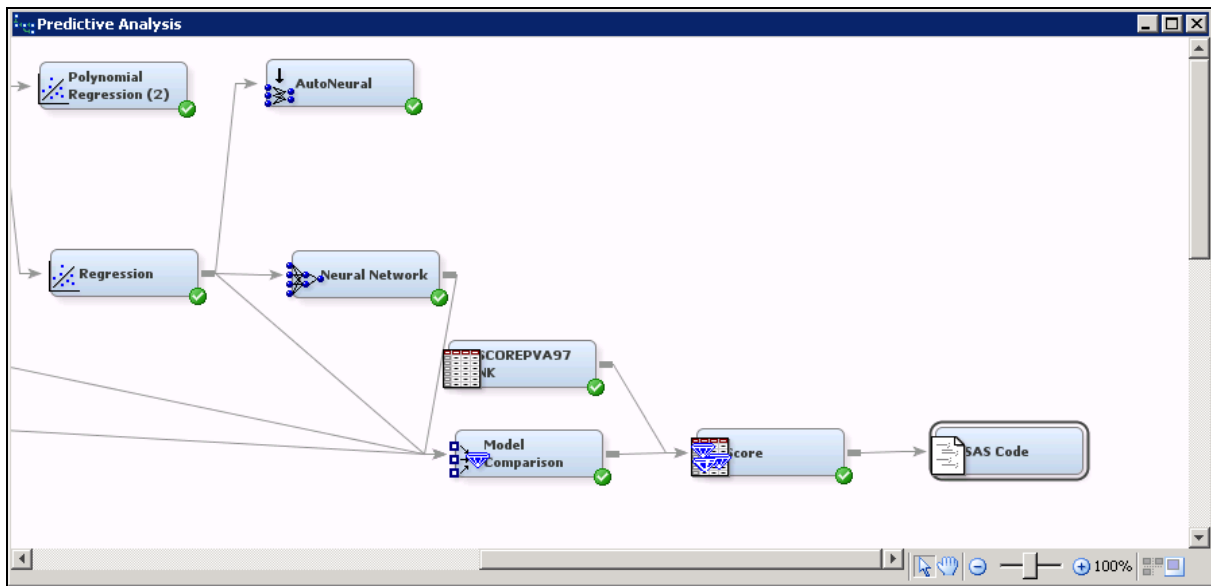
Property	Value
Rows	Unknown
Columns	59
Library	EMWS
Member	SCORE_SCORE
Type	VIEW
Sample Method	Random
Fetch Size	Max
Fetches Rows	20000
Random Seed	12345


Obs #	Control Nu...	Target Gift ...	Target Gift ...	Gift Count 3...	Gift Count A...	Gift Count ...	Gift Count ...	Gift Amount...
100033265		0	.	7	10	6	8	\$10.00
200029879		0	.	3	3	1	1	\$16.00
300006622		0	.	2	12	0	6	\$15.00
400088185		0	.	2	7	2	6	\$23.00
500116411		0	.	1	1	1	1	\$20.00
600146237		1	\$20.00	4	6	4	4	\$16.00
700047124		0	.	6	14	3	6	\$3.00
800187178		0	.	1	1	1	1	\$25.00
900140150		0	.	4	4	3	3	\$12.00
1000040007		0	.	3	8	3	5	\$19.00
1100036815		0	.	2	13	1	9	\$17.00
1200044662		0	.	2	20	2	15	\$5.00

This scored table is situated on the SAS Foundation server in the current project directory. You might want to place a copy of this table in another location. The easiest way to do this is with a SAS code node.

4. Close the Explore window.
5. Select the **Utility** tab.
6. Drag a **SAS Code** node into the Diagram workspace.

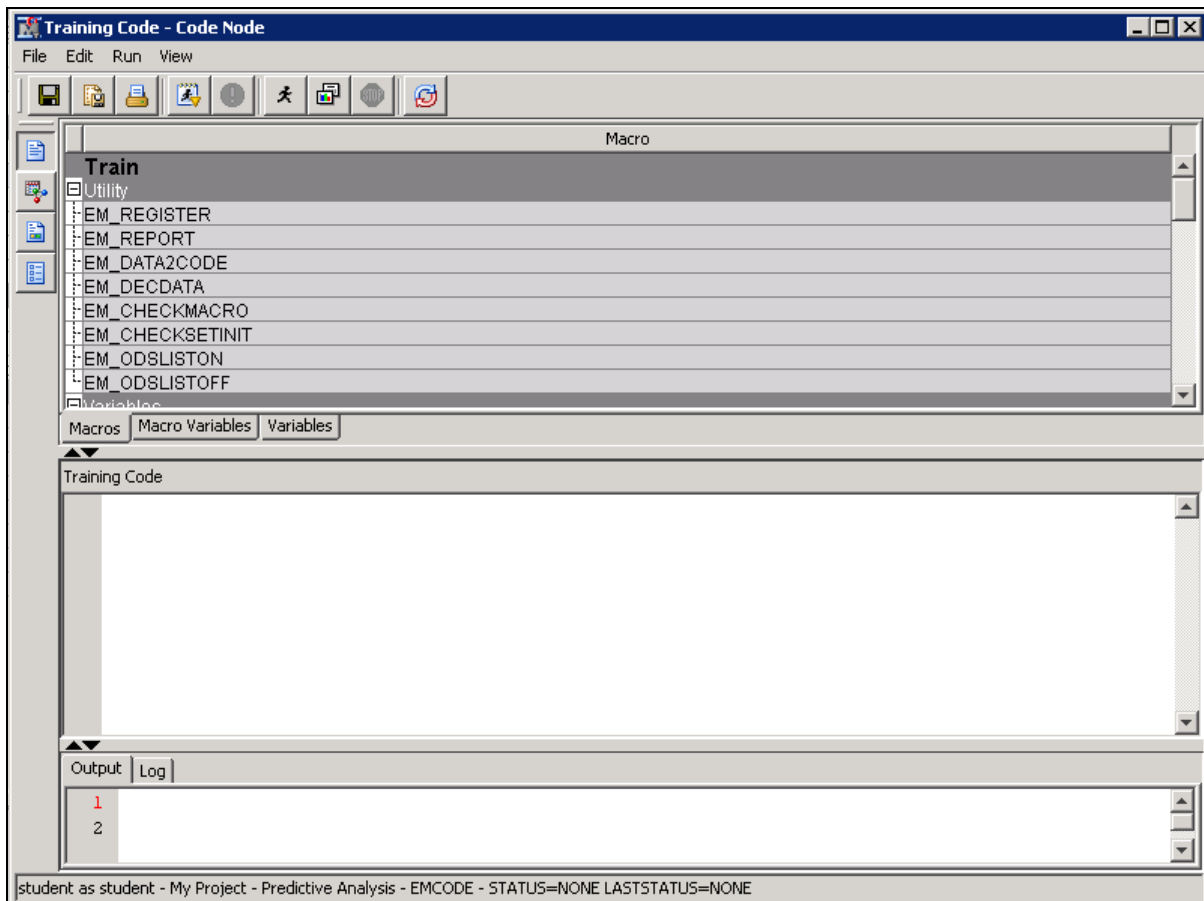
7. Connect the **Score** node to the **SAS Code** node as shown.



8. Select **Code Editor** ⇒  in the SAS Code node's Properties panel.

Property	Value
General	
Node ID	EMCODE
Imported Data	...
Exported Data	...
Notes	...
Train	
Variables	...
Code Editor	...
Tool Type	Utility
Data Needed	No
Rerun	No
Use Priors	Yes
Score	
Advisor Type	Basic
Publish Code	Publish
Code Format	Datastep

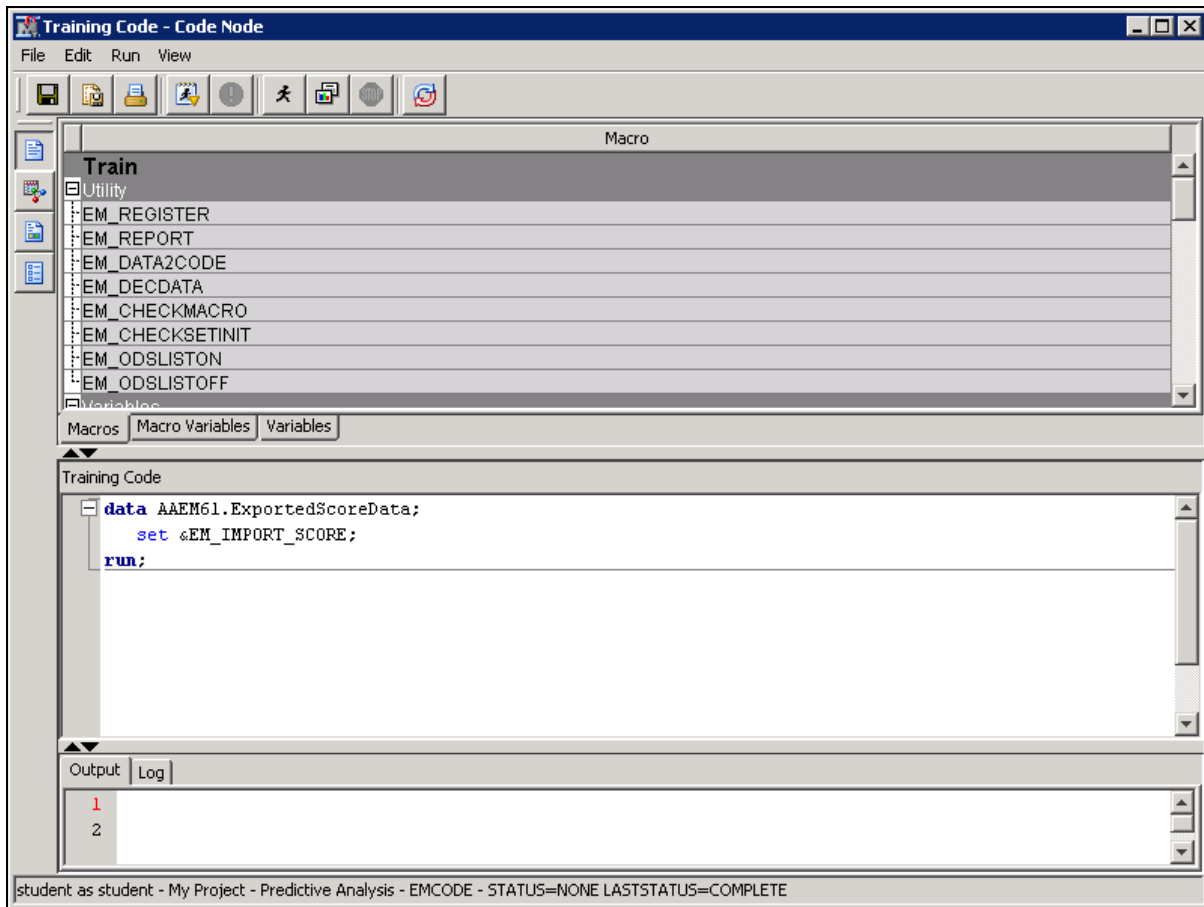
The Training Code window opens.



The Training Code window enables you to add new functionality to SAS Enterprise Miner by accessing the scripting language of SAS.

9. Type the following program in the Training Code window:

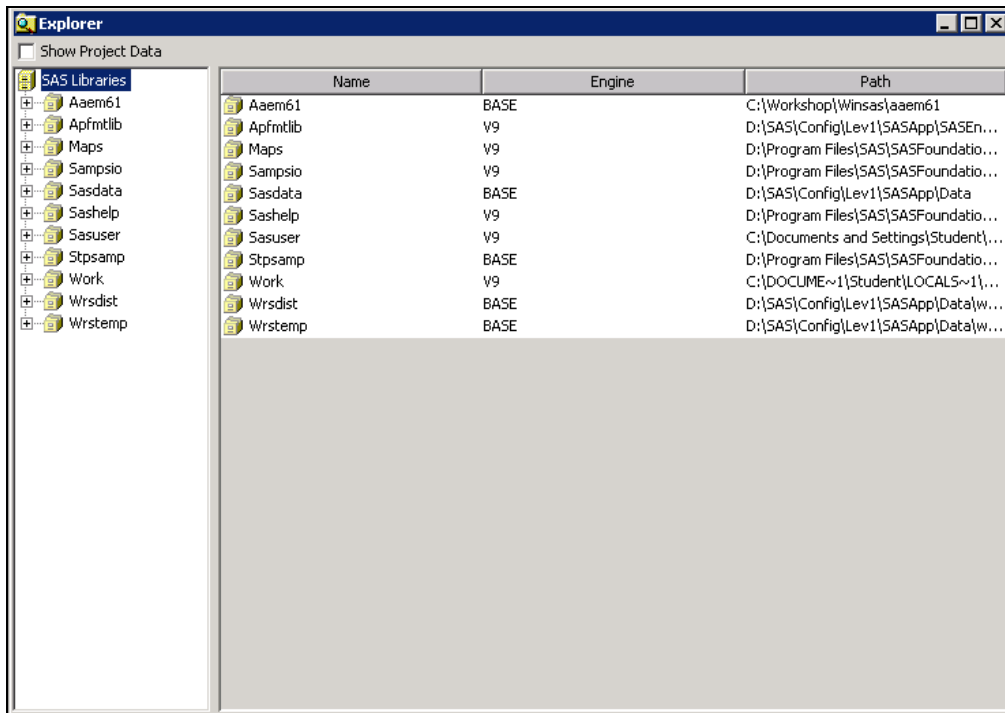
```
data AAEM61.ExportedScoreData;  
    set &EM_IMPORT_SCORE;  
run;
```



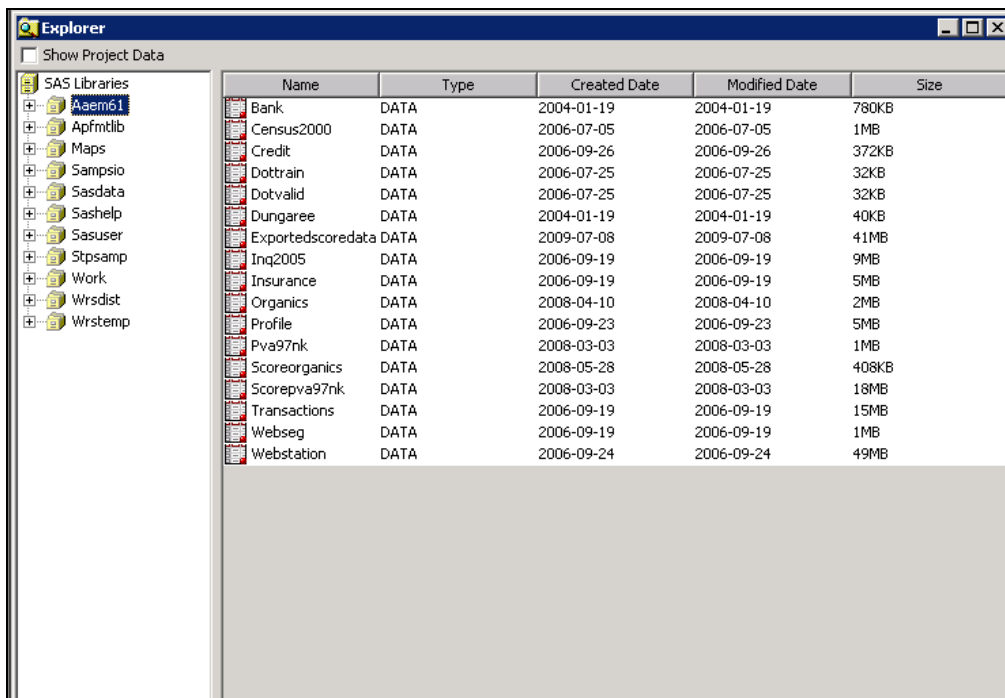
This program creates a new table named **ExportedScoreData** in the AAEM61 library.

10. Close the Training Code window and save the changes.
11. Run the SAS Code node. You do not need to view the results.

12. Select **View** ⇒ **Explorer...** from the SAS Enterprise Miner menu bar. The Explorer window opens.



13. Select the **Aaem61** library.



The Aaem61 library contains the **Exportedscoredata** table created by your SAS Code node. You can modify the SAS Code node to place the scored data in any library that is visible to the SAS Foundation server.

14. Close the Explorer window.

7.3 Score Code Modules

Model deployment usually occurs outside of SAS Enterprise Miner and sometimes even outside of SAS. To accommodate this need, SAS Enterprise Miner is designed to provide score code modules to create predictions from properly prepared tables. In addition to the prediction code, the score code modules include all the transformations that are found in your modeling process flow. You can save the code as a SAS, C, or Java program.



Creating a SAS Score Code Module

The SAS Score Code module is opened by default when you open the Score node.

1. Open the Score node Results window.
2. Maximize the SAS Code window.

The SAS Code window shows the SAS DATA step code that is necessary to append predictions from the selected model (in this case, the Regression Model) to a score data set. Each node in the process flow can contribute to the DATA step code. The following list describes some highlights of the generated SAS code:

- Go to line 12. This code removes the spurious zero from the median income input.

```
*-----*;  
* TOOL: Extension Class;  
* TYPE: MODIFY;  
* NODE: Repl;  
*-----*;  
* ;  
* Variable: DemMedIncome ;  
* ;  
Label REP_DemMedIncome = "Replacement: DemMedIncome " ;  
REP_DemMedIncome =DemMedIncome ;  
if DemMedIncome ne . and DemMedIncome <1 then REP_DemMedIncome = . ;
```

- Go to line 36. This code takes the log transformation of selected inputs.

```
*-----*;  
* TRANSFORM: GiftAvg36 , log(GiftAvg36 + 1);  
*-----*;  
label LOG_GiftAvg36 = 'Transformed: Gift Amount Average 36 Months';  
if GiftAvg36 + 1 > 0 then LOG_GiftAvg36 = log(GiftAvg36 + 1);  
else LOG_GiftAvg36 = .;  
*-----*;  
* TRANSFORM: GiftAvgAll , log(GiftAvgAll + 1);  
*-----*;  
label LOG_GiftAvgAll = 'Transformed: Gift Amount Average All Months';  
if GiftAvgAll + 1 > 0 then LOG_GiftAvgAll = log(GiftAvgAll + 1);  
else LOG_GiftAvgAll = .;  
.  
.  
.
```


- Go to line 84. This code replaces the levels of the StatusCat96NK input.

```

*-----*;
* TOOL: Extension Class;
* TYPE: MODIFY;
* NODE: Repl2;
*-----*;

* ;
* Defining New Variables;
* ;
Length REP_StatusCat96NK $5;
Label REP_StatusCat96NK = "Replace:Status Category 96NK";
REP_StatusCat96NK= StatusCat96NK;

* ;
* Replace Specific Class Levels ;
* ;
length _UFormat200 $200;
drop _UFORMAT200;
_UFORMAT200 = " ";
* ;
* Variable: StatusCat96NK;
* ;
if strip(StatusCat96NK) = "A" then
REP_StatusCat96NK="A";
if strip(StatusCat96NK) = "S" then
REP_StatusCat96NK="A";
if strip(StatusCat96NK) = "F" then
REP_StatusCat96NK="N";
if strip(StatusCat96NK) = "N" then
REP_StatusCat96NK="N";
if strip(StatusCat96NK) = "E" then
REP_StatusCat96NK="L";
if strip(StatusCat96NK) = "L" then
REP_StatusCat96NK="L";

```

- Go to line 118. This code replaces missing values and creates missing value indicators.

```

*-----*;
* TOOL: Imputation;
* TYPE: MODIFY;
* NODE: Impt;
*-----*;
*;
*MEAN-MEDIAN-MIDRANGE AND ROBUST ESTIMATES;
*;
label IMP_DemAge = 'Imputed: Age';
IMP_DemAge = DemAge;
if DemAge = . then IMP_DemAge = 59.262912087912;
label IMP_LOG_GiftAvgCard36 = 'Imputed: Transformed: Gift Amount Average Card 36 Months';
IMP_LOG_GiftAvgCard36 = LOG_GiftAvgCard36;
if LOG_GiftAvgCard36 = . then IMP_LOG_GiftAvgCard36 = 2.5855317177381;
label IMP_REP_DemMedIncome = 'Imputed: Replacement: DemMedIncome';
IMP_REP_DemMedIncome = REP_DemMedIncome;
if REP_DemMedIncome = . then IMP_REP_DemMedIncome = 53570.8504928806;
*;
*INDICATOR VARIABLES;
*;
label M_DemAge = "Imputation Indicator for DemAge";
if DemAge = . then M_DemAge = 1;
else M_DemAge= 0;
label M_LOG_GiftAvgCard36 = "Imputation Indicator for LOG_GiftAvgCard36";
if LOG_GiftAvgCard36 = . then M_LOG_GiftAvgCard36 = 1;
else M_LOG_GiftAvgCard36= 0;
label M_REP_DemMedIncome = "Imputation Indicator for REP_DemMedIncome";
if REP_DemMedIncome = . then M_REP_DemMedIncome = 1;
else M_REP_DemMedIncome= 0;

```

- Go to line 160. This code comes from the Regression node. It is this code that actually adds the predictions to a Score data set.

```

*-----*
* TOOL: Regression;
* TYPE: MODEL;
* NODE: Reg;
*-----*
*****
*** begin scoring code for regression;
*****

length _WARN_ $4;
label _WARN_ = 'Warnings' ;

length I_TargetB $ 12;
label I_TargetB = 'Into: TargetB' ;
*** Target Values;
array REGDRF [2] $12 _temporary_ ('1' '0' );
label U_TargetB = 'Unnormalized Into: TargetB' ;
*** Unnormalized target values;
ARRAY REGDRU[2] _TEMPORARY_ (1 0);

drop _DM_BAD;
_DM_BAD=0;

*** Check DemMedHomeValue for missing values ;
if missing( DemMedHomeValue ) then do;
    substr(_warn_,1,1) = 'M';
    _DM_BAD = 1;
end;

*** Check GiftTimeLast for missing values ;
if missing( GiftTimeLast ) then do;
    substr(_warn_,1,1) = 'M';
    _DM_BAD = 1;
end;

.
.
.

```

- Go to line 290. This block of code comes from the Model Comparison node. It adds demi-decile bin numbers to the scored output. For example, bin 1 corresponds to the top 5% of the data as scored by the Regression model, bin 2 corresponds to the next 5%, and so on.

```
*-----*;  
* TOOL: Model Compare Class;  
* TYPE: ASSESS;  
* NODE: MdlComp;  
*-----*;  
if (P_TargetB1 ge 0.09198928166881) then do;  
  b_TargetB = 1;  
end;  
else  
  if (P_TargetB1 ge 0.08014055773524) then do;  
    b_TargetB = 2;  
  end;  
else  
  if (P_TargetB1 ge 0.07027014765811) then do;  
    b_TargetB = 3;  
  end;  
  .  
  .  
  .
```

- Go to line 380. This block of code comes from the Score node. It adds the following standardized variables to the scored data set:

EM_CLASSIFICATION	Prediction for TargetB
EM_DECISION	Recommended Decision for TargetB
EM_EVENTPROBABILITY	Probability for Level 1 of Target
EM_PROBABILITY	Probability of Classification
EM_PROFIT	Expected Profit for TargetB
EM_SEGMENT	Segment

```

*-----*;
* TOOL: Score Node;
* TYPE: ASSESS;
* NODE: Score;
*-----*;
*-----*;
* Score: Creating Fixed Names;
*-----*;
LABEL EM_SEGMENT = 'Segment';
EM_SEGMENT = b_TargetB;
LABEL EM_EVENTPROBABILITY = 'Probability for level 1 of TargetB';
EM_EVENTPROBABILITY = P_TargetB1;
LABEL EM_PROBABILITY = 'Probability of Classification';
EM_PROBABILITY = max(
P_TargetB1
,
P_TargetB0
);
.
.
.

```



To use this code, you must embed it in a DATA step. The easiest way to do this is by saving it to as a SAS code file and including it in your DATA step.

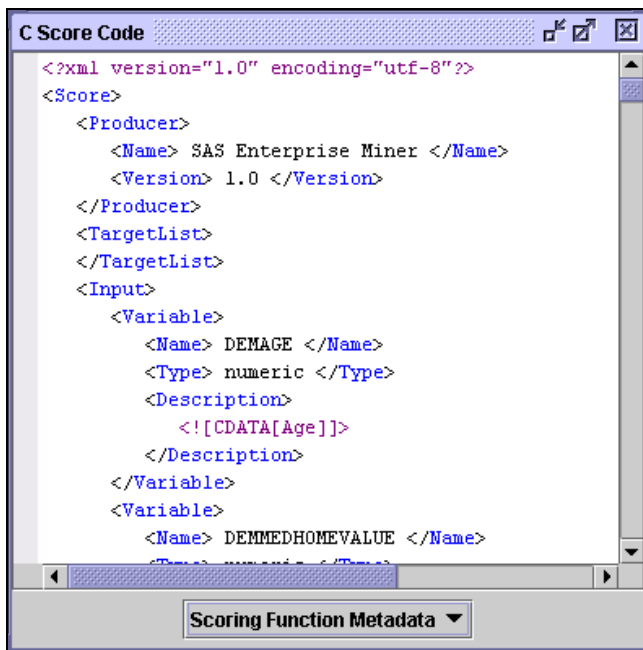
- Select **File** ⇒ **Save As...** to save this code to a location of your choice.



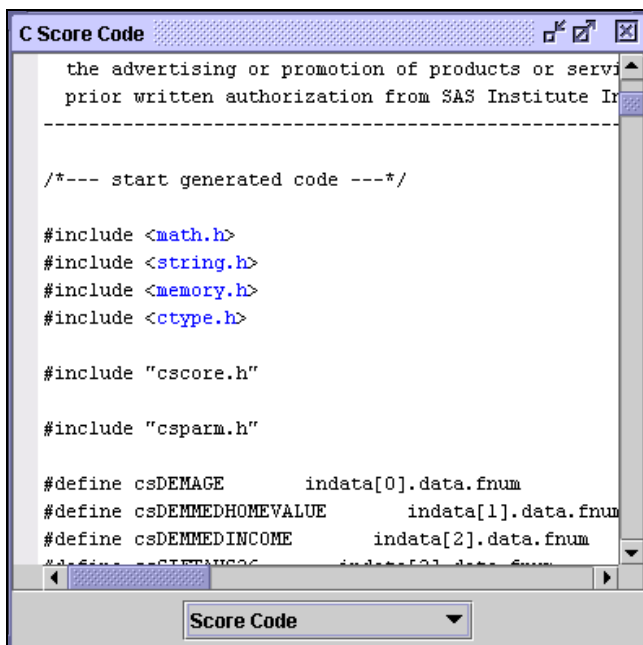
Creating Other Score Code Modules

To access scoring code for languages other than SAS, use the following procedure:

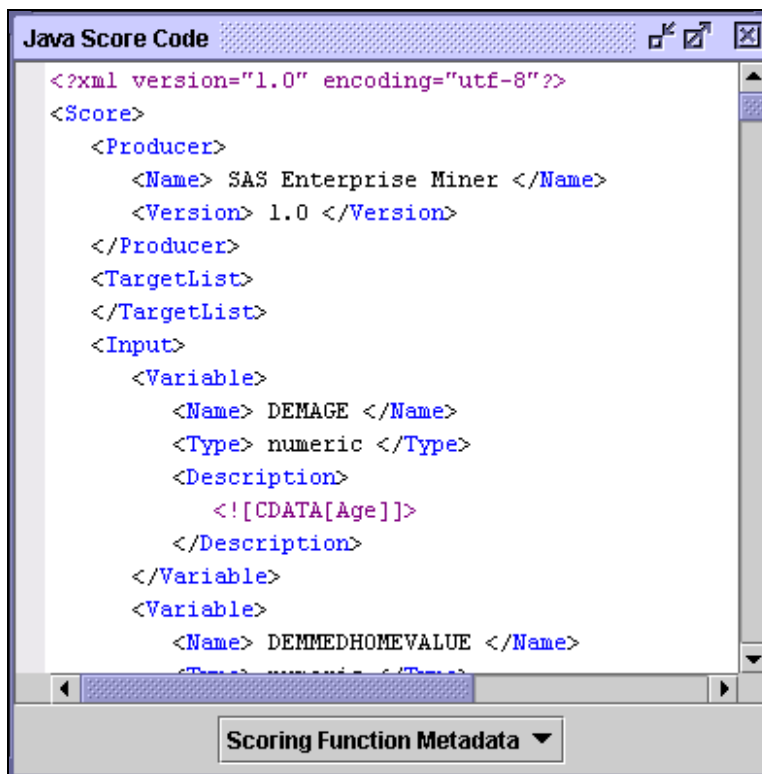
1. Select **View** ⇒ **Scoring** ⇒ **C Score Code**. The C Score Code window opens.



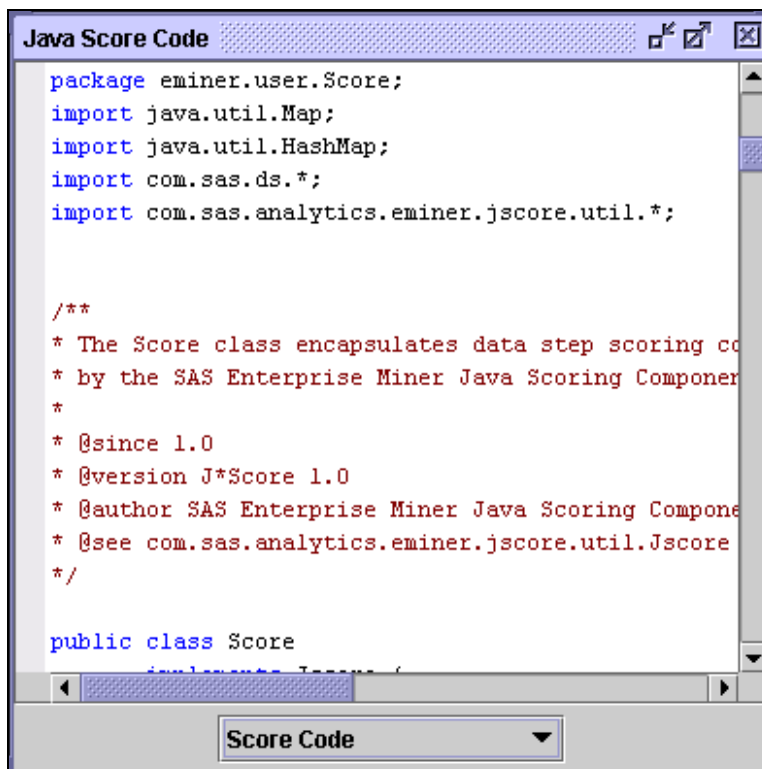
There are four parts to the C Score Code window. The actual C score code part is accessible from the menu at the bottom of the C Score Code window.



2. Select **View** ⇒ **Scoring** ⇒ **Java Score Code**. The Java Score Code window opens.



There are five parts to the Java Score Code window. The actual Java score code part is accessible from the menu at the bottom of the Java Score Code window.





Exercises

1. Scoring Organics Data

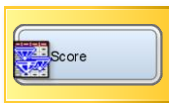
- a. Create a Score data source for the **ScoreOrganics** data.
- b. Score the **ScoreOrganics** data using the model selected with the Model Comparison node.

7.4 Chapter Summary

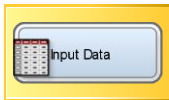
The Score node is used to score new data inside SAS Enterprise Miner and to create scoring modules for use outside SAS Enterprise Miner. The Score node adds predictions to any data source with a role of Score. This data source must have the same inputs as the training data. A scored data source is stored within the project directory. You can use a SAS Code tool to relocate it to another library.

The Score tool creates score code modules in the SAS, C, and Java languages. These score code modules can be saved and used outside of SAS Enterprise Miner.

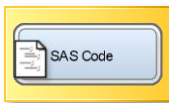
Model Implementation Tools Review



Add predictions to Score data sources; and create SAS, C, and Java score code modules.



Create a Score data source.



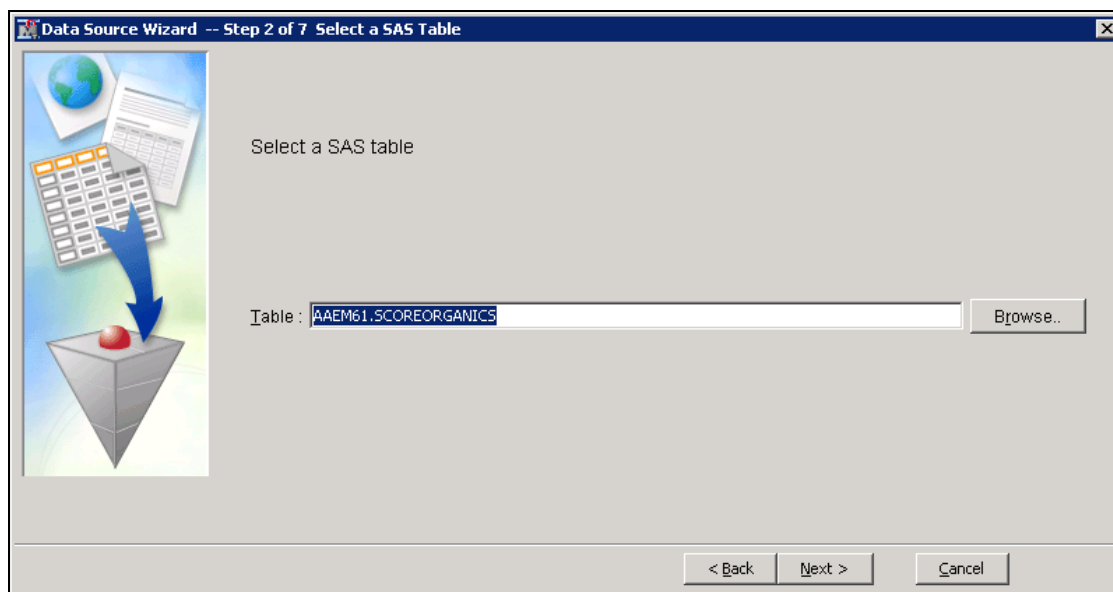
Use SAS scripting language to export scored data outside a SAS Enterprise Miner project.

7.5 Solutions to Exercises

1. Scoring Organics Data

a. Create a Score data source for the **ScoreOrganics** data.

- 1) Select **File** ⇒ **New** ⇒ **Data Source**.
- 2) Proceed to Step 2 of the Data Source Wizard by selecting **Next >**.
- 3) Select the **AAEM61.SCOREORGANICS** data set.



- 4) Proceed to the final step of the Data Source Wizard.

- 5) Select **Score** as the role.

Data Source Wizard -- Step 6 of 7: Data Source Attributes

You may change the name and the role, and can specify a population segment identifier for the data source to be created.

Name : SCOREORGANICS

Role : Score

Segment :

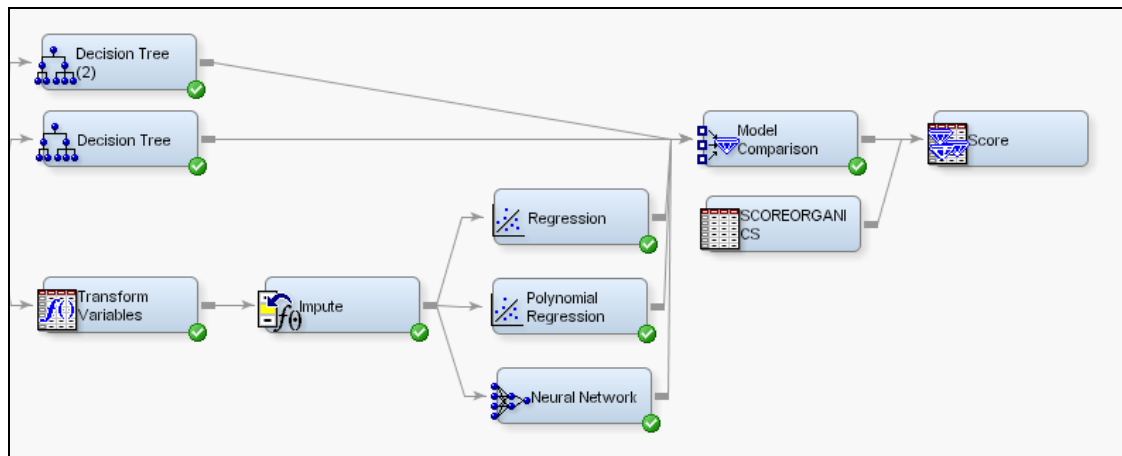
Notes :

< Back Next > Cancel

- 6) Select **Finish**.

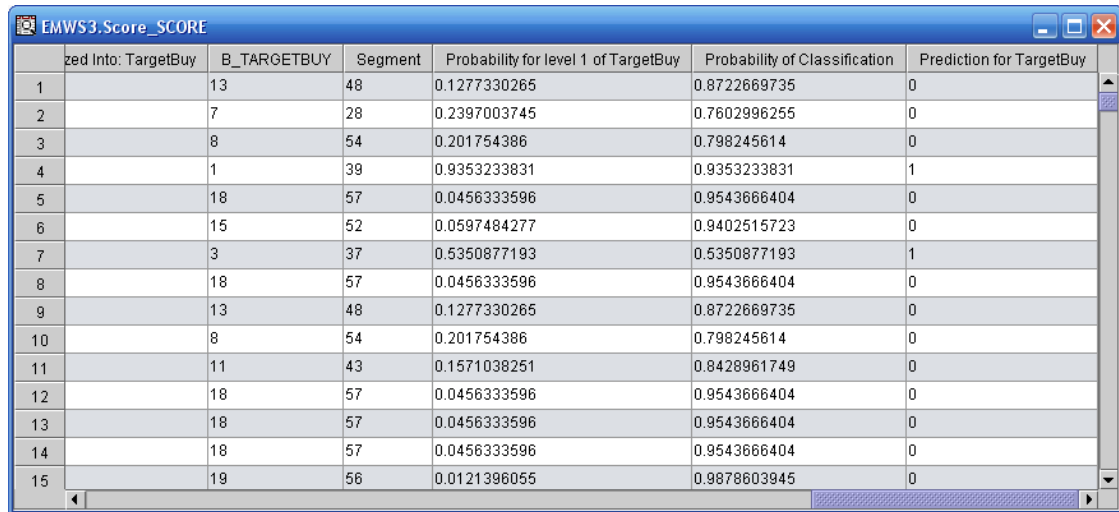
- b. Score the **ScoreOrganics** data using the model selected with the Model Comparison node.

- 1) Connect a **Score** tool to the **Model Comparison** node.
- 2) Connect a **ScoreOrganics** data source to the **Score** node.



- 3) Run the Score node.

4) Browse the Exported data from the Score node to confirm the scoring process.



	zed Into: TargetBuy	B_TARGETBUY	Segment	Probability for level 1 of TargetBuy	Probability of Classification	Prediction for TargetBuy
1		13	48	0.1277330265	0.8722669735	0
2		7	28	0.2397003745	0.7602996255	0
3		8	54	0.201754386	0.798245614	0
4		1	39	0.9353233831	0.9353233831	1
5		18	57	0.0456333596	0.9543666404	0
6		15	52	0.0597484277	0.9402515723	0
7		3	37	0.5350877193	0.5350877193	1
8		18	57	0.0456333596	0.9543666404	0
9		13	48	0.1277330265	0.8722669735	0
10		8	54	0.201754386	0.798245614	0
11		11	43	0.1571038251	0.8428961749	0
12		18	57	0.0456333596	0.9543666404	0
13		18	57	0.0456333596	0.9543666404	0
14		18	57	0.0456333596	0.9543666404	0
15		19	56	0.0121396055	0.9878603945	0

A successfully scored data set features predicted probabilities and prediction decisions in the last three columns.