Chapter 2
Introduction to SAS Software

In this chapter, we will briefly overview the various types of SAS software that can be useful for credit risk modeling. It is not our aim to provide an exhaustive discussion on all functionality and options available, but rather to give a quick introduction on how to get started with each of the solutions. For more detailed information, we refer to the SAS website (http://www.sas.com), SAS training and books (http://support.sas.com/learn/), and SAS support (http://support.sas.com/).

SAS VERSUS OPEN SOURCE SOFTWARE

The popularity of open source analytical software such as R and Python has sparked the debate about the added value of SAS, which is a commercial tool. In fact, both commercial software as well as open source software have their merits, which should be thoroughly evaluated before any analytical software decision is made.

First of all, the key advantage of open source software is that it is obviously available for free, which significantly lowers the entry barrier to use it. However, this clearly poses a danger as well, since anyone can contribute to it without any quality assurance or extensive prior testing. In heavily regulated environments such as credit risk (e.g., Basel Accord), insurance (e.g., Solvency Accord, XXX and AXXX reserving) and pharmaceutics (e.g., Food and Drug Administration regulation), the analytical models are subject to external supervisory review because of their strategic impact to society, which is now bigger than ever before. Hence, in these settings many firms prefer to rely on mature commercial solutions that have been thoroughly engineered, extensively tested, validated, and documented. Many of these solutions also include automatic reporting facilities to generate compliance reports in each of the settings mentioned. Open source software solutions do not come with any kind of quality control or warranty, which increases the risk when using them in a regulated environment.

Another key advantage of commercial software like SAS is that the software offered is no longer centered on dedicated analytical workbenches such as data preprocessing and data mining, but on well-engineered business-focused solutions that automate the end-to-end activities. As an example, consider credit risk modeling, which starts from framing the business problem and continues to data preprocessing, analytical model development, backtesting and benchmarking, stress testing, and regulatory capital calculation. To automate this entire chain of activities using open source software would require various scripts, likely originating from heterogeneous sources, to be matched and connected together, resulting in a possible melting pot of software in which the overall functionality could become unstable and/or unclear.

Contrary to open source software, commercial software vendors also offer extensive help

facilities such as FAQs, technical support hot lines, newsletters, and professional training courses. Another key advantage of commercial software vendors is business continuity—more specifically, the availability of centralized research and development (R&D) teams (as opposed to worldwide, loosely connected open source developers) who follow up on new analytical and regulatory developments. This provides a better guarantee that new software upgrades will provide the facilities required. In an open source environment, you would need to rely on the community to voluntarily contribute, which provides less of a guarantee.

A general disadvantage of commercial software is that it usually comes in prepackaged, black box routines (e.g., the PROCs in Base SAS), which, although extensively tested and documented, cannot be inspected by the more sophisticated data scientist. This is in contrast to open source solutions, which provide full access to the source code of each of the scripts contributed. To address this issue, SAS offers multiple programming environments within statistical procedures and the DATA step environment so that users can self-program applications, including estimation, simulation, and forecasting procedures.

Given this discussion, it is clear that both commercial software and open source software have their strengths and weaknesses. It is likely that they will continue to coexist, and interfaces should be provided for them to collaborate, as is the case for both SAS and R/Python.

BASE SAS

Base SAS is a fourth-generation programming language (4 GL) for data access, transformation, and reporting and is the foundation for all other SAS software. It includes the following features:

- A programming language
- A web-based programming interface
- A centralized metadata repository to store data definitions
- A macro facility
- Integration with big data solutions such as Hadoop and MapReduce

You can start SAS 9.4 by clicking Start (in Windows) ⇨ All Programs ⇨ SAS ⇨ SAS 9.4. You then encounter the windows shown in Exhibit 2.1.

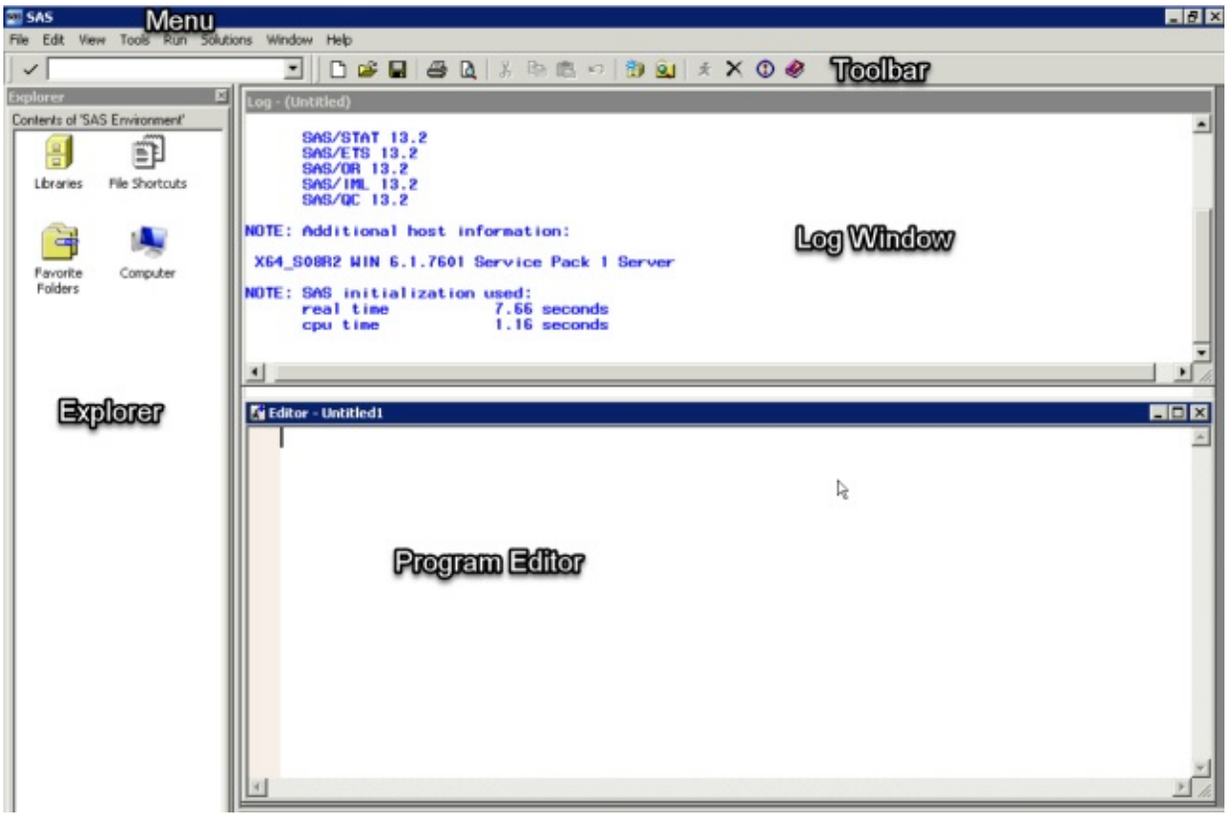

Exhibit 2.1 Start Screen of Base SAS 9.4

The interface is divided into multiple components:

- **Menu**: Here you find all the functionality to edit, view, and run your SAS programs, together with some extra solutions and help functionality.
- **Toolbar**: This provides a selection of shortcut buttons to frequently used menu items.
- **Explorer**: Here you can explore the libraries that you have defined (or that have already been predefined) and browse folders on your computer.
- **Program Editor**: This is where you will write your SAS programs.
- **Log window**: Here you will see errors or warnings appear as you execute your SAS programs.
- **Result window** (hidden behind the Explorer): Summary with links to generated outputs.
- **Output** (hidden behind the Editor): Outputs from procedures.

Let's now enter the following program code in the Program Editor:

```
LIBNAME DATA “C:\Users”;
RUN;
```

This first statement will create a SAS library DATA, which is a shortcut notation to a physical directory on disk (in our case C:\Users). Throughout the book, we capitalize SAS commands and present user input in lowercase letters. Generally speaking, SAS commands end with a semicolon (;) and code sections with either the RUN; command for Base SAS, the QUIT

command for PROC IML (see later discussion), or the %MEND command for macros (see later discussion).

SAS allows performing data manipulation using data steps:

```
DATA example;
SET data.mortgage;
/*Example for deletion of observations*/
IF FICO_orig_time< 500 THEN DELETE;
/*Example for generation of new variables*/
IF FICO_orig_time> 500 THEN FICO_cat=1;
IF FICO_orig_time> 700 THEN FICO_cat=2;
/*Example for data filtering*/
WHERE default_time=1;
/*Example for dropping of variables*/
DROP status_time;
RUN;
```

You can run it by clicking the running man icon on the toolbar or selecting Run, Submit from the menu above. Data steps start with the DATA command for the name of the new data set, followed by the SET command for the name of the original data set. The preceding code changes the original data set data.mortgage and generates a new data set mortgage by deleting observations, generating new variables, filtering the data, and dropping a variable.

Furthermore, SAS offers a set of built-in procedures (PROC …). The following statement computes the mean, standard deviation, minimum, and maximum for the variables default_time, FICO_orig_time, ltv_orig_time, and gdp_time of the mortgage data set:

```
PROC MEANS DATA=data.mortgage;
VAR default_time FICO_orig_time ltv_orig_time gdp_time;
RUN;
```

The outcome will be as shown in Exhibit 2.2.

Exhibit 2.2 Output PROC MEAN

The MEANS Procedure					
Variable	**N**	**Mean**	**Std Dev**	**Minimum**	**Maximum**
default_time	622489	0.0243506	0.1541354	0	1.0000000
FICO_orig_time	622489	673.6169217	71.7245579	400.0000000	840.0000000
LTV_orig_time	622489	78.9754596	10.1270521	50.1000000	218.5000000
gdp_time	622489	1.3810318	1.9646446	−4.1467109	5.1324642

SAS/STAT

SAS offers a range of statistical procedures. These procedures generally estimate the parameters of parametric and semiparametric models. A first example is the linear regression

model offered by PROC REG:

```
PROC REG DATA=data.mortgage;
MODEL default_time = FICO_orig_time ltv_orig_time gdp_time;
RUN;
```

The resulting parameter estimates in short form are shown in Exhibit 2.3.

Exhibit 2.3 Output PROC REG

The REG Procedure					
Model: MODEL1					
Dependent Variable: default_time					
Parameter Estimates					
Variable	**DF**	**Parameter Estimate**	**Standard Error**	**t Value**	**Pr > |t|**
Intercept	1	0.07957	0.00259	30.71	<.0001
FICO_orig_time	1	−0.00011544	0.00000275	−42.02	<.0001
LTV_orig_time	1	0.00038077	0.00001944	19.59	<.0001
gdp_time	1	−0.00545	0.00009914	−55.02	<.0001

MACROS IN BASE SAS

In SAS, you can define functions within the Macro language. Here you can see a macro that defines input arguments, which are passed to a regression model. Macros commence with the %MACRO command and conclude with the %MEND. The merit of the following macro is that you don't have to replicate PROC REG, as different covariate combinations are explored and you only need to change the variables included:

```
%MACRO example(datain, lhs, rhs);
PROC REG DATA=&datain;
MODEL &lhs = &rhs;
RUN;
%MEND example;
%example(datain=data.mortgage, lhs=default_time,
rhs=FICO_orig_time );
%example(datain=data.mortgage, lhs=default_time,
rhs=FICO_orig_time ltv_orig_time);
%example(datain=data.mortgage, lhs=default_time,
rhs=FICO_orig_time ltv_orig_time gdp_time);
```

The resulting output is shown in Exhibits 2.4, and 2.6.

Exhibit 2.4 Examples Macro 1

The REG Procedure					
Model: MODEL1					
Dependent Variable: default_time					
Parameter Estimates					
Variable	**DF**	**Parameter Estimate**	**Standard Error**	**t Value**	**Pr > |t|**
Intercept	1	0.10289	0.00184	55.85	<.0001
FICO_orig_time	1	−0.00011660	0.00000272	−42.87	<.0001

Exhibit 2.5 Examples Macro 2

The REG Procedure					
Model: MODEL1					
Dependent Variable: default_time					
Parameter Estimates					
Variable	**DF**	**Parameter Estimate**	**Standard Error**	**t Value**	**Pr > |t|**
Intercept	1	0.06835	0.00259	26.39	<.0001
FICO_orig_time	1	−0.00010867	0.00000275	−39.50	<.0001
LTV_orig__time	1	0.00036981	0.00001948	18.98	<.0001

Exhibit 2.6 Examples Macro 3

The REG Procedure					
Model: MODEL1					
Dependent Variable: default_time					
Parameter Estimates					
Variable	**DF**	**Parameter Estimate**	**Standard Error**	**t Value**	**Pr > |t|**
Intercept	1	0.07957	0.00259	30.71	<.0001
FICO_orig_time	1	−0.00011544	0.00000275	−42.02	<.0001
LTV_orig_time	1	0.00038077	0.00001944	19.59	<.0001
gdp_time	1	−0.00545	0.00009914	−55.02	<.0001

SAS OUTPUT DELIVERY SYSTEM (ODS)

SAS offers an output delivery system (ODS) that transforms outputs into SAS data sets that can then be transformed, fed into second-stage processes, or exported to a comma-separated values (CSV) file (and Excel). Here you can see an example for the preceding regression model

where we generate a SAS data set named parameters that includes the parameter estimates of the model. PROC EXPORT exports these parameter estimates into a CSV file in the specified location path 'C:\Users\export.csv'.

```
ODS LISTING CLOSE;
ODS OUTPUT PARAMETERESTIMATES=parameters;
PROC REG DATA=DATA.mortgage;
MODEL default_time = FICO_orig_time ltv_orig_time gdp_time;
RUN;
ODS OUTPUT CLOSE;
ODS LISTING;
PROC EXPORT DATA=parameters REPLACE DBMS=CSV OUTFILE="C:\Users\export.csv";
RUN;
```

SAS/IML

SAS offers its own programming language, IML (Interactive Matrix Language), which is particularly powerful and flexible for matrix operations. The fundamental object of the language is a data matrix. You can use SAS/IML software interactively (at the statement level) to see results immediately, or you can submit blocks of statements or an entire program. It is also possible to encapsulate a series of statements by defining a module that can then be called later to execute all of its statements. Built-in operators and call routines are available to perform complex tasks in numerical linear algebra such as matrix inversion or the computation of eigenvalues. You can define your own functions and subroutines by using SAS/IML modules and perform operations on a single value, or take advantage of matrix operators to perform operations on an entire data matrix.

The SAS/IML language contains statements that enable data management. You can read, create, and update SAS data sets in SAS/IML software without using the DATA step.

You can program with the many features for arithmetic and character expressions in SAS/IML software. SAS/IML allows you to access a wide variety of built-in functions and subroutines designed to make your programming fast, easy, and efficient. Because SAS/IML software is part of the SAS system, you can access SAS data sets or external files with an extensive set of data processing commands for data input and output, and you can edit existing SAS data sets or create new ones.

SAS/IML software has a complete set of control statements, such as DO/END, START/FINISH, iterative DO, IF-THEN/ELSE, GOTO, LINK, PAUSE, and STOP, giving you all of the commands necessary for execution control and program modularization.

As a simple example, consider the following program taken from the SAS manual (SAS Institute Inc. 2015). It implements a numerical algorithm that estimates the square root of a number, accurate to three decimal places. The algorithm is implemented as the function MySqrt that performs the necessary calculations.

```
PROC IML;                    /* begin IML session */
START MySqrt(x);             /* begin module */
```

```
   y = 1;                    /* initialize y */
   DO UNTIL (w<1e-3);         /* begin DO loop */
      z = y;                 /* set z=y */
      y = 0.5#(z+x/z);       /* estimate square root */
      w = ABS(y-z);          /* compute change in estimate */
   END;                      /* end DO loop */
   RETURN(y);                /* return approximation */
FINISH;
t = MySqrt({3,4,7,9});       /* call function MySqrt  */
s = SQRT({3,4,7,9});         /* compare with true values */
diff = t - s;                /* compute differences */
PRINT t s diff;              /* print matrices */
QUIT;
```

As just illustrated, you can then call the MySqrt module to estimate the square root of several numbers given in a matrix literal (enclosed in braces) and print the results, as shown in Exhibit 2.7.

Exhibit 2.7 Output PROC IML

t	s	diff
1.7320508	1.7320508	0
2	2	2.22E−15
2.6457513	2.6457513	4.678E−11
3	3	1.397E−9

In this book, we will use IML primarily in the chapter on correlations, where we implement some numerical routines for estimating correlations and program Monte Carlo simulations for loss distributions, which can conveniently be coded and run via IML. The stress testing chapter also has PROC IML examples for modeling credit portfolio loss distributions.

SAS STUDIO

SAS Studio is a developmental web application for SAS that you access through your web browser. With SAS Studio, you can access your data files, libraries, and existing programs, and write new programs. It is also possible to use the predefined tasks in SAS Studio to generate SAS code. When you run a program or task, SAS Studio processes the SAS code on a SAS server. The SAS server can be a server in a cloud environment, a server in your local environment, or SAS installed on your local machine. After the code is processed, the results are returned to SAS Studio in your browser.

SAS ENTERPRISE MINER

SAS Enterprise Miner is the flagship data mining tool offered by SAS. It is characterized by an easy-to-use, drag-and-drop interface and works using a distributed client/server architecture. It

allows for the building of analytical models using the SEMMA methodology: sampling, exploration, modification, modeling, and assessment. It also provides:

- Open source integration (e.g., with R)
- In-database and in-Hadoop scoring to deal with massive data sets
- Parallelized grid computing
- Support for XML and SAS macros

In what follows, we provide a brief overview of how to work with SAS Enterprise Miner.

Start Enterprise Miner by clicking Start (in Windows) ⇨ All Programs ⇨ SAS ⇨ SAS Enterprise Miner Client 14.1. Enter the user name and password as shown in Exhibit 2.8.

Exhibit 2.8 Log on Screen of SAS Enterprise Miner

Click Log On to arrive at the screen shown in Exhibit 2.9.

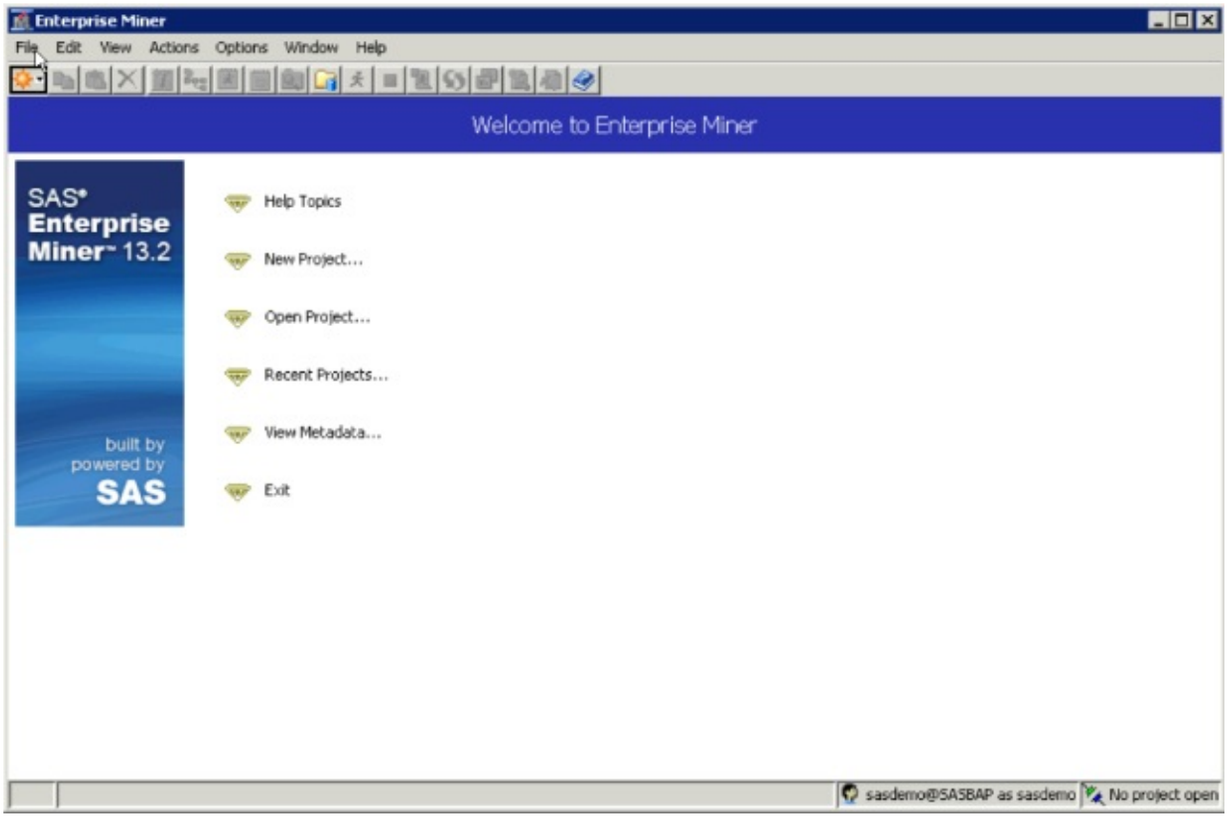

Exhibit 2.9 Welcome Screen of SAS Enterprise Miner

We can choose to create a new project, open an existing or a recent project, inspect metadata that was defined earlier, or simply exit. Let's create a new project with the details shown in Exhibit 2.10.

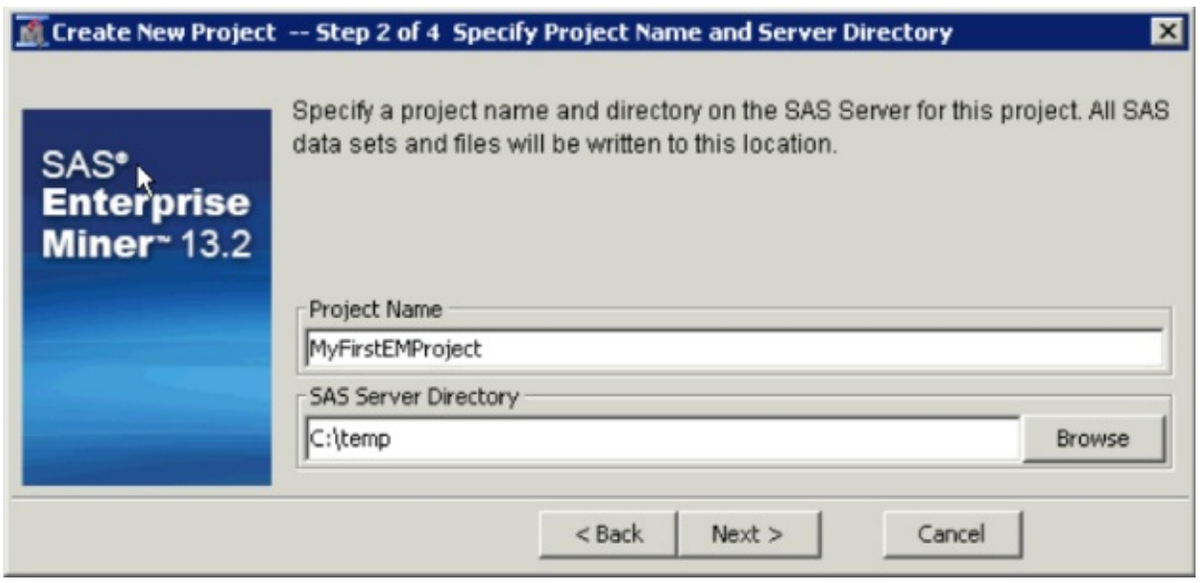

Exhibit 2.10 Creating a New Project in SAS Enterprise Miner

Finish the wizard by clicking Next and Finish, which will give the result shown in Exhibit 2.11.

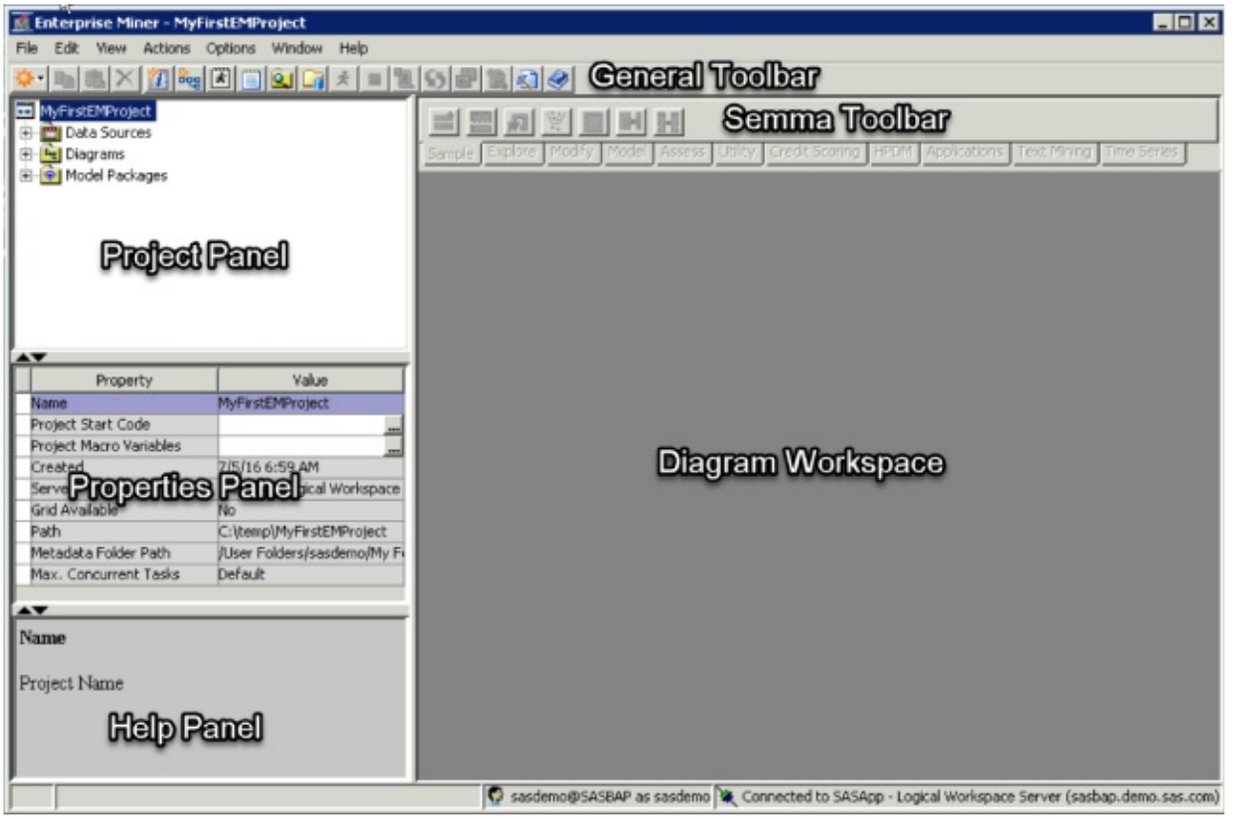

Exhibit 2.11 Start Screen of SAS Enterprise Miner

The interface is divided into multiple components:

- **General toolbar**: This toolbar provides a set of common utilities to assist in building your projects.
- **SEMMA toolbar**: This toolbar is a graphical set of node icons and tools for building process flow diagrams in the diagram workspace. To display the text name of any node or tool icon, position your mouse pointer over the icon. The nodes are grouped according to the SEMMA methodology (sampling, exploration, modification, modeling, and assessment).
- **Project panel**: This panel is used to manage and view data sources, diagrams, model packages, and list users.
- **Properties panel**: This panel is used to view and edit the settings of any object that you select (including data sources, diagrams, nodes, results, and users).
- **Diagram workspace**: This is used to build, edit, run, and save process flow diagrams. This is where you graphically build, order, and sequence the nodes that you use to analyze your data and generate reports.
- **Help panel**: This panel displays a short description of the property that you select in the properties panel. Extended help can be found in the Help Topics selection from the Help main menu.

We will now first create a library to the physical directory on disk containing our data sets. In

the properties panel, click on the button next to the Project Start Code property, and enter the information shown in Exhibit 2.12.

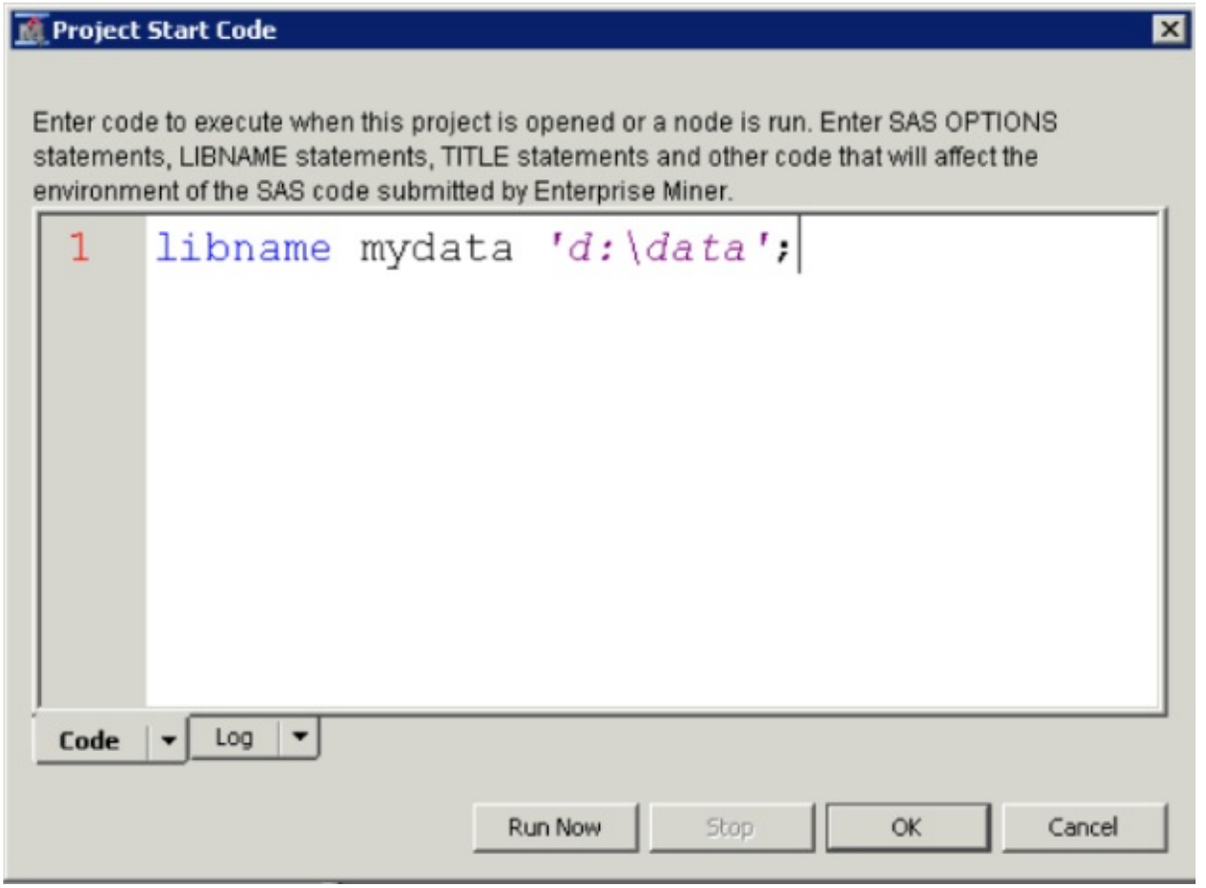

Exhibit 2.12 Creating a SAS Library in SAS Enterprise Miner

Click the Run Now button to run the statement. The mydata library has now been successfully created.

We can now right-click the Data Sources folder in the project panel (or select File ⇨ New ⇨ Data Source) to open the Data Source Wizard. In step 2 of this wizard, we select the HMEQ data set from the mydata library, as shown in Exhibit 2.13.

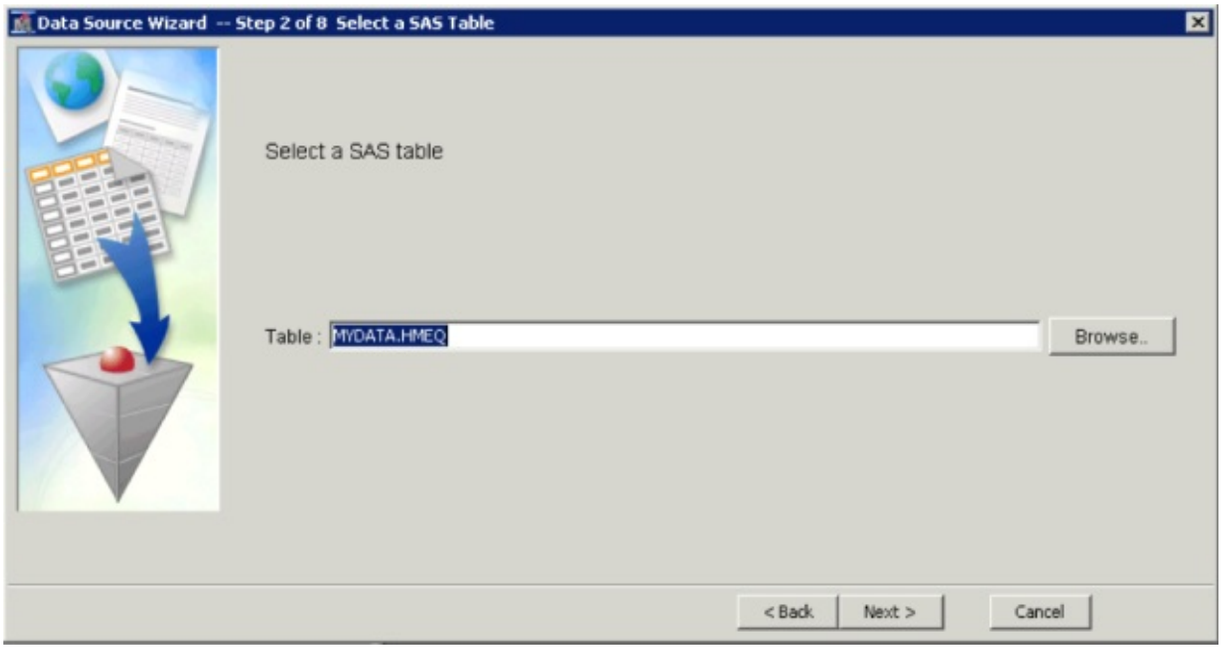

Exhibit 2.13 Selecting the HMEQ Data Set from the Mydata Library

Click Next to proceed through steps 3 and 4. In step 5 of the wizard, you can set the measurement level and measurement role for each of the variables as shown in Exhibit 2.14.

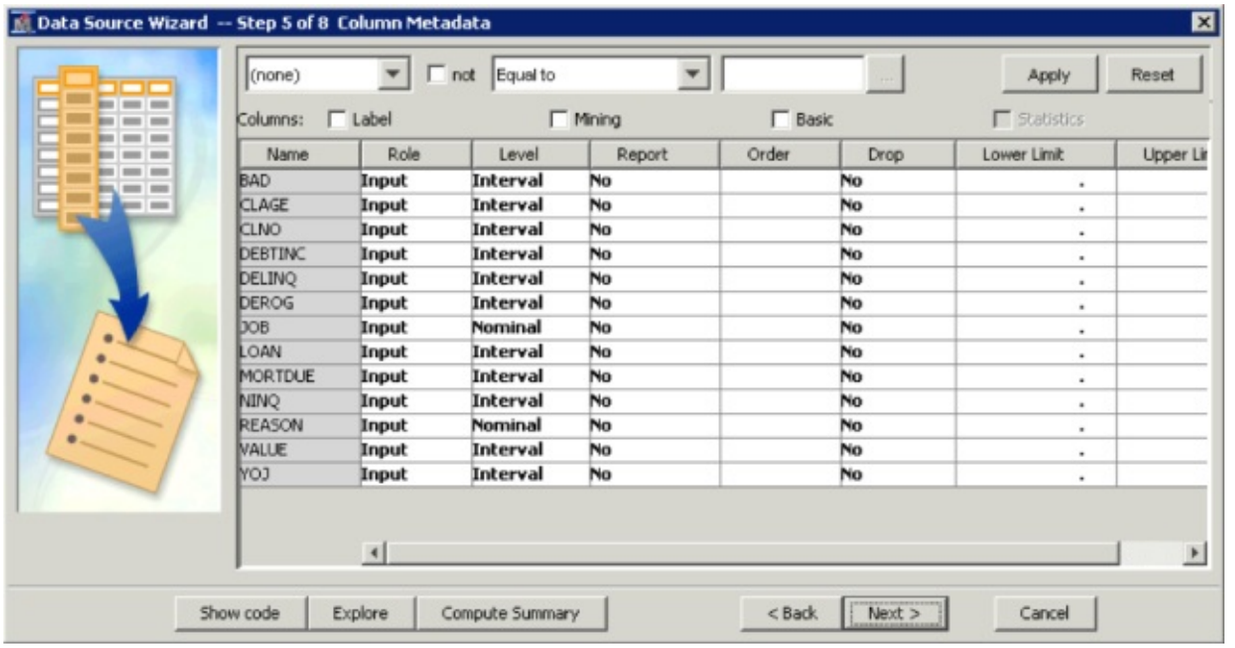

Exhibit 2.14 Specifying the Measurement Level and Measurement Role for the Variables

Let's set the measurement role of the BAD variable to target, as this is our target default indicator. We can accept all other suggested settings for both the measurement role and level. We can now create our first SAS Enterprise Miner diagram by right-clicking Diagrams in the project panel and creating a new diagram MyFirstEMDiagram, as shown in Exhibit 2.15.

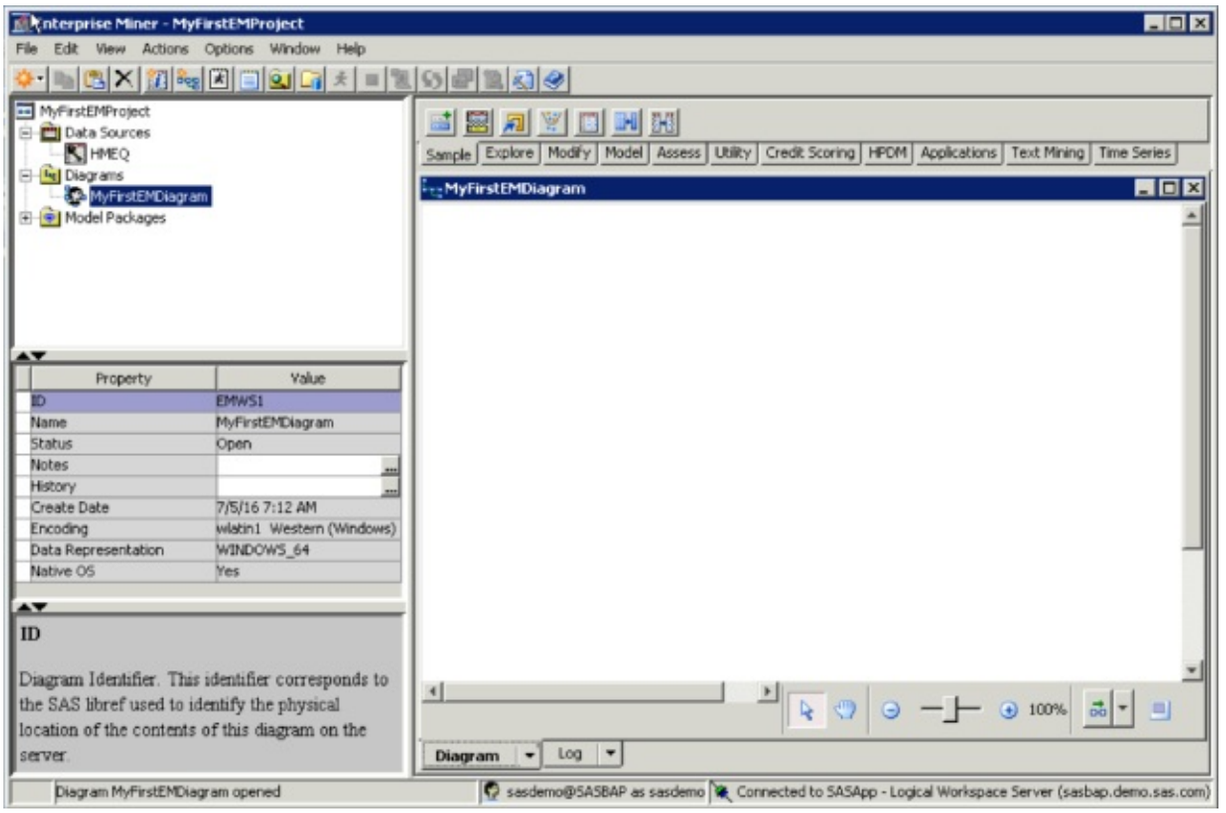

Exhibit 2.15 Creating a New Diagram

The SEMMA toolbar has been activated. We can now drag and drop the HMEQ data set from the project panel to the diagram workspace as shown in Exhibit 2.16.

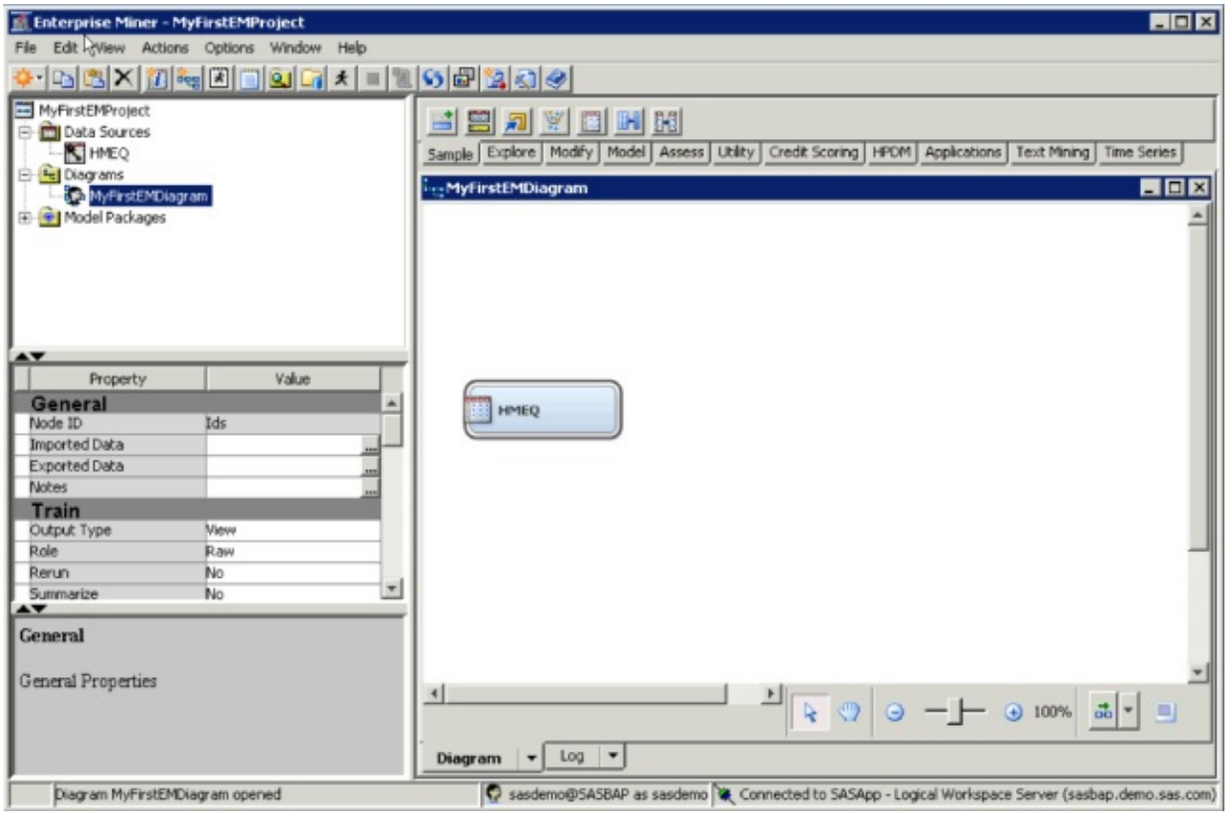

Exhibit 2.16 Adding the HMEQ Data to the Diagram Workspace

We are now ready to start analyzing this data set using nodes provided in the SEMMA toolbar. Go to the Explore tab, add a MultiPlot node to the diagram workspace, and connect it to the HMEQ data set as shown in Exhibit 2.17.

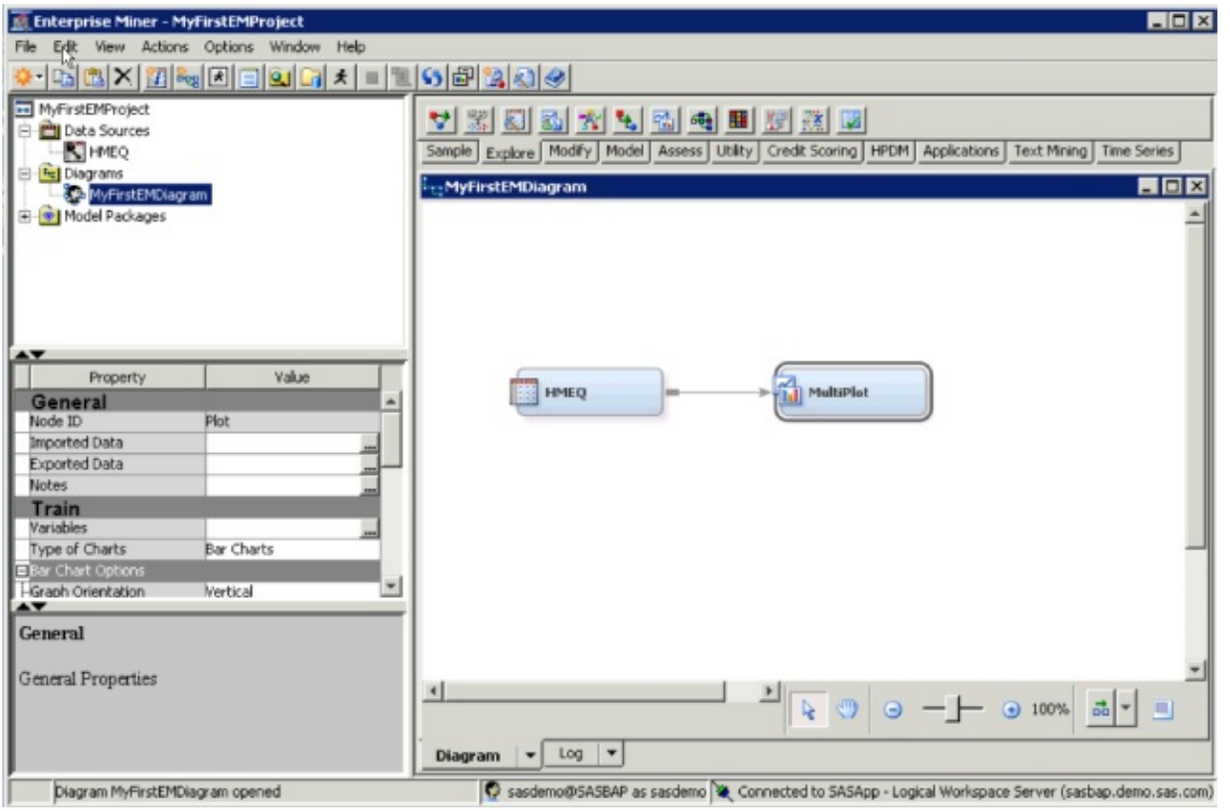

Exhibit 2.17 Adding a Multiplot Node to the Diagram Workspace

Right-click the MultiPlot node and click Run. After the run has finished, select Results. Inspect the graphs and output generated by this node.

OTHER SAS SOLUTIONS FOR CREDIT RISK MANAGEMENT

SAS Credit Scoring for banking is a SAS solution to develop, validate, deploy, and track credit scorecards. Its primary focus is on probability of default (PD) risk. It is tightly integrated with Base SAS and SAS Enterprise Miner.

SAS Credit Risk Management for banking is an end-to-end solution to estimate PD, loss given default (LGD), and exposure at default (EAD) and combine them into regulatory capital. It features an extensive catalog of prebuilt regulatory reports about risk-weighted assets (RWA), expected losses, unexpected losses, capital charges, and more. It also provides facilities to build customized reports. Each report can be inspected with roll-up and drill-down facilities by using SAS online analytical processing (OLAP) capabilities. The solution also assists in defining and performing stress testing.

SAS Model manager is a tool that supports the backtesting, benchmarking, and life cycle

management of credit risk models. It provides a centralized model repository storing models as they progress through their lifecycle, making the analytical process fully traceable, which is handy for audit and compliance checking in a Basel environment.

Going forward, we focus on providing examples in Base SAS and SAS Enterprise Miner which we believe are the most useful tools for credit analysts.

REFERENCE

SAS Institute Inc. (2015), *SAS/IML 14.1 User's Guide: Technical Report*. Cary, NC: SAS Institute.