Chapter 2
Preparing Data for Data Mining

2.1 Introduction

Data are the backbone of data mining and knowledge discovery; however, real-world business data usually are not available in data-mining-ready form. The biggest challenge for data miners, then, is preparing data suitable for modeling. Many businesses maintain central data storage and access facilities called data warehouses. Data warehousing is defined as a process of centralized data management and allows analysts to access, update, and maintain the data for analysis and reporting. Thus, data warehouse technology improves the efficiency of extracting and preparing data for data mining. Popular data warehouses use relational databases (e.g., Oracle, Informix, Sybase), and the PC data format (spreadsheets and MS Access). Roughly 70% of data mining operation time is spent on preparing the data obtained from different sources; therefore, considerable time and effort should be spent on preparing data tables to be suitable for data mining modeling.

2.2 Data Requirements in Data Mining

Summarized data are not suitable for data mining because information about individual customers or products is not available. For example, to identify profitable customers, individual customer records that include
demographic information are necessary to profile or cluster customers based on their purchasing patterns. Similarly, to identify the characteristics of profitable customers in a predictive model, target (outcome or response) and input (predictor) variables should be included. Therefore, for solving specific business objectives, suitable data must be extracted from data warehouses or new data collected that meet the data mining requirements.

2.3 Ideal Structures of Data for Data Mining

The rows (observations or cases) and columns (variables) format, similar to a spreadsheet worksheet file, is required for data mining. The rows usually contain information regarding individual customers or consumer products. The columns describe the attributes (variables) of individual cases. The variables can be continuous or categorical. Total sales per product, number of units purchased by each customer, and annual income per customer are some examples of continuous variables. Gender, race, and age group are considered categorical variables. Knowledge about the possible maximum and minimum values for the continuous variables can help to identify and exclude extreme outliers from the data. Similarly, knowledge about the possible levels for categorical variables can help to detect data entry errors and anomalies in the data.

Constant values in continuous (e.g., zip code) or categorical (state code) fields should not be included in any predictive or descriptive data mining modeling because these values are unique for each case and do not help to discriminate or group individual cases. Similarly, unique information about customers, such as phone numbers and Social Security numbers, should also be excluded from predictive data mining; however, these unique value variables can be used as ID variables to identify individual cases and exclude extreme outliers. Also, it is best not to include highly correlated (correlation coefficient >0.95) continuous predictor variables in predictive data mining, as they can produce unstable predictive models that work only with the particular sample used.

2.4 Understanding the Measurement Scale of Variables

The measurement scale of the target and input variables determines the type of modeling technique that is appropriate for a specific data mining project; therefore, understanding the nature of the measurement scale of variables used in modeling is an important data mining requirement. The variables can be generally classified into continuous or categorical.
Continuous variables are numeric variables that describe quantitative attributes of the cases and have a continuous scale of measurement. Means and standard deviations are commonly used to quantify the central tendency and dispersion. Total sales per customers and total manufacturing costs per products are examples of interval scales. An interval-scale target variable is a requirement for multiple regression and neural net modeling.

Categorical variables can be further classified as:

- **Nominal**, a categorical variable with more than two levels. Mode is the preferred estimate for measuring the central tendency, and frequency analysis is the common form of descriptive technique. Different kinds of accounts in banking, telecommunication services, and insurance policies are some examples of nominal variables. Discriminant analysis and decision tree methods are suitable for modeling nominal target variables.
- **Binary**, a categorical variable with only two levels. Sale vs. no sale and good vs. bad credit are some examples of binary variables. Logistic regression is suitable for modeling binary target variables.
- **Ordinal**, a categorical or discrete rank variable with more than two levels. Ordinal logistic regression is suitable for modeling ordinal variables.

2.5 Entire Database vs. Representative Sample

To find trends and patterns in business data, data miners can use the entire database or randomly selected samples from the entire database. Although using the entire database is currently feasible with today’s high-powered computing environment, using randomly selected representative samples in model building is more attractive due to the following reasons:

- Using random samples allows the modeler to develop the model from training or calibration samples, validate the model with a holdout “validation” dataset, and test the model with another independent test sample.
- Mining a representative random sample is easier and more efficient and can produce accurate results similar to those produced when using the entire database.
- When samples are used, data exploration and visualization help to gain insights that lead to faster and more accurate models.
- Representative samples require a relatively shorter time to cleanse, explore, and develop and validate models. They are therefore more cost effective than using entire databases.
2.6 Sampling for Data Mining

The sample used in modeling should represent the entire database because the main goal in data mining is to make predictions about the entire database. The size and other characteristics of the selected sample determine whether the sample used in modeling is a good representation of the entire database. The following types of sampling are commonly practiced in data mining:

- **Simple random sampling.** This is the most common sampling method in data mining. Each observation or case in the database has an equal chance of being included in the sample.
- **Cluster sampling.** The database is divided into clusters at the first stage of sample selection and a few of those clusters are randomly selected based on random sampling. All the records from those randomly selected clusters are included in the study.
- **Stratified random sampling.** The database is divided into mutually exclusive strata or subpopulations; random samples are then taken from each stratum proportional to its size.

2.6.1 Sample Size

The number of input variables, the functional form of the model (liner, nonlinear, models with interactions, etc.) and the size of the databases can influence the sample size requirement in data mining. By default, the SAS Enterprise Miner software takes a simple random sample of 2000 cases from the data table and divides it into TRAINING (40%), VALIDATION (30%), and TEST (30%) datasets. If the number of cases is less than 2000, the entire database is used in the model building. Data analysts can use these sampling proportions as a guideline in determining sample sizes; however, depending on the data mining objectives and the nature of the database, data miners can modify sample size proportions.

2.7 SAS Applications Used in Data Preparation

SAS software has many powerful features available for extracting data from different database management systems (DBMS). Some of the features are described in the following section. Readers are expected to have a basic knowledge in using SAS to perform the following operations. *The Little SAS Book* can serve as an introductory SAS guide to become familiar with the SAS systems and SAS programming.

© 2003 by CRC Press LLC
2.7.1 Converting Relational DBMS into SAS Datasets

2.7.1.1 Instructions for Extracting SAS Data from Oracle Database Using the SAS SQL Pass-Through Facility

If you have SAS/ACCESS software installed for your DBMS, you can extract DBMS data by using the PROC SQL (SAS/BASE) pass-through facility. The following SAS code can be modified to create an SAS data "SAS_data_name" from the Oracle database "tbl_name" to extract all the variables by inputting the username, password, file path, oracle filename, and the SAS dataset name:

```sas
PROC SQL;
CONNECT TO oracle(USER = <user> ORAPW = <password> PATH = "mypath");
CREATE TABLE sas_data_name AS
  SELECT *
  FROM CONNECTION TO oracle
  (SELECT * FROM tbl_name);
DISCONNECT FROM oracle;
QUIT;
```

Users can find additional SAS sample files in the SAS online site, which provides instructions and many examples to extract data using the SQL pass-through facility.

2.7.1.2 Instructions for Creating SAS Dataset from Oracle Database Using SAS/ACCESS and the Libname Statement

In SAS version 8.0, an Oracle database can be identified directly by associating it with the LIBNAME statement if the SAS/ACCESS software is installed. The following SAS code illustrates the DATA step with LIBNAME that refers to the Oracle database:

```sas
LIBNAME myoralib ORACLE
  USER = <user>
  PASSWORD = <password>
  PATH = "mypath"
  SCHEMA = hrdept
  PRESERVE_COL_NAMES = yes;
PROC CONTENTS DATA = myoralib.orafilename;
TITLE "The list of variable names and characteristics in the Oracle data";
RUN;
```
2.7.2 Converting PC-Based Data Files

MS Excel, Access, dBase, Lotus worksheets, and tab-delimited and comma-separated are some of the popular PC data files used in data mining. These file types can be easily converted to SAS datasets by using the PROC ACCESS or PROC IMPORT procedures in SAS. A graphical user interface (GUI)-based import wizard is also available in SAS to convert a single PC file type to an SAS dataset, but, before converting the PC file types, the following points should be considered:

- Be aware that the maximum number of rows and columns allowed in an Excel worksheet is 65,536 x 256.
- Check to see that the first row of the worksheet contains the names of the variables stored in the columns. Select names that are valid SAS variable names (one word, maximum length of 8 characters). Also, do not have any blank rows in the worksheet.
- Save only one data table per worksheet. Name the data table to "sheet1" if you are importing an MS Access table.
- Be sure to close the Excel file before trying to convert it in SAS, as SAS cannot read a worksheet file that is currently open in Excel. Trying to do so will cause a sharing violation error.
- Assign a LIBNAME before importing the PC file into an SAS dataset to create a permanent SAS data file. For information on the LIBNAME statement and making permanent SAS data files, refer to *The Little SAS Book*.
- Make sure that each column in a worksheet contains either numeric or character variables. Do not mix numeric and character values in the same column. The results of most Excel formulas should import into SAS without a problem.

2.7.2.1 Instructions for Converting PC Data Formats to SAS Datasets Using the SAS Import Wizard

The SAS import wizard available in the SAS/ACCESS module can be used to import or export Excel 4, 5, 7 (95), 98, and 2000 files, as well as Microsoft Access files in version 8.0. The GUIs in the import wizard guide users through menus and provide step-by-step instructions for transferring data between external data sources and SAS datasets. The types of files that can be imported depend on the operating system and the SAS/ACCESS engines installed. The steps involved in using the import wizard for importing a PC file follow:
1. **Select the PC file type.** The import wizard can be activated by using the pull-down menu, selecting FILE, and then clicking IMPORT. For a list of available data sources from which to choose, click the drop-down arrow (Figure 2.1). Select the file format in which your data are stored. To read an Excel file, click the black triangle and choose the type of Excel file (4.0, 5.0, 7.0 (95), 97, and 2000 spreadsheets). You can also select other PC file types, such as MS Access (97 and 2000 tables), dBASE (5.0, IV, III+, and III files), Lotus (1–2–3 WK1, WK3, and WK4 files), or text files such as tab-delimited and comma-separated files. After selecting the file type, click the NEXT button to continue.

2. **Select the PC file location.** In the import wizard’s Select file window, type the full path for the file or click BROWSE to find the file. Then click the NEXT button to go to the next screen. On the second screen, after the Excel file is chosen, the OPTIONS button becomes active. The OPTIONS button allows the user to choose which worksheet to read (if the file has multiple sheets), to specify whether or not the first row of the spreadsheet contains the variable names, and to choose the range of the worksheet to be read. Generally, these options can be ignored.

![Figure 2.1](image)

*Figure 2.1* Screen copy of SAS IMPORT (version 8.2) showing the valid file types that can be imported to SAS datasets.

© 2003 by CRC Press LLC
3. Select the temporary or permanent SAS dataset name. The third screen prompts for the SAS data file name. Select the LIBRARY (the alias name for the folder) and member (SAS dataset name) for your SAS data file. For example, to create a temporary data file called “fraud”, choose “WORK” for the LIBRARY and “fraud” as the valid SAS dataset name for the member. When you are ready, click FINISH, and SAS will convert the specified Excel spreadsheet into an SAS data file.

4. Perform a final check. Check the LOG window for a message indicating that SAS has successfully converted the Excel file to an SAS dataset. Also, compare the number of observations and variables in the SAS dataset with the source Excel file to make sure that SAS did not import any empty rows or columns.

2.7.2.2 Converting PC Data Formats to SAS Datasets Using the EXCELSAS Macro

The EXCELSAS macro application can be used as an alternative to the SAS import wizard to convert PC file types to SAS datasets. The SAS procedure PROC IMPORT is the main tool if the EXCELSAS macro is used with post-SAS version 8.0. PROC IMPORT can import a wide variety of types and versions of PC files. However, if the EXCELSAS macro is used in SAS version 6.12, then PROC ACCESS will be selected as the main tool for importing only limited PC file formats. See Section 2.7.2.3 for more details regarding the various PC data formats that can be imported using the EXCELSAS macro. The advantages for using the EXCELSAS macro over the import wizard include:

- Multiple PC files can be converted in a single operation.
- A sample printout of the first 10 observations is produced in the output file.
- The characteristics of the numeric and character variables and number of observations in the converted SAS data file are reported in the output file.
- Descriptive statistics of all the numeric variables and the frequency information of all character variables are reported in the output file.
- Options for saving the output tables in WORD, HTML, PDF, and TXT formats are available.

Software requirements for using the EXCELSAS macro include:

- The SAS/CORE, SAS/BASE, and SAS/ACCESS interface to PC file formats must be licensed and installed at your site.

© 2003 by CRC Press LLC
The EXCELSAS macro has been tested only in the Windows (Windows 98 and later) environment. However, to import DBF, CSV, and tab-delimited files in the Unix platform, the EXCELSAS macro could be used with minor modification in the macro-call file (see the steps below).

An active Internet connection is required for downloading the EXCELSAS macro from the book website if the companion CD-ROM is not available.

SAS version 8.0 or above is recommended for full utilization.

### 2.7.2.3 Steps Involved in Running the EXCELSAS Macro

1. Prepare the PC data file by following the recommendations given in Section 2.7.2.

2. **If the companion CD-ROM is not available**, first verify that the Internet connection is active. Open the Excelsas.sas macro-call file in the SAS PROGRAM EDITOR window. The Appendix provides instructions for downloading the macro-call and sample data files from the book website. **If the companion CD-ROM is available**, the Excelsas.sas macro-call file can be found in the mac-call folder on the CD-ROM. Open the Excelsas.sas macro-call file in the SAS PROGRAM EDITOR window. Click the RUN icon to submit the macro-call file Excelsas.sas to open the MACRO window called EXCELSAS.

3. Input the appropriate parameters in the macro-call window by following the instructions provided in the EXCELSAS macro help file (see Section 2.7.2.4). After inputting all the required macro parameters, check whether the cursor is in the last input field (#6) and that the RESULTS VIEWER window is closed, then hit the ENTER key (not the RUN icon) to submit the macro.

4. Examine the LOG window for any macro execution errors only in the DISPLAY mode. If any errors in the LOG window are found, activate the PROGRAM EDITOR window, resubmit the Excelsas.sas macro-call file, check the macro input values, and correct any input errors. Otherwise, activate the PROGRAM EDITOR window, resubmit the Excelsas.sas macro-call file, and change the macro input (#6) value from DISPLAY to any other desirable format (see Section 2.7.2.4). The PC file will be imported to a temporary (if macro input #4 is blank or WORK) or permanent (if a LIBNAME is specified in macro input option #4) SAS dataset. The output, including the first 10 observations of the imported SAS data, characteristics of numeric and character variables, simple statistics for numeric variables, and frequency information for the character variables, will be saved in the user-specified format in the user-specified folder as a single file.
2.7.2.4 Help File for SAS Macro EXCELSAS: Description of Macro Parameters

1. **Macro-call parameter:** Input PC file type (required parameter).
   **Descriptions and explanation:** Include the type of PC file being imported.
   **Options/explanations:**
   - Pre-version 8.0
     - *Excel* — (XLS) files; Excel 95, Excel5, Excel4
     - *Lotus* — (WK4) files
     - *dBase* — (III and IV) files
   - Version 8.0 and after
     - *Excel* — (XLS) files; all types of Excel
     - *Lotus* — (WK4) files
     - *dBase* — (III and IV) files
     - *Access* — (mdb) files; 97 and 2000 files
     - *Tab* — (TAB) tab-delimited files
     - *CSV* — (CSV) comma-delimited files

2. **Macro-call parameter:** Input folder name containing the PC file (required parameter).
   **Descriptions and explanation:** Input the location (path) of folder name containing the PC file. If the field is left blank, SAS will look in the default HOME folder.
   **Options/explanations:**
   - Possible values
     - a:\ — A drive
     - c:\excel\ — folder named “Excel” in the C drive (be sure to include the back-slash at the end of folder name)

3. **Macro-call parameter:** Input PC file names (required statement).
   **Descriptions and explanation:** List the names of PC files (without the file extension) being imported. The same file name will be used for naming the imported SAS dataset. If multiple PC files are listed, all of the files can be imported in one operation.
   **Options/examples:**
   - BASEBALL CRIME
   - customer99
   Use a short file name (eight characters or less in pre-8.0 versions).

4. **Macro-call parameter:** Optional LIBNAME.
   **Descriptions and explanation:** To save the imported PC file as a permanent SAS dataset, input the preassigned library (LIBNAME) name. The predefined LIBNAME will tell SAS in which folder to
save the permanent dataset. If this field is left blank, a temporary data file will be created.

**Option/example:**

SASUSER

The permanent SAS dataset is saved in the library called SASUSER.

5. **Macro-call parameter:** Folder to save SAS output (optional).

**Descriptions and explanation:** To save the SAS output files in a specific folder, input the full path of the folder. The SAS dataset name will be assigned to the output file. If this field is left blank, the output file will be saved in the default folder.

**Options/explanations:**

Possible values

- c:\output\ — folder named “OUTPUT”
- s:\george\ — folder named “George” in network drive S

Be sure to include the back-slash at the end of the folder name.

6. **Macro-call parameter:** Display or save SAS output (required statement).

**Descriptions and explanation:** Option for displaying all output files in the OUTPUT window or saving as a specific format in a folder specified in option #5.

**Options/explanations:**

Possible values

- **DISPLAY:** Output will be displayed in the OUTPUT window. System messages will be displayed in LOG window.
- **WORD:** Output will be saved in the user-specified folder and viewed in the results VIEWER window as a single RTF format (version 8.0 and later) or saved only as a text file in pre-8.0 versions.
- **WEB:** Output will be saved in the user-specified folder and viewed in the results VIEWER window as a single HTML file (version 8.0 and later) or saved only as a text file in pre-8.0 versions.
- **PDF:** Output will be saved in the user-specified folder and viewed in the results VIEWER window as a single PDF file (version 8.2 and later) or saved only as a text file in pre-8.2 versions.
- **TXT:** Output will be saved as a TXT file in all SAS versions. No output will be displayed in the OUTPUT window.

**Note:** All system messages will be deleted from the LOG window at the end of macro execution if DISPLAY is not selected as the macro input in option #6.
2.7.2.5 Importing an Excel File Called “fraud” to a Permanent SAS Dataset Called “fraud”

Source file: fraud.xls; MS Excel sheet 2000
Variables: Daily retail sales, number of transactions, net sales, and manager on duty in a small convenience store
Number of observations: 923

1. Open the Excel file “fraud” and make sure that all the specified data requirements reported in Section 2.7.2 are satisfied. The screen copy of the Excel file with the required format is shown in Figure 2.2. Close the “fraud” worksheet file and exit from Excel.
2. Open the EXCELSAS macro-call window in SAS (see Figure 2.3); input the appropriate macro-input values by following the suggestions given in the help file in Section 2.7.2.4. Submit the EXCELSAS macro to import the “fraud” Excel worksheet to a permanent SAS dataset called “fraud”.
3. A printout of the first 10 observations including all variables in the SAS dataset “fraud” is displayed (Table 2.1). Examine the printout to see whether SAS imported all the variables from the Excel worksheet correctly.
4. Examine the PROC CONTENTS display of all the variables in the SAS dataset called “fraud”. Table 2.2 shows the characteristics of all numeric variables, and Table 2.3 shows the character variables.
5. Examine the simple descriptive statistics for all the numeric variables (Table 2.4). Note that the variables YEAR, WEEK, and DAY are treated as numeric. Total number of observations in the dataset is 923. Confirm that three observations in VOIDS and TRANSAC and two observations in NETSALES are missing in the Excel file. Also, examine the minimum and the maximum numbers for all the numeric variables and verify that no unusual or extreme values are present.
6. Examine the frequency information (Tables 2.5 to 2.7) for all the character variables. Make sure that character variable levels are entered consistently. SAS systems consider uppercase and lowercase data values differently. For example, April, april, and APRIL are considered different data values. The frequency information for MGR (manager on duty) indicated that managers mgr_a and mgr_e were on duty relatively fewer times than the other three managers (Table 2.8). This information should be considered in modeling.
2.7.3 SAS Macro Applications: Random Sampling from the Entire Database Using the SAS Macro RANSPLIT

The RANSPLIT macro can be used to obtain TRAINING, VALIDATION, and TEST samples from the entire database. The SAS data step and the RANUNI function are the main tools in the RANSPLIT macro. The advantages of using the RANSPLIT macro are:

Figure 2.2  Screen copy of MS Excel 2000 worksheet “fraud.xls” opened in Office 2000; shows the required structure of the PC spreadsheet.

© 2003 by CRC Press LLC
The distribution pattern among the TRAINING, VALIDATION, and TEST samples for user-specified numeric variables can be examined graphically by box plots to confirm that all three sample distributions are similar.

A sample printout of the first 10 observations can be examined from the TRAINING sample.

Options for saving the output tables and graphics in WORD, HTML, PDF, and TXT formats are available.

Figure 2.3 Screen copy of EXCELTOSAS macro-call window showing the macro-call parameters required to import PC file types to SAS datasets.

- The distribution pattern among the TRAINING, VALIDATION, and TEST samples for user-specified numeric variables can be examined graphically by box plots to confirm that all three sample distributions are similar.
- A sample printout of the first 10 observations can be examined from the TRAINING sample.
- Options for saving the output tables and graphics in WORD, HTML, PDF, and TXT formats are available.

Software requirements for using the RANSPLIT macro include:

- SAS/CORE, SAS/BASE, and SAS/GRAPH must be licensed and installed at the site.
- SAS version 8.0 and above is recommended for full utilization.
- An active Internet connection is required for downloading the RANSPLIT macro from the book website if the companion CD-ROM is not available.

© 2003 by CRC Press LLC
Table 2.1  Macro EXCELSAS: PROC PRINT Output, First 10 Observations

<table>
<thead>
<tr>
<th>YEAR</th>
<th>MONTH</th>
<th>WEEK</th>
<th>DAY</th>
<th>DOFWEK</th>
<th>VOIDS</th>
<th>NETSALES</th>
<th>TRANSAC</th>
<th>MGR</th>
</tr>
</thead>
<tbody>
<tr>
<td>1998</td>
<td>January</td>
<td>1</td>
<td>2</td>
<td>Fri</td>
<td>1008.75</td>
<td>1443</td>
<td>139</td>
<td>mgr_a</td>
</tr>
<tr>
<td>1998</td>
<td>January</td>
<td>1</td>
<td>3</td>
<td>Sat</td>
<td>10.00</td>
<td>1905</td>
<td>168</td>
<td>mgr_b</td>
</tr>
<tr>
<td>1998</td>
<td>January</td>
<td>2</td>
<td>4</td>
<td>Sun</td>
<td>9.00</td>
<td>1223</td>
<td>134</td>
<td>mgr_b</td>
</tr>
<tr>
<td>1998</td>
<td>January</td>
<td>2</td>
<td>5</td>
<td>Mon</td>
<td>7.00</td>
<td>1280</td>
<td>146</td>
<td>mgr_c</td>
</tr>
<tr>
<td>1998</td>
<td>January</td>
<td>2</td>
<td>6</td>
<td>Tue</td>
<td>15.00</td>
<td>1243</td>
<td>129</td>
<td>mgr_b</td>
</tr>
<tr>
<td>1998</td>
<td>January</td>
<td>2</td>
<td>7</td>
<td>Wed</td>
<td>14.00</td>
<td>871</td>
<td>135</td>
<td>mgr_a</td>
</tr>
<tr>
<td>1998</td>
<td>January</td>
<td>2</td>
<td>8</td>
<td>Thu</td>
<td>4.00</td>
<td>1115</td>
<td>105</td>
<td>mgr_c</td>
</tr>
<tr>
<td>1998</td>
<td>January</td>
<td>2</td>
<td>9</td>
<td>Fri</td>
<td>33.21</td>
<td>1080</td>
<td>109</td>
<td>mgr_c</td>
</tr>
<tr>
<td>1998</td>
<td>January</td>
<td>2</td>
<td>10</td>
<td>Sat</td>
<td>8.00</td>
<td>1796</td>
<td>156</td>
<td>mgr_b</td>
</tr>
<tr>
<td>1998</td>
<td>January</td>
<td>3</td>
<td>11</td>
<td>Sun</td>
<td>13.00</td>
<td>1328</td>
<td>132</td>
<td>mgr_c</td>
</tr>
</tbody>
</table>

Table 2.2  Macro EXCELSAS: PROC CONTENTS Output, Numeric Variable Description

<table>
<thead>
<tr>
<th>Obs</th>
<th>NAME</th>
<th>TYPE</th>
<th>LENGTH</th>
<th>VARNUM</th>
<th>LABEL</th>
<th>NPOS</th>
<th>NOBS</th>
<th>ENGINE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>DAY</td>
<td>1</td>
<td>8</td>
<td>4</td>
<td>DAY</td>
<td>16</td>
<td>923</td>
<td>V8</td>
</tr>
<tr>
<td>5</td>
<td>NETSALES</td>
<td>1</td>
<td>8</td>
<td>7</td>
<td>NETSALES</td>
<td>32</td>
<td>923</td>
<td>V8</td>
</tr>
<tr>
<td>6</td>
<td>TRANSAC</td>
<td>1</td>
<td>8</td>
<td>8</td>
<td>TRANSAC</td>
<td>40</td>
<td>923</td>
<td>V8</td>
</tr>
<tr>
<td>7</td>
<td>VOIDS</td>
<td>1</td>
<td>8</td>
<td>6</td>
<td>VOIDS</td>
<td>24</td>
<td>923</td>
<td>V8</td>
</tr>
<tr>
<td>8</td>
<td>WEEK</td>
<td>1</td>
<td>8</td>
<td>3</td>
<td>WEEK</td>
<td>8</td>
<td>923</td>
<td>V8</td>
</tr>
<tr>
<td>9</td>
<td>YEAR</td>
<td>1</td>
<td>8</td>
<td>1</td>
<td>YEAR</td>
<td>0</td>
<td>923</td>
<td>V8</td>
</tr>
</tbody>
</table>

© 2003 by CRC Press LLC
Table 2.3  Macro EXCELSAS: PROC CONTENTS Output, Character Variable Descriptions

<table>
<thead>
<tr>
<th>Obs</th>
<th>NAME</th>
<th>TYPE</th>
<th>LENGTH</th>
<th>VARNUM</th>
<th>LABEL</th>
<th>FORMAT</th>
<th>NPOS</th>
<th>NOBS</th>
<th>ENGINE</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>DOFWEEK</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>DOFWEEK</td>
<td>$</td>
<td>57</td>
<td>923</td>
<td>V8</td>
</tr>
<tr>
<td>3</td>
<td>MGR</td>
<td>2</td>
<td>5</td>
<td>9</td>
<td>MGR</td>
<td>$</td>
<td>60</td>
<td>923</td>
<td>V8</td>
</tr>
<tr>
<td>4</td>
<td>MONTH</td>
<td>2</td>
<td>9</td>
<td>2</td>
<td>MONTH</td>
<td>$</td>
<td>48</td>
<td>923</td>
<td>V8</td>
</tr>
</tbody>
</table>

Table 2.4  Macro EXCELSAS: PROC MEANS Output, Simple Statistics and Numeric Variables

<table>
<thead>
<tr>
<th>Variable</th>
<th>Label</th>
<th>N</th>
<th>Mean</th>
<th>Std Dev</th>
<th>Minimum</th>
<th>Maximum</th>
</tr>
</thead>
<tbody>
<tr>
<td>Year</td>
<td>YEAR</td>
<td>923</td>
<td>1998.88</td>
<td>0.7867336</td>
<td>1998.00</td>
<td>2000.00</td>
</tr>
<tr>
<td>Week</td>
<td>WEEK</td>
<td>923</td>
<td>3.0270856</td>
<td>1.3215726</td>
<td>1.0000000</td>
<td>6.0000000</td>
</tr>
<tr>
<td>Day</td>
<td>DAY</td>
<td>923</td>
<td>15.7941495</td>
<td>8.7590603</td>
<td>1.0000000</td>
<td>31.0000000</td>
</tr>
<tr>
<td>Voids</td>
<td>VOIDS</td>
<td>923</td>
<td>69.6595543</td>
<td>183.0534292</td>
<td>0</td>
<td>1752.45</td>
</tr>
<tr>
<td>Net sales</td>
<td>NETSALES</td>
<td>923</td>
<td>1324.33</td>
<td>471.5667690</td>
<td>7.0000000</td>
<td>4114.00</td>
</tr>
<tr>
<td>Transactions</td>
<td>TRANSAC</td>
<td>923</td>
<td>132.2576087</td>
<td>33.0792886</td>
<td>10.0000000</td>
<td>259.0000000</td>
</tr>
</tbody>
</table>

© 2003 by CRC Press LLC
Table 2.5  Macro EXCELSAS: PROC FREQ Output, Frequency and Character Variable MONTH

<table>
<thead>
<tr>
<th>MONTH</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>April</td>
<td>89</td>
</tr>
<tr>
<td>August</td>
<td>88</td>
</tr>
<tr>
<td>December</td>
<td>57</td>
</tr>
<tr>
<td>February</td>
<td>83</td>
</tr>
<tr>
<td>January</td>
<td>83</td>
</tr>
<tr>
<td>July</td>
<td>87</td>
</tr>
<tr>
<td>June</td>
<td>88</td>
</tr>
<tr>
<td>March</td>
<td>92</td>
</tr>
<tr>
<td>May</td>
<td>88</td>
</tr>
<tr>
<td>November</td>
<td>53</td>
</tr>
<tr>
<td>October</td>
<td>58</td>
</tr>
<tr>
<td>September</td>
<td>57</td>
</tr>
</tbody>
</table>

Table 2.6  Macro EXCELSAS: PROC FREQ Output: Frequency and Character Variable DOFWEEK

<table>
<thead>
<tr>
<th>DOFWEEK</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fri</td>
<td>133</td>
</tr>
<tr>
<td>Mon</td>
<td>133</td>
</tr>
<tr>
<td>Sat</td>
<td>130</td>
</tr>
<tr>
<td>Sun</td>
<td>137</td>
</tr>
<tr>
<td>Thu</td>
<td>128</td>
</tr>
<tr>
<td>Tue</td>
<td>129</td>
</tr>
<tr>
<td>Wed</td>
<td>133</td>
</tr>
</tbody>
</table>

Table 2.7  Macro EXCELSAS: PROC FREQ Output, Frequency and Character Variable MGR

<table>
<thead>
<tr>
<th>MGR</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>mgr_a</td>
<td>38</td>
</tr>
<tr>
<td>mgr_b</td>
<td>204</td>
</tr>
<tr>
<td>mgr_c</td>
<td>258</td>
</tr>
<tr>
<td>mgr_d</td>
<td>408</td>
</tr>
<tr>
<td>mgr_e</td>
<td>15</td>
</tr>
</tbody>
</table>

© 2003 by CRC Press LLC
Table 2.8  Macro RANSPLIT: PROC PRINT Output, First 10 Observations, Training Data

<table>
<thead>
<tr>
<th>Obs</th>
<th>YEAR</th>
<th>MONTH</th>
<th>WEEK</th>
<th>DAY</th>
<th>DOFWEEK</th>
<th>VOID</th>
<th>NETSALES</th>
<th>TRANSAC</th>
<th>MGR</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1999</td>
<td>May</td>
<td>4</td>
<td>17</td>
<td>Mon</td>
<td>45</td>
<td>1148.00</td>
<td>117</td>
<td>mgr_c</td>
</tr>
<tr>
<td>2</td>
<td>1998</td>
<td>December</td>
<td>2</td>
<td>7</td>
<td>Mon</td>
<td>12.5</td>
<td>1208.25</td>
<td>130</td>
<td>mgr_c</td>
</tr>
<tr>
<td>3</td>
<td>1998</td>
<td>July</td>
<td>3</td>
<td>15</td>
<td>Wed</td>
<td>0.0</td>
<td>930.25</td>
<td>89</td>
<td>mgr_d</td>
</tr>
<tr>
<td>4</td>
<td>2000</td>
<td>July</td>
<td>3</td>
<td>10</td>
<td>Mon</td>
<td>0.0</td>
<td>1900.97</td>
<td>163</td>
<td>mgr_b</td>
</tr>
<tr>
<td>5</td>
<td>1999</td>
<td>November</td>
<td>2</td>
<td>11</td>
<td>Thu</td>
<td>1601.5</td>
<td>785.00</td>
<td>113</td>
<td>mgr_d</td>
</tr>
<tr>
<td>6</td>
<td>1998</td>
<td>January</td>
<td>2</td>
<td>7</td>
<td>Wed</td>
<td>14</td>
<td>871.00</td>
<td>135</td>
<td>mgr_a</td>
</tr>
<tr>
<td>7</td>
<td>1999</td>
<td>August</td>
<td>2</td>
<td>8</td>
<td>Sun</td>
<td>16</td>
<td>1439.20</td>
<td>126</td>
<td>mgr_c</td>
</tr>
<tr>
<td>8</td>
<td>1999</td>
<td>February</td>
<td>4</td>
<td>25</td>
<td>Thu</td>
<td>4.75</td>
<td>751.50</td>
<td>83</td>
<td>mgr_d</td>
</tr>
<tr>
<td>9</td>
<td>1998</td>
<td>February</td>
<td>2</td>
<td>8</td>
<td>Sun</td>
<td>7.0</td>
<td>2103.00</td>
<td>187</td>
<td>mgr_b</td>
</tr>
<tr>
<td>10</td>
<td>2000</td>
<td>August</td>
<td>5</td>
<td>27</td>
<td>Sun</td>
<td>5.0</td>
<td>1329.94</td>
<td>121</td>
<td>mgr_b</td>
</tr>
</tbody>
</table>
2.7.3.1 Steps Involved in Running the RANSPLIT Macro

1. Prepare the SAS dataset (permanent or temporary) and examine the variables.

2. If the companion CD-ROM is not available, first verify that the Internet connection is active. Open the Ransplit.sas macro-call file in the SAS PROGRAM EDITOR window. The Appendix provides instructions for downloading the macro-call and sample data files from the book website. If the companion CD-ROM is available, the Ransplit.sas macro-call file can be found in the mac-call folder on the CD-ROM. Open the Ransplit.sas macro-call file in the SAS PROGRAM EDITOR window. Click the RUN icon to submit the macro-call file Ransplit.sas to open the macro-call window called RANSPLIT (Figure 2.4).

3. Input the appropriate parameters in the macro-call window by following the instructions provided in the RANSPLIT macro help file (Section 2.7.3.2). After inputting all the required macro parameters, be sure the cursor is in the last input field and that the RESULTS VIEWER window is closed, then hit the ENTER key (not the RUN icon) to submit the macro.

Figure 2.4 Screen copy of RANSPLIT macro-call window showing the macro-call parameters required to split the database into TRAINING, VALIDATION, and TEST samples.

© 2003 by CRC Press LLC
4. Examine the LOG window (only in DISPLAY mode) for any macro execution errors. If any errors appear in the LOG window, activate the PROGRAM EDITOR window, resubmit the Ransplit.sas macro-call file, check the macro input values, and correct any input errors.

5. Save the output files. If no errors are found in the LOG window, activate the PROGRAM EDITOR window, resubmit the Ransplit.sas macro-call file, and change the macro input value from DISPLAY to any other desirable format (see Section 2.7.3.2). If the sample size input for the validation sample is blank, a random sample with a user-specified sample size will be saved as TRAINING and the leftover observations in the database will be saved as VALIDATION datasets. If sample sizes are specified for both TRAINING and VALIDATION input, random samples with user-specified sample sizes will be saved as TRAINING and VALIDATION samples and the leftover observations will be saved as the TEST sample. The new SAS datasets will be saved as temporary (if macro input option #9 is blank or WORK) or permanent files (if a LIBNAME is specified in macro input option #9). The printout of the first 10 observations of the TRAINING SAS data and box plots illustrating distribution patterns among the TRAINING, VALIDATION, and TEST samples for user-specified numeric variables can be saved in a user-specified format in the user-specified folder.

2.7.3.2 Help File for SAS Macro RANSPLIT: Description of Macro Parameters

1. **Macro-call parameter**: Input the SAS data set (required parameter).
   **Descriptions and explanation**: Include the SAS dataset name, temporary or permanent (LIBNAME.sas_data_name) of the database you would like to draw samples from.
   **Options/explanations**:
   - fraud (temporary SAS data called “fraud”)
   - gf.fraud (permanent SAS data called “fraud” saved in the predefined SAS library called “GF”)

2. **Macro-call parameters**: Input numeric variable names (optional parameter).
   **Descriptions and explanation**: Input names of the numeric variables. Distribution aspects of the specified numeric variables are compared among different samples by box plots.
   **Options/example**: fraud net sales
3. **Macro-call parameter:** Input observation number in train data (required statement).
   **Descriptions and explanation:** Input the desired sample size number for the TRAINING data. Usually 40% of the database equivalent to 2000 observations is selected.
   **Options/example:** 2000 1400 400

4. **Macro-call parameter:** Observation number in validation data (optional parameter).
   **Descriptions and explanation:** Input the desired sample size number for the VALIDATION data. Usually 30% of the database equivalent to roughly 1000 observations is selected for validation. The leftover observations in the database after the TRAINING and VALIDATION samples are selected will be included in the TEST sample. If this field is left blank, all of the leftover observations in the database after the TRAINING sample is selected will be included in the VALIDATION set.
   **Options/example:** 1000 300

5. **Macro-call parameter:** Folder to save SAS output (optional statement).
   **Descriptions and explanation:** To save the SAS output files in a specific folder, input the full path of the folder. If this field is left blank, the output file will be saved in the default folder.
   **Options/explanations:**
   - Possible values
     - c:\output\ — folder named “OUTPUT”
     - s:\george\ — folder named “George” in network drive S
   - Be sure to include the back-slash at the end of the folder name.

6. **Macro-call parameter:** Folder to save SAS graphics (optional).
   **Descriptions and explanation:** To save the SAS graphics files in EMF format suitable for including in PowerPoint presentations, specify the output format as TXT in version 8.0 or later. In pre-8.0 versions, all graphic format files will be saved in a user-specified folder. If the graphics folder field is left blank, the graphics file will be saved in the default folder.
   **Options/explanations:**
   - Possible values
     - c:\output\ — folder named “OUTPUT”

7. **Macro-call parameter:** Display or save SAS output (required statement).
**Descriptions and explanation:** Option for displaying all output files in the OUTPUT window or save as a specific format in a folder specified in option #5.

**Options/explanations:**
Possible values

- **DISPLAY:** Output will be displayed in the OUTPUT window. All SAS graphics will be displayed in the GRAPHICS window. System messages will be displayed in the LOG window.

- **WORD:** Output and all SAS graphics will be saved together in the user-specified folder and will be displayed in the VIEWER window as a single RTF format file (version 8.0 and later) or saved only as a text file, and all graphics files in CGM format will be saved separately in a user-specified folder (macro input option #6) in pre-8.0 version SAS.

- **WEB:** Output and graphics are saved in the user-specified folder and are viewed in the results VIEWER window as a single HTML file (version 8.1 and later) or saved only as a text file, and all graphics files in GIF format will be saved separately in a user-specified folder (macro input option #5) in pre-8.0 versions.

- **PDF:** Output and graphics are saved in the user-specified folder and are viewed in the results VIEWER window as a single PDF file (version 8.2 and later) or saved only as a text file, and all graphics files in the PNG format will be saved separately in a user-specified folder (macro input option #6) in pre-8.2 versions.

- **TXT:** Output will be saved as a TXT file in all SAS versions. No output will be displayed in the OUTPUT window. All graphic files will be saved in the EMF format in version 8.0 and later or CGM format in pre-8.0 versions in a user-specified folder (macro input option #6).

*Note:* System messages are deleted from the LOG window if DISPLAY is not selected as the input.

8. **Macro-call parameter:** $\zeta$th number of run (required statement).

**Descriptions and explanation:** SAS output files will be saved by forming a file name from the original SAS dataset name and the counter number provided in macro input option #8. For example, if the original SAS dataset name is “fraud” and the counter number included is 1, the SAS output files will be saved as “fraud1.*” in the user-specified folder. By changing the counter numbers, users can avoid replacing the previous SAS output files with the new outputs.

**Options/explanations:** Numbers 1 to 10 and any letters are valid.
9. **Macro-call parameter**: Optional LIBNAME for creating permanent SAS data.

**Descriptions and explanation**: To save the TRAINING, VALIDATION, and TEST datasets as permanent SAS datasets and input the preassigned library (LIBNAME) name. The predefined LIBNAME will tell SAS in which folder to save the permanent datasets. If this field is left blank, temporary WORK data files will be created for all samples.

**Options/example**:

```
SASUSER
```

The permanent SAS dataset is saved in the library called SASUSER.

---

2.7.3.3 *Drawing TRAINING (400), VALIDATION (300), and TEST (All Leftover Observations) Samples from the Permanent SAS Dataset Called “fraud”*

- **Source file**: Permanent SAS data set “fraud” located in the library “GF”
- **Variables**: Daily retail sales, number of transactions, net sales, and manager on duty in a small convenience store
- **Number of observations**: 923

1. Open the RANSPLIT macro-call window in SAS (see Figure 2.4), input the appropriate macro input values by following the suggestions given in the help file in Section 2.7.3.2. Submit the RANSPLIT macro, and SAS will randomly split the entire database into three samples and save these TRAIN (400 observations), VALIDATION (300 observations), and TEST (leftover observations) as permanent SAS datasets in the LIBRARY called “GF”.

2. The output file shows a list of the first 10 observations from the train dataset (Table 2.8). This dataset will be used in calibrating or training the models. Examine the contents and characteristics of the variables in the SAS data set called “fraud”.

3. The distribution pattern among the TRAINING, VALIDATION, and TEST samples for one of the numeric variables NETSALES can be graphically examined by the box plot (Figure 2.5) created by the RANSPLIT SAS macro. A box plot shows the distribution pattern and the central tendency of the data. The line between the lowest adjacent limit and the bottom of the box represents one fourth of the data. One fourth of the data fall between the bottom of the
box and the median, and another one fourth between the median and the top of the box. The line between the top of the box and the upper adjacent limit represents the final one fourth of the observations. For more information about interpreting the box plot, see Chapter 3. The box plot confirmed that the distribution showed a similar pattern for NETSALES among the TRAINING, VALIDATION, and TEST samples and confirmed that the random sampling was successful.

2.8 Summary

Data mining and knowledge discovery are driven by massive amounts of data. Business databases are growing at exponential rates because of the multitude of data that exist. Today, organizations are accumulating vast and growing amounts of data in various formats and in different databases. Dynamic data access is critical for data navigation applications, and the ability to store large databases is critical to data mining. The data may exist in a variety of formats such as relational databases, mainframe systems, or
flat files; therefore, in data mining, it is common to work with data from several different sources. Roughly 70% of the time spent data mining is in preparing the data. The methods of extracting and preparing suitable data for data mining are covered in this chapter. Calibrating the prediction model using the TRAINING sample, validating the model using the VALIDATION sample, and fine-tuning the model using the TEST data are briefly addressed. The steps involved in applying the user-friendly SAS macro applications for importing PC worksheet files into SAS datasets and randomly splitting the entire database into TRAIN, VALIDATION, and TEST data are shown by using the example of a small business dataset called “fraud”.

References


Suggested Reading


© 2003 by CRC Press LLC