Paper 190-25

Using a Fuzzy Merge to Combine Customer Performance Information Across Accounts

Helen-Jean Talbott, CitiFinancial, Credit Policy Department, Baltimore, MD

ABSTRACT

One of our primary goals for credit policy is to establish criteria which select applicants that have the highest probability of repaying their loans to our company. These criteria rely on models developed using information from loan applications (such as income and homeownership) and from our experience with them as customers (whether they paid as agreed). One of the first steps in the model development process involves sampling relevant data and classifying the accounts based on performance (payment history). Customers frequently renew and refinance loans within the sample period and it is important to track the performance of these customers from one loan to the next. Matching the old loan with the subsequent loan becomes difficult because the closing of one loan and the opening of the next loan can occur several days apart and the customer may have additional loans open with the company. Simple match-merge techniques will not work because there are not common values of a key variable for matching between the two accounts. The purpose of this paper is to explain how to use a fuzzy merge to match the old loan with the subsequent loan in order to combine the performance data between the two accounts.

INTRODUCTION

Our company lends money to people for many different purposes. One of the primary goals of our Credit Policy Department is to establish criteria which select applicants that have the highest probability of repaying their loans to our company. These criteria rely on statistical models that are developed using information from loan applications (such as the customer's income and whether they own or rent their home) and from our experience with them as customers (whether they paid as agreed). These models must be redeveloped every few years so that they reflect the characteristics of the current population of customers.

One of the first steps in the model development process involves sampling relevant data and classifying the accounts as "good", "bad", or "indeterminate" based on performance (payment history). Customers frequently renew and refinance loans within the sample period. It is important to track the performance of these customers from one loan to the next so that we can relate the performance over time to the information we had when the first loan was made. Matching the old loan with the subsequent loan becomes difficult because the closing of one loan and the opening of the next loan can occur several days apart and the customer may have additional loans open with the company. Simple match-merge techniques will not work in this situation because there are not common values of a key variable, such as account number or prior account number, for matching between the two accounts.

The purpose of this paper is to explain how to use a fuzzy merge to match the old loan with the subsequent loan in order to combine the performance data between the two accounts.

FUZZY MERGES

Fuzzy merge techniques are a good solution to this problem of combining customer performance information across accounts. Traditional match-merging in SAS® data steps will only match observations that have identical values for the key variables. For our account-matching problem, we have identical values for social security number, but no link between the account numbers. The dates the accounts closed and opened are close, but seldom exactly the same.

In fuzzy merges, observations are combined based on having close or similar values rather than identical values for the key variables. Typical examples of key variables used in fuzzy merges are times, dates, and names. Each fuzzy merge solution must be carefully tuned to the requirements of the merge. Each solution is different and it is essential to understand the nature of the observations being matched in order to write the code needed to perform a fuzzy merge. In this paper, the fuzzy merge deals with matching observations based on determining which date is the most recent to another date. Other fuzzy merges, such as those requiring matching on names that sound similar, require other techniques. Please refer to papers written by Malachy J. Foley (cited in the references) to see further examples of using SAS software to perform fuzzy merges.

EXPLANATION OF PROGRAM

The following program was developed using MVS SAS release 6.09 Enhanced on a mainframe computer. For this model development effort, the old loans are sales finance loans that are used to borrow money for consumer goods such as televisions and furniture. Later, these customers often want to borrow more money for larger purchases, such as a house. In that case, the sales finance loan is closed and "converted" into a real estate loan. The goal of this program is to identify which real estate account originated from the sales finance account so that the performance of new account can be associated with the sales finance account for model development. The general approach used in this program is to interleave the sales finance and real estate accounts in order by date, examine the accounts to determine which accounts match, create a new key variable for matching, and then perform match-merging based on the new key variable.

JCL for this program includes DD statements referencing the input files. File SCORECARD.SAMPLE contains all the sales finance accounts selected to be the sample for the model development. Some of these sales finance accounts were closed and converted to a real estate account. These subsequent real estate accounts will be found in the file ACCOUNT.MASTER which contains information on all the accounts for the company.

```
--- more JCL statements ---

//IN1      DD DSN=SCORECARD.SAMPLE.DATA,
//            DISP=SHR
//IN2      DD DSN=ACCOUNT.MASTER.DATA,
//            DISP=SHR
//SYSIN    DD *
*****************************************;
* BDC203.SCORE.CNTL(SFCONV)               *;
*****************************************;
* IN1      DD DSN=SCORECARD.SAMPLE.DATA   *;
* IN2      DD DSN=ACCOUNT.MASTER.DATA     *;
*                                         *;
*                      DEC. 28, 1999      *;
*                      HELEN-JEAN TALBOTT *;
*                                         *;
* SF SCORECARD SAMPLE                     *;
* PROGRAM TO MATCH CLOSED SF ACCOUNTS WITH *;
* RE ACCOUNTS BOOKED IN SIMILAR TIME      *;
* PERIOD  (SF CONVERSIONS)                *;
*                                         *;
*****************************************;
```

The first step selects the sales finance accounts that were closed (OPEN='0') due to conversion (PO_RE='CONV'). The variables OFFICE and ACCTN identify the office within the company and the customer account number at the time the account was closed. The new variable SORTDATE is set equal to the date the loan closed (PO_DY) and will be used to put the accounts in proper order during the fuzzy merge. The variable containing the customer name (CNAME) is renamed to OLDNAME. Later in the program the customer name on the old account is compared with the customer name on the new account to validate the matching of the two accounts. The resulting data set (CONV) is sorted by customer social security number (SSN) and SORTDATE.

```
DATA CONV;
   *** WHERE: ACCOUNTS CLOSED DUE TO
              CONVERSION ***;
   SET IN1.A(WHERE=(OPEN='0' AND
                    PO_RE='CONV')
             KEEP=OPEN PO_RE PO_DY OFFICE
                  ACCTN SSN CNAME);

   SORTDATE=PO_DY;

   RENAME CNAME=OLDNAME;

PROC SORT DATA=CONV;
   BY SSN SORTDATE;
```

The second step selects a group of real estate accounts (PRODUCT = 'RE') for matching to the closed accounts selected above. These accounts were opened after the beginning of the scorecard sample period ('01NOV96'D <= NOTE_DY) and their source of business shows that they were converted from sales finance accounts (SOB='SFCONV'). The variables DLQ30, DLQ60, and DLQ90 are performance indicators that show how many times the account has been delinquent 30, 60, and 90 days, respectively. The new variable SORTDATE is set equal to the date the account was opened (NOTE_DY) because we want to match the opening of these real estate accounts as closely as possible to the closing of the accounts selected in first step above. The resulting data set (BOOKED) is sorted by SSN and SORTDATE.

```
DATA BOOKED;
   *** WHERE: ACCOUNTS OPEN IN SAMPLE PERIOD
              AND SOURCE IS SFCONV ***;
   SET IN2.A(WHERE=('01NOV96'D <= NOTE_DY
                    AND
                    PRODUCT = 'RE'
                    AND
                    SOB='SFCONV')
             KEEP=NOTE_DY PRODUCT SOB SSN
                  CNAME DLQ30 DLQ60 DLQ90);

   SORTDATE=NOTE_DY;

PROC SORT DATA=BOOKED;
   BY SSN SORTDATE;
```

The third step determines which accounts are matches. Data sets CONV and BOOKED are combined by interleaving based on customer social security number and SORTDATE. This interleaving puts the accounts in correct order for each customer, with the closed sales finance account first followed by real estate accounts in the order the accounts were opened. The interleaving also creates FIRST.SSN that will be used later in the data step.

```
DATA CONV2(KEEP=PO_RE PO_DY OFFICE ACCTN SSN
               OLDNAME SORTDATE MATE)
     BOOKED2(KEEP=NOTE_DY PRODUCT SOB SSN
                  CNAME SORTDATE MATE
                  DLQ30 DLQ60 DLQ90);
   SET CONV(IN=X)
   BOOKED(IN=Y);
   BY SSN SORTDATE;
```

Accounts are labeled by the variable FILE to show whether the observation is a one of the closed sales finance accounts (FILE='C') or one of the real estate accounts (FILE='B'). Two variables (FOUND and MATE) are initialized to zero using the retain statement. FOUND is a flag that shows whether the sales finance account has been located in the interleaved collection of accounts. MATE is a counter and serves to assign the same sequential number to the sales finance account and matching real estate account. Later in the program, MATE will serve as a new key variable for match-merging.

```
IF X THEN FILE='C';
IF Y THEN FILE='B';

RETAIN FOUND 0
       MATE 0;
```

Processing begins for the first observation for the customer (as identified by FIRST.SSN). The flag FOUND is set to zero because at this point the sales finance account has not been located. If this observation is a real estate account (FILE='B') then the observation is discarded because that loan would have been opened prior to the sales finance account. If this observation is the sales finance account (FILE='C') then the observation is output to file CONV2 and the FOUND flag is set to 1. Program returns to fetch the next observation.

```
IF FIRST.SSN THEN DO;
   FOUND=0;
   IF FILE='B' THEN DELETE;
   IF FILE='C' THEN DO;
      OUTPUT CONV2;
      FOUND=1;
      RETURN;
      END;
   END;
```

Additional observations for the same customer are processed.
If the sales finance account has not been located (FOUND=0), then the same actions occur as described above: real estate accounts are discarded, and if the sales finance account is located, it is output to data set CONV2 and the FOUND flag is set to 1. Program returns to fetch the next observation.

If the sales finance account has been located (FOUND=1), then the action depends on whether the observation is sales finance or real estate.

If the account is real estate (FILE='B'), then the match for the sales finance account now has been found. The observation is output to data set BOOKED2, the FOUND flag is reset to zero, the MATE counter is incremented (MATE+1) and the program returns to fetch the next observation. Both the sales finance account and its matching real estate account will have the same value for MATE and this is important in the merge in the next step. Any additional real estate accounts encountered sequentially in the next few observations will be discarded because the match has already been found (this action is controlled by the FOUND=0 section of code described above).

If the account is another sales finance account for the same customer (FILE='C'), then the sales finance account is output to data set CONV2. The FOUND flag is set to 1 and this enables the program to look for new matches in the real estate accounts when the program returns to fetch the next observation.

```
ELSE DO;
   IF FOUND=0 THEN DO;
      IF FILE='B' THEN DELETE;
      IF FILE='C' THEN DO;
         OUTPUT CONV2;
         FOUND=1;
         RETURN;
         END;
      END;
```

```
      IF FOUND=1 THEN DO;
         IF FILE='B' THEN DO;
            OUTPUT BOOKED2;
            FOUND=0;
            MATE+1;
            RETURN;
            END;
         IF FILE='C' THEN DO;
            OUTPUT CONV2;
            FOUND=1;
            RETURN;
            END;
         END;
      END;
```

The fourth step merges the performance information from the real estate accounts (identified in the third step) with the corresponding sales finance accounts. Data sets CONV2 and BOOKED2 are each sorted by customer social security number and MATE. Next, these two data sets are merged by SSN and MATE. Variable BETWEEN is created to examine the length of time between the sales finance account closing date and the real estate opening date. Variable NAMEAGRE examines whether the first 5 characters of the customer name agree comparing the old and new accounts. Distributions for BETWEEN and NAMEAGRE are reported in the FREQ procedure at the end of the program. Data set COMBINE contains the sales finance accounts with the performance data obtained from the matching real estate accounts. Data sets NOMATE and MATE were created at the same time as COMBINE to get a measure of how well we were able to find matches for the sales finance accounts.

```
PROC SORT DATA=CONV2;
   BY SSN MATE;

PROC SORT DATA=BOOKED2;
   BY SSN MATE;

DATA COMBINE NOMATE MATE;
   MERGE CONV2(IN=X)
         BOOKED2(IN=Y);
   BY SSN MATE;
   IF X;

   *** TIME BETWEEN PAID OUT AND BOOKED ***;
   BETWEEN=(NOTE_DY-PO_DY)/30;

   *** DO NAMES MATCH? ***;
   IF SUBSTR(OLDNAME,1,5) = SUBSTR(CNAME,1,5)
   THEN NAMEAGRE='YES';
   ELSE NAMEAGRE='NO';

   OUTPUT COMBINE;
   IF NOT Y THEN OUTPUT NOMATE;
   ELSE OUTPUT MATE;

PROC FORMAT;
   VALUE TIMES
   LOW-<10 = '< 10'
   10-<30  = '10 - 29'
   30-<60  = '30 - 59'
   60-<90  = '60 - 89'
   90-HIGH = '90+';

PROC FREQ DATA=MATE;
   TABLES BETWEEN/LIST MISSPRINT;
   TABLES NAMEAGRE/LIST MISSPRINT;
   FORMAT BETWEEN TIMES.;
TITLE1 'MATCH CLOSED SALES FINANCE ACCOUNTS T
O SUBSEQUENT LOANS';
TITLE2 'SF BOOKED NOV96 - MAY97, OCT98 PERFOR
MANCE DATA';
TITLE3 '(1) TIME BETWEEN LOANS, (2) CHECK IF
CUSTOMER NAMES MATCH';
FOOTNOTE 'CREDIT POLICY MIS REPORT H757';

RUN;
```

CONCLUSION

This paper demonstrates how to use fuzzy merge techniques to combine customer performance information across accounts. Traditional match-merging techniques alone could not be used to solve this problem because the key variable (loan open and close dates) did not match exactly for the observations being merged. The approach used in this paper was to interleave the accounts in order by date, examine the accounts to determine which accounts matched, create a new key variable for matching, and then perform the match-merge based on the new key variable. The process worked smoothly and the desired results were obtained.

REFERENCES

Foley, Malachy J. (1997), "Advanced Match-Merging: Techniques, Tricks, and Traps" in Proceedings of the Twenty-Second Annual SAS Users Group International Conference, p 199-206.

Foley, Malachy J. (1998), "Match-Merging: 20 Some Traps and How to Avoid Them" in Proceedings of the Twenty-Third Annual SAS Users Group International Conference, p 277-286.

Foley, Malachy J. (1999), "Fuzzy Merges: Examples and Techniques" in Proceedings of the Twenty-Fourth Annual SAS Users Group International Conference, p 296-305.

SAS Institute Inc. (1990), SAS Language, Cary, NC: SAS Institute Inc., 1042 pp.

ACKNOWLEDGMENTS

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Helen-Jean Talbott
CitiFinancial
300 St. Paul Place, BSP10A
Baltimore, MD 21202 USA