Paper 3103-2015

Macro to compute best transform variable for the model

Nancy Hu, Discover Financial Service

ABSTRACT

This study is intended to assist Analysts to generate the best of variables using simple arithmetic operators (square root, log, loglog, exp and rcp) and such as monthly amount paid, daily number of received customer service calls, or annual number total sales for a specific product. During a statistical data modeling process, Analysts are often confronted with the task of computing derived variables using the existing available variables. The advantage of this methodology is that the new variables may be more significant than the original ones. This paper gives a new way to compute all the possible variables using a set of math transformation. The codes include many SAS® features that are very useful tools for SAS programmers to incorporate in their future codes Such as %SYSFUNC, SQL, %INCLUDE, CALL SYMPUT, %MACRO, SORT, CONTENTS, MERGE, MACRO _NULL_, as well as %DO …%TO … and many more. I demonstrate the syntax and mechanics of the macro using following examples.

INTRODUCTION

These codes are used for logistical regression model.

There are four steps to transformation one variable into 5 derived variables and pick the best one based on the information value and store it into final data.

Table 1, sale data –age_of_house. sold indiator, loation, bed_room, style, listdaystosale and location.

sold indicator	size_sqft	location	bed_rooms	bath_room	style	listdaysTosale	location	age_of_house
yes	2393	chicago	3	2	condo	130	suburbur	50
no	3628	Houston	5	3	single family	200	city	20
yes	2200	Washtington	3	2	condo	100	city	60
no	2000	chicago	3	2	town house	120	city	60
yes	1890	Houston	4	3	single family	100	suburbur	87
yes	1600	Washtington	2	1	town house	80	city	70
yes	2400	chicago	3	1	single family	110	suburbur	50
yes	2500	Houston	4	2	single family	120	suburbur	35
yes	3600	Washtington	3	2	town house	130	city	50
yes	3500	chicago	6	2	single	250	suburbur	25

					family			
yes	3400	Houston	6	3	single family	160	city	25
yes	843	Washtington	2	1	condo	60	suburbur	50

Step 1, Five mathematical function of macro to transform one variable into 5 new variables.

macro 1, Log transformation. See figure 1.a

```
%MACRO LOG_FUNC(VN,LO_LIM=,HI_LIM=,D=&DELTA);

/* FUNCTION VNL=LOG(VN-LO_LIM+D) */
%LET VNN1 = &VN_6&SX1;
IF &VN < 0 THEN &VNN1 = 0;
ELSE &VNN1 = LOG(&VN+&D);

%IF &HI_LIM NE %THEN %DO;
      IF &VN > &HI_LIM THEN &VNN1 = LOG(&HI_LIM+&D);
%END;
IF LENG <= 34 THEN VARLAB1 = SUBSTR(VARLAB,1,LENG)||' (LOG)';
ELSE VARLAB1 = SUBSTR(VARLAB,1,34)||' (LOG)';

%MEND LOG FU
```

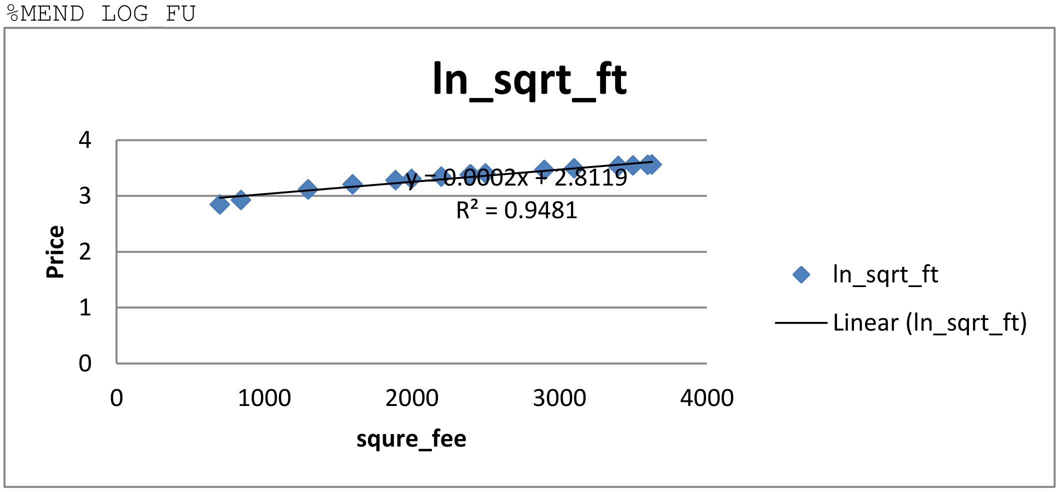

Figure 1-a, see log transformation.

Macro 2, Log log transformation.

```
%MACRO LLG_FUNC(VN,LO_LIM,HI_LIM=,D=&DELTA);

/* FUNCTION VNL=LOG(LOG(VN+&D+1)) */
%LET VNN1 = &VN_6&SX2;
 IF &VN < 0 THEN &VNN1 = LOG(LOG(3));
ELSE &VNN1 = LOG(LOG(&VN+&D+2));
%IF &HI_LIM NE %THEN %DO;
      IF &VN > &HI_LIM THEN &VNN1 =LOG(LOG(&HI_LIM+&D+2));
%END;
IF LENG <= 33 THEN VARLAB2 = SUBSTR(VARLAB,1,LENG)||' (LLOG)';
```

```
ELSE VARLAB2 = SUBSTR(VARLAB,1,33)||" (LLOG)";

%MEND LLG_FUNC;
```

Macro 3, Exponential transformation.

```
%MACRO EXP_FUNC(VN,LO_LIM=,HI_LIM=,A=0,B=);

/* FUNCTION VNE=EXP(A*VN) */ ;
%LET VNN1 = &VN_6&SX3;
%LET RANGE = %SYSEVALF(&B-&A);
%IF &RANGE = 0 %THEN %LET RANGE = 1;
IF &VN < 0 THEN &VNN1 = -1;
      ELSE &VNN1 = -EXP((-1) * &VN / &RANGE );
%IF &HI_LIM NE %THEN %DO;
      IF &V N > &HI_LIM THEN &VNN1 = -EXP((-1) * &HI_LIM / &RANGE );
%END;

&VNN1 = ROUND(&VNN1,0.00001);
IF LENG <= 34 THEN VARLAB3 = SUBSTR(VARLAB,1,LENG)||' (EXP)';
ELSE VARLAB3 = SUBSTR(VARLAB,1,34)||' (EXP)';

%MEND EXP_FUNC;
```

Macro 4, Square Root transformation.

```
%MACRO SQR_FUNC(VN,LO_LIM=,HI_LIM=,A=0);

%LET VNN1 = &VN_6&SX4;IF &VN < 0 THEN &VNN1 = 0;
ELSE &VNN1 = SQRT(&VN+&A);
%IF &HI_LIM NE %THEN %DO;
      IF &VN > &HI_LIM THEN &VNN1 = SQRT(&HI_LIM+&A);
%END;
IF LENG <= 33 THEN VARLAB4=SUBSTR(VARLAB,1,LENG)||' (SQRT)';
ELSE VARLAB4 = SUBSTR(VARLAB,1,33)||' (SQRT)';

%MEND SQR_FUNC;
```

Macro 5, RCP transformation see figure 1-b.

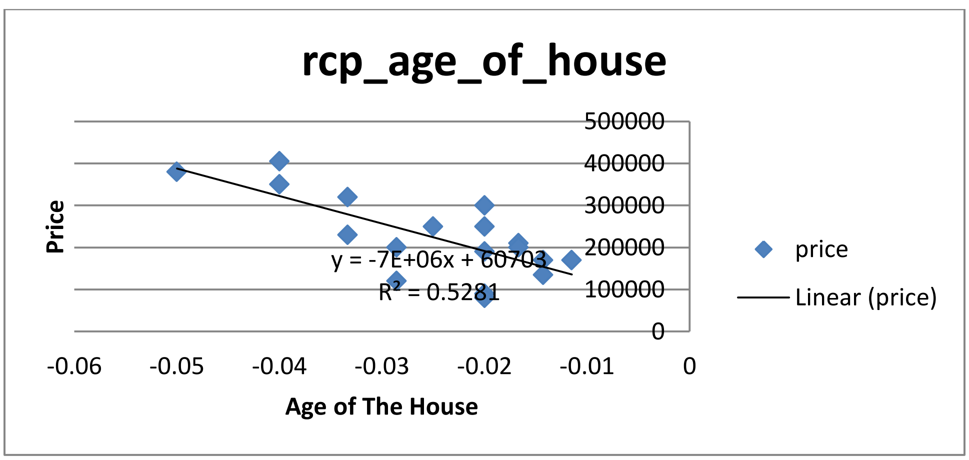

Figure 1-b, RCP function.

```
%MACRO RCP_FUNC(VN,LO_LIM=,HI_LIM=,D=&DELTA);
```

```
%VNAME_N(&VN,VN_6)
%LET VNN1 = &VN_6&SX5;
 IF &VN < 0 THEN &VNN1 = -1;
      ELSE &VNN1 = -1 / (&VN+&D);
%IF &HI_LIM NE %THEN %DO;
      IF &VN > &HI_LIM THEN &VNN1 = -1 / (&HI_LIM+&D);
%END;
IF LENG <= 34 THEN VARLAB5 = SUBSTR(VARLAB,1,LENG)||' (RCP)';
      ELSE VARLAB5 = SUBSTR(VARLAB,1,34)||' (RCP)';

%MEND RCP_FUNC;
```

Step 2, After variable transforming, call mapp macro to get each variable into same label and short type of variable.

```
%MACRO MAPP(DATA_IN=,DATA_OUT=,VARLIST=,PCNTILE=);
%GLOBAL VNN1;
DATA &DATA_OUT;
LENGTH VARLAB VARLAB1-VARLAB5 $40;
SET &DATA_IN;
%DO II=1 %TO &N_VAR;
      %IF (&&VAR&II NE &BAD AND &&VAR&II NE &WEIGHT_) %THEN %DO; CALL
LABEL(&&VAR&II,VARLAB);
            LENG=LENGTH(VARLAB);%LOG_FUNC(&&VAR&II,LO_LIM=&&P&II._&LEFT_PCT,H
      I_LIM=&&P&II._&RITE_PCT);
            %LET VAR&II._1=&VNN1;
            %LLG_FUNC(&&VAR&II,LO_LIM=&&P&II._&LEFT_PCT,HI_LIM=&&P&II._&RITE_
      PCT);
            %LETVAR&II._2=&VNN1;
            %EXP_FUNC(&&VAR&II,LO_LIM=&&P&II._&LEFT_PCT,
HI_LIM=&&P&II._&RITE_PCT,B=&&P&II._&RITE_PCT);
            %LET VAR&II._3=&VNN1;
            %SQR_FUNC(&&VAR&II,LO_LIM=&&P&II._&LEFT_PCT,HI_LIM=&&P&II._&RITE_
      PCT);
            %LET VAR&II._4=&VNN1;
            %RCP_FUNC(&&VAR&II,LO_LIM=&&P&II._&LEFT_PCT,HI_LIM=&&P&II._&RITE_
      PCT);
            %LET VAR&II._5=&VNN1;
                  %DO JJ=1 %TO 6;
                  CALL SYMPUT("LAB&&II._&JJ",VARLAB&JJ);
                  %END;
      %END;
%END;
%IF &DROP_ALL=1 %THEN %DO;
      DROP &VARLIST LENG VARLAB VARLAB1-VARLAB5;
%END;
%ELSE %DO;
      DROP LENG VARLAB VARLAB1-VARLAB5;
%END;
RUN;

%MEND MAPP;
```

This mapp macro can transfer one variable into 5 new derived variables with 5 new label see table 1A.

Example 2, Table 1A,

Original varible	log	loglog	sqrt	exp	rcp
a	a_l	a_LL	a_s	a_e	a_r
b	b_l	b_LL	b_s	b_e	b_r

Step 3, call logistical variable and pick the most significantly variables. See example3.

```
%MACRO SLCTVAR(SDATA,INDSN,DROPFILE,NUMKP=1,SLEVEL=0.05);

DATA SUBSET;
SET &INDSN;
      %IF &PORTION NE %THEN %DO;
            IF RANUNI(374521) <= &PORTION THEN OUTPUT;
      %END;
       RUN;
      %DO II=1 %TO &N_VAR;
            %IF &&P&II._&RITE_PCT NE . %THEN %DO;
                  PROC LOGISTIC DATA=SUBSET(KEEP=&BAD &&VAR&II &&VAR&II._1
            &&VAR&II._2 &&VAR&II._3 &&VAR&II._4 &&VAR&II._5 &WEIGHT_)
            OUTEST=EST1 NOPRINT;
                  MODEL &BAD =&&VAR&II &&VAR&II._1 &&VAR&II._2 &&VAR&II._3
            &&VAR&II._4 &&VAR&II._5 / SELECTION=FORWARD STOP=&NUMKP
            SLE=&SLEVEL;

            %IF &WEIGHT_ NE %THEN %DO;
                         WEIGHT &WEIGHT_;
            %END;
            RUN;

      DATA ONEVAR(KEEP=DROPVAR); LENGTH DROPVAR $8;
      SET EST1(WHERE=(_TYPE_='PARMS'));
      ARRAY VNNS &&VAR&II._1 &&VAR&II._2 &&VAR&II._3 &&VAR&II._4 &&VAR&II._5;
            DO K=1 TO DIM(VNNS);

                  IF VNNS{K} <= .Z THEN DO;
                  CALL VNAME(VNNS{K},DROPVAR); OUTPUT;
                  END;
            END;
      RUN;
      %END;
      %ELSE %DO;
      DATA ONEVAR(KEEP=DROPVAR);
      LENGTH DROPVAR $8;
      ARRAY VNNS &&VAR&II &&VAR&II._1 &&VAR&II._2 &&VAR&II._3 &&VAR&II._4
      &&VAR&II._5;
            DO K=1 TO DIM(VNNS);
                  VNNS{K} = 0;
                  CALL VNAME(VNNS{K},
                  DROPVAR);
                   OUTPUT;
            END; STOP;
            RUN;
```

```
      %END;
      PROC APPEND BASE=DROPLIST DATA=ONEVAR FORCE;
RUN;

%END;

DATA _NULL_;
      SET DROPLIST END=LAST;
      FILE "DROPFILE.dat";
       IF _N_=1 THEN PUT '%MACRO DROPLIST;';
      PUT DROPVAR;
      IF LAST THEN PUT '%MEND DROPLIST;';
RUN;

%INCLUDE "dropfile.dat";

DATA &SDATA;
      SET &INDSN;
      LABEL %DO II=1 %TO &N_VAR; &&VAR&II._1="&&LAB&II._1"
&&VAR&II._2="&&LAB&II._2" &&VAR&II._3="&&LAB&II._3" &&VAR&II._4="&&LAB&II._4"
&&VAR&II._5="&&LAB&II._5" %END;
      %STR(;);
      DROP %DROPLIST;
 RUN;

%MEND SLCTVAR;
```

Example 2, dropplist of this macro.

droplist
a_l
b_s
c_e
d_r

Step 4, call lgt_plot and calculate each variable for the information value and log odds.

```
%macro lgt_plot(dsn,bad,varname);
data temp;
set &dsn (keep=&bad &varname) end=last;retain n_bads 0;
      format odds 6.2 ;
       length odds_a $5;
      level=&varname;
      if &bad then n_bads=n_bads+1;
      if last then do;
            call symput('n_bads',trim(left(n_bads)));
            call symput('n_obs',trim(left(_n_)));
            odds=(n_bads)/(_n_-n_bads+0.1);
            odds_a=trim(left(odds));
            call symput('o_odds', odds_a);
      end;
```

```
      drop n_bads odds;
run;

proc means data=temp noprint;
      var &varname &bad;class level;
      output out=temp2 min=min min2 sum=&varname &bad max=max max2 ;
run;

sql noprint;
      select count(*) into :macro1 from temp;
quit;

data temp3;
set temp2(firstobs=2) end=last;
      format odds 3. logodds log_b_g 6.2 info 7.5;
      retain tot_info 0;
      odds=(&bad+0.1)/(s_wt-&bad+0.1);
      logodds=log(odds);
      WOE=logodds;
      log_b_g=logodds;

      p_good=(s_wt&bad+0.1)/(&n_obs&n_bads+0.1);p_bad=(&bad+0.1)/(&n_bads+0.1
);

      if info = . then info = 0;
      tot_info = tot_info + info;
      if last then do;
      if tot_info <= 0.00001 then tot_info = 0.00001;
      tot_info = round(tot_info,0.00001);
      call symput('tot_info',tot_info);
      end;

      data temp3;
      set temp3;
      format info_rt 4.1;
      info_rt = info/(&tot_info);
run;

proc means data=temp3 noprint;
      var logodds &varname;
      output out=null css=logodds &varname;
run;

proc corr data=temp3 noprint outp=corr1;
      var logodds &varname;
run;

Data out.corr1;
      set corr1;
      length var $32 corr_a $5;
      format corr &varname 4.2;
      if _type_='CORR' and logodds = 1;
      var="&varname";
      corr=&varname;
      info=&tot_info;
      corr_a=trim(left(corr));
```

```
      call symput("&varname",trim(left(corr_a)));
      keep var corr info;
run;

%mend lgt_plot(dsn,bad,varname);
```

Example3, final variable list and related measurement

variable	information value	correlation
size_sqft_r	0.2	0.45
location	0.1	0.35
bed_rooms_l	0.3	0.2
bath_room	0.5	0.6
style	0.6	0.6
listdaysTosale	0.7	0.5
age_of_house	0.1	0.2
price	0.1	0.1

This final list of significant variable list can use to plug in the final logistical model. This example shows the ranking order of the derived variables with dependent variable.

CONCLUSION

This method of computing the derived variables can be useful for modeling purposes and save time during the variable creation stage. These macros are very straightforward but the user needs a strong understanding of SAS programming especially in SAS MACRO in order to fully take advantage of the codes

Although not shown in the examples above, the macro will work for nonlinear variable (i.e. linear from the alternative model. In addition to normal bug fixing, future work on the macro may include improving the overall coding efficiency and enhancing the handling of missing values. Fitting complex models and/or models to large data sets is the chief challenge because of the iterative nature of the macro. My hope is that in the not-too-distance future, the parametric will be built-in component.

REFERENCES

SAS Institute Inc. 2011. *SAS/CONNECT® 9.3 User's Guide*. Cary, NC: SAS Institute Inc. Available at http://support.sas.com/documentation/cdl/en/connref/63066/PDF/default/connref.pdf

SAS Institute Inc. 2012. *SAS® 9.3 SQL Procedure User's Guide*. Cary, NC: SAS Institute Inc. Available at http://support.sas.com/documentation/cdl/en/sqlproc/63043/PDF/default/sqlproc.pdf

SAS Institute Inc. 2012. *SAS® 9.3 Language Reference: Concepts*. Cary, NC: SAS Institute Inc. Available at http://support.sas.com/documentation/cdl/en/lrcon/65287/PDF/default/lrcon.pdf

Dickstein, Craig, Pass, Ray, & Davis, Michael. 2007. "DATA Step vs. PROC SQL: What's a neophyte to do?". *Proceedings of the SAS Global Forum 2007 Conference.* Cary, NC. Available at http://www2.sas.com/proceedings/forum2007/237-2007.pdf

Lafler, Kirk Paul. 2014. "Powerful and Hard-to-Find PROC SQL Features". *Proceedings of the SAS Global Forum 2014 Conference.* Cary, NC. Available at http://support.sas.com/resources/papers/proceedings14/1240-2014.pdf

ACKNOWLEDGMENTS

We appreciate the feedback, suggestions and encouragement from my co-worker, Ming Zhang, Alicia, Anani and Joel. We also thank my Manager, Mamta Kakkar and Yang and Kotha, for their support and patience.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the authors at nanyhu@discover.com