Paper 7860-2016

Finding and Evaluating Multiple Candidate Models for Logistic Regression

Bruce Lund
Magnify Analytic Solutions, a Division of Marketing Associates, LLC
Detroit MI, Wilmington DE, and Charlotte NC

ABSTRACT

Logistic regression models are commonly used in direct marketing and consumer finance applications. In this context the paper discusses two topics about the fitting and evaluation of logistic regression models. Topic #1 is a comparison of two methods for finding multiple candidate models. The first method is the familiar “best subsets” approach. Then best subsets is compared to a proposed new method based on combining models produced by backward and forward selection plus the predictors considered by backward and forward. This second method uses HPLOGISTIC with selection of models by SBC (Schwarz Bayes). Topic #2 is a discussion of model evaluation statistics to measure predictive accuracy and goodness-of-fit in support of the choice of a final model. Base SAS® and SAS/STAT are used.

INTRODUCTION

This paper focuses on fitting of binary logistic regression models for direct marketing, customer relationship management, credit scoring, or other applications where (i) samples for modeling are large, (ii) there are many predictors, and (iii) the emphasis is on using the models to score future observations.

There are two main topics:

Topic 1: Finding a large number of promising candidate models for comparison and evaluation

Two methods are discussed:

METHOD1: Best Subsets (SELECTION=SCORE) of PROC LOGISTIC

METHOD2: PROC HPLOGISTIC using METHOD=BACKWARD and using METHOD=FORWARD and collecting the models that are created as predictors are either selected or considered

Either method can produce far more models than are needed for comparative evaluation on a validation data set. To reduce the number of models to a manageable amount, the models are ranked by the Schwarz Bayes criterion (SBC).[1] The best 10 to 30 models might then be selected for evaluation. However, METHOD1 cannot handle a large number of nominal or discrete variables that would be designated as class variables. We provide a work-around, but this work-around has limitations.

Topic 2: Evaluating the best candidate models on a validation data set and final model selection

ASSUMPTIONS:

It is assumed there is an abundant population from which to sample and that large sub-samples have been selected for the **training, validation, and test data sets**. It is also assumed that a large set of potential predictor variables X_1 - X_K have been given values as of an observation date. Finally, it is assumed that the **target Y** has been carefully defined and assigned a value of 0 or 1 where this value is determined by whether an event occurred within a specified time period following the observation date.

SCHWARZ BAYES CRITERION

In direct marketing, customer relationship management, or credit scoring (where there are large training data sets and many candidate predictors) it is easy and tempting to over fit a logistic regression model. It is natural to want to use all the information in the fitting of models that has been uncovered through data discovery. But the impact of using “all the data” in fitting a logistic model may be the creation of data management overhead with either minimal or no benefit for out-of-sample prediction by the model.

[1] In the case of Best Subsets the ranking is done by using a proxy for SBC.

Of course, a purpose of a validation data set is to detect over-fitting. But a "penalized measure of fit" can help to detect over-fitting before proceeding to validation. A well-known penalized measure of fit for logistic regression models is the Schwarz Bayes criterion (SBC).[2] The formula is given below:

$$SBC = - 2 * LL + \log(n) * K$$

where LL is log likelihood of the logistic model, K is degrees of freedom in the model (including the intercept) and n is the sample size.

The theory supporting the Schwarz Bayes criterion is complex, both conceptually and mathematically. For a logistic modeling practitioner the primary practical consequence of the theory is that a model with smaller SBC value is preferred over a model with a larger SBC value.[3]

PRE-MODELING DECISIONS AND DEGREES OF FREEDOM

The value of "K" (degrees of freedom in the model) for computing the SBC of a logistic regression model depends on decisions that the modeler makes before actual model fitting begins. Specifically, the modeler decides how to transform a predictor X where this decision is based on a preliminary bivariate analysis of X against the target Y. It is the use of the preliminary analysis that affects the degrees of freedom. In this regard two topics are discussed in the following sections.

1. The weight-of-evidence (WOE) transformation of a nominal or discrete predictor
2. A non-linear transformation of a continuous predictor X, such as log(X)

TRANSFORMING BY WOE AND DEGREES OF FREEDOM

A character or numeric predictor C with L levels (distinct values) can be entered into a logistic regression model with a CLASS statement or as a collection of dummy variables.[4] Typically, L is 15 or less.

```
PROC LOGISTIC; CLASS C; MODEL Y = C <and other predictors>;
      or
PROC LOGISTIC; MODEL Y = C_dum_<k> where k = 1 to L-1  <and other predictors>;
```

These two models produce exactly the same probabilities.

An alternative to CLASS / DUMMY coding of C is the weight-of-evidence (WOE) transformation of C.

For the predictor C and target Y of TABLE 1 the weight-of-evidence transformation (or recoding) of C is given by the right-most column in the table.

TABLE 1 – WEIGHT OF EVIDENCE TRANSFORMATION OF C

C	Y = 0 "B_k"	Y = 1 "G_k"	Col % Y=0 "b_k"	Col % Y=1 "g_k"	**WOE= Log(g_k/b_k)**
C1	2	1	0.250	0.125	-0.69315
C2	1	1	0.125	0.125	0.00000
C3	5	6	0.625	0.750	0.18232

The formula for the transformation is: If C = "C_k" then C_woe = log(g_k / b_k) for k = 1 to L where g_k, b_k > 0.

WOE coding is preceded by binning (or coarse classing) of the levels of predictor C. A discussion of binning is given by Finlay (2010 p. 146). A SAS macro for binning is given in Lund and Brotherton (2013).

WOE Coding vs. CLASS / DUMMY Coding

If C is the only predictor in a logistic regression model, then the same probabilities are produced by the two models: (i) the WOE transformed predictor, and (ii) the CLASS / DUMMY coding.

[2] SBC is also called BIC (Bayes Information Criterion). "BIC" appears in PROC HPLOGISTIC output.

[3] An introductory discussion of "Information Criteria" (SBC, AIC, and more) is given by Dziak, et al. (2012).

[4] "CLASS C;" creates a coefficient in the model for each of L-1 of the L levels. The modeler's choice of "reference level coding" determines how the L^{th} level enters into the calculation of the model scores. See SAS/STAT(R) 14.1 User's Guide (2015), LOGISTIC procedure, CLASS statement.

Now other predictors X1 – XK are added to form the models (A) and (B):

```
Model A: PROC LOGISTIC; CLASS C; MODEL Y = C X1 - XK;
Model B: PROC LOGISTIC; MODEL Y = C_woe X1 - XK;
```

In this case:

- Models (A) and (B) produce different probabilities
- Model (A) has greater log-likelihood (better fit).

Model (A) has better fit because the L-1 coefficients for C allow for greater interaction (real or spurious) with other predictors in the logistic regression model than does the single coefficient for C_woe.[5]

See **Appendix A** for several examples.

Despite the short-fall in fit of WOE transformations, the choice of using WOE vs. CLASS / DUMMY is not a pivotal decision in enabling the building of good logistic regression models. The pros and cons of the two choices are discussed by Finlay (2010, pp. 155-159) and Thomas (2009, pp. 77-78). Siddiqi recommends the use of WOE coding for credit risk models where he shows how WOE coding naturally supports score-card development. See Siddiqi (2006 pp. 91-92, 116).

Degrees of Freedom for WOE Coded Variable

A WOE predictor could not be assigned more degrees of freedom than the corresponding CLASS / DUMMY predictor since a WOE recoding produces a model with less fit. But should it have less d.f. when considering the equality with CLASS / DUMMY coding in the case of the one-variable model?

This leads to a working rule:

If C has L levels, then its WOE transformation adds L-1 degrees of freedom to a model.

Corrected degrees of freedom for a WOE coded predictor must be taken into account when computing Schwarz Bayes criterion (SBC) of a logistic regression model. This is especially a requirement when SBC is used to rank multiple candidate models and this ranking is used to eliminate low-ranked models from further study. Generally, the working rule will over-penalize (inflate SBC) in the case of WOE predictors.[6]

TRANSFORMATIONS OF CONTINUOUS PREDICTORS AND THE DEGREES OF FREEDOM

Often a modeler performs a preliminary bivariate analysis of a continuous predictor X against the target variable Y. This analysis may lead to the selection of a transformation for predictor X, such as "log(X)", to be used when fitting the logistic regression model.

A question arises: If log(X) is selected by this preliminary analysis and entered into a logistic regression model, are 2 degrees of freedom being used (one for the coefficient and one for the choice of transformation)?

I think the answer is "yes" and this "yes" answer affects the number of degrees of freedom when computing the Schwarz Bayes criterion (SBC) of a logistic regression model.[7] However, in SAS programs which are presented in this paper, the user will have a choice of making or not making this adjustment.

MULTIPLE PROMISING CANDIDATE MODELS FOR COMPARISON AND EVALUATION

The building of a logistic regression model is often an iterative process of fitting models and looking at the results. It is better to replace the iterative process with an automated and structured process.

[5] I believe the log-likelihoods of models (A) and (B) are equal if C_woe is uncorrelated with each of X1 to XK.

[6] Since LL for models with WOE predictors (vs. CLASS) can be less, is there a general rule for assigning fewer than L-1 d.f. to WOE's. The question is content dependent and cannot be answered without knowledge of other predictors in the model. Perhaps a heuristic formula might be developed of the following form: $\text{d.f.} = 1 + (L-2)^p$ where $L > 1$ and $0 < p \leq 1$ and the exponent "p" somehow depends on the count of predictors in the model. A further complex refinement to the formula might somehow involve the variance inflation factor (VIF) of a WOE predictor. Development and implementation of a VIF refinement would be a challenge.

[7] The degrees of freedom question is discussed in a survey article by Babyak (2004, see p. 417).

A strategy for automated, structured model development involves these two steps:

Step 1: Use an automated process to find "**M**" promising candidate models where M might be between 10 and 30. These models are found by modeling on the training data set.

Step 2: Use a structured evaluation of these M models on the validation data set with regard to:
(i) parsimonious fit, (ii) goodness-of-fit, (iii) predictive accuracy, and, (iv) more subjectively, satisfying business requirements.

Following Steps 1 and 2, the "final" model is selected and a final measure of performance of this model is made on the test data set. The following section of the paper addresses Step 1.

Two automated processes for finding multiple candidate models are presented next.

METHOD 1: PROC LOGISTIC USING BEST SUBSETS AND PENALIZED SCORE CHI-SQUARE [8]

The Best Subsets method of finding multiple candidate models is realized by using PROC LOGISTIC with SELECTION option "SCORE". SCORE has 3 options: START, STOP, BEST.

```
PROC LOGISTIC; MODEL Y = <X's> / SELECTION = SCORE  START=s1  STOP=s2  BEST=b;
```

The SELECTION options "START" and "STOP" restrict the models to be considered to those where the number of predictors is between s1 and s2. Then for each k in [s1, s2] the option "BEST" will produce the b "best" models having k predictors. These b "best" models are the ones with highest score chi-square.[9]

The score chi-square is reported by PROC LOGISTIC in the report "Testing Global Null Hypothesis: BETA = 0". The score chi-square closely approximates the likelihood ratio chi-square.

Example of START, STOP, BEST:

If there are 4 predictors, X1 – X4 and START=1, STOP=4, and BEST=3, then a total of 10 models is selected as shown:

- From the four 1 variable models: {X1}, {X2}, {X3}, {X4}, take the 3 with highest score chi-square
- From the six 2 variable models: {X1 X2}, {X1 X3}, {X1 X4}, {X2 X3}, {X2 X4}, {X3 X4}, take the 3 with highest score chi-square
- From the four 3 variable models: {X1 X2 X3}, {X1 X2 X4}, {X1 X3 X4}, {X2 X3 X4}, take the 3 with highest score chi-square
- From the one 4 variable model: {X1 X2 X3 X4}, take the 1 model

The Computational Short-Cut Provided by Score Chi-Square

The SELECTION = SCORE can efficiently find the "best" models as directed by START, STOP, BEST because the score chi-square is computed for the models without actually solving the likelihood equations to find the maximum likelihood. When running SELECTION = SCORE the model coefficients are not found and log likelihood (LL) statistics for the model are not computed. This includes statistics that are derived from LL including SBC.

Penalized Score Chi-Square

A "best" model is best only within a set of models with the same number of predictors. For example, {X1} might have the largest score chi-square among 1 variable models but {X1, X2} is guaranteed to have a larger score chi-square simply because a predictor was added to the model.

To make the models comparable, a penalty term is needed to reflect the number of degrees of freedom in the model and the sample size. In the absence of SBC a substitute is "ScoreP", a penalized score chi-square defined by:

$$\text{ScoreP} = -\text{Score Chi-Sq} + \log(n) * K$$

where K is the degrees of freedom in the model (counting the intercept) and n is the sample size.

[8] This approach generally follows the approach of SAS Institute (2012, section 3.5) but with modifications.
[9] See Hosmer et al. (2013 p.15 and p. 42) for a brief discussion of score chi-square.

If all possible models are ranked by ScoreP and also by SBC, there is no guarantee that the rankings will be the same but the rankings will be very similar.[10]

Problem and Solution but Another Problem

The SELECTION = SCORE option does not support CLASS statements. A solution is to convert class variables into a set of dummy variables. But with even a modest number of class variables the conversion to dummies could make the total number of predictors to be 100 or more. Run time for PROC LOGISTIC increases exponentially as the number of predictors for SELECTION = SCORE becomes 50 and greater. This makes large scale dummy variable conversion not practical when using SELECTON = SCORE.[11]

A work-around is to transform class variables to weight-of-evidence coded predictors before running SELECTION = SCORE. The run-time issue might be solved but the calculation of ScoreP for the models from SELECTION = SCORE must take into account the implicit degrees of freedom of WOE predictors.

Best Subsets and Handling WOE Coded Predictors

Here is a three-step approach which addresses the degree of freedom issue for WOE coded predictors:

A. The modeler defines a SAS FORMAT that assigns L-1 degrees of freedom to a WOE coded predictor having L levels. In EXAMPLE 1, which follows below, this format is named “$df”.

B. A value for BEST is required so that *all* models between START and STOP are produced. All models are needed to insure that the best ScoreP model can be identified in Step C. For large K this requirement forces the modeler to restrict the values of START and STOP to avoid the generation of a count of models of the order of 2^K. The count of models equals $\binom{K}{\text{Start}}$ + … + $\binom{K}{\text{Stop}}$

C. In a following DATA step the ScoreP is computed while taking into account the degrees of freedom provided by a call to the format (discussed in Step A). This step changes the contribution to the ScoreP penalty term for each WOE coded predictor from log(n) to log(n)*(L-1). This follows the “working rule” of assigning L-1 d.f. to a WOE-coded predictor. Generally, this rule will over-penalize ScoreP in the case of WOE-coded predictors.

Finally, ScoreP can rank the models, and the best **M** (lowest ScoreP) can be selected for evaluation on the validation sample.

EXAMPLE 1: BEST SUBSETS

EXAMPLE 1 gives an example of the three-step approach. The data set in this example is “example1”. It has 100 observations. The first 4 observations are shown below and the complete “example1” data set is given in **Appendix B**. There is one character predictor C with 7 levels, a 0/1 target Y, and 3 numeric predictors X1, X2, X8. Of course, “example1” is much smaller than encountered in practical applications.

```
DATA example1;
input C$ Y X1 X2 X8 @@;
datalines;
D 0 10.2 6 0.8   A 1 12.2 6 0.6   D 1  7.7 1 0.6   G 1 10.9 7 0.2
<96 more observations>
;
```

[10] The connection between SBC and ScoreP is given here:
Likelihood ratio chi-square is:
LR $\chi2$ = $-2*LL_r - (-2*LL_f)$ where “r” is intercept-only model and “f” is model with covariates.
SBC = $-2*LL_f + \log(n)*K$ = - LR $\chi2$ + $\log(n)*K$ $-2*LL_r$
The SELECTION option gives Score $\chi2$. But: LR $\chi2$ ~ Score $\chi2$
So now, SBC ~ - Score $\chi2$ + $\log(n)*K$ $-2*LL_r$
Definition: ScoreP = - Score $\chi2$ + $\log(n)*K$ … by dropping the constant $-2*LL_r$

[11] SAS Institute (2012, chapter 3, page 78). SAS Training suggests running a preliminary PROC LOGISTIC with SELECTION = BACKWARD FAST to reduce the count of predictors to the point where SELECTION=SCORE can be run. But BACKWARD is unlikely to select all the dummies (or none of the dummies) associated with a class variable. This distorts the results of final binning of class variables. See Siddiqi (2006, pp 77–87) for the importance of final binning in credit modeling.

1. C_woe is coded.

```
* WOE coding for any class variables is completed first;
DATA example1; SET example1;
if C in ( "A" ) then C_woe = -0.809318612 ;
if C in ( "B" ) then C_woe = 0.1069721196 ;
if C in ( "C" ) then C_woe = 0.6177977433 ;
if C in ( "D" ) then C_woe = -0.403853504 ;
if C in ( "E" ) then C_woe = -1.145790849 ;
if C in ( "F" ) then C_woe = -0.703958097 ;
if C in ( "G" ) then C_woe = 1.4932664807 ;
run;
```

2. A format associates C_woe with its d.f. of 6. The format is enclosed in the macro %UPCASE_FORMAT to insure that the values to be formatted are in upper case. Upper case is useful for down-stream processing. In the case of "example1" there is only one predictor "C_woe" where degrees of freedom will need to be corrected before the ScoreP calculation. But if, for example, X1 is the result of a transformation that was found by a pre-modeling data analysis, then another row, "X1" = "2", could be entered in the format.

```
%MACRO UPCASE_FORMAT;
PROC FORMAT LIBRARY = Work;
VALUE $df
/* enter VARIABLES and DF ... converts to UPCASE */
%UPCASE
(
"C_woe" = "6"
)
Other = "1"
;
run;
%MEND;
%UPCASE_FORMAT;
```

3. %LET statements specify START, STOP, the data set name, the target, and the predictors. Further processing determines the number of predictors NPRED and computes a default value for BEST.

```
/* ++++++++++++++   PARAMETERS   +++++++++++++++++++++++++++ */
/* Set Parameters for SELECTION = SCORE and ScoreP calculations */
%LET DATASET = example1;
%LET INPUT = X1 X2 X8 C_woe;
%LET Y = Y;
* Number of model candidates printed by PROC PRINT statements;
%LET Pobs = 4;
* Determination of the number of predictor variables;
DATA _NULL_; SET &DATASET(obs=1);
  ARRAY Vars {*} &INPUT;
  NPRED = DIM(Vars);
  CALL SYMPUT('NPRED',NPRED);
/* Set START, STOP, BEST parameters for SELECTION = SCORE
   START is defaulted to 1. Change LET to override.
   STOP is set at the maximum number of variables. Change LET to override.
   BEST equals COMB(NPRED, FLOOR(NPRED/2)). All subsets are produced.
   ScoreP is computed using format-corrected DFs for all Models */
%LET START = 1;
%LET STOP = &NPRED;
/* END: ++++++++++   PARAMETERS   +++++++++++++++++++++++++++ */
DATA _NULL_;
  BEST = COMB(&NPRED, FLOOR(&NPRED/2)); PUT BEST= ;
  CALL SYMPUT('BEST',BEST);
run;
```

4. After PROC LOGISTIC is run, the ODS "Bestsubsets" data set is written to "Score_Out" and printed.

```
ODS OUTPUT Nobs = Nobs;
ODS OUTPUT Bestsubsets = Score_Out;
ODS EXCLUDE Bestsubsets;  /* Suppresses print out from PROC LOGISTIC */
PROC LOGISTIC DATA = &DATASET DESCENDING; MODEL &Y = &INPUT
  /SELECTION= SCORE START= &START STOP= &STOP BEST= &BEST;
PROC PRINT DATA = Score_Out(obs = &Pobs);
  VAR NumberOfVariables ScoreChiSq VariablesInModel;
Title1 "First &Pobs Observations";
Title2 "Before computing DF-corrected ScoreP";
```

TABLE 2 – FIRST 4 OBS. FROM PROC LOGISTIC SELECTION=SCORE ODS OUTPUT

Obs	NumberOfVariables	ScoreChiSq	VariablesInModel
1	1	12.3744	C_woe
2	1	4.2986	X8
3	1	3.9882	X2
4	1	1.2090	X1

5. A DATA step reads "Score_Out" and writes out "SCORE_Out2". During the DATA step processing the correct d.f. for C_woe is computed by usage of the format call. Associating the format to C_woe requires tricky string manipulation. Then ScoreP is computed.

   TABLE 3 shows all the one-predictor models. The model with highest score chi-square of 12.3744 is {C_woe}. But the lowest (i.e. best) ScoreP model with one predictor is {X8} with ScoreP = 4.9117. This illustrates that the value of option BEST in SELECTION = SCORE must be large enough to produce all possible models so that the lowest (best) ScoreP model can be found.

```
DATA _null_; SET Nobs;
  IF label = "Number of Observations Used";
  CALL SYMPUT('obs', N);
DATA _null_;
  IF (CEXIST("WORK.formats.df.formatc")) THEN CALL SYMPUT('df_exists', "Y");
   ELSE CALL SYMPUT('df_exists', "N");
DATA _null_;
  IF "&df_exists"="Y" THEN CALL SYMPUT('df_call',"+(put(varname,$df.)-1)");
   ELSE CALL SYMPUT('df_call', "");
DATA _null_;
  PUT "&df_call";
DATA Score_Out2; SET Score_Out;
  DROP I varname control_var;
  LABEL DF = "DF with Intercept";
  DF = numberofvariables + 1;
  DF_add = 0;
  DO I = 1 TO &NPRED;
    varname = upcase(left(trim(scan(VariablesInModel,I,' '))));
    IF varname > '' THEN DF_add = DF_add &df_call;
    END;
  DF = DF + DF_add;
  ScoreP = -scorechisq + log(&obs) * DF;
PROC PRINT DATA = Score_Out2(obs = &Pobs) LABEL;
Title1 "First &Pobs Observations";
Title2 "Correct DF and ScoreP";
Title3 "Before sorting by ScoreP";
run;
```

TABLE 3 – FIRST 4 OBS. WITH ScoreP

Obs	NumberOfVariables	ScoreChiSq	VariablesInModel	DF with Intercept	ScoreP
1	1	12.3744	C_woe	7	19.8618
2	1	4.2986	X8	2	4.9117
3	1	3.9882	X2	2	5.2221
4	1	1.2090	X1	2	8.0014

6. Finally, the models are sorted by ScoreP. The best "ScoreP" model is {X2, X8} with ScoreP = 4.8656.

```
PROC SORT DATA = Score_Out2 OUT = Score_Out3; BY ScoreP;
PROC PRINT DATA = Score_Out3(obs = &Pobs) LABEL;
Title1 "First &Pobs Observations";
Title2 "Sorted by ScoreP";
```

TABLE 4 – FIRST 4 OBS. WITH SORTED ScoreP

Obs	NumberOfVariables	ScoreChiSq	VariablesInModel	DF with Intercept	ScoreP
1	2	8.9499	X2 X8	3	4.8656
2	1	4.2986	X8	2	4.9117
3	1	3.9882	X2	2	5.2221
4	3	10.4563	X1 X2 X8	4	7.9644

In Summary, METHOD1 has Weaknesses:

These weaknesses are: (1) imprecise nature of d.f. correction for WOE-coded variables, (2) use of ScoreP as a proxy for SBC, and (3) restrictions on "START" and "STOP" to avoid generation of a count of models on the order of 2^K. Perhaps METHOD2 will be a better way to find candidate models.

METHOD2: BACKWARD AND FORWARD SELECTIONS (B-F)

Unlike PROC LOGISTIC, PROC HPLOGISTIC does not offer the option SELECTION = SCORE (as of SAS/STAT 14.1). But PROC HPLOGISTIC might provide a substitute to "Best Subsets" by a process I'll call "**B-F**" for Backward and Forward Selections. B-F is a proposal that needs simulation testing before it should be used for practical applications.

In B-F the first step is running PROC HPLOGISTIC with SELECTION METHOD = BACKWARD with the selection of predictors-to-remove being determined by the predictor which gives best (lowest) SBC after removal. Then the second step is running PROC HPLOGISTIC with SELECTION METHOD = FORWARD with the selection of predictors-to-enter being determined by the predictor which gives best SBC when entered. All models and their SBC's from "removal" and "entry" are captured. Also all candidate predictors for removal and entry lead to models, and all these additional models and their SBC's are captured. CLASS statements may be used, and SBC is correctly computed when class variables appear.

Estimated SBC: The SBC from removal and entry of predictors by HPLOGISTIC is an estimated SBC.[12] The estimated SBC and true SBC can be modestly different. Also models in common to the Backward and Forward steps often have different estimated SBC's. The best combination of the two estimated SBC's seems to be the "average". The minimum estimated SBC is also reported. The average estimated SBC is used to rank the models in the report at the end of the B-F process.

Estimated SBC and Transformed Predictors:

If no WOE-coded predictors or transformed continuous predictors are used in modeling, then the degrees of freedom for estimated SBC of the models are correctly calculated by HPLOGISTIC. In this case the B-F process is *fully successful* at ranking the models by average estimated SBC.

If WOE-coded predictors or transformed continuous predictors appear in a model, there are issues:

[12] SAS/STAT 14.1 User's Guide, High-Performance Procedures p. 434: … if you specify SELECT=AIC, AICC, or BIC, the selection criteria are estimated … and hence do not match the values that are computed when that model is fit outside of the selection routine.

1. A WOE-coded predictor can be entered as a class variable so that degrees of freedom are better accounted for when calculating estimated SBC of the model. Usage of the CLASS statement decreases -2*LL of the model (vs. entering WOE-coded predictor as a numeric predictor) but the increase in the penalty term (increase = log(n)*(L-1) – log(n)*1) has a compensating effect.

   When re-fitting the selected M candidate models from B-F on the validation data set, the usage of numeric WOE predictors (replacing their CLASS designation) will modestly change the models. This is satisfactory for modelers who want to use WOE predictors in their models.[13]

2. The degrees of freedom adjustment for transformed continuous predictors [e.g. Log(X)] cannot be completed before BACKWARD and FORWARD predictor selections are made by PROC HPLOGISTIC. This is a deficiency in B-F with no work-around. As a result, the selection of these predictors will be favored. Corrected estimated SBC's will be computed in a later DATA step for those models with a transformed continuous predictor.

Number of Models Produced by B-F

If it is assumed that the removal of predictors by BACKWARD is the reverse of entry of predictors by FORWARD, then the combined number of models is K*(K-1) + 1 where K is the number of predictors.[14] For each k between 1 and K-1 there are K models giving a total of (K-1)*K. Then the full model adds one more. This is appealing in that B-F finds K models for each k where 1 ≤ k < K, among which is the model on the BACKWARD (and FORWARD) path.[15]

EXAMPLE 2: BACKWARD AND FORWARD SELECTIONS (B-F)

In EXAMPLE 2 the "example1" data set with predictors X1, X2, X8, C_woe and target Y are re-used to illustrate B-F. We pick C_woe in order to illustrate B-F for a WOE-coded predictor. To account for degrees of freedom, C_woe will be entered as a class variable in the PROC HPLOGISTIC's.

SAS code for this example has been uploaded to the **SAS Global Forum web-site.**

FORMAT: If transformed continuous predictors appear in models found by B-F processing, then the estimated SBC for these models can be corrected with help from FORMAT $df. In EXAMPLE 2 there are no degrees of freedom corrections. In this case FORMAT $df can be omitted with no downstream impact.

```
%MACRO UPCASE_FORMAT;
PROC FORMAT LIBRARY = Work;
VALUE $df
/* enter VARIABLES and DF ... converts to UPCASE */
%UPCASE
(

)
Other = "1";
run;
%MEND;
%UPCASE_FORMAT;
```

BACKWARD: PROC HPLOGISTIC is run with METHOD = BACKWARD. The option "SELECT = SBC" specifies that the predictor to be removed by BACKWARD is the one that gives the lowest estimated SBC. The "CHOOSE = SBC" is not relevant to this discussion. The "STOP=NONE" directs BACKWARD to continue to the end. "DETAILS = ALL" is required to generate the ODS data sets, discussed next.

```
ods output SelectionDetails = seldtl_b;
ods output CandidateDetails = candtl_b;
```

[13] The SBC's arising from fitting the M candidate models on the validation data set should be recomputed with a degree of freedom adjustment for WOE predictors or any transformed continuous predictors.

[14] This formula ignores the "Intercept Only" Model.

[15] Contact the author for a proof of these assertions.

```
PROC HPLOGISTIC DATA= example1 NAMELEN = 32;
   CLASS C_woe;
   MODEL Y (descending) = X1 X2 X8 C_woe;
   SELECTION METHOD = BACKWARD
      (SELECT = SBC CHOOSE = SBC STOP = NONE) DETAILS=ALL;
```

The ODS statements collect information about the predictors that are removed by the BACKWARD option (see SelectionDetails) as well as the predictors that were candidates for removal at each step (see CandidateDetails). In addition to the step number and the predictor name, the estimated SBC value that is obtained is the estimated SBC for the model after removal of a predictor or candidate predictor.

```
DATA canseldtl_b; MERGE seldtl_b candtl_b; BY step;
PROC PRINT DATA=canseldtl_b;
```

TABLE 5 shows the results of printing data set “canseldtl_b”. The Criterion is “estimated SBC”. The predictors selected “for removal” are given in the “Effect Removed” column. Predictors considered as “candidates for removal” are listed in the “Effect” column. For example, Obs #3 shows the estimated SBC of **141.73** that would result if X1 (instead of C_woe) were removed in Step 1. Obs #2 shows the lower estimated SBC of **129.31** that results from removing C_woe in Step 1. C_woe was entered as a class variable and the correct degrees of freedom (d.f. = 6) were counted in the computation of estimated SBC.

The best model is {X2, X8} with an estimated SBC of **128.19**. The model {X2, X8} is the result of removing C_woe and X1 in step 1 and step 2.

TABLE 5 – MODELS FROM METHOD=BACKWARD (BEFORE CORRECTION FOR DF)

Obs	Step	Number In Model	Effect Removed	Effect	Criterion (estimated SBC)
1	0	5			.
2	1	4	C_woe	C_woe	**129.31**
3	1	4	C_woe	X1	**141.73**
4	1	4	C_woe	X2	146.51
5	1	4	C_woe	X8	146.83
6	2	3	X1	X1	**128.19**
7	2	3	X1	X8	131.61
8	2	3	X1	X2	131.74
9	3	2	X2	X2	128.38
10	3	2	X2	X8	128.70
11	4	1	X8	X8	128.22

BACKWARD modeling (from predictors selected for removal or considered as a candidate for removal) will create K*(K+1) / 2 models where K is the number of predictors in the MODEL statement. But for even modest size K the model count of K*(K+1) / 2 is far less than the 2^K - 1 of all possible models.

FORWARD: To supplement the K*(K+1)/2 models from BACKWARD, PROC HPLOGISTIC can be run again with FORWARD and SELECT = SBC. The ODS statements collect information about predictors that are entered by the FORWARD option (see SelectionDetails) and predictors that were candidates for entry at each step (see CandidateDetails).

```
ods output SelectionDetails = seldtl_f;
ods output CandidateDetails = candtl_f;
PROC HPLOGISTIC DATA = example1 NAMELEN = 32;
   CLASS C_woe;
   MODEL Y (descending) = X1 X2 X8 C_woe;
   SELECTION METHOD = FORWARD
      (SELECT = SBC CHOOSE = SBC STOP = NONE) DETAILS=ALL;
DATA canseldtl_f; MERGE seldtl_f candtl_f;
BY step;
run;
```

Processing “canseldtl_b” and “canseldtl_f”: The information in data sets “canseldtl_b” and “canseldtl_f” is processed to produce a variable called VariablesInModel whose values are the list of predictors in the model. Each observation corresponds to a model and has an associated SBC. The same model (same VariablesInModel) can appear in both “canseldtl_b” and “canseldtl_f”.

Degrees of Freedom and Correction to Estimated SBC: In a DATA step corrections to estimated SBC (for example, for transformed continuous predictors) can be made by applying the FORMAT $df to the uncorrected SBC.

Deduped Models and Average Estimated SBC: The models from BACKWARD and FORWARD are deduped (with respect to having the same variables-in-model). For duplicate models the average estimated SBC is computed as well as minimum estimated SBC. Models are sorted by average estimated SBC. With a small change to the B-F program another sort could be made by minimum estimated SBC.

In TABLE 6 the best model according to average estimated SBC is {X2, X8}. The true SBC appears in the “memo” column. The estimated and true SBC’s give the same ranking except for Obs 7 and 8.

TABLE 6 – MODELS FROM B-F

Obs	VariablesInModel from consolidated BACKWARD and FORWARD	Average Estimated SBC (ranking)	Minimum Estimated SBC	# of Models in Average	MEMO: True SBC
1	X2 X8	128.237	128.185	2	128.212
2	Intercept Only	128.321	128.218	2	128.425
3	X8	128.554	128.377	2	128.672
4	X2	128.869	128.695	2	129.014
5	X1 X2 X8	130.239	129.307	2	131.167
6	X1 X2	131.608	131.608	1	131.932
7	X1	131.822	131.822	1	132.096
8	X1 X8	131.922	131.744	2	131.819
9	X2 X8 C_WOE	141.778	141.734	2	141.740
10	X8 C_WOE	142.610	142.610	1	142.961
11	C_WOE	143.541	143.541	1	143.767
12	X1 X2 X8 C_WOE	145.570	145.558	2	145.582
13	X1 X8 C_WOE	146.515	146.515	1	147.016
14	X1 X2 C_WOE	146.834	146.834	1	147.454

Comments: Candidate Models from Backward and Forward (B-F)

- If BACKWARD and FORWARD follow the same sequence of variable removals and, in reverse order, entries, then the total number of candidate models (ignoring the “Intercept Only” Model) is:

  $$K*(K-1) + 1$$

  For large K this is roughly double the number from BACKWARD alone. For each k between 1 and K-1 there are K models giving a total of (K-1)*K. Then the full model adds one more.

  Without the assumption that the BACKWARD and FORWARD selections are equal, a non-tight upper-bound for the number of models is simply the sum of BACKWARD and FORWARD models:

  $$K*(K+1)/2 + K*(K+1)/2 = K*(K+1)$$

- Using average estimated SBC or minimum estimated SBC to rank the models the modeler can decide on a cut-off of “**M**” models for a final evaluation on a validation data set.
- Because of the high-performance feature of HPLOGISTIC the number of predictors to be offered to the fitting of the model is less of an issue.
- Does B-F find the best model with respect to (actual) SBC? It is very doubtful there is an analytic answer to this question and the recourse is to run extensive simulations to test this conjecture.
- A process that is similar to B-F could be centered around SELECTION METHOD = STEPWISE. A speculation is that STEPWISE could go “off-track” as more and more predictors are added. The

candidate models with many predictors found by STEPWISE might have higher SBC than the models with many predictors found by BACKWARD. This speculation has not been tested.

Comments: Fitting the M candidate models on the validation data set

- The usage of WOE coded variables as class variables is a convenience in that it places the WOE predictor variable name within the value of "VariablesInModel". See, for example, {X1 X2 C_WOE} in the last row of TABLE 6. Later, using a DATA step the values of "VariableInModel" can be put into macro variables by way of SYMPUT to automate the re-fitting of the M candidate models on the validation data set. This is done by macro %CANDIDATE_STATS to be discussed in the final section.

EVALUATION OF THE CANDIDATE MODELS AND FINAL MODEL SELECTION

Now that the predictors for M candidate models have been specified by Best Subsets or B-F, these M models will be fitted on the validation data set. There are at least four aspects of evaluation and ranking of the models after fitting them to the validation data set:

- Business Requirements: For example, it may be of interest to have a model which finds the best 10% of a population for direct marketing. Performance of the model outside the top decile might be of less interest. Additionally, a model might be required to have predictors that are understood by the business and which have the expected relationships to the target.
- Specification: Goodness-of-fit (GOF) statistics measure if the model was correctly specified. These statistics might reveal the existence of outliers or the need to add interactions or higher order terms. But GOF statistics are of less interest to the modeler than the measures of predictive accuracy and SBC ranking.
- Predictive Accuracy: Measures of predictive accuracy include the c-statistic, several R-squares, and coefficient of discrimination.
- Parsimonious Fit: The models with the best parsimonious fit are those with lowest SBC.

A macro called %CANDIDATE_STATS provides measures of Specification, Accuracy, and Parsimonious Fit for the collection of candidate models that are found by Best Subsets or B-F or by some other method. This macro will be applied to the data sets of EXAMPLE 3.

In fact, if the modeler has, perhaps, 7 predictors (i.e. 127 models), then an automated process of finding candidate models might be skipped and, instead, the modeler would run all models, of interest, through %CANDIDATE_STATS. For example, the modeler might look at only models with at least 3 predictors.

EXAMPLE 3: %CANDIDATE_STATS

A training and validation data set are created by the SAS program below:

```
DATA example3T example3V;
DO ID = 1 to 2000;
  cumLogit = ranuni(3);
  e = 1*log(cumLogit/(1-cumLogit));
  X1 = rannor(3);
  X2 = (ranuni(3) < .2);
  H = X1*X2; /* "hidden" */
  W  = floor(20*ranuni(3) + 1);
  X3 = log(W); /* Count as 2 d.f. */
  Y_star = 0.2*X1 + 0.5*X2 + 2.0*H + W**(0.10) + 0.5*e;
  Y = (Y_star > 1);
  IF ID <= 1000 THEN OUTPUT example3T; /* Training */
   ELSE OUTPUT example3V; /* Validation */
  END;
```

It is assumed: Variables X1 X2 X3 are the predictors selected by the modeler and X3 was the result of selecting the transformation of "Log" so that 2 d.f. are associated with X3.

In the true model for Y an interaction H = X1*X2 is included. Also, instead of X3 = log(W) in the true model, the fractional polynomial W**(0.10) is used in the true model for Y.

The “Model_Dataset” (collection of models with lists of variables in the models) that is required by %CANDIDATE_STATS can be obtained from Best Subsets or B-F.[16] But instead, the seven possible models are simply read into the data set “Models” as shown:

```
DATA Models;
Length VIM $50;
INFILE datalines delimiter=',';
INPUT VIM $;
DATALINES;
X1,
X2,
X3,
X1 X2,
X1 X3,
X2 X3,
X1 X2 X3,
;
```

A format may be required to associate corrected degrees of freedom to predictors in these models. It is assumed in this example that X3 requires 2 degrees of freedom.

```
%MACRO UPCASE_FORMAT;
PROC FORMAT LIBRARY = Work;
VALUE $df
/* enter VARIABLES and DF ... MACRO converts to UPCASE */
%UPCASE
(
"X3" = "2"
)
Other = "1"
;
run;
%MEND;
%UPCASE_FORMAT;
```

The macro call for %CANDIDATE_STATS is shown next:

```
%CANDIDATE_STATS(7, Models, VIM, Y, , example3T, example3V);
```

Parameters for %CANDIDATE_STATS:

Parameter	**From EXAMPLE 3**	**Description**
1. M_ModelsX	7	The first **M_ModelsX** observations from **Model_Dataset** will be processed. If **M_ModelsX** exceeds the count of observations in **Model_Dataset**, it is reset to equal this observation count.
2. Model_Dataset	Models	A data set where each observation describes a candidate model by specifying its predictors. These predictors are listed in the variable in **Model_Dataset** that is identified by parameter **VariablesInModel**.
3. VariablesInModel	VIM	Variable name in **Model_Dataset** that contains the list of predictors in the candidate model. (Predictors are separated by spaces).
4. Target	Y	Variable in training data set **FitData** that gives the target variable for the models specified by **VariablesInModel. Target** has values 0 and 1.
5. Class	“space”	“Space” or Predictors that are separated by spaces. Predictors are treated as class variables for every occurrence within the **VariablesInModel** column
6. FitData	example3T	The training data set on which the models from **Model_Dataset** are fit.
7. ScoreData	example3V	The validation data set on which the fitted models are evaluated.

[16] From Example 1 the data set “Scoreout2” with variable “VariablesInModel” could be used in %CANDIDATE_STATS. In this example there is only the “training” data set “example1”. This data set would have to be entered for both the FITDATA and SCOREDATA parameters (described in the Parameter definitions).

Report from %CANDIDATE_STATS

The report produced by %CANDIDATE_STATS is shown below in TABLE 7.

The sources of the evaluation statistics in the %CANDIDATE_STATS report are described below:

- SBC is obtained from PROC LOGISTIC option FITSTAT. SBC, after d.f. correction, is called SBC_add_DF. (A correction for degrees of freedom can be applied to WOE-coded predictors, transformed continuous predictors, or to any other predictor as determined by the modeler. The correction is implemented in the macro by using the FORMAT $df.)
- AUC (c-statistic) is obtained from PROC LOGISTIC option FITSTAT.
- R-Square and Max Rescaled R-Square are obtained from PROC LOGISTIC option FITSTAT.
- Coefficient of Discrimination is found by a calculation in the macro on data set **SCOREDATA**. For discussion of the coefficient of discrimination see Allison (2014).
- The two-tail probability for the standardized Pearson statistic is found by a calculation on **SCOREDATA**. A two-tail probability which is less than a pre-set alpha value is reason to question whether the model is correctly specified. But GOF statistics seem of limited value for evaluating predictive models.

Insights into measures of goodness-of-fit and predictive accuracy for logistic regression models are provided by Allison (2014).

TABLE 7 - EVALUATION OF CANDIDATE MODELS FROM EXAMPLE 6 (some columns omitted)

Model #	Variables in Model	DF Added	SBC	SBC_ add_DF	AUC	R-Square	Max-Rescaled R-Square	Coeff of Discrim	z-value of Std Pearson	2-Tail Prob of z-value
1	**X1**	**0**	**1263.66**	**1263.66**	0.6751	0.083	0.113	0.084	0.5699	0.5687
2	X2	0	1350.50	1350.50	0.5095	0.000	0.000	0.000	-0.7074	0.4793
3	X3	1	1343.82	1350.72	0.5513	0.007	0.009	0.005	-0.9619	0.3361
4	X1 X2	0	1270.52	1270.52	0.6750	0.083	0.113	0.084	0.5081	0.6114
5	**X1 X3**	**1**	**1263.45**	**1270.35**	0.6827	0.090	0.122	0.090	0.4575	0.6473
6	X2 X3	1	1350.15	1357.06	0.5501	0.007	0.010	0.006	-1.2729	0.2031
7	X1 X2 X3	1	1270.21	1277.12	0.6831	0.090	0.122	0.090	0.3899	0.6966

Discussion of Results in Table 7:

After adding one d.f. to X3, the model {X1 X3} loses its position as best SBC model to model {X1}.

The Coefficient of Discrimination is closely aligned with R-Square. The best R-Square models are {X1 X3} and {X1 X2 X3}

The AUC (c-statistic) falls below 0.70 for each model, showing that no model has strong predictive accuracy (Hosmer, et al. 2013, p. 177).

The standardized Pearson statistic is not significant for any model despite the omission in these models of the X1 and X2 interaction which is present in the true model.[17] A “two-tail probability” which is less than a pre-set alpha value is reason to question whether the model is correctly specified.

[17] Formulas for standardized Pearson statistic appear in Hosmer et al. (2013 p 203). See examples on pp 206-207. %CANDIDATE_STATS reproduces the examples on pp 206-207 [once a few covariate patterns are slightly perturbed to make each covariate pattern have just one observation]. With some additional effort the standardized sum-of-squares statistic could be added to the GOF statistics. Formulas are given in Hosmer, et al. (2013, p. 203).

CONCLUSIONS

WOE and transformations of continuous predictors come at the “price” of degrees of freedom that must be accounted for when ranking models by penalized measures of fit such as SBC.

Best Subsets and B-F can automate the production of candidate models on the training data set where the ranking of these candidate models relies on ScoreP (Best Subsets) or estimated SBC (B-F). It is very possible the best SBC model will be found by B-F even though not all models are produced by this method.

When entering the model evaluation phase, the subjective but expert judgment of the modeler, guided by the summary report of TABLE 7, can lay the foundation for final model selection.

SAS Global Forum 2016, Las Vegas, NV

SAS CODE AND MACROS DISCUSSED IN THIS PAPER

SAS code for EXAMPLE 2 has been uploaded to the **SAS Global Forum web-site.** SAS code for macro %CANDIDATE_STATS is available from the author.

REFERENCES

Allison, P. (2014). Measures of Fit for Logistic Regression, *Proceedings of the SAS Global Forum 2014 Conference*, Paper 1485-2014.

Babyak, M., (2004). What You See May Not Be What You Get: A Brief, Nontechnical Introduction to Overfitting in Regression-Type Models, *Psychosomatic Medicine*, 66: 411-421.

Dziak, J, Coffman, D., Lanza, S., Li, R. (2012). Sensitivity and Specificity of Information Criteria, The Methodology Center, Pennsylvania State University, Technical Report Series #12-119. https://methodology.psu.edu/media/techreports/12-119.pdf

Finlay, S. (2010). *Credit Scoring, Response Modelling and Insurance Rating*, New York, Palgrave MacMillan.

Hosmer D., Lemeshow S., and Sturdivant R. (2013). *Applied Logistic Regression, 3rd Ed.,* John Wiley & Sons, New York.

Lund B. and Brotherton D. (2013). Information Value Statistic, *MWSUG 2013, Proceedings*, Midwest SAS Users Group, Inc., paper AA-14.

SAS Institute (2012). *Predictive Modeling Using Logistic Regression: Course Notes*, Cary, NC, SAS Institute Inc.

Siddiqi, N. (2006). *Credit Risk Scorecards*, Hoboken, NJ, John Wiley & Sons, Inc.

Thomas, L. (2009). *Consumer Credit Models*, Oxford, UK, Oxford University Press.

CONTACT INFORMATION

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

Bruce Lund
Magnify Analytic Solutions, a Division of Marketing Associates, LLC
777 Woodward Ave, Suite 500
Detroit, MI, 48226
blund.data@gmail.com or blund@magnifyas.com

All code in this paper is provided by Marketing Associates, LLC. "as is" without warranty of any kind, either express or implied, including but not limited to the implied warranties of merchantability and fitness for a particular purpose. Recipients acknowledge and agree that Marketing Associates shall not be liable for any damages whatsoever arising out of their use of this material. In addition, Marketing Associates will provide no support for the materials contained herein.

APPENDIX A: CLASS VARIABLES PRODUCE BETTER FIT THAN WOE VARIABLES

Here are examples that show that CLASS coding produces better fit than WOE coding in models which have other predictors.

```
data test;
do i = 1 to 500;
   z = rannor(1);
   C1 = (-2 <= z <= 2) * z;
   C1 = floor(C1);
   C2 = floor(5*ranuni(1));
   X1 = ranpoi(1, 2); /* random Poisson with seed 1 and mean 2 */
   X2 = ranuni(1);
   Y_star = C1 + 0.2*C2 + X1 + 0.1*X2 + 0.3*rannor(1);
   Y = (Y_star > 1);
   output;
   end;
run;

data test2; set test;
if C1 in ( 0 ) then C1_woe = 0.7537248551 ;
if C1 in ( 1 ) then C1_woe = 3.2496813411 ;
if C1 in ( -1 ) then C1_woe = -0.24808805 ;
if C1 in ( -2 ) then C1_woe = -1.740455431 ;

if C2 in ( 0 ) then C2_woe = -0.568030985 ;
if C2 in ( 1 ) then C2_woe = 0.0135637098 ;
if C2 in ( 2 ) then C2_woe = 0.2221084617 ;
if C2 in ( 3 ) then C2_woe = 0.2792668755 ;
if C2 in ( 4 ) then C2_woe = 0.1431347012 ;
run;

proc logistic data = test2 desc; model y = c1_woe x1 x2;                /* #1 WOE */
proc logistic data = test2 desc; class c1; model y = c1 x1 x2;          /* #1 CLASS */
proc logistic data = test2 desc; model y = c2_woe x1 x2;                /* #2 WOE */
proc logistic data = test2 desc; class c2; model y = c2 x1 x2;          /* #2 CLASS */
proc logistic data = test2 desc; model y = c1_woe c2_woe x1 x2;         /* #3 WOE */
proc logistic data = test2 desc; class c1 c2; model y = c1 c2 x1 x2;    /* #3 CLASS */
run;
```

TABLE 8 - BETTER FIT (smaller -2*log-likelihood) FOR CLASS VARIABLES

MODEL Number	WOE: -2*Log(L)	CLASS: -2*Log(L)
1	198.663	192.509
2	372.849	372.020
3	178.865	147.931

Comment: Based on several examples, if C_woe and X are uncorrelated, then MODEL Y = C_woe X has the same log-likelihood as CLASS C; MODEL Y = C X.

APPENDIX B: DATA SET FOR EXAMPLE 1 AND EXAMPLE 2

```
data example1;
  input C$ Y X1 X2 X8 @@;
  datalines;
 D  0  10.2  6  0.8     A  1  12.2  6  0.6     D  1   7.7  1  0.6     G  1  10.9  7  0.2
 E  0  17.3  6  0.4     A  0  18.7  4    1     B  0   7.2  1  0.8     D  0   0.1  3  0.8
 B  1   2.4  4    0     G  0  15.6  7    1     G  0  11.1  3  0.6     A  0     4  6  0.8
 A  0   6.2  2  0.2     B  0   3.7  3  0.4     A  1   9.2  3    0     F  0    14  3    0
 E  1  19.5  6  0.6     C  0    11  3  0.8     B  0  15.3  7  0.4     B  1   7.4  4  0.2
 A  0  11.4  2  0.4     C  1  19.4  1    0     F  0   5.9  4  0.4     F  1  15.8  6  0.6
 B  0    10  3  0.4     E  0  15.7  1  0.2     F  0    11  5    1     G  1  16.8  0  0.8
 D  1    11  4  0.6     G  1   4.8  7    1     G  1  10.4  5    0     F  0  12.7  7    1
 F  0   6.8  1  0.8     E  0   8.8  0  0.4     B  1   0.2  0  0.2     G  1   4.6  7  0.4
 G  1   2.3  2  0.6     B  0  10.8  3  0.8     B  0   9.3  2  0.8     A  0   9.2  6  0.6
 D  0   7.4  0  0.8     F  0  18.3  3  0.2     A  0   5.3  4  0.2     C  0   2.6  5    0
 A  0  13.8  4  0.4     B  1  12.4  6  0.2     B  0   1.3  1  0.6     A  0  18.2  7    1
 G  0   5.2  2  0.8     F  1   9.4  2  0.4     G  1  10.4  2  0.2     G  0    13  1  0.6
 A  0  17.9  4  0.6     D  1  19.4  6  0.2     B  0  19.6  3  0.6     B  1     6  2  0.4
 F  0  13.8  1  0.8     B  0  14.3  4  0.6     E  0  15.6  0  0.4     D  0    14  2  0.6
```

```
  C  1    9.4  5    0     B  0  13.2  1  0.2     A  0  13.5  5  0.4     E  0   2.6  4  0.4
  E  0   12.4  3  0.8     D  0   7.6  2  0.8     B  0  12.7  1  0.4     C  1  10.7  4  0.8
  B  0   10.1  2  0.4     C  1  16.6  1  0.6     B  1   0.2  3  0.4     C  0  10.8  4  0.4
  A  0    7.1  4    0     D  0  16.5  1    0     B  0  17.1  7  0.6     D  0   4.3  1    0
  B  0     15  2    0     F  0  19.7  3  0.6     B  1   2.8  6    1     F  0  16.6  3  0.6
  E  0   11.7  5  0.8     A  0  15.6  3  0.8     C  1   5.3  6  0.4     B  1   8.1  7  0.6
  B  0   14.8  2    0     D  0   7.4  4  0.6     D  0   4.8  3  0.2     A  0   4.5  0  0.8
  D  0    6.9  6  0.4     B  0   4.7  4    1     B  1   7.5  4  0.4     C  0   6.1  0  0.8
  C  0   18.3  1    1     A  0  16.4  7  0.4     B  0   9.4  2  0.4     A  1  17.9  4  0.2
  B  0   14.9  3  0.2     C  0  12.7  0  0.8     A  0   5.4  4  0.4     G  1  12.1  4  0.4
;
DATA example1; SET example1;
if C in ( "A" ) then C_woe = -0.809318612 ;
if C in ( "B" ) then C_woe = 0.1069721196 ;
if C in ( "C" ) then C_woe = 0.6177977433 ;
if C in ( "D" ) then C_woe = -0.403853504 ;
if C in ( "E" ) then C_woe = -1.145790849 ;
if C in ( "F" ) then C_woe = -0.703958097 ;
if C in ( "G" ) then C_woe = 1.4932664807 ;
run;
```