SESUG 2023 Paper 142

Binning Predictors for Logistic Regression

Bruce Lund, Statistical Trainer, Novi MI

ABSTRACT

Binary logistic models for credit risk and direct marketing campaigns are generally built on large samples and with many classification predictors. It is normal and prudent to bin (combine) the levels of a classification predictor to achieve a smaller number of binned levels. This reduces parameters, eliminates low frequency levels, and may achieve desirable relationships versus the target, such as monotonicity between an ordered binned predictor and the sample odds of the target. Although controversial, continuous numeric predictors are also binned by some modelers. This paper presents two SAS® macros that perform binning. %NOD_BIN applies to nominal predictors. The second, %ORDINAL_BIN, applies to ordered predictors. Both methods, according to the modeler's option, maximize information value or minimize entropy when combining, and they handle zero-bins without recourse to adding 0.5 to the bin. %ORDINAL_BIN provides optimal results as follows: given k denoting a number of bins, the macro is guaranteed to find: (a) the optimal IV or entropy binning and (b), if it exists, the optimal monotonic binning. With limitations, these macros can be applied to continuous numeric predictors as well. This presentation uses Base SAS and SAS/STAT®.

INTRODUCTION

Let a classification variable X have L levels (i.e. distinct values) where L is typically ≤ 20. As a classification variable, X can be character or numeric and X might be nominal or ordered. It is assumed that X is being considered as a predictor for a binary logistic regression model with target Y.

The goal of binning a predictor X with respect to a binary target Y is to simplify X by combining some of the levels while maintaining most of the power of X to predict Y.

For example, suppose X has levels A, B, C, D. After one step the levels of the binned X might become {A,B}, {C}, {D}. The first bin contains the levels A and B. After another step the binned X might become {A,B,D}, {C}. Here, the first bin contains 3 levels A, B, and D.

Let k satisfy 2 ≤ k ≤ L. A **k-bin solution** is an *eligible assignment* (see below) of the L levels into k bins (each bin has at least one level). If the k-binned X has almost as much power in predicting Y (as measured by information value or entropy, see the discussion below) as the unbinned X, then the binning was successful in simplifying X.[1]

An **eligible** assignment of levels to bins is defined in terms of the ordering of the levels of X. There are two alternatives:

(i) Solutions with **ordered bins**: the levels within a bin are adjacent (no gaps) with respect to the ordering of X

(ii) Solutions with **unrestricted bins**: the levels within a bin are unrestricted with respect to any ordering of X

[1] Two approaches of how to use a k-binned X in a model are: (1) as a group variable (via a CLASS variable) or (2) as a weight of evidence coded variable. The pros and cons of (1) and (2) are not discussed in this paper.

For example: If X has 4 ordered levels A, B, C, D, then the 2-bin solutions with ordered bins are {A} {B C D}, {A B} {C, D}, and {A B C} {D}. In contrast, a 2-bin solution which is not ordered is {A C} {B D}.

To preserve the power of binned X to predict Y, the binning is performed using a binning algorithm. The purpose of such an algorithm is to conduct the binning process so as to optimize a measure of the power of the binned X to predict Y.

Two such measures of power are Entropy and Information Value (IV).

TWO MEASURES OF POWER OF BINNED X TO PREDICT Y

Entropy

Assume X has been binned to create variable X_binned. The X-saturated [2] logistic model for X_binned is:

PROC LOGISTIC; CLASS X_binned; MODEL Y = X_binned;

If there are k bins ($k \geq 2$), the formula for Log-Likelihood for X-saturated model with k-levels is given by:

$$\text{Log}(L) = \sum_{j=1}^{k} G_j * \log(G_j / (G_j + B_j)) + B_j * \log(B_j / (G_j + B_i))$$

where G_j = count of Y=1 for cell j and B_j = count of Y=0 for cell j and $1 \leq j \leq k$

Entropy with base e is given by this formula:

Entropy = - (Log-Likelihood) / n

where log-likelihood is from the X-saturated Logistic Model and n is the sample size.

Whenever two levels of X_binned are combined, the entropy is non-decreasing. The goal of an algorithm with the objective function of entropy is to minimize the increase of entropy as the binning proceeds.

Equivalent to entropy is -2*Log(L), with Log(L) as defined above. In all reports, macros %NOD_BIN and %ORDINAL_BIN display **-2*Log(L)** rather than entropy. See the discussion below about these macros.

Information Value

Information Value is most easily explained by giving an example. Simply work through the columns from left to right in Table 1 to obtain the IV of X with target Y. The same calculations apply to any X_binned.

X	Frequencies		Col % Y=0 "b_k"	Col % Y=1 "g_k"	X_woe: Log(g_k/b_k)	g_k - b_k	IV Terms: (g_k - b_k) * Log(g_k/b_k)
	Y = 0	Y = 1					
X1	2	1	25.0%	12.5%	-0.69315	-0.125	0.08664
X2	1	1	12.5%	12.5%	0.00000	0	0.00000
X3	5	6	62.5%	75.0%	0.18232	0.125	0.02279
SUM	8	8	100%	100%		**IV =**	0.10943

Table 1. Example of calculation of IV for predictor X and target Y

[2] Non-standard terminology: I use "X-saturated" to indicate there is a parameter for each level of X_binned.

Information Value is used by modelers who work on credit-risk applications. In a well-known book by Naeem Siddiqi, Table 2 is presented to give guidelines for the usage of IV in evaluating X_binned.

IV Range	Interpretation
IV < 0.02	"Not Predictive"
IV in [0.02 to 0.1)	"Weak"
IV in [0.1 to 0.3)	"Medium"
IV ≥ 0.3	"Strong"

Table 2. Siddiqi (2017). *Intelligent Credit Scoring, 2nd Ed*. p. 179

Whenever two levels of X_binned are combined, then information value is non-increasing.[3]

MONOTONIC ORDERED BIN SOLUTION

An ordered bin solution is **monotonic** if the ordered bins are monotonic versus the odds of the target where odds = (count Y=1 / count Y=0) given X=x, or equivalently, the event-rate of the target.

For example, consider a predictor X with four levels: 1, 2, 3, 4 and target counts as shown:

X	Y=0	Y=1	Odds	Odds after BINNING
1	2	1	.50	.667
2	1	1	1	
3	3	1	.333	.333
4	4	1	.25	.250

An ordered 3-bin solution for X is {1 2}, {3}, {4}. This solution is monotonic because odds are monotonic: 0.667, 0.333, 0.250. If the odds are monotonic, then the event-rates are also monotonic and conversely. Here, the event-rates are: 2/5, 1/4, 1/5.

Given k > 2 there may not exist a monotonic solution. Consider the table below:

X	Y=0	Y=1
1	1	4
2	5	4
3	5	3
4	1	4

The reader can check that no monotonic 3-bin solution exists.

TWO SAS MACROS FOR BINNING ARE GIVEN IN THIS PAPER

Two SAS macros are presented for binning a classification predictor X for a binary target Y.[4]

1. **%NOD_BIN**: Modeler specifies whether solutions are ordered (=J) in the ordering of predictor X or unrestricted (=A). If (A), then any pair of levels of X can be combined. If (J), then only a pair composed of adjacent levels in the ordering of X can be combined. The designation "J" or "A" will be called the "MODE".

[3] For a proof see Lund B. and Brotherton D. (2013). Information Value Statistic, *MWSUG 2013, Proceedings*.

[4] A different method of binning is based on decision trees. A decision tree is used by the Interactive Grouping Node in the Credit Scoring Application in SAS Enterprise Miner. Decision tree binning is not discussed in this paper.

Starting with L levels of X, an eligible pair (using "A" or "J") is selected for combining so as to maximize IV. This is a stepwise process. The steps are repeated until there are 2 bins. For each k in the range $2 \le k \le L$ the algorithm reports a k-bin solution.

Alternatively, -2*Log(L) may be minimized at each step.

The method of stepwise maximization of IV and the method of stepwise minimization of -2*Log L can lead to different solutions. An example is given in the Appendix. Moreover, %NOD_BIN algorithm, using either IV or -2*Log(L) can lead to suboptimal solutions. An example is given in the Appendix.

Although %NOD_BIN can be applied to ordered X using mode "J", it should not be used when $L \le 20$. In this case the macro %ORDINAL_BIN should be used. For each k, %ORDINAL_BIN is guaranteed to find the optimal solution (whether in terms of IV or -2*Log(L)). See the discussion below.

2. **%ORDINAL_BIN**: This macro is applied only to ordered X and finds ALL k-bin solutions with ordered bins where $2 \le k \le L$. The algorithm to produce the binning solutions is simply the complete enumeration of all solutions. Therefore, for each k, %ORDINAL_BIN finds the optimal solution.

   In particular, if a monotonic ordered solution exists for a given k, then %ORDINAL_BIN finds the optimal monotonic solution for that k.

   To illustrate, consider the simple case of a predictor with three levels: 1, 2, 3. There are 2 ordered bin solutions with 2 bins: {1 2} {3} and {1} {2 3} and one solution with 3 bins: {1} {2} {3}. %ORDINAL_BIN finds them all, finds which are monotonic with respect to odds of the target, and computes IV and -2*Log(L) for each bin solution.

   %ORDINAL_BIN is restricted to X where $L \le 20$. This is due to long run-times for greater values of L. The section below discusses the reasons for these long run-times.

NUMBER OF BIN SOLUTIONS FOR ORDERED X

For ordered X with L levels the total number of ordered bin solutions across k for $2 \le k \le L$ is $2^{(L-1)} - 1$.

Here is the explanation of this formula. For k = L there is 1 solution, for k = L-1 there are L-1 solutions, and, in general, for k = K, there are $\binom{L-1}{L-K}$ solutions. By the binomial formula $\sum_{k=2}^{L}\binom{L-1}{L-k} = 2^{(L-1)} - 1$.

For %ORDINAL_BIN, the run-time doubles with each added level since the number of solutions doubles.

%ORDINAL_BIN was run on a data set with 20 levels for X giving $2^{(L-1)} - 1 = 524{,}287$ binning solution. But the run-time was not long. However, for L = 25, there would be 16,777,215 solutions.

If the number of levels L of ordinal X is over 20, then preliminary binning is needed to reduce L to a manageable number. Preliminary binning can be performed by PROC HPBIN or PROC RANK.

HOW TO SELECT A BINNING SOLUTION FOR THE LOGISTIC MODEL

Usually, the modeler will require that no bin appears with a small count in the final binning solution. Once this requirement has been met, then the user is guided by various statistics in selecting a k-bin solution. One approach is to monitor the change in -2*Log(L) or IV

during the binning steps. If IV or -2*Log(L) has a large change in one binning step to the next step, then it is reasonable to stop at the current step.

For ordinal binning by %ORDINAL_BIN, the user may require a monotonic solution. In this case, the first k that includes a k-bin monotonic solution is often selected. But it is possible for a monotonic solution from a smaller k to have higher IV (or lower -2*Log(L)).

Some statistical guidelines for the "stopping k" are provided by %NOD_BIN when ***MODE***=A. These are discussed in the section on %NOD_BIN.

The final stopping point ultimately relies on expert and subjective judgment of the modeler.

SOME OF THE FEATURES OF %ORDINAL_BIN

BINS OF X WITH A ZERO COUNT FOR Y=0 OR Y=1

For a predictor X there may be a bin with a count of zero for either Y=0 or Y=1. Such a bin will be called a zero-bin. IV cannot be computed due to the logarithm in the formula. But -2*Log(L) can be computed since terms involving 0*Log(0) in the formula can be set to zero.

%ORDINAL_BIN does not report binning solutions with a zero-bin. A "solution" with a zero-bin, most likely, could not be implemented in a logistic model because maximum likelihood estimation would not converge due to quasi-separation. Furthermore, IV would not be computed. A zero-bin with a large number of events or non-events might be unusual and should be analyzed separately.

Below there is an example where X has 4 levels and two zero-bins. The complete enumeration of binning solutions k = 4, 3, 2 is shown below. Only the 2-bin solution {1,2}, {3,4} is reported by %ORDINAL_BIN.

k=4

X	Y=0	Y=1
1	1	0
2	2	1
3	3	1
4	4	0

k=3

X_bin	Y=0	Y=1
1,2	3	1
3	3	1
4	4	0

X_bin	Y=0	Y=1
1	1	0
2,3	5	2
4	4	0

X_bin	Y=0	Y=1
1	1	0
2	2	1
3,4	7	1

k=2

X_bin	Y=0	Y=1
1,2,3	6	2
4	4	0

X_bin	Y=0	Y=1
1,2	3	1
3,4	7	1

X_bin	Y=0	Y=1
1	1	0
2,3,4	9	2

Table 3. Only the X_bin with bins {1 2}, {3 4} will be reported by %ORDINAL_BIN

SOLUTIONS WITH SMALL BIN COUNTS MAY BE OMITTED FROM REPORTS

The user may specify the value of a parameter called ***MIN_PCT*** which gives the minimum percent (0 to 25) of the total sample that the bin must contain in order for the solution to be reported. Alternatively, parameter ***MIN_NUM*** may be specified which gives minimum count requirement for a bin.

A FREQUENCY VARIABLE MAY BE SPECIFIED

If the input dataset has been summarized, there is a parameter ***W*** (Freq) that designates the frequency variable. The entry for ***W*** is a SAS variable with positive integer values.

SOLUTIONS ARE RANKED BY IV (OR -2*LOG(L)) IN THE PRINTED REPORTS

The Tables below are based on dataset "level_4" where X is predictor, Y is target, and W is frequency.

```
DATA level_4;
do X = 1 to 4;
  Y = 1;
  W = floor(200*ranuni(1)) + 1;
  output;
  Y = 0;
  W = floor(250*ranuni(1)) + 1;
  output;
  end;
run;
```

	X				
Y	1	2	3	4	Total
0	243	65	243	133	684
1	37	80	185	109	411
Total	280	145	428	242	1095

Table 4. Counts from dataset Level_4

Here is the macro call for %ORDINAL_BIN that produces most of the Tables below.

```
%ORDINAL_BIN (
  DATASET = level_4,
  X = X,
  TARGET = Y,
  W = W,
  RANKING = IV,
  ORDER = D,
  MISS = ,
  SUMMARYONLY = ,
  N_BEST = 3,
  N_MONO = ,
  MIN_PCT = ,
  MIN_NUM = ,
  MIN_BIN = 3,
  MAX_BIN = 3,
  NOPRINT_WOE = ,
  PRINT1_WOE = 3,
  PRINT2_WOE = 3,
  RUN_TITLE = TEST A,
  DELETE_PRIOR =
);
```

There are 3 solutions with 3 bins since L = 4. The modeler obtains the report of the best 3 IV solutions, where each has 3 bins, by these parameters: ***N_BEST*** = 3***, MAX_BIN*** = 3, ***MIN_BIN*** = 3.

Top 3 solutions are displayed in descending IV order. Solution_num = 3 is the only monotonic solution for k = 3. The column "turns" counts number of times the odds changes "directions" (increase v. decrease).

Obs	BINS	missing	best_ rank	best_ mono	solution_ num	turns	IV	minus2LL	L1	L2	L3
1	3	N	*		1	1	0.479	1336.9	1	2	3+4
2	3	N	*		2	1	0.456	1342.9	1	2+3	4
3	3	N	*	*	3	0	0.120	1418.7	1+2	3	4

Table 5. The "*" in "best_mono" column indicates the best IV (-2*Log(L)) monotonic solution at k = 3

The modeler can print all solutions within ***MIN_BIN*** to ***MAX_BIN*** by assigning ***N_BEST*** a high number.

PROVIDING SAS CODE TO IMPLEMENT A BINNING SOLUTIONS IN A LOGISTIC MODEL

The modeler may obtain SAS code for "Binned weight of evidence" or "Binned classes" for selected "k". For the dataset "level_4" the SAS code is shown below for the best IV 3-bin solution. This is obtained by setting ***N_BEST*** = 1, ***PRINT1_WOE*** = 3, ***PRINT2_WOE*** = 3. The modeler would copy and paste the code below into the user's SAS program.

solution_num	WOE Coding
1	if X in (1) then X_B_woe = -1.372778828 ;
1	if X in (2) then X_B_woe = 0.7170040679 ;
1	if X in (3,4) then X_B_woe = 0.2633553271 ;
solution_num	BIN Coding
1	if X in (1) then X_B = 1 ;
1	if X in (2) then X_B = 2 ;
1	if X in (3,4) then X_B = 3 ;

Table 6. SAS code when *N_BEST* =1 for k = 3

OPTIONALLY, MISSING VALUES OF X ARE INCLUDED AS AN UNORDERED LEVEL

The user may specify that the missing level of X is included in the binning. The usual complete enumeration of all solutions for non-missing levels is carried out and the contribution to these solutions by the missing level is then added to IV or -2*Log(L). Binning reports, which are sorted by IV or - 2*Log(L), include the contribution from missing. If missing is a zero-bin, then the missing level is ignored.

SOME OF THE FEATURES OF %NOD_BIN

First, a brief discussion is given of some of the parameters:

THE REQUIRED PARAMETERS:

X: SAS variable: Numeric (".” or 0 to 999) | Character) … with some restrictions

TARGET: SAS variable with exactly 2 non-missing Levels and no Missing

W: A frequency variable with positive integer levels | 1 (1 if no frequency variable is used)

METHOD: Method of binning: IV | LL (where LL is -2*Log-Likelihood for X-saturated logistic model.

MODE: A | J: A for unrestricted binning or J for restricted to maintain ordering of X within bins

ORDER: If D, then higher level of ***TARGET*** is set to G ("good") and lower level of ***TARGET*** is set to B ("bad"). G appears in the numerator of "odds" and weight-of-evidence expressions. If A, then the lower level of ***TARGET*** is set to G, higher is set to B.

A parameter that is not required is ***VERBOSE***, but it is commonly set to YES, unless the number of levels of X is large, which would cause the summary report to wrap around the page.

VERBOSE: YES | <any other>. YES displays all steps of binning as part of the SUMMARY REPORT

Details regarding macro parameters are given in Documentation available from the author.

Here is an example of %NOD_BIN with the required parameters, in addition to ***VERBOSE***.

```
%NOD_BIN(
  DATASET = level_4,
  X =  X,
  TARGET =  Y,
  ZERO_ONE = YES,
  W = W ,
  METHOD = IV,
  MODE = A,
  ORDER = D,
  MISS = ,
  MIN_PCT = ,
  MIN_NUM = ,
  MIN_BIN = ,
  MAX_BIN = ,
  VERBOSE = YES,
  VERBOSE2 = ,
  LL_STAT = ,
  WOE = ,
  ADD = ,
  RUN_TITLE = Required Parameters and VERBOSE
);
```

The macro call sets ***MODE***=A (unrestricted binning) and ***METHOD***=IV. There is a frequency variable is W. The ***ORDER*** D is descending which specifies that the count of Y=1 forms the numerator of the odds.

The reports are shown below.

Required Parameters and VERBOSE
Dataset= level_4, Predictor= X, Target= Y, Zero_one= YES, Freq= W, Method= IV, Mode= A, Miss= , Order= D Min_Pct= 0, Min_Num= 0, ADD= N/A
Summary Report

k	REASON collapse row k to k-1	ZERO CELL	IV	Like-Ratio Chi_Sq	-2*Log L	X_STAT	L1	L2	L3	L4
4		NO	0.480	112.552	1336.654	0.655	1	2	3	4
3		NO	0.479	112.345	1336.861	0.652	1	2	3+4	
2		NO	0.456	106.244	1342.962	0.633	1	2+3+4		

Table 7.

Log-odds Ratio with 95% CI
Consider stopping at k if +/- 2SD interval after collapse omits zero

k	collapsing_to	LO Ratio after collapse	LO Ratio Std Dev	LOminus2SD	LOplus2SD
4	3	-0.074	0.162	-0.398	0.250
3	2	0.454	0.184	0.085	0.822

Table 8.

EXPLANATION OF THE COLUMNS IN TABLE 7

REASON: This column reports the reason for a combine that relate to zero-bins and small-bin counts. A space indicates that the combine was based on normal processing (maximize IV or minimize -2*Log(L)).

ZERO CELL: If YES, then at this step, one of the bins was a zero-bin

-2*LOG L: Value from Logistic Model: CLASS X_binned; MODEL Y = X_binned;

LIKELIHOOD-RATIO CHI_SQ: -2*(G*Log(G/N) + B*Log*(B/N)) - (-2*Log(L)). Depends on current k. Notation: N = sample size. G = count of goods in the sample. B = count of bads in the sample.

X_STAT: The c-Statistic for Logistic Model: CLASS X_binned; MODEL Y = X_binned;

TABLE 8 GIVES A TEST FOR STOPPING WHEN MODE=A

When ***MODE***=A, a test of whether to stop the binning process is given in Table 8. The notation B_i is used for count of bads in the i^{th} bin and G_i is used for count of goods in the i^{th} bin. If levels i and j are selected to be combined, then their log-odds ratio, LO, is given by $\log((G_i / B_i) / (G_j / B_j))$. The approximate standard deviation of the LO is $LO\_SD = SQRT (1/G_i + 1/B_i + 1/G_j + 1/B_j)$.

Assuming bin counts in rows i and j are large, then LO is normally distributed and an approximate 95% confidence interval is:

LO +/- 2 * LO_SD (approximate 95% confidence interval for true LO).

If LO = 0, then $G_i / Bi = G_j / Bj$ and the combining of i and j is a good decision. The more that LO deviates from 0, the more that IV decreases (which is undesirable) if a combine is performed. A potential guideline for stopping the binning process is when the interval (LO ± 2 * LO_SD) does not include 0.

In Table 8 the confidence interval range for k=3 does not include 0. Therefore, the combine to two levels ({1} and {2 3 4}) is not recommended.

FOR MODE=A THERE IS A SECOND STATISTICAL TEST FOR STOPPING

To perform this test the macro parameter ***LL_STAT*** is set to YES. Now the summary report has additional columns: "Nested ChiSq" and "Pr > ChiSq". Nested ChiSq and the associated right tail probability Pr > ChiSq perform a test of whether the coefficients of dummy variables for the bins about to the combined are statistically equal. The coefficients are those in this logistic model:

PROC LOGISTIC; CLASS X_binned; MODEL Y=X_binned;.

The combine should not be performed if the coefficients of the dummy variables are statistically unequal at the modeler's preferred value of alpha.

According to this test the combine from k=3 to k=2 should not be made since the right tail probability is 0.01 which indicates that the coefficients for {1} and {2 3 4} are unequal.

k	IV	Like-Ratio Chi_Sq	-2*Log L	Nested ChiSq	Pr > ChiSq	L1	L2	L3	L4
4	0.480	112.55	1336.65	N/M	N/M	1	2	3	4
3	0.479	112.35	1336.86	0.21	0.65	1	2	3+4	
2	0.456	106.24	1342.96	6.10	0.01	1	2+3+4		

Table 9. Some columns are omitted to save space on this page.

See the Appendix for SAS code to conduct a formal test of the equality of these coefficients using PROC LOGISTIC. %NOD_BIN performs an equivalent calculation.

LL_STAT is ignored if ***MODE***=J. I felt that the Nested ChiSq was not meaningful since backward selection of dummy variables is compromised by the use of ***MODE***=J. This was also the reason for restricting Table 8 to ***MODE***=J. But in my next version, I will report Table 8 for both modes.

PROBLEM WITH ZERO-BINS AND HOW HANDLED BY %NOD_BIN

Since IV involves a logarithm, IV cannot be computed when there is a zero-bin. A way to avoid this problem is add a small positive number to either G_j or B_j (depending on which is zero). The ***METHOD***=LL also involves logarithms, but a 0 would appear in a term 0*Log(0). This term can simply be set to zero.

PARAMETER ADD= 0.5 | 0.0001 | "space"

When using %NOD_BIN the modeler may specify the parameter ***ADD*** to have value of 0.5 or 0.0001 whether or not there actually are zero-bins in the dataset. If a zero-bin is encountered, then ***ADD*** is applied. Either 0.5 or 0.0001 is added to G_j or B_j, whichever is zero for the bin.

The computational problem with IV is avoided by using ***ADD***. For small samples, adjustment by 0.5 can have an undue influence on the values of IV. In the current version of %NOD_BIN, 0*Log(0) is not set to zero. For ***METHOD***=LL the usage of ***ADD***=0.0001 provides a good approximation.

AFTER ASSIGNING A VALUE FOR ADD

After assigning 0.5 or 0.0001, a bin could be encountered with either $0 < G_j < 1.0$ or $0 < B_j < 1.0$. This bin is immediately forced to combine with another bin according to either

the IV or LL criterion. This prevents a bin which initially was a zero-bin from continuing into the later steps of binning while retaining a low frequency.[5]

ALTERNATIVE TO THE ADD PARAMETER WHEN METHOD IS IV

X_STAT is the model c (or c-statistic) for the one-variable logistic model:

PROC LOGISTIC; CLASS X_binned; MODEL Y = X_binned;

X_STAT can be computed by the formula shown below: [6]

$$X\_STAT = 0.5 * \{ \sum_{i=1}^{L-1} \sum_{j=i+1}^{L} | B_i*G_j - B_j*G_i | / M + 1 \}$$

where $G = \sum_{j=1}^{L} G_j$ and $B = \sum_{j=1}^{L} B_j$ and $M = G * B$

As bins are combined, X_STAT is decreasing (unless the odds for the two bins to be combined are equal in which case X_STAT is unchanged).

When ***ADD***="space", then X_STAT is used to combine zero-bins. If %NOD_BIN detects a zero-bin, the macro forces a combine of that bin with some other bin via X_STAT maximization.[7]

Here is an example: A zero-bin occurs for X=4. The binning of the zero-bin is by X_STAT maximization.

```
DATA test;
input x y w;
datalines;
1 1 2
1 0 5
2 1 1
2 0 6
3 1 5
3 0 4
4 1 4
;

PROC FREQ data = test; WEIGHT w;
TABLES x*y / norow nocol nopercent;
```

x	y		
	0	1	Total
1	5	2	7
2	6	1	7
3	4	5	9
4	0	4	4
Total	15	12	27

Table. 10

[5] The Macro programming does not guarantee that the bin that is combined at a given step gives the overall best IV (or entropy) among all remaining bins with G < 1 or B < 1. It is simply the first encountered zero-bin that is processed.

[6] See Lund (2019), Logistic Regression, Basics and Beyond, MWSUG 2019. https://www.mwsug.org/proceedings/2019/SP/MWSUG-2019-SP-026.pdf

[7] Macro programming does not guarantee that the bin that is combined at a given step would have given the overall best X_STAT among all zero-bins. First encountered zero-bin is processed.

```
%NOD_BIN(
  DATASET = test ,
  X =  x,
  TARGET =  y,
  ZERO_ONE = YES,
  W = w ,
  METHOD = IV ,
  MODE = A ,
  ORDER = D ,
  MISS = ,
  MIN_PCT =  ,
  MIN_NUM =  ,
  MIN_BIN = ,
  MAX_BIN = ,
  VERBOSE = YES ,
  VERBOSE2 =  ,
  LL_STAT =  ,
  WOE =  ,
  ADD =  , /* space implies the use of X_STAT for zero bins */
  RUN_TITLE = X-Stat for Zero Bin);
```

X-Stat for Zero Bin

k	REASON collapse row k to k-1	ZERO CELL	IV	Like-Ratio Chi_Sq	-2*Log L	X_STAT	L1	L2	L3	L4
4		YES	N/M	N/M	N/M	0.8056	1	2	3	4
3	X_STAT	NO	1.112	6.9302	30.1657	0.7611	1	2	3+4	
2		NO	1.020	6.4994	30.5965	0.7417	1+2	3+4		

Table 11. ADD="space"

For comparison, %NOD_BIN is now run twice more using ***ADD***=0.0001 and 0.5.

k	REASON collapse row k to k-1	ZERO CELL	IV	Like-Ratio Chi_Sq	-2*Log L	X_STAT	L1	L2	L3	L4
4		NO	4.286	10.6110	26.4850	0.8056	1	2	3	4
3	MIN NUM	NO	1.112	6.9301	30.1659	0.7611	1	2	3+4	
2		NO	1.020	6.4993	30.5968	0.7417	1+2	3+4		

Table 12A. ADD=0.0001 for Zero Bin

k	REASON collapse row k to k-1	ZERO CELL	IV	Like-Ratio Chi_Sq	-2*Log L	X_STAT	L1	L2	L3	L4
4		NO	1.349	8.0542	29.6222	0.7849	1	2	3	4
3	MIN NUM	NO	1.006	6.3731	31.3033	0.7487	1	2	3+4	
2		NO	0.916	5.9423	31.7341	0.7298	1+2	3+4		

Table 12B. ADD=0.5 for Zero Bin

In Table 11 the binning from k=4 to k=3 was due to X_STAT maximization. In Table 12A the ADD of 0.0001 was added to B_4 and in Table 11B the ADD of 0.5 was added to B_4

In Tables 12A and 12B the REASON = MIN NUM in step 3 refers to the forced binning of level 4 as a consequence of $B_4 < 1$. Based on IV, level 4 was combined with level 3 when going to Step k=3. All statistics in Tables 12A and 12B are incorrect due to use of ADD. Statistics in Table 11 are correct.

In this contrived, very small example the binning solutions for k=3 and k=2 are the same in Table 11 as in Tables 12A and 12B. I can't say with any confidence how often these solutions could differ.

In my judgment the X_STAT method of handling zero-bins is the better approach when using ***METHOD***=IV. However, if ***METHOD***=LL, then ***ADD***=0.0001 is a valid approach.

%NOD_BIN: FORCING COMBINES OF BINS WITH LOW FREQUENCY

The following dataset is used in this section. This dataset has low frequency bins as well as zero-bins.

```
DATA SMALL;
INPUT X $ Y W;
DATALINES;
1 0 3
1 1 2
2 0 6
2 1 1
3 0 4
3 1 8
4 0 10
5 1 6
6 0 10
6 1 10
;
run;
PROC FREQ DATA = SMALL;
TABLES X*Y / norow nocol nopercent;
WEIGHT W;
run;
```

Here is the frequency table of X vs. Y.

	Y			
X	0	1	Total	
1	3	2	5	(NOTE: bin count < 6)
2	6	1	7	
3	4	8	12	
4	10	0	10	(zero-bin)
5	0	6	6	(zero-bin)
6	10	10	20	
Total	33	27	60	

Table 13. Zero-bins and low frequency bins

Perhaps the modeler requires any bin to have at least 6 observations. The macro parameter ***MIN_NUM*** is used to specify this minimum. Any bin with a count of 5 or fewer is forced to combine before further steps.

But this is a situation where there are also zero-bins.

First, X_STAT is used to combine zero-bins. Then ***MIN_NUM*** is applied.

```
%NOD_BIN(
  DATASET = SMALL,
  X =  X,
  TARGET =  Y,
  ZERO_ONE = YES,
  W = W ,
  METHOD = IV,
  MODE = A,
  ORDER = D,
  MISS = ,
  MIN_PCT =  ,
  MIN_NUM = 6,
  MIN_BIN =  ,
  MAX_BIN =  ,
  VERBOSE = YES,
  VERBOSE2 =  ,
  LL_STAT =  ,
  WOE =  ,
  ADD =  ,
  RUN_TITLE = X-Stat for Zero Bin and MIN_NUM < 6);
```

k	REASON collapse row k to k-1	ZERO CELL	IV	Like-Ratio Chi_Sq	-2*Log L	X_STAT	L1	L2	L3	L4	L5	L6
6		YES	N/M	N/M	N/M	0.824	1	2	3	4	5	6
5	X_STAT	YES	N/M	N/M	N/M	0.810	1	2	3+5	4	6	
4	X_STAT	NO	1.746	N/M	61.13	0.805	1	2+4	3+5	6		
3	MIN NUM	NO	1.735	21.28	61.29	0.799	1+6	2+4	3+5			
2		NO	1.432	17.26	65.32	0.724	1+6+3+5	2+4				

Table 14. X-Stat for Zero-Bin and MIN_NUM < 6

Combining X=3 and X=5 at k=5 is due to X_STAT processing of X=5 zero-bin.

Likewise, this is true for combining X=2 and X=4 at k=4.

Then X=1 is combined with X=6 since bin count for X=1 is less than 6.

SHOWING THE USEFULNESS OF BINNING WITH %ORDINAL_BIN

Here is an example to illustrate that the preparation of classification predictor is greatly facilitated by binning. This example uses %ORDINAL_BIN. The data come from a HELOC (home equity) credit-risk example. See https://community.fico.com/s/explainable-machine-learning-challenge?tabset-3158a=2

A predictor is "NumTradesOpeninLast12M". It is ordered with 18 levels and has a missing level (-9). The target is called RiskPerformance with levels “good” and “bad”.

The overarching question:

Is there an ordered bin solution of X which is monotonic versus the event rate of the target, and which has a good IV?

Monotonic binning for this predictor certainly makes sense. Credit risk modelers expect more Trades Open in last 12 Months to be associated with higher Bad rates.

Here the table of NumTradesOpeninLast12M vs. RiskPerformance.

	RiskPerformance		
NumTradesOpeninLast12M	Bad	Good	Total
(no credit bureau) .	291	165	456
0	565	618	1183
1	569	627	1196
2	511	477	988
3	343	274	617
4	210	162	372
5	97	100	197
6	57	44	101
7	40	17	57
8	18	9	27
9	8	5	13
10	8	2	10
11	4	0	4
12	1	0	1
13	2	0	2
14	2	0	2
16	1	0	1
17	1	0	1
19	1	0	1
Total	2729	2500	5229

Table 15. Table of NumTradesOpeninLast12M by RiskPerformance

In a DATA Step, -9 is reset to . (numeric missing). Minimum bin size is set at 20. Parameter ***MISS*** is set to MISS, and parameter ***SUMMARYONLY*** is set to YES (to greatly reduce the printout).

```
DATA TRAINx; SET HELOC_2.TRAIN;
if NumTradesOpeninLast12M = -9 then NumTradesOpeninLast12M = .; run;
%ORDINAL_BIN(
DATASET=TRAINx, X=NumTradesOpeninLast12M, TARGET=RiskPerformance,
W=1, RANKING=IV, ORDER=D, MISS=MISS, SUMMARYONLY=YES, N_BEST=,
N_MONO=, MIN_PCT=, MIN_NUM=20, MIN_BIN=, MAX_BIN=, NOPRINT_WOE=,
PRINT1_WOE=, PRINT2_WOE=, RUN_TITLE=HELOC Train, DELETE_PRIOR=);
```

Obs	bins_in _solution	Solution _num	IV	minus 2LL	turns	missing	best _rank	best _mono
1	10	1	0.0625	7161.3	5	Y	*	
2	9	1	0.0625	7161.3	4	Y	*	
3	8	1	0.0624	7161.4	4	Y	*	
4	7	1	0.0621	7161.7	2	Y	*	
5	6	1	0.0611	7163.1	2	Y	*	
6	6	4	0.0599	7164.6	0	Y		*
7	5	1	0.0598	7164.7	0	Y	*	*
8	4	1	0.0580	7167.1	0	Y	*	*
9	3	1	0.0516	7175.2	0	Y	*	*
10	2	1	0.0393	7188.7	0	Y	*	*

Table 16. Step by step history of binning is excluded in order to fit to the page.

A promising solution occurs at k=6. There is a monotonic binning solution (solution_num=4) with strong IV of 0.0599. This IV is higher than the IV's of the monotonic solutions which follow at k=5 through k=2.

The first binning solution occurs at k = 10 (and not k=18) since there are zero-bins in all solutions with k > 10.

The monotonic solution at k=6, as well as WOE SAS code, is obtained by re-running %ORDINAL_BIN with ***MIN_BIN***=6, ***MAX_BIN***=6 and ***PRINT1_WOE***=6, ***PRINT2_WOE***=6

```
%ORDINAL_BIN(
DATASET=TRAINx, X=NumTradesOpeninLast12M, TARGET=RiskPerformance,
W=1,RANKING=IV, ORDER=D, MISS=MISS, SUMMARYONLY=, N_BEST=,
N_MONO=, MIN_PCT=, MIN_NUM=20, MIN_BIN=6, MAX_BIN=6, NOPRINT_WOE=,
PRINT1_WOE=6, PRINT2_WOE=6, RUN_TITLE=HELOC Train, DELETE_PRIOR=);
```

Bins	Sol _num	IV	-2LL	L1	L2	L3	L4	L5	L6
6	4	0.060	7164.6	0+1	2	3+4+5	6	7+8+9	10+11+12+13+14+16+17+19

Table 17. Best Monotonic Bin Solution

4	if NumTradesOpeninLast12M in (0,1) then NumTradesOpeninLast12M_B_woe = 0.1810288345 ;
4	if NumTradesOpeninLast12M in (2) then NumTradesOpeninLast12M_B_woe = 0.0187914105 ;
4	if NumTradesOpeninLast12M in (3,4,5) then NumTradesOpeninLast12M_B_woe = -0.105193692 ;
4	if NumTradesOpeninLast12M in (6) then NumTradesOpeninLast12M_B_woe = -0.171217124 ;
4	if NumTradesOpeninLast12M in (7,8,9) then NumTradesOpeninLast12M_B_woe = -0.668023028 ;
4	if NumTradesOpeninLast12M in (10,11,12,13,14,16,17,19) then NumTradesOpeninLast12M_B_woe = -2.214940583 ;
4	if NumTradesOpeninLast12M= . then NumTradesOpeninLast12M_B_woe = -0.479733283 ;

Table 18. WOE values are monotonic vs. binned X for the non-missing bins

MONTONIC BINNING FOR A CONTINUOUS NUMERIC X

A continuous numeric predictor X has many levels, although "many" is subjective … perhaps as few as 10 but, also, as many as thousands.

It is sometimes a normal practice of a business to bin any continuous numeric X when preparing to fit a logistic model. This practice is founded on the idea that a binned predictor (now with multiple degrees of freedom) will better track the event-rate.

Also, sometimes, the business may believe that "more of X" should imply "high (lower) event-rate". In this situation a **monotonic binning solution** of X is required. Then the ordered bins of X must be monotonic with respect to the event-rate of the target.

%MONOBIN MACRO FOR MONOTONIC BINNING

WenSui Liu developed a SAS macro named %MONOBIN [8] that performs monotonic binning for numeric predictors with few or many levels.

The core of this macro is a usage of PROC TRANSREG.

The HELOC dataset is monotonically binned by %MONOBIN after the rows with a missing predictor value are removed. A numeric target Y is needed for the macro call.

```
DATA TRAINx; SET HELOC_2.TRAIN;
if NumTradesOpeninLast12M = -9 then delete;
Y = (RiskPerformance = "Bad");
run;
```

%MONOBIN(data = TRAINx, y = Y, x = NumTradesOpeninLast12M);

Lower	Upper	#Bads	#Freq	BadRate	WoE	IV
0	1	1134	2379	0.47667	-0.13655	0.009293
2	2	511	988	0.51721	0.02569	0.000137
3	5	650	1186	0.54806	0.14967	0.005547
6	6	57	101	0.56436	0.21570	0.000978
7	9	66	97	0.68041	0.71250	0.009829
10	19	20	22	0.90909	2.25942	0.016600
		2438	4773			0.042384

Table 19. %MONOBIN Binning for NUMTRADESOPENINLAST12M

This is the same monotonic solution as produced by %ORDINAL_BIN. The missing level can be appended to Table 19 by a separate calculation.

But %ORDINAL_BIN has the advantages of displaying non-monotonic solutions which might be good alternatives and has the option to exclude solutions with small bin counts from the reporting.

EXAMPLE OF CONTINUOUS NUMERIC X WITH MANY LEVELS

The German Credit dataset [9] is often used as an example for fitting a Probability of Default (PD) model. It has 1000 rows. Target is binary with levels 0 and 1 where 1 is a payment default. There are 3 continuous numeric predictors, including AGE of borrower in years, and 17 classification predictors.

Our focus is on AGE. AGE has 53 levels, 5 zero-bins, and no missing values.

The modeler may try transforms of a predictor X (e.g. Log(X), X and X^2) when fitting a model. But neither linear X or a common transform (such as Log(X) or a polynomial in X)

[8] https://statcompute.wordpress.com/2017/09/24/granular-monotonic-binning-in-sas/
[9] See https://archive.ics.uci.edu/dataset/144/statlog+german+credit+data.

can guaranteed a monotonic relationship between the ordered levels of X and the event-rate (or odds) of the target.

In fact, the idea of a monotonic relationship is ill-defined if some of the levels of X have only 1 or very few cases.

Setting aside the 5 zero-bins, AGE is not monotonic vs. the event-rate for the German Credit dataset.

%ORDINAL_BIN is limited by program design to L ≤ 20. Running %ORDINAL_BIN with L = 53 is out of the question.

In contrast %MONOBIN has handle L = 53 without a problem.

Here is the %MONOBIN macro call. There are only 3 parameters.

```
%MONOBIN(data = GERMAN.BANK, y = Y, x = AGE);
```

The result of binning includes a bin consisting of AGE=19 with only 2 cases. This is not satisfactory.

Lower	Upper	#Bads	#Freq	BadRate	WoE	IV
62	75	7	38	0.18421	-0.64078	0.01343
35	61	101	414	0.24396	-0.28378	0.03135
30	34	55	177	0.31073	0.05061	0.00046
26	29	57	181	0.31492	0.07007	0.00090
20	25	79	188	0.42021	0.52540	0.05654
19	19	1	2	0.50000	0.84730	0.00161
		300	1000			0.10429

Table 20. %MONOBIN Binning for AGE

%MONOBIN would need re-programming to enforce a minimum bin count for the final solution. But, here, post-processing to combine of AGE=19 with AGE=20_to_25 easily removes the problem. The IV is slightly lower after the combine.

Lower	Upper	#Bads	#Freq	BadRate	WoE	IV
62	75	7	38	0.18421	-0.64078	0.01343
35	61	101	414	0.24396	-0.28378	0.03135
30	34	55	177	0.31073	0.05061	0.00046
26	29	57	181	0.31492	0.07007	0.00090
19	25	80	190	0.42105	0.52884	0.05792
		300	1000			0.10406

Table 21. %MONOBIN Binning for AGE after post-processing

An alternative is pre-processing of AGE to recode 19 as 20 before running %MONOBIN. In general such pre-processing requires ad hoc judgments by the modeler.

After such pre-processing the same binning result is obtained.

%NOD_BIN FOLLOWED BY %ORDINAL_BIN FOR MONOTONIC BINNING

%NOD_BIN with specification ***MODE***=J finds an ordered solution for each k. When ***MODE***=J, an X with a large numbers of levels can be run through %NOD_BIN within a short time period. %NOD_BIN has been run successfully on a predictor X with 100 levels.

However, %NOD_BIN does not guarantee that the optimal solution is found for any of the k's except k = L-1.

%NOD_BIN combined with %ORDINAL_BIN provides an approach to binning ordinal X when L ≤ 100. Preliminary binning by %NOD_BIN provides an ordered bin solution with L = 20. This 20-bin solution can then be processed by %ORDINAL_BIN.

This two-step approach was applied to AGE from German Credit. A 20-bin solution from %NOD_BIN was found where, additionally, ***MIN_NUM*** was set at 10 to combine low frequency bins.

Zero-bins were handled by X_STAT maximization.

Then the 20-bin solution was processed by %ORDINAL_BIN to find good monotonic binning solutions. Here is the code for the two-step process.

```
%NOD_BIN(DATASET=GERMAN.BANK, X=AGE, TARGET=Y, ZERO_ONE=YES,
 W=1, METHOD=IV, MODE=J, ORDER=D, MISS= , MIN_PCT= , MIN_NUM=10,
 MIN_BIN=20, MAX_BIN=20, VERBOSE= , VERBOSE2= , LL_STAT= ,
 WOE= WOE, ADD= , RUN_TITLE= 20 bin solution for ORDINAL_BIN);
```

%NOD_BIN provides the 20-bin solution as SAS code:

```
DATA TEMP; SET GERMAN.BANK;
 if AGE in ( 19,20,21 ) then AGE_B = 001 ;
 if AGE in ( 22,23,24,25 ) then AGE_B = 002 ;
 if AGE in ( 26,27 ) then AGE_B = 003 ;
 if AGE in ( 28,29 ) then AGE_B = 004 ;
 if AGE in ( 30,31,32 ) then AGE_B = 005 ;
 if AGE in ( 33,34 ) then AGE_B = 006 ;
 if AGE in ( 35,36 ) then AGE_B = 007 ;
 if AGE in ( 37 ) then AGE_B = 008 ;
 if AGE in ( 38 ) then AGE_B = 009 ;
 if AGE in ( 39,40,41 ) then AGE_B = 010 ;
 if AGE in ( 42,43,44 ) then AGE_B = 011 ;
 if AGE in ( 45,46,47,48 ) then AGE_B = 012 ;
 if AGE in ( 49 ) then AGE_B = 013 ;
 if AGE in ( 50 ) then AGE_B = 014 ;
 if AGE in ( 51,52 ) then AGE_B = 015 ;
 if AGE in ( 53,54 ) then AGE_B = 016 ;
 if AGE in ( 55,56,57 ) then AGE_B = 017 ;
 if AGE in ( 58,59,60,61 ) then AGE_B = 018 ;
 if AGE in ( 62,63,64 ) then AGE_B = 019 ;
 if AGE in ( 65,66,67,68,70,74,75 ) then AGE_B = 020 ;
```

Now %ORDINAL_BIN is applied to the predictor AGE_B.

```
%ORDINAL_BIN(DATASET=TEMP, X=AGE_B, TARGET=Y, W=1, RANKING=IV, ORDER=D,
MISS=, SUMMARYONLY=YES, N_BEST=, N_MONO=, MIN_PCT=, MIN_NUM=, MIN_BIN=,
MAX_BIN=, NOPRINT_WOE=, PRINT1_WOE=, PRINT2_WOE=,
RUN_TITLE=20 bin solution for ORDINAL_BIN, DELETE_PRIOR=);
```

Here is the report from %ORDINAL_BIN:

Obs	bins_in_solution	solution_num	IV	minus2LL	turns	missing	best_rank	best_mono
1	20	1	0.23602	1175.43	16	N	*	
2-13	Obs 2 to 13 are omitted.							
14	7	1	0.16953	1187.42	4	N	*	
15	6	1	0.15168	1190.63	2	N	*	
16	6	5204	0.08337	1204.10	0	N		*
17	5	1	0.13436	1193.70	2	N	*	
18	5	205	0.10406	1199.87	0	N		*
19	4	1	0.12394	1196.28	1	N	*	
20	4	12	0.10402	1199.88	0	N		*
21	3	1	0.10015	1200.60	0	N	*	*
22	2	1	0.07317	1206.09	0	N	*	*

Table 22.

A promising monotonic solution is found at k=5 with solution_num = 205. The IV is 0.10406. There is a monotonic solution for k=6 but the IV is much lower at IV = 0.08337.

The bins that comprise the k=5 solution are obtained by an additional run of %ORDINAL_BIN. Set ***MIN_BIN***=5 and ***MAX_BIN***=5 to exclude other k. To obtain SAS code for the 5-bin monotonic solution, set ***PRINT1_WOE***=5 and ***PRINT2_WOE***=5.

```
%ORDINAL_BIN(DATASET=TEMP, X=AGE_B, TARGET=Y, W=1, RANKING=IV, ORDER=D,
MISS=, SUMMARYONLY=, N_BEST=, N_MONO=, MIN_PCT=, MIN_NUM=, MIN_BIN=5,
MAX_BIN=5, NOPRINT_WOE=, PRINT1_WOE=5, PRINT2_WOE=5,
RUN_TITLE=20 bin solution for ORDINAL_BIN, DELETE_PRIOR= );
```

solution_num	BIN Coding	Counts	Age Range
205	if AGE_B in (1,2) then AGE_B_B = 1 ;	190	19-25
205	if AGE_B in (3,4) then AGE_B_B = 2 ;	181	26-29
205	if AGE_B in (5,6) then AGE_B_B = 3 ;	177	30-34
205	if AGE_B in (7,8,9,10,11,12,13,14,15,16,17,18) then AGE_B_B = 4 ;	414	35-61
205	if AGE_B in (19,20) then AGE_B_B = 5 ;	38	62+

Table 23. Columns "Counts" and "Age Range" were added by a manual process.

This table is identical to Table 21 from %MONOBIN. Admittedly, the method of using the "%NOD_BIN 20-bin solution followed by %ORDINAL_BIN" is a bit messy and is a somewhat ad hoc approach.

But this method has the advantage of displaying other monotonic solutions and alternative non-monotonic solutions. The ***MIN_NUM*** parameter in %NOD_BIN can prevent solutions with small bin counts.

HOW MONOTONIC BINNING WORKS WITH TRANSREG

An explanation of how PROC TRANSREG implements monotonic regression is given in *SAS/STAT® 15.1 User's Guide The TRANSREG Procedure*, pp. 10357-10358.[10] The predictor X is numeric and the target Y is numeric and not restricted to be binary.

I think, in principle, %MONOBIN, using TRANSREG, accomplishes this:

a) Identifies all monotonic binning of X v. binary Y … I'll call them generically: X_bin
b) For each, R-Square is computed for the linear model which can be depicted by:

PROC GLM; CLASS X_bin; MODEL Y= X_bin;

c) The optimal solution is the one with largest R-square.[11]

HISTORY OF THE TWO BINNING MACROS AND REQUESTS FOR MACRO CODE

%NOD_BIN appeared as %BEST_COLLAPSE in Lund and Brotherton (2013), Information Value Statistic, *MWSUG*. Complete code appears in the Appendix of that paper. There were 272 lines of code with 20 lines of comments.

%ORDINAL_BIN was introduced in 2016 and also %BEST_COLLAPSE became %NOD_BIN. See Lund (2016), Weight of Evidence Coding and Binning of Predictors in Logistic Regression, *MWSUG*.

Today, %ORDINAL_BIN has about 1,600 lines of code of which about 350 are comments. %NOD_BIN also has about 1,600 lines of which about 300 are comments. Much of the new code is devoted to input parameter error checking and the rest to new functionality.

I think the current macro code would not be accessible to a new user without a considerable effort. The code becomes a "black box". So cautioned, I will provide the macro code and user documentation to any requester.

SESUG 2023, Charlotte, NC

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author Bruce Lund at: blund_data@mi.rr.com or blund.data@gmail.com.

[10] https://support.sas.com/documentation/onlinedoc/stat/151/transreg.pdf

[11] R-square, when applied to a binary target, is not the same measure as either IV or entropy. I assume it would be strongly correlated. I have not investigated the correlation between R-square and IV or entropy.

APPENDIX

EXAMPLE TO SHOW SUBOPTIMAL %NOD_BIN WITH MODE=A AND METHOD=IV

An example is given for ***MODE***=A and ***METHOD***=IV where the binning solution by %NOD_BIN is non-optimal for, at least, level k= 2. In general, there is no indicator provided by %NOD_BIN that a k-bin solution is suboptimal. Suboptimal solutions can also occur for ***METHOD***=LL.

The example "INCOME1" given in this Appendix was found by chance. The IV value for the 2-bin solution is 0.0844. See the %NOD_BIN Summary Report. In the 2-bin solution the first bin is {01, 02, 03} and the second is {04, 05, 06, 07, 08, 09, 10, 11, 12}.

A second %NOD_BIN is run on data set "INCOME2". In INCOME2 the levels "01", "02", "03", "04" are forced together in the DATA Step which creates INCOME2. A better solution (not necessarily optimal) for k= 2 is {01, 02, 03, 04} and {05, 06, 07, 08, 09, 10, 11, 12}. This is shown in the second %NOD_BIN Summary Report where IV is 0.0888.

The %NOD_BIN algorithm on INCOME1 "goes bad" at k= 5 when "04" is combined with "05". Once combined, "04" cannot be split off to join {01, 02, 03}. In passing, it is noted that the modeler would likely stop the combining process at k= 7 based on the drop off in IV from k= 7 to k=6.

```
DATA Income;
INPUT Income Y W @@;
DATALINES;
01 0 1393 01 1 218
02 0 6009 02 1 890
03 0 5083 03 1 932
04 0 4519 04 1 1035
05 0 8319 05 1 2284
06 0 4841 06 1 1593
07 0 2689 07 1 1053
08 0 2090 08 1 872
09 0 729  09 1 311
10 0 292  10 1 136
11 0 253  11 1 120
12 0 294  12 1 142
;
DATA INCOME1; SET Income;
   length Income_c $2;
   Income_c = put(income,Z2.);
run;

%NOD_BIN(
DATASET = INCOME1, X = Income_c, TARGET = Y, ZERO_ONE = YES,
W = W, METHOD = IV, MODE = A, ORDER = D, MISS = , MIN_PCT = ,
MIN_NUM = , MIN_BIN = , MAX_BIN = , VERBOSE = YES, VERBOSE2 = ,
LL_STAT = , WOE = , ADD = , RUN_TITLE = Suboptimum Solution);

DATA INCOME2; SET income;
  length Income_c $11;
  Income_c = put(income,Z2.);
  if Income_c in ("01" "02" "03" "04") then Income_c = "01_02_03_04";
run;
```

```
%NOD_BIN(
DATASET = INCOME2, X = Income_c, TARGET = Y, ZERO_ONE = YES,
W = W, METHOD = IV, MODE = A, ORDER = D, MISS = , MIN_PCT = ,
MIN_NUM = , MIN_BIN = , MAX_BIN = , VERBOSE = YES, VERBOSE2 = ,
LL_STAT = , WOE = , ADD = ,
RUN_TITLE = Force combine of 01 02 03 04 and get Better Solution);
```

Suboptimum Solution
Dataset= Income1, Predictor= income_c, Target= Y, Zero_one= YES, Freq= W,
Method= IV, Mode= A, Miss= , Order= D
Min_Pct= 0, Min_Num= 0, ADD= N/A
Summary Report

k	IV	-2* Log L	X_ STAT	L1	L2	L3	L4 to L12 omitted
12	0.1214	46223.0	0.5980	01	02	03	
11	0.1214	46223.0	0.5980	01	02	03	
10	0.1214	46223.0	0.5979	01	02	03	
9	0.1214	46223.1	0.5979	01	02	03	
8	0.1214	46223.6	0.5978	01+02	03	04	
7	0.1211	46225.5	0.5975	01+02	03	04	
6	0.1205	46231.0	0.5971	01+02	03	04	
5	0.1179	46250.1	0.5946	01+02	03	**04+05**	
4	0.1151	46267.8	0.5928	01+02+03	04+05	06	
3	0.1103	46307.4	0.5890	01+02+03	04+05	06+07+08+09+ 10+11+12	
2	**0.0844**	46512.9	0.5646	01+02+03	04+05+06+07+ 08+09+10+11+12		

Table 24. A Suboptimum Solution with METHOD=IV

Force combine of 01 02 03 04 and get Better Solution
Dataset= Income2, Predictor= income_c, Target= Y, Zero_one= YES, Freq= W,
Method= IV, Mode= A, Miss= , Order= D
Min_Pct= 0, Min_Num= 0, ADD= N/A
Summary Report

k	IV	-2* Log L	X_ STAT	L1 (forced at start)	L2	L3	L4 to L12 omitted
9	0.1092	46304.3	0.5908	01_02_03_04	05	06	
8	0.1092	46304.4	0.5908	01_02_03_04	05	06	
7	0.1092	46304.4	0.5907	01_02_03_04	05	06	
6	0.1092	46304.5	0.5907	01_02_03_04	05	06	
5	0.1090	46306.4	0.5904	01_02_03_04	05	06	
4	0.1083	46311.9	0.5900	01_02_03_04	05	06	
3	0.1053	46335.3	0.5868	01_02_03_04	05+06	07+08+09+ 10+11+12	
2	**0.0888**	46468.6	0.5725	01_02_03_04	05+06+07+08+ 09+10+11+12		

Table 25. A Better Solution with METHOD=IV after forced combine of 01 02 03 04

EXAMPLE TO SHOW %NOD_BIN MAY HAVE DIFFERENT BINS FOR METHOD=IV VS. LL

%NOD_BIN with ***METHOD***=LL may be run and compared to ***METHOD***=IV. The reader can verify that the 5-bin solutions are different.

```
%NOD_BIN(
DATASET = Income1, X = Income_c, TARGET = Y, ZERO_ONE = YES,
W = W, METHOD = LL, MODE = A, ORDER = D, MISS = , MIN_PCT = ,
MIN_NUM = , MIN_BIN = , MAX_BIN = , VERBOSE = YES, VERBOSE2 = ,
LL_STAT = , WOE = , ADD = , RUN_TITLE = Different LL solution);
```

PARAMETER LL_STAT IN %NOD_BIN WHEN MODE=A

When ***LL_STAT***=YES is selected by the user, then a nested ChiSq and the associated right tail probability Pr > ChiSq perform a test of whether the coefficients of dummy variables for the bins about to the combined are statistically equal. It is best to give an example to illustrate this feature:

```
DATA TEST_A;
INPUT X $ Y W @@;
datalines;
AA   0 500  C 1 310
AA   1 330  D 0 400
BA   0 300  D 1 210
BA   1 270  E 0 550
C    0 400  E 1 400
;
```

Summary Report
Consider stopping at k+1 if (Pr > ChiSq) < 0.05, or other alpha, at k

k	IV	Like-Ratio Chi_Sq	-2*Log L	Nested ChiSq	Pr > ChiSq	L1	L2	L3	L4	L5
5	0.0264	23.36	4955.65	N/M	N/M	AA	BA	C	D	E
4	0.0259	22.96	4956.06	0.402	0.526	AA	BA	C+E	D	
3	0.0236	20.89	4958.13	2.069	0.150	AA+C+E	BA	D		
2	0.0170	14.98	4964.04	5.910	0.015	AA+C+E+BA	D			

Table 26. With LL_STAT=YES

Because Pr > ChiSq is 0.015 for k = 2 (well below 0.05), the combine of {AA C E} with {BA} should not be performed.

Now, the %NOD_BIN result is compared with PROC LOGISTIC.

```
DATA dummies; SET TEST_A;
dum_AA_C_E = (X = "AA" or X = "C" or X = "E");
dum_BA = (X = "BA");
dum_D = (X = "D");
PROC LOGISTIC DATA = dummies;
MODEL Y = dum_AA_C_E dum_BA /* dum_D is reference */;
TEST dum_AA_C_E = dum_BA;
FREQ W;
run;
```

Linear Hypotheses Testing Results			
Label	Wald Chi-Square	DF	Pr > ChiSq
Test 1	5.9300	1	0.015

Table 27. PROC LOGISTIC TEST of EQUALITY of COEFFICIENTS

The coefficients of the dummy variables for {AA C E} and {BA} are statistically unequal.