

# Entropy-based Measures of Weight of Evidence and Information Value for Variable Reduction and Segmentation for Continuous Dependent Variables

Alec Zhixiao Lin, PayPal Credit, Timonium, MD

## ABSTRACT

Weight of Evidence (WOE) and Information Value (IV) have become important tools for analyzing and modeling binary outcomes such as default in payment, response to a marketing campaign, etc. This application encounters difficulty when dealing with continuous outcomes because non-occurrence is either unquantifiable or non-existent. Going back to the fundamentals of Information Theory, this paper suggests a set of alternative formulae that attempts to dichotomize high occurrences and low occurrences in order to expand the use of WOE and IV to continuous outcomes such as sales volume, loss amount, etc. A SAS<sup>®</sup> macro program is provided that will efficiently evaluate the predictive power of continuous, ordinal and categorical variables together and yield useful suggestions for variable reduction/selection, segmentation and subsequent linear or logistic regressions.

## INTRODUCTION

Based on Information Theory conceived in later 1940s and initially developed for scorecard development, Weight of Evidence (WOE) and Information Value (IV) have been gaining increasing attention in recent years for such uses as segmentation and variable reduction.

As the calculation of WOE and IV requires a contrast between occurrence and non-occurrence (usually denoted by 1 and 0), their use has largely been limited to the analysis of and modeling for binary outcomes such as default in payment, response to a marketing campaign, etc. Using risk analysis as example, the following is how WOE and IV are calculated:

$$WOE = \left[ \ln \left( \frac{\%Bad_i}{\%Good_i} \right) \right] \times 100$$

$$IV = \sum_{i=1}^n \left( (\%Bad_i - \%Good_i) \times \ln \left( \frac{\%Bad_i}{\%Good_i} \right) \right)$$

Can we use the above formulae for a continuous outcome? In an example of sales analysis in which Sales Volumes is the target of interest, can rewrite the formula of WOE to the following?

$$WOE = \left[ \ln \left( \frac{\%Sales_i}{\%NonSales_i} \right) \right] \times 100$$

There are two problems: 1) NonSales is qualitative and unquantifiable. 2) If every person in the sample has made sales, NonSales becomes non-existent. In order to get around this, an alternative set of formulae has been suggested as follows by replacing %NonSales<sub>i</sub> with %Records<sub>i</sub>:

$$WOE = \left[ \ln \left( \frac{\%Sales_i}{\%Records_i} \right) \right] \times 100$$

$$IV = \sum_{i=1}^n \left( (\%Sales_i - \%Records_i) \times \ln \left( \frac{\%Sales_i}{\%Records_i} \right) \right)$$

But the formulae above still have a flaw: %Records includes records with sales and those with no sales, so the dichotomy between the numerator and the denominator is less than clean.

## QUANTIFYING HIGH OCCURRENCES AND LOW OCCURRENCES

Suppose a company has 500 salespersons and the sales volume for each person is \$440, table 1 uses 10 persons to illustrate how a contrasting pair of Positive Sales and Negative Sales can be created.

Salesperson	Sales Volume	Sales Volume Standardized	Positive Sales	Negative Sales
A	\$ 1,250.0	\$ 810.0	\$ 810.0	\$ -
B	\$ 150.0	\$ (290.0)	\$ -	\$ 290.0
C	\$ 700.0	\$ 260.0	\$ 260.0	\$ -
D	\$ 100.0	\$ (340.0)	\$ -	\$ 340.0
E	\$ 200.0	\$ (240.0)	\$ -	\$ 240.0
F	\$ 475.0	\$ 35.0	\$ 35.0	\$ -
G	\$ 600.0	\$ 160.0	\$ 160.0	\$ -
H	\$ 390.0	\$ (50.0)	\$ -	\$ 50.0
I	\$ 450.0	\$ 10.0	\$ 10.0	\$ -
J	\$ 550.0	\$ 110.0	\$ 110.0	\$ -
avg sales volume	\$ 440.0			

**Table1. An Example of Sales Volume**

Using average sales as a benchmark, we can judge whether a salesperson has above-par or below-par performance:

High occurrence: If  $Sales_i > AvgSales$  then Positive Sales =  $Sales_i - AvgSales$

Low occurrence: If  $Sales_i \leq AvgSales$  then Negative Sales =  $AvgSales - Sales_i$

Please note that AvgSales is the average sales based on the entire sales force, not on the segment chosen for illustration in the table above. High occurrence ( $PosSales > 0$ ) suggests extra sales above average while low occurrence ( $NegSales > 0$ ) indicates the opposite. Users can also use median sales or a combination of the median and the average as the benchmark. For each segment, we can compute the following:

%PosSales<sub>i</sub>: Positive Sales of a segment over all Positive Sales in the sample

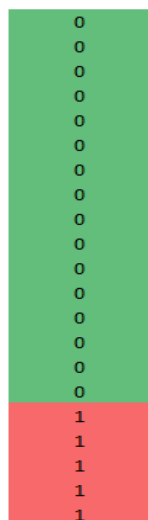
%NegSales<sub>i</sub>: Negative Sales of a segment over all Positive Sales in the sample

By directly contrasting high occurrence ( $PosSales > 0$ ) and low occurrence ( $NegSales > 0$ ), the above formulae are more rigorously consistent with the concept of entropy commonly employed in Information Theory: the juxtaposition of two opposite forces in one equation. With this “quantified” dichotomy, we suggest that the formulae for WOE and IV be rewritten as the following:

$$WOE = \left[ \ln \left( \frac{\%PosSales_i}{\%NegSales_i} \right) \right] \times 100$$

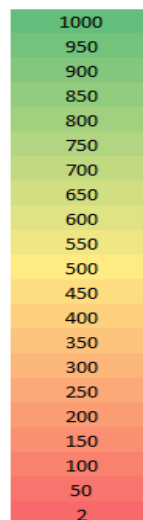
$$IV = \sum_{i=1}^n \left( (\%PosSales_i - \%NegSales_i) \times \ln \left( \frac{\%PosSales_i}{\%NegSales_i} \right) \right)$$

We need to point out one critical difference between the original formulae for binary outcomes and the new formulae for continuous outcomes: For a binary outcome, each record weighs and contributes to WOE and IV equally since each has the outcome coded as either 0 or 1. In the case of a continuous outcome, records of very high or of very low occurrences contribute more and hence exert more impact. The rule of thumb of judging whether variables have strong ( $IV > 0.3$ ), median ( $0.1 < IV \leq 0.3$ ), weak ( $0.02 < IV \leq 0.1$ ) or no ( $IV \leq 0.02$ ) predictive power, as suggested by Siddique for binary outcomes, does not apply to the situation of continuous outcomes.



**Figure 1: a Binary Outcome**

- There is a clear-cut dichotomy between occurrence and non-occurrence.
- Each record is weighed equally and contributes to WOE and IV equally.



**Figure 2: a Continuous Outcome**

- No clear-cut dichotomy.
- Records on the top weigh more in PosSales than those in the middle. Records at the bottom weigh more in NegSales than those in the middle. That is, records of higher or lower occurrences contribute more to WOE and IV.

In general, variables of higher IVs are considered to be more predictive. This suggests that for numerous variables in a data set, we can compare their IVs to determine which ones to retain or to delete. However, IV depends on the contrast in performance between different segments only. In the following highly stylized example, data in Table 2 suggest a good monotonicity. After swapping the values of Sales between some rows, the pattern of monotonicity is totally disrupted, but IV remains the same.

Score Band	# Records	%Records	Sales
1-50	100	10.0%	1,116
51-100	100	10.0%	871
101-150	100	10.0%	693
151-200	100	10.0%	522
201-250	100	10.0%	421
251-300	100	10.0%	363
301-350	100	10.0%	305
351-400	100	10.0%	247
401-450	100	10.0%	174
451-500	100	10.0%	131
Total	1,000	100.0%	484

**Table 2. WOE and Good Monotonicity**

Score Band	# Records	%Records	Sales
1-50	100	10.0%	421
51-100	100	10.0%	871
101-150	100	10.0%	174
151-200	100	10.0%	363
201-250	100	10.0%	305
251-300	100	10.0%	247
301-350	100	10.0%	1,116
351-400	100	10.0%	693
401-450	100	10.0%	131
451-500	100	10.0%	522
Total	1,000	100.0%	484

**Table 3. WOE and Bad Monotonicity**

In order to provide additional suggestions for linear or logistic regressions, we have also included the followings statistics in the SAS programs provided:

- Gini coefficients for numeric independent variables, regardless whether the outcome is binary or continuous. A higher Gini coefficient suggests a higher potential for the variable to be useful in a linear regression.
- Chi-square statistics for categorical independent variables and a binary outcome;
- F-statistics from Tukey test for categorical independent variables and a continuous outcome.

Chi-square and F-statistics lend insights on how to transform a categorical variable into a class variable.

## THE SAS PROGRAMS

This paper provides a SAS macro program that can be used to assess the predictive power of continuous, ordinal and categorical independent variables together. Apart from greatly expediting the process of variable reduction/selection, the multiple SAS outputs generated by the programs can provide useful suggestions for segmentation, scorecard development, imputation for missing values, flooring and ceiling variables, creating class variables, etc.

Even though discussion of this paper focuses on continuous outcomes, the SAS programs retain the flexibility of analyzing binary outcomes as well. If a target variable is binary, the programs revert to the calculation of original WOE and IV.

We suggest doing the following simple preliminary preparations for the data:

- Run PROC CONTENTS to separate categorical and numerical variables. All categorical variables should be expressed as characters.
- Convert ordinal variables to character and lump them together with categorical variables.
- Run PROC MEANS to screen out variables with no coverage or with a uniform value (when MIN=MAX). You can also consider deleting those variables with very low coverage. These simple measures help to reduce the file size and to shorten the processing time.

The SAS programs provided in the appendices contain four parts and could look intimidating to some, but one does not need to digest the long codes in order to use them. Programs from Appendix 1a, Appendix 1b and Appendix 2 will suffice for most users. The following is how you can make use of them:

- Save Appendix 1a and Appendix 1b to a designated folder without making any changes.
- Save Appendix 2 to a designated folder. First-time users only need to make limited changes to the macro values listed in Part I at the beginning of the program.

```

                ** Part I - SPECIFY DATA SET AND VARIABLES;
%let numfile=C:/SAS/appendix 1a.sas;    /* for processing numeric variables */
%let charfile=C:/SAS/appendix 1b.sas;    /* for processing categorical and
                                         ordinal variables */
libname your_lib "c:/sas/mydata/data";    /* the libname */
%let libdata=your_lib;                    /* libname defined in the previous line */
%let inset=yousasfile;                    /* your SAS data set */
%let vartxt=xchar1 xchar2 ...;            /* all categorical/ordinal variables.
    Variables should be in character format only. Set vartxt= if none.
    Ensure no duplications in variables names */
%let varnum=xnum1 xnum2 ...;              /* all numeric variables. Set varnum= if none.
    Ensure no duplications in variables names */
%let target=sales_amt;                    /* target variable (y) */
%let ytype=contin; /* continu=continuous outcome, binary=binary outcome */
%let outname=% Sales Volume;              /* label of target y for summary outputs */
%let targetvl=15.2; /* change to percent8.3 or others for binary outcomes */
%let libout=C:/SAS/output;                /* folder for outputting summaries */

```

Once you have gained some familiarity with the programs and the associated SAS outputs, you can make changes to items in Part II as you see fit:

```

                **Part II - CHANGE THE FOLLOWING AS YOU SEE FIT;
%let tiermax=10; /* maximum number of bins for continuous variables */
%let ivthresh=0; /* For a continuous outcome, always set this to 0.
    For a binary outcome, set ivthresh=0.03 or higher */
%let mean_median_combo=1; /* 1=mean as cutoff, 0=median as cutoff. You can
    choose any number between for combining mean and median */
%let capy=98; /* outlier cap for continuous outcomes.
    It has no impact on a binary outcome */
%let tempmiss=-1000000000000; /* missing values for numeric variables */
%let charmiss=_MISSING_; /* missing values for categorical variables */
%let formtall=12.3; /* format accommodating all numeric variables.
    Extend to more digits after decimal to accommodate very small values */
%let outgraph=gh_&target; /* name pdf graph for predictors */
%let ivout=iv_&target; /* name output file for IV */
%let woeout=woe_&target; /* name of output file for WOE */

```

(See the rest of the program in Appendix 2.)

There is no need to change any code from Part III on.

## SAS OUTPUTS

The program generates several output summaries in a designated folder. These summaries provide a useful guidance for variable reduction, segmentation, scorecard development, etc.

- 1) A flow table of IV for all variables, ranked from highest to lowest, as illustrated by Table 4. Please note the following when reading the output:
  - The program evaluates numeric, ordinal and categorical variables together. It will automatically combine the summaries for numeric variables and for categorical/ordinal variables together and rank their IVs from highest to lowest.
  - % Obs Missing refers to the coverage by the variable;
  - If a numeric variable is high on IV Rank but low on Gini coefficient (xnum4 and xnum111, for example), it usually suggests a lack of linearity.
  - Variables of the same or very similar IV usually denote very similar activities and exhibit a very high correlation.

Variable	Variable Type	IV	IV Rank	% Obs Missing	Chi-Sq, Tukey or Gini
xnum12	num	0.6804	1	0.0%	num-4
xnum7	num	0.6319	2	0.0%	num-5
xnum4	num	0.4789	3	0.0%	num-22
xnum111	num	0.4758	4	0.0%	num-23
xnum67	num	0.4278	5	17.7%	num-3
xnum55	num	0.4191	6	0.0%	num-26
xnum38	num	0.4058	7	0.0%	num-24
xchar5	char	0.3250	8	0.0%	char-1
xnum23	num	0.2962	9	0.0%	num-37
xnum24	num	0.2942	10	0.0%	num-38

**Table 4. IV List**

- 2) A flow table of WOE for all variables, ranked in a descending order by IV. While preserving all contents in the flow table of IV, the flow table of WOE expands to bins within each variable.

Variable	type	IV	ivrank	# Records	% Records	Sales Volume	WOE	% Missing Values	Tier/Bin	bin min	bin max
xnum2	num	0.106142	1	2,568	1.2%	316	76.37	1.2%	-1E+12		
xnum2	num	0.106142	1	21,987	9.9%	894	(27.54)	1.2%	46	0	71
xnum2	num	0.106142	1	21,931	9.9%	902	(28.47)	1.2%	99	72	130
xnum2	num	0.106142	1	21,860	9.9%	901	(28.37)	1.2%	168	131	211
xnum2	num	0.106142	1	22,029	9.9%	848	(22.32)	1.2%	261	212	317
xnum2	num	0.106142	1	21,878	9.9%	778	(13.74)	1.2%	381	318	453
xnum2	num	0.106142	1	21,933	9.9%	674	0.68	1.2%	537	454	634
xnum2	num	0.106142	1	21,881	9.9%	583	15.17	1.2%	751	635	888
xnum2	num	0.106142	1	21,954	9.9%	510	28.59	1.2%	1056	889	1261
xnum2	num	0.106142	1	21,947	9.9%	416	48.84	1.2%	1534	1262	1912
xnum2	num	0.106142	1	21,929	9.9%	321	74.86	1.2%	2590	1913	9993
xchar18	char	0.014396	18	5,981	2.7%	752	10.25	0.0% BOTH_CONTROLLED			
xchar18	char	0.014396	18	17,615	7.9%	719	(5.80)	0.0% BOTH_ISP			
xchar18	char	0.014396	18	12,054	5.4%	771	12.75	0.0% EQUAL_CONTROLLED			
xchar18	char	0.014396	18	26,590	12.0%	747	(9.61)	0.0% EQUAL_ISP			
xchar18	char	0.014396	18	156,466	70.5%	639	5.97	0.0% NONE			
xchar18	char	0.014396	18	3,091	1.4%	1,331	67.37	0.0% NO_EMAILS			
xchar18	char	0.014396	18	95	0.0%	1,294	64.56	0.0% NO_IPS			
xchar18	char	0.014396	18	5	0.0%	393	54.60	0.0% _MISSING_			

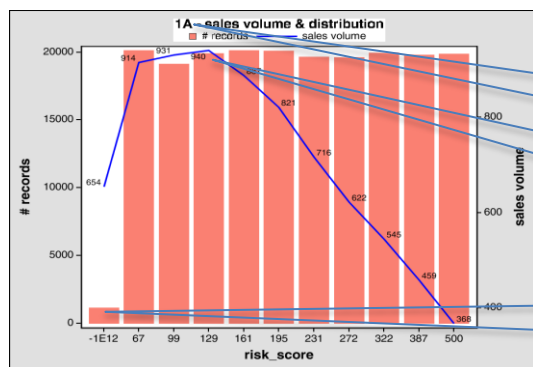
**Table 5. WOE List**

We offer a few tips on how to read and use the table:

- For a numeric variable exhibiting a nonlinear pattern, we can use WOE, Bin Min and Bin max to transform it into derived variable for regression. Please note that the column Tier/Bin is the median values of a numeric variable for the associated bin.
- For a categorical or ordinal variable, use the value from the column Tier/Bin to transform it into a derived variable for regression.

3) A PDF collage of WOE for each variable. Modelers will find these graphs to be useful for such purposes as examining variable behavior, imputing missing values, etc..

Outputs for numeric variables and for categorical/ordinal variables are slightly different. The following is an example of graphic outputs for a numeric variable.

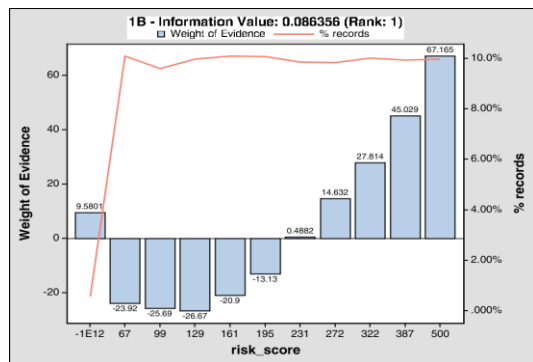


**Figure 3-1: Variable Behavior & Distribution**

All charts are ordered by IV from highest to lowest. '1' means it is the most predictive power among all variables.

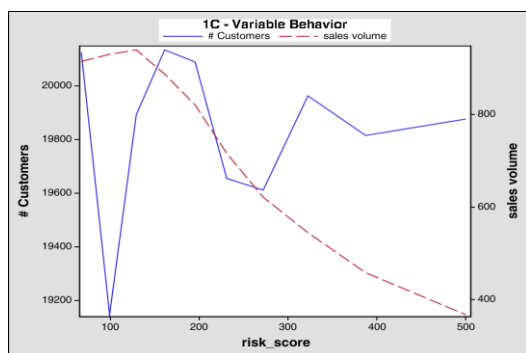
One can floor the risk\_score as follows:  
if risk\_score < 130 then risk\_score=130.

risk\_score=. can be imputed as risk\_score=260  
because of their similarity in behavior in terms of the target outcome.



**Figure 3-2: Graphic Representation of WOE**

- Each value in x axis denotes the median value of the bin.
- An equal distance between bins do not suggest an equal difference between two medians.



**Figure 3-3: Variable Behavior & Distribution**

This chart shares some similarity to Figure 3-1, but with the following differences:

- The distance between two points in X axis is the actual difference in value;
- Missing values are not plotted.

**Figure 3. Graphs for a Numeric Variable**

We now illustrate the use of graphic for categorical and ordinal variables in Figure 4-1 to Figure4-3. The first two are similar to what we have seen for numeric variables, but the third graph presents a somewhat different view.

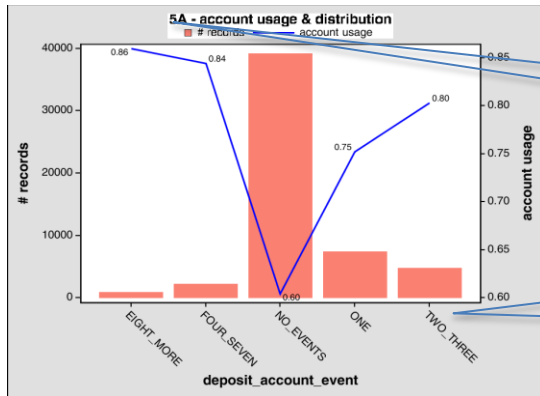


Figure 4-1: Variable Behavior & Distribution

'5' suggests that the variable has the 5<sup>th</sup> biggest IV.

Values in X axis are ordered alphabetically by the flags.

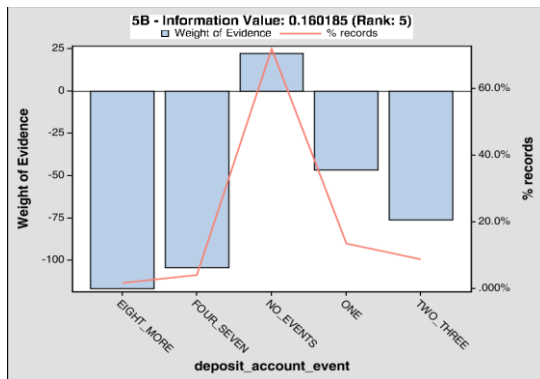


Figure 4-2: Graphic Representation of WOE

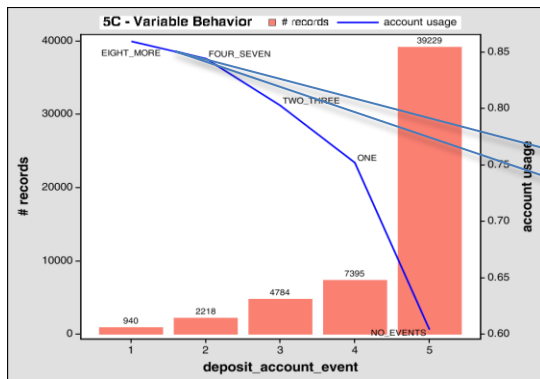


Figure 4-3: Variable Behavior & Distribution

Flags for the categorical variables are ranked in a descending order by the target outcome. This helps modelers to group flags.

- The first two flags exhibits very similar behavior, they can be binned together for regression or segmentation.
- One can also consider translating the categorical value into a numerical tier for modeling.

Figure 4. Graphs for a Categorical Variable

## IN CASE OF VERY BIG DATA

Processing numerous variables for a large data set often runs into the problem of insufficient system memory. To partially overcome this problem, the SAS program in APPENDIX 2 automatically splits a data set into ten subsets by variables for separate processing and then integrates all at the end. But sometimes even this manoeuvre is not adequate. One simple solution is to run the same processing (Appendix 2) on a reduced sample. We can also consider screening out variables of extremely low IVs in a broad-brushed manner. The program in Appendix 3 is for this purpose and works in the follow way:

- It uses DO LOOP to accommodate as many variables as possible.
- Only an IV list will be generated at the end. The list ranks IVs of all variables from highest to lowest. No PDF collages will be generated in order to expedite the process.

The following is an example of dividing 3000 variables into 30 lists for consecutive processing.



```

%let numfile=C:/SAS/appendix 2a.sas;
%let charfile=C:/SAS/appendix 2b.sas;
libname your_lib "C:/SAS/data";
%let libdata=your_lib;                                /* libname */
%let inset=yoursasdata;                               /* input SAS data set */
%let target=sales_amt;                                /* target variable (y). */
%let ytype=continuu; /* continu=continuous outcome, binary=binary outcome */
%let outname=Sales Volume;                            /* label of target y */
%let targetvl=15.2;                                   /* format of dependent variable */
%let libout=C:/SAS/output;                            /* output folder */

                ** partitioned lists of all variables;
%let listnum=30;                                       /* # expand to as many lists as needed */
%let varlist1=X1 X2 ... X100;
%let varlist2=X101 X102 ... X200;
%let varlist3=X201 X201 ... X300;
.....
%let varlist30=X2901 X2902 ... X3000; /* correspond to macro value for listnum */
(See the rest of the program in Appendix 3.)

```

After deleting variables with very low IVs, users can run the program from Appendix 2 for a closer examination on retained variables.

## CONCLUSION

Even though Weight of Evidence and Information Value were developed for evaluating binary outcomes, we can expand their use to the analysis and modeling for continuous dependent variables. We hope this paper can provide some useful suggestions in this direction.

## ACKNOWLEDGEMENTS

I would like to thank team of Risk Management of PayPal Credit led by Shawn Benner for their continuous support.

## REFERENCES

- Anderson, Raymond (2007) "The Credit Scoring Toolkit: Theory and Practice for Retail Credit Risk Management and Decision Automation", OUP Oxford.
- Gleick, James (2012), "The Information: A History, A Theory, A Flood", Vintage.
- Lin, Alec Zhixiao (2013) "Variable Reduction in SAS by Using Information Value and Weight of Evidence" <http://support.sas.com/resources/papers/proceedings13/095-2013.pdf#!>, proceeding in SUGI Conference 2013.
- Siddiqi, Naeem (2006) "Credit Risk Scorecards: Developing and Implementing Intelligent Credit Scoring", SAS Institute.

## CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the authors at

Alec Zhixiao Lin

Credit Modeling Manager

PayPal Credit

9690 Deereco Road, Suite 110

Timonium, MD 21093

Email: [alecindc@gmail.com](mailto:alecindc@gmail.com)

Web: [www.linkedin.com/pub/alec-zhixiao-lin/25/708/261/](http://www.linkedin.com/pub/alec-zhixiao-lin/25/708/261/)

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® Indicates USA registration.



Other brand and product names are registered trademarks or trademarks of their respective companies.

### APPENDIX 1a (for evaluating numerical variables)

```
data num_modified;
set &libdata..&inset(keep=&target &varnum);
%macro tempnum;
%do i=1 %to 10;
if ranuni(123) < 0.5 then insertnumtemp&i=1; else insertnumtemp&i=0;
%end;
%mend;
%tempnum;
attrib _all_ label=''; run;

ods output nlevels=checkfreq;
proc freq data=num_modified nlevels;
tables &varnum insertnumtemp1-insertnumtemp10 / noprint; run;
ods output close;

data varcnt; set checkfreq; varcnt+1; run;
proc univariate data=varcnt noprint;
var varcnt;
output out=pctscore pctlpts=0 10 20 30 40 50 60 70 80 90 100
pctlpre=pct_; run;

data _null_;
set pctscore;
call symputx('start1', 1);
call symputx('end1', int(pct_10)-1);
call symputx('start2', int(pct_10));
call symputx('end2', int(pct_20)-1);
call symputx('start3', int(pct_20));
call symputx('end3', int(pct_30)-1);
call symputx('start4', int(pct_30));
call symputx('end4', int(pct_40)-1);
call symputx('start5', int(pct_40));
call symputx('end5', int(pct_50)-1);
call symputx('start6', int(pct_50));
call symputx('end6', int(pct_60)-1);
call symputx('start7', int(pct_60));
call symputx('end7', int(pct_70)-1);
call symputx('start8', int(pct_70));
call symputx('end8', int(pct_80)-1);
call symputx('start9', int(pct_80));
call symputx('end9', int(pct_90)-1);
call symputx('start10', int(pct_90));
call symputx('end10', pct_100); run;

proc sql noprint; select tablevar into :varmore separated by ' ' from varcnt;
quit;
proc sql; create table vcnt as select count(*) as vcnt from varcnt; quit;
data _null_; set vcnt; call symputx('vmcnt', vcnt); run;

proc sql noprint; select tablevar into :v1-:v&vmcnt from varcnt; quit;
proc sql noprint; select max(varcnt), compress('&x'||put(varcnt, 10.))
into :varcount, :tempvar separated by ' ' from varcnt order by varcnt; quit;
proc sql noprint; select tablevar into :x1-:x&end10 from varcnt; quit;
```

```

proc sql noprint; select count(*) into :obsent from num_modified; quit;

%macro stkorig;
%do i=1 %to &vmcnt;
data v&i;
length tablevar $ 32;
set num_modified(keep=&&v&i rename=(&&v&i=origvalue));
tablevar="&&v&i";
format tablevar $32.; run;

proc rank data=v&i groups=&tiermax out=v&i;
by tablevar;
var origvalue;
ranks rankvmore; run;

proc means data=v&i median mean min max nway noprint;
class tablevar rankvmore;
var origvalue;
output out=vmoreranked&i(drop=_type_ _freq_)
      median=med_origv
      mean=mean_origv
      min=binmin
      max=binmax; run; %end;
%mend;
%stkorig;

data stackorig; set vmoreranked1-vmoreranked&vmcnt; run;

proc rank data=num_modified groups=&tiermax out=try_model(keep=&tempvar
&target);
var &varmore; ranks &varmore; run;

%macro outshell;
%do i=1 %to &varcount;
%macro ybinamax;
%if &ytype=binary %then %do;
proc univariate data=num_modified noprint;
var &target;
output out=pctoutlier(keep=pcty_100 rename=(pcty_100=capy))
      pctlpts=100
      pctlpre=pcty_; run; %end;
%else %do;
proc univariate data=num_modified noprint;
var &target;
output out=pctoutlier(keep=pcty_&capy rename=(pcty_&capy=capy))
      pctlpts=&capy
      pctlpre=pcty_; run; %end;
%end;
%ybinamax;

data try_model;
if _n_=1 then set pctoutlier;
set try_model;
if &&x&i=. then &&x&i=&tempmiss;
if &target > capy then &target=capy; run;

```

```
proc sql noprint; select count(*) into :nonmiss from try_model where &&x&i ne
&tempmiss; quit;
```

```
%macro binary;
```

```
proc sql noprint;
select sum(case when &target=1 then 1 else 0 end), sum(case when &target=0
then 1 else 0 end), count(*)
into :tot_bad, :tot_good, :tot_both
from try_model; quit;
```

```
proc sql;
create table woe&i as
(select "&&x&i" as tablevar,
&&x&i as tier,
count(*) as cnt,
count(*)/&tot_both as cnt_pct,
sum(case when &target=0 then 1 else 0 end) as sum_good,
sum(case when &target=0 then 1 else 0 end)/&tot_good as dist_good,
sum(case when &target=1 then 1 else 0 end) as sum_bad,
sum(case when &target=1 then 1 else 0 end)/&tot_bad as dist_bad,
log((sum(case when &target=0 then 1 else 0 end)/&tot_good)/(sum(case
when &target=1 then 1 else 0 end)/&tot_bad))*100 as woe,
((sum(case when &target=0 then 1 else 0 end)/&tot_good)-(sum(case
when &target=1 then 1 else 0 end)/&tot_bad))
*log((sum(case when &target=0 then 1 else 0
end)/&tot_good)/(sum(case when &target=1 then 1 else 0 end)/&tot_bad)) as
pre_iv,
sum(case when &target=1 then 1 else 0 end)/count(*) as outcomesum

from try_model
group by "&&x&i", &&x&i)
order by &&x&i; quit;
%mend;
```

```
%macro continu;
```

```
proc means data=try_model mean median;
var &target;
output out=check_mean_median(drop=_type_ _freq_)
mean=ymeanvalue
median=ymedianvalue; run;
```

```
data mean_median_combo(keep=ymeanmedvalue);
set check_mean_median;
ymeanmedvalue=sum(&mean_median_combo*ymeanvalue, (1-
&mean_median_combo)*ymedianvalue);
call symputx('ymeanmed', ymeanmedvalue); run;
```

```
data check_mod;
if _N_ = 1 then set mean_median_combo;
set try_model;
```

```
ytarget_good_temp=&target - ymeanmedvalue;
if &target > ymeanmedvalue then do; ytarget_good=&target-ymeanmedvalue;
ytarget_bad=0; end;
else if &target le ymeanmedvalue then do; ytarget_bad=ymeanmedvalue-&target;
ytarget_good=0; end; run;
```

```

proc sql noprint; select sum(ytarget_bad), sum(ytarget_good), count(*)
    into :tot_bad, :tot_good, :tot_both from check_mod; quit;

proc sql;
create table woe&i as
(select "&x&i" as tablevar,
    &x&i as tier,
    count(*) as cnt,
    count(*)/&tot_both as cnt_pct,
    sum(ytarget_bad) as sum_bad,
    sum(ytarget_bad)/&tot_bad as dist_bad,
    log((sum(ytarget_bad)/&tot_bad)/(sum(ytarget_good)/&tot_good))*100 as
woe,
    (sum(ytarget_bad)/&tot_bad-sum(ytarget_good)/&tot_good)

    *log((sum(ytarget_bad)/&tot_bad)/(sum(ytarget_good)/&tot_good)) as
pre_iv,
    sum(&target)/count(*) as outcomesum
    from check_mod
    group by "&x&i", &x&i)
order by &x&i; quit;
%mend;

%&ytype;

proc rank data=temp out=temp; var &x&i; ranks rank_x;
run;

proc means data=temp sum mean nway noprint;
class rank_x;
var &target;
output out=temp2(drop=_type_ rename=(freq=cntweight))
    sum=ycount
    mean=ymean; run;

proc sort data=temp2; by rank_x; run;

data table; set temp2;
retain ysum weightsum;
ysum+ycount; weightsum+cntweight; rank_x_negative=-1*rank_x; run;

proc sort data=table; by rank_x_negative; run;

data table; set table;
by rank_x_negative;
if _n_=1 then do; ytotal=ysum; counttotal=weightsum; end;
retain ytotal counttotal; run;
proc sort data=table; by rank_x; run;

data table; set table;
y_cum_pct=ysum/ytotal; pop_cum_pct=weightsum/counttotal; run;

data ginidat; set table;
xlag = lag(pop_cum_pct); ylag = lag(y_cum_pct);
columna =y_cum_pct*xlag; columnb =pop_cum_pct*ylag;
retain suma sumb;
suma + columna; sumb + columnb;

```

```

gini=abs(suma-sumb); run;

data gini_x(keep=gini); set ginidat; if y_cum_pct=1; run;

data gini&i;
if _N_ = 1 then set gini_x;
set gini_x;
tablevar="&x&i"; run;

proc sql;
create table iv&i as select "&x&i" as tablevar,
                           sum(pre_iv) as iv,
                           (1-&nonmiss/&obsct) as pct_missing
from woe&i; quit; %end;
%mend outshell;
%outshell;

%macro stackset;
%do j=1 %to 10;
data tempiv&j;
length tablevar $ 32;
set iv&start&j-iv&end&j;
format tablevar $32.; run;

data tempwoe&j;
length tablevar $ 32;
set woe&start&j-woe&end&j;
format tablevar $32.; run;

data tempgini&j;
length tablevar $ 32;
set gini&start&j-gini&end&j;
format tablevar $32.; run;
%end;
%mend;
%stackset;

data &libdata..ivall; set tempiv1-tempiv10; if substr(tablevar, 1,
14)='insertnumetemp' then delete; run;
data &libdata..woeall; set tempwoe1-tempwoe10; if substr(tablevar, 1,
14)='insertnumetemp' then delete; run;
data &libdata..giniall; set tempgini1-tempgini10; if substr(tablevar, 1,
14)='insertnumetemp' then delete; run;
proc sort data=&libdata..giniall; by descending gini; run;

proc sort data=&libdata..ivall; by descending iv; run;
data &libdata..ivall; set &libdata..ivall; ivrank+1; run;
proc sort data=&libdata..ivall nodupkey out=ivtemp(keep=iv); by descending
iv; run;
data ivtemp; set ivtemp; ivtier+1; run;
proc sort data=ivtemp; by iv; run;
proc sort data=&libdata..ivall; by iv; run;
data &ivout; merge &libdata..ivall ivtemp; by iv; run;
proc sort data=&ivout; by tablevar; run;
proc sort data=&libdata..woeall; by tablevar; run;
proc sort data=&libdata..giniall; by tablevar; run;

```

```

data &libdata..iv_woe_all;
length tablevar $ 32;
merge &ivout &libdata..woeall &libdata..giniall;
by tablevar; run;

proc sort data=&libdata..iv_woe_all; by tablevar tier; run;
proc sort data=stackorig; by tablevar rankvmore; run;

data &libdata..iv_woe_all2;
length tablevar $ 32;
merge &libdata..iv_woe_all(in=t) stackorig(in=s rename=(rankvmore=tier));
by tablevar tier;
if t;
if s then tier=med_origv; run;

proc sort data=&libdata..iv_woe_all2; by ivrank tier; run;

%let retvar=tablevar iv ivrank ivtier tier cnt cnt_pct dist_bad woe
outcomesum pct_missing binmin binmax;

data &libdata..&woeout(keep=&retvar);
retain &retvar;
set &libdata..iv_woe_all2;

format binmin &formtall;
format binmax &formtall;
informat binmin &formtall;
informat binmax &formtall;

label tablevar="Variable";
label iv="IV";
label ivrank="IV Rank";
label tier="Tier/Bin";
label cnt="# Customers";
label cnt_pct="% CustoeMrs";
label woe="Weight of Evidence";
label outcomesum="&outname";
label pct_missing="% Missing Values";
label binmin="bin min";
label binmax="bin max"; run;

proc sort data=&ivout nodupkey; by tablevar; run;
proc sort data=&libdata..giniall nodupkey; by tablevar; run;
proc sort data=&libdata..&woeout; by tablevar; run;

data &libdata..&ivout;
length tablevar $ 32;
merge &ivout(in=t keep=tablevar iv ivrank ivtier pct_missing)
&libdata..giniall(keep=tablevar gini);
by tablevar;
if t; run;

proc sort data=&libdata..&ivout; by descending gini; run;
proc sort data=&libdata..&ivout nodupkey out=ivtemp(keep=gini); by descending
gini; run;
data ivtemp; set ivtemp; ginitier+1; run;
proc sort data=ivtemp; by gini; run;

```

```

proc sort data=&libdata..&ivout; by gini; run;
data &ivout; merge &libdata..&ivout ivtemp; by gini; run;
proc sort data=&ivout; by tablevar; run;

data &libdata..&ivout;
length type $ 4;
set &ivout;
by tablevar;
type="num ";
format type $4.;
informat type $4.; run;

proc sort data=&libdata..&ivout(drop=ivrank)
out=&ivout._num(rename=(ivtier=ivrank ginitier=power_order)); by ivtier; run;
data &ivout._num(keep=iv ivrank pct_missing tablevar type gini power_order);
set &ivout._num; run;

proc sort data=&ivout._num; by tablevar; run;
proc sort data=&libdata..&woeout; by tablevar; run;

data &libdata..&woeout._t;
merge &libdata..&woeout(in=t)
&ivout._num(keep=tablevar power_order);
by tablevar;
if t; run;
proc sort data=&libdata..&woeout._t(drop=ivrank)
out=&woeout._num(rename=(ivtier=ivrank)); by ivtier tablevar; run;

%macro to_excel(data_sum);
PROC EXPORT DATA=&data_sum
OUTFILE="&libout/&data_sum"
label DBMS=tab REPLACE; run;
%mend;

proc sort data=&woeout._num; by tablevar tier; run;
data &woeout._num(rename=(tier_txt=tier));
set &woeout._num;
by tablevar;
if first.tablevar=1 then seqnum=0; seqnum+1; tier_txt=STRIP(PUT(tier,
best32.)); tier_num=tier; type="num "; output;
drop tier; run;

```

## APPENDIX 1b (for evaluating categorical and ordinal variables)

```

data inset_categorical;
set &libdata..&inset(keep=&target &vartxt);
%macro tempchar;
%do i=1 %to 10;
if ranuni(123) < 0.5 then insertchartemp&i='A'; else insertchartemp&i='B';
%end;
%mend;
%tempchar;
attrib _all_ label=''; run;

ods output nlevels=checkfreq;
proc freq data=inset_categorical nlevels;
tables &vartxt insertchartemp1-insertchartemp10 / noprint; run;

```



```

ods output close;

data varcnt; set checkfreq; varcnt+1; run;

proc univariate data=varcnt noprint;
    var varcnt;
    output out=pctscore pctlpts=0 10 20 30 40 50 60 70 80 90 100
        pctlpre=pct_; run;

data _null_;
set pctscore;
call symputx('start1', 1);
call symputx('end1', int(pct_10)-1);
call symputx('start2', int(pct_10));
call symputx('end2', int(pct_20)-1);
call symputx('start3', int(pct_20));
call symputx('end3', int(pct_30)-1);
call symputx('start4', int(pct_30));
call symputx('end4', int(pct_40)-1);
call symputx('start5', int(pct_40));
call symputx('end5', int(pct_50)-1);
call symputx('start6', int(pct_50));
call symputx('end6', int(pct_60)-1);
call symputx('start7', int(pct_60));
call symputx('end7', int(pct_70)-1);
call symputx('start8', int(pct_70));
call symputx('end8', int(pct_80)-1);
call symputx('start9', int(pct_80));
call symputx('end9', int(pct_90)-1);
call symputx('start10', int(pct_90));
call symputx('end10', pct_100); run;

proc sql noprint; select tablevar into :varmore separated by ' ' from varcnt;
quit;
proc sql noprint; create table vcnt as select count(*) as vcnt from varcnt;
quit;
data _null_; set vcnt; call symputx('vmcnt', vcnt); run;
proc sql noprint; select tablevar into :v1-:v&vmcnt from varcnt; quit;
proc sql noprint; select max(varcnt), compress('&x'||put(varcnt, 10.))
into :varcount, :tempvar separated by ' ' from varcnt order by varcnt; quit;
proc sql noprint; select tablevar into :x1-:x&end10 from varcnt; quit;
proc sql noprint; select count(*) into :obs cnt from inset_categorical; quit;

%macro stkchar;
%do i=1 %to &vmcnt;
data v&i;
length tablevar $ 32;
set inset_categorical(keep=&v&i rename=(&v&i=origvalue));
tablevar="&v&i";
format tablevar $32.; run;

proc sql;
create table vmoreranked&i as
select tablevar, origvalue as rankvmore, count(*) as totalcnt
from v&i
group by 1, 2; quit;

```

```

data vmoreranked&i; length rankvmore $ 12; set vmoreranked&i; run; %end;
%mend;
%stkchar;

data stackorig;
length rankvmore $ 32;
set vmoreranked1-vmoreranked&vmcnt;
format rankvmore $32.;
informat rankvmore $32.; run;

%macro outshell;
%do i=1 %to &varcount;
%macro ybinamax;
%if &ytype=binary %then %do;
proc univariate data=inset_categorical noprint;
    var &target;
    output out=pctoutlier(keep=pcty_100 rename=(pcty_100=capy))
        pctlpts=100
        pctlpre=pcty_; run; %end;
%else %do;
proc univariate data=inset_categorical noprint;
    var &target;
    output out=pctoutlier(keep=pcty_&capy rename=(pcty_&capy=capy))
        pctlpts=&capy
        pctlpre=pcty_; run; %end;
%end;
%ybinamax;

data try_model;
length &&x&i $ 32;
if _n_=1 then set pctoutlier;
set inset_categorical;
if &&x&i=' ' then &&x&i="&charmmiss";
if &target > capy then &target=capy;
format &&x&i $32.;
informat &&x&i $32.; run;

proc sql noprint; select count(*) into :nonmiss from try_model where &&x&i ne
"&charmmiss"; quit;

%macro binary;
proc sql noprint;
select sum(case when &target=1 then 1 else 0 end), sum(case when &target=0
then 1 else 0 end), count(*)
    into :tot_bad, :tot_good, :tot_both
from try_model; quit;

proc sql;
create table woe&i as
(select "&&x&i" as tablevar,
    &&x&i as tier,
    count(*) as cnt,
    count(*)/&tot_both as cnt_pct,
    sum(case when &target=0 then 1 else 0 end) as sum_good,
    sum(case when &target=0 then 1 else 0 end)/&tot_good as dist_good,
    sum(case when &target=1 then 1 else 0 end) as sum_bad,
    sum(case when &target=1 then 1 else 0 end)/&tot_bad as dist_bad,

```

```

        log((sum(case when &target=0 then 1 else 0 end)/&tot_good)/(sum(case
when &target=1 then 1 else 0 end)/&tot_bad))*100 as woe,
        ((sum(case when &target=0 then 1 else 0 end)/&tot_good)-(sum(case
when &target=1 then 1 else 0 end)/&tot_bad))
        *log((sum(case when &target=0 then 1 else 0
end)/&tot_good)/(sum(case when &target=1 then 1 else 0 end)/&tot_bad)) as
pre_iv,
        sum(case when &target=1 then 1 else 0 end)/count(*) as outcomesum

from try_model
group by "&x&i", &x&i)
order by &x&i; quit;

ods results off;
ods listing close;
ods output chisq=CHISQDAT_temp;
proc freq data=try_model;
tables &x&i*&target / NOCOL NOPERCENT CHISQ noprint; run;
ods listing;

data chisquare&i(keep=tablevar value rename=(value=chisquare));
set chisqdat_temp;
IF STATISTIC='Chi-Square';
table=substr(Table,7,49);
tablevar=substr(table, 1, index(table, "*")-2); run;
%mend;

%macro continu;
proc means data=try_model mean median;
var &target;
output out=check_mean_median(drop=_type_ _freq_)
    mean=ymeanvalue
    median=ymedianvalue; run;

data mean_median_combo(keep=ymeanmedvalue);
set check_mean_median;
ymeanmedvalue=sum(&mean_median_combo*ymeanvalue, (1-
&mean_median_combo)*ymedianvalue);
call symputx('ymeanmed', ymeanmedvalue); run;

data check_mod;
if _N_ = 1 then set mean_median_combo;
set try_model;

if &target > ymeanmedvalue then do; ytarget_good=&target-ymeanmedvalue;
ytarget_bad=0; end;
else if &target le ymeanmedvalue then do; ytarget_bad=ymeanmedvalue-&target;
ytarget_good=0; end; run;

proc sql noprint; select sum(ytarget_bad), sum(ytarget_good), count(*)
into :tot_bad, :tot_good, :tot_both from check_mod; quit;

proc sql;
create table woe&i as
(select "&x&i" as tablevar,
    &x&i as tier,
    count(*) as cnt,

```

```

        count(*)/&tot_both as cnt_pct,
        sum(ytarget_bad) as sum_bad,
        sum(ytarget_bad)/&tot_bad as dist_bad,
        log((sum(ytarget_bad)/&tot_bad)/(sum(ytarget_good)/&tot_good))*100 as
woe,
        (sum(ytarget_bad)/&tot_bad-sum(ytarget_good)/&tot_good)
*log((sum(ytarget_bad)/&tot_bad)/(sum(ytarget_good)/&tot_good)) as pre_iv,
        sum(&target)/count(*) as outcomesum
    from check_mod
    group by "&x&i", &x&i)
order by &x&i; quit;

data woe&i;
length tier $ 32;
set woe&i;
format tier $32.;
informat tier $32.; run;

data check_for_anova(keep=&x&i); set check_mod; run;

proc contents data=check_for_anova out=check_contents(keep=name); run;
proc sql noprint; select name into :anovalab from check_contents; quit;
%put &anovalab;

ods results on;
PROC ANOVA DATA=check_mod
    outstat=chisquaretemp(where=(tablevar ne 'ERROR')
        keep=_source_ F rename=(_source_=tablevar F=chisquare)
    );
    CLASS &x&i;
    MODEL &target=&x&i;
    MEANS &x&i/TUKEY CLDIFF; run;

data chisquare&i;
length tablevar $ 32.;
set chisquaretemp;
tablevar="&anovalab";
format tablevar $32.;
informat tablevar $32.; run;
%mend;
%&ytype;

proc sql;
create table iv&i as select "&x&i" as tablevar,
                        sum(pre_iv) as iv,
                        (1-&nonmiss/&obsct) as pct_missing
from woe&i; quit; %end;
%mend outshell;
%outshell;

%macro stackset;
%do j=1 %to 10;
data tempiv&j;
length tablevar $ 32;
set iv&start&j-iv&end&j;
format tablevar $32.; run;

```

```

data tempwoe&j;
length tablevar $ 32;
length tier $ 32;
set woe&&start&j-woe&&end&j;
format tablevar $32.;
format tier $32.; run;

data tempchisquare&j;
length tablevar $ 32;
set chisquare&&start&j-chisquare&&end&j;
format tablevar $32.; run; %end;
%mend;
%stackset;

data &libdata..ivall; set tempiv1-tempiv10; if substr(tablevar, 1,
14)='insertchartemp' then delete; run;
data &libdata..woeall; set tempwoe1-tempwoe10; if substr(tablevar, 1,
14)='insertchartemp' then delete; run;
data &libdata..chisquareall; set tempchisquare1-tempchisquare10; if
substr(tablevar, 1, 14)='insertchartemp' then delete; run;

proc sort data=&libdata..chisquareall nodupkey
out=chisquare_nodup(keep=chisquare); by descending chisquare; run;
data chisquare_nodup; set chisquare_nodup; power_order+1; run;
proc sort data=chisquare_nodup; by chisquare; run;
proc sort data=&libdata..chisquareall; by chisquare; run;

data &libdata..chisquareall;
merge &libdata..chisquareall chisquare_nodup;
by chisquare; run;

proc sort data=&libdata..ivall; by descending iv; run;
data &libdata..ivall; set &libdata..ivall; ivrank+1; run;
proc sort data=&libdata..ivall nodupkey out=ivtemp(keep=iv); by descending
iv; run;
data ivtemp; set ivtemp; ivtier+1; run;
proc sort data=ivtemp; by iv; run;
proc sort data=&libdata..ivall; by iv; run;

data ivouttmp;
length type $ 4;
merge &libdata..ivall ivtemp; by iv;
ivrank=ivtier;
type='char';
format type $4.;
informat type $4.;
drop ivtier; run;

proc sort data=ivouttmp nodupkey; by tablevar; run;
proc sort data=&libdata..woeall; by tablevar; run;
proc sort data=&libdata..chisquareall nodupkey; by tablevar; run;
proc sort data=&libdata..ivall nodupkey; by tablevar; run;

data &libdata..ivout;
merge ivouttmp &libdata..chisquareall;
by tablevar; run;

```

```

proc sort data=&libdata..ivall; by tablevar; run;

data &libdata..iv_woe_all;
length tablevar $ 32;
merge &libdata..ivall &libdata..woeall &libdata..chisquareall;
by tablevar;
type="char"; run;

proc sort data=&libdata..iv_woe_all; by tablevar tier; run;
proc sort data=stackorig; by tablevar rankvmore; run;

data &libdata..iv_woe_all2;
length tablevar $ 32;
merge &libdata..iv_woe_all(in=t) stackorig(in=s rename=(rankvmore=tier));
by tablevar tier; if t; run;

proc sort data=&libdata..iv_woe_all2; by ivrank tier; run;

%let retvar=tablevar iv ivrank tier cnt cnt_pct dist_bad woe outcomesum
pct_missing type chisquare power_order;
data &libdata..&woeout(keep=&retvar);
retain &retvar;
set &libdata..iv_woe_all2;

format binmin &formtall;
format binmax &formtall;
informat binmin &formtall;
informat binmax &formtall;

label tablevar="Variable";
label iv="IV";
label ivrank="IV Rank";
label tier="Tier/Bin";
label cnt="# Customers";
label cnt_pct="% CustoeMrs";
label woe="Weight of Evidence";
label outcomesum="&outname";
label pct_missing="% Missing Values"; run;

proc sort data=&libdata..ivout; by tablevar; run;
data &libdata..&ivout; set &libdata..ivout; by tablevar; run;
proc sort data=&libdata..&woeout out=&woeout._char; by ivrank tablevar; run;
proc sort data=&libdata..&ivout out=&ivout._char(keep=iv ivrank pct_missing
tablevar type chisquare power_order); by ivrank; run;

```

## APPENDIX 2

**\*\* Make sure you have saved Appendix 1a & Appendix 1b into a SAS folder;**

```

** Part I - SPECIFY THE DATA SET AND VARIABLES;

%let numfile=C:/SAS/Appendix 1a.sas; /* to process numeric variables */
%let charfile=C:/SAS/Appendix 1b.sas; /* to process character variables */
libname your_lib "C:/SAS/data"; /* the libname */
%let libdata=your_lib; /* libname defined in the previous line */
%let inset=yousasfile; /* your SAS data set */
%let vartxt= xchar1 xchar2 ...; /* all categorical & ordinal variables.

```

```

Variables should be in character format only. Set vartxt= (blank) if none.
                                Ensure no duplications in variables names */
%let varnum= xnum1 xnum2 ...; /* all numeric variables. Set varnum= if none.
                                Ensure no duplications in variables names */
%let target=sales_amt; /* target variable (y) */
%let ytype=contin; /* continu=continuous outcome, binary=binary outcome */
%let outname=% Sales Volume; /* label of target y for summary outputs */
%let targetvl=15.2; /* change to percent8.3 or others for binary outcomes */
%let libout=C:/SAS/output; /* folder for outputting summaries */

**Part II - CHANGE THE FOLLOWING AS YOU SEE FIT;
%let tiermax=10; /* maximum number of bins for continuous variables */
%let ivthresh=0; /* For a continuous outcome, always set this to 0.
                  For a binary outcome, set ivthresh=0.02 or higher */
%let mean_median_combo=1; /* benchmark for judging high or low occurrence.
                           1=100% mean, 0=100% median, 0.5=avg of mean & median.
                           One can choose any ratio deemed suitable.
                           It has no impact on binary outcomes */
%let capy=98; /* outlier cap for continuous outcomes.
               It has no impact on a binary outcome */
%let tempmiss=-1000000000000; /* for missing numeric values */
%let charmiss=_MISSING_; /* for missing categorical/ordinal values */
%let formtall=12.3; /* format accommodating all numeric variables.
                    Extend to more digits after decimal to accommodate very small values */
%let outgraph=gh_&target; /* pdf collage of variable behavior */
%let ivout=iv_&target; /* output file for IV */
%let woeout=woe_&target; /* output file for WOE */

**Part III - DO NOT CHANGE ANYTHING FROM HERE ON;
%macro kickchar;
%let charcount = %sysfunc(countw(&vartxt dummyfill));
%if &charcount > 1 %then %do; %include "&charfile"; %end;
%mend;
%kickchar;

%macro kicknum;
%let numcount = %sysfunc(countw(&varnum dummyfill));
%if &numcount > 1 %then %do; %include "&numfile"; %end;
%mend;
%kicknum;

%macro to_excel(data_sum);
PROC EXPORT DATA=&data_sum OUTFILE="&libout/&data_sum" label DBMS=tab
REPLACE; run;
%mend;

%macro chknum(datanum);
%if %sysfunc(exist(&datanum)) %then %do;
data ivnum; set &ivout._num; run;
data woenum; set &woeout._num; run; %end;
%else %do;
data ivnum; set _null_; run;
data woenum; set _null_; run; %end;
%mend;
%chknum(&ivout._num);

%macro chkchar(datachar);

```



```

%if %sysfunc(exist(&datachar)) %then %do;
    data ivchar; set &ivout._char; run;
    data woechar; set &woeout._char; run; %end;
%else %do;
    data ivchar; set _null_; run;
    data woechar; set _null_; run; %end;
%mend;
%chkchar(&ivout._char);

data ivout; set ivnum ivchar; run;
proc sort data=ivout; by descending iv; run;
data ivout; set ivout(drop=ivrank); ivrank+1; run;
data woeout; set woenum woechar; run;
proc sort data=woeout; by tablevar; run;
proc sort data=ivout; by tablevar; run;
data woeout; merge woeout(drop=ivrank) ivout; by tablevar; run;
proc sort data=woeout; by ivrank
%macro selnum(datanum);
%if %sysfunc(exist(&datanum)) %then %do;
seqnum
%end;
%mend;
%selnum(&ivout._num);
; run;
proc sort data=ivout; by ivrank; run;

%let ivvar=tablevar type iv ivrank pct_missing separate_rank;
data &ivout(keep=&ivvar);
retain &ivvar;
set ivout;
if type="char" then separate_rank=compress("char-"||power_order);
if type="num" then separate_rank=compress("num-"||power_order);
if power_order=. then separate_rank="NA"; run;

%let woevar=tablevar type iv ivrank cnt cnt_pct outcomesum woe pct_missing
tier separate_rank;
data &woeout(keep=&woevar);
retain &woevar
%macro selnum3(datanum);
%if %sysfunc(exist(&datanum)) %then %do;
binmin binmax
%end;
%mend;
%selnum3(&ivout._num);
;
set woeout(drop=dist_bad);
%macro selnum2(datanum);
%if %sysfunc(exist(&datanum)) %then %do;
drop seqnum tier_num; %end;
%mend;
%selnum2(&ivout._num);
if type="char" then separate_rank=compress("char-"||power_order);
if type="num" then separate_rank=compress("num-"||power_order);
if power_order=. then separate_rank="NA";
run;
%to_excel(&woeout);
%to_excel(&ivout);

```

```

proc sql noprint;
select count(distinct ivrank) into :cntgraph
from woeout
where iv > &ivthresh; quit;
data _null_; call symputx('endlabel', &cntgraph); run;
proc sql noprint;
select tablevar, iv
into :tl1-:tl&endlabel, :ivr1-:ivr&endlabel
from &ivout
where ivrank le &cntgraph
order by ivrank; quit;
proc template;
define style myfont;
parent=styles.default;
style GraphFonts /
'GraphDataFont'=("Helvetica",8pt)
'GraphUnicodeFont'=("Helvetica",6pt)
'GraphValueFont'=("Helvetica",9pt)
'GraphLabelFont'=("Helvetica",12pt,bold)
'GraphFootnoteFont' = ("Helvetica",6pt,bold)
'GraphTitleFont'=("Helvetica",10pt,bold)
'GraphAnnoFont' = ("Helvetica",6pt)
; end; run;

ods results on;
ods pdf file="&libout/&outgraph..pdf" style=myfont;
%macro drgraph;
%do j=1 %to &cntgraph;
proc sql noprint; select sum(case when type='char' then 0 else 1 end) into
:numchar from woeout where ivrank=&j; quit;
%macro gnum;
%if &numchar > 0 %then %do;
proc sgplot data=woeout(where=(ivrank=&j));
vbar tier_num / response=cnt nostatlabel nooutline
fillattrs=(color="salmon");
vline tier_num / response=outcomesum datalabel y2axis lineattrs=(color="blue"
thickness=2) nostatlabel;
label cnt="# records";
label outcomesum="&outname";
label tier_num="&tl&j";
keylegend / location = outside position = top noborder
title = "&j.A - &outname & distribution";
format cnt_pct percent7.4;
format outcomesum &targetvl;
format cnt 15.0; run;

proc sgplot data=woeout(where=(ivrank=&j));
vbar tier_num / response=woe nostatlabel;
vline tier_num / response=cnt_pct y2axis lineattrs=(color="salmon"
thickness=2) nostatlabel;
label woe="Weight of Evidence";
label cnt_pct="% records";
label outcomesum="&outname";
label tier_num="&tl&j";
keylegend / location = outside position = top noborder
title = "&j.B - Information Value: &ivr&j (Rank: &j)";
format cnt_pct percent7.4;

```

```

format cnt 15.0; run;

proc sgplot data=woeout(where=(ivrank=&j and tier_num ne &tempmiss));
series x=tier_num y=cnt;
series x=tier_num y=outcomesum / y2axis;
label woe="Weight of Evidence";
label cnt_pct="% records";
label outcomesum="&outname";
label tier_num="&&tl&j";
keylegend / location = outside
position = top
noborder
title = "&j.C - Variable Behavior";
footnote "Missing numeric values are not drawn.";
format cnt_pct percent7.4;
format outcomesum &targetvl;
format cnt 15.0; run; %end;
%mend ghnum;
%ghnum;

%macro ghchar;
%if &numchar=0 %then %do;
proc sgplot data=woeout(where=(ivrank=&j));
vbar tier / response=cnt nostatlabel nooutline fillattrs=(color="salmon");
vline tier / response=outcomesum datalabel y2axis lineattrs=(color="blue"
thickness=2) nostatlabel;
label cnt="# records";
label outcomesum="&outname";
label tier="&&tl&j";
keylegend / location = outside
position = top
noborder
title = "&j.A - &outname & distribution";
format cnt_pct percent7.4;
format outcomesum &targetvl; run;

proc sgplot data=woeout(where=(ivrank=&j));
vbar tier / response=woe nostatlabel;
vline tier / response=cnt_pct y2axis lineattrs=(color="salmon" thickness=2)
nostatlabel;
label woe="Weight of Evidence";
label cnt_pct="% records";
label outcomesum="&outname";
label tier="&&tl&j";
keylegend / location = outside
position = top
noborder
title = "&j.B - Information Value: &&ivr&j (Rank: &j)";
format cnt_pct percent7.4; run;

proc sort data=woeout(where=(ivrank=&j)) out=chargraph2; by descending
outcomesum; run;
data chargraph2; set chargraph2; yrank+1; run;
proc sgplot data=chargraph2;
vbar yrank / response=cnt datalabel nostatlabel nooutline
fillattrs=(color="salmon");

```

```

vline yrank / response=outcomesum datalabel=tier y2axis
lineattrs=(color="blue" thickness=2) nostatlabel;
label cnt="# records";
label outcomesum="&outname";
label yrank="&&t1&j";
keylegend / location = outside
position = top
noborder
title = "&j.C - Variable Behavior";
format cnt_pct percent7.4;
format outcomesum &targetvl; run; %end;
%mend ghchar;
%ghchar;
%end;
%mend drgraph;
%drgraph;
ods pdf close;

```

### Appendix 3 (for very big data)

**\*\* Make sure you have saved Appendix 1a & Appendix 1b into a SAS folder;**

```

** Part I - SPECIFY THE DATA SET AND VARIABLES;
%let numfile=C:/SAS/Appendix 1a.sas; /* to process numeric variables */
%let charfile=C:/SAS/Appendix 1b.sas; /* to process character variables */
libname your_lib "C:/SAS/data";
%let libdata=your_lib; /* libname */
%let inset=yoursasdata; /* input SAS data set */
%let target=sales_amt; /* target variable (y) */
%let ytype=continuu; /* continu=continuous outcome, binary=binary outcome */
%let outname=Sales Amt; /* label of target y */
%let targetvl=15.2; /* format of dependent variable */
%let libout=C:/SAS/output; /* output folder */

** partitioned lists of all variables;
%let listnum=30; /* # expand to as many lists as needed */
%let varlist1=X1 X2 ... X100;
%let varlist2=X101 X102 ... X200;
%let varlist3=X201 X202 ... X300;
.....
%let varlist30=X2901 X2902 ... X3000; /* correspond to macro value for listnum */

**Part II - CHANGE THE FOLLOWING AS YOU SEE FIT;
%macro getiv;
%do z=1 %to &listnum;
%let vartxt=;
%let varnum=&&varlist&z;
%let tiermax=10;
%let ivthresh=0;
%let capy=98;
%let mean_median_combo=1;
%let tempmiss=-1000000000000; /* missing values for numeric variables */
%let charmiss=_MISSING_; /* missing values for categorical variables */
%let formtall=12.3; /* format accommodating all numeric variables.
Extend to more digits after decimal to accommodate very small values */
%let outgraph=ghpart&z; /* name of pdf graph for top predictors */
%let ivout=ivpart&z; /* name of output file for Information Value */
%let woeout=woepart&z; /* name of output file for Weight of Evidence */

```

```

        ** NO NEED TO CHANGE ANYTHING FROM HERE ON;
%macro kickchar;
    %let charcount = %sysfunc(countw(&vartxt dummyfill));
    %if &charcount > 1 %then %do; %include "&charfile"; %end;
%mend;
%kickchar;

%macro kicknum;
    %let numcount = %sysfunc(countw(&varnum dummyfill));
    %if &numcount > 1 %then %do; %include "&numfile"; %end;
%mend;
%kicknum;

%macro to_excel(data_sum);
PROC EXPORT DATA=&data_sum OUTFILE="&libout/&data_sum" label DBMS=tab
REPLACE; run;
%mend;

%macro chknum(datanum);
    %if %sysfunc(exist(&datanum)) %then %do;
        data ivnum; set &ivout._num; run; %end;
    %else %do;
        data ivnum; set _null_; run; %end;
%mend;
%chknum(&ivout._num);

%macro chkchar(datachar);
    %if %sysfunc(exist(&datachar)) %then %do;
        data ivchar; set &ivout._char; run; %end;
    %else %do;
        data ivchar; set _null_; run; %end;
%mend;
%chkchar(&ivout._char);

data ivout; set ivnum ivchar; run;
proc sort data=ivout; by descending iv; run;
data &ivout; set ivout(drop=ivrank); ivrank+1; run;
%to_excel(&ivout);
data &libdata..&ivout; set &ivout; run; %end;
%mend;
%getiv;

data iv_temp;
set &libdata..ivpart1-&libdata..ivpart&listnum; run;
proc sort data=iv_temp nodupkey out=ivrank; by iv; run;
proc sort data=ivrank(keep=iv); by descending iv; run;
data ivrank; set ivrank; ivrank+1; run;
proc sort data=ivrank; by iv; run;
proc sort data=iv_temp; by iv; run;

data &libdata..iv_for_all(keep=tablevar type iv ivrank);
merge iv_temp(drop=ivrank) ivrank; by iv; run;

proc sort data=&libdata..iv_for_all out=iv_for_all; by ivrank; run;
%to_excel(iv_for_all);

```